NAVIgator: Exploring the Voltage Limits of AMD NAVI GPUs for Energy Efficient Computing

Maria Trakosa Odysseas Chatzopoulos George Papadimitriou Dimitris Gizopoulos
University of Athens, Greece
{mariatrak | od.chatzopoulos | georgepap | dgizop}@di.uoa.gr

***Abstract*—As semiconductor fabrication scales to smaller technology nodes, process variation has become a significant challenge, affecting power consumption, thermal behavior, and voltage stability in microprocessors and GPUs. Conservative voltage guardbands are traditionally used to ensure reliable operation under worst-case process, voltage, and temperature (PVT) variations, but they lead to excessive power consumption. Reducing the supply voltage, while maintaining a fixed frequency, has emerged as a promising technique for improving energy efficiency without sacrificing computational correctness and performance. While extensive research has been conducted on reducing the voltage levels in CPUs and NVIDIA GPUs, AMD GPUs remain relatively unexplored, particularly in terms of process variation. This variation, inherent in semiconductor manufacturing, results in differences in power efficiency, thermal characteristics, and voltage stability even among identical GPUs from the same production batch. In this paper, we present an extensive study on voltage scaling beyond nominal conditions for three modern AMD NAVI GPUs (i.e., RX 7600 XT, 7700 XT, and 7800 XT) executing both conventional benchmarks and PyTorch-based machine learning workloads. We evaluate and present power savings and execution stability under undervolted conditions, highlighting the impact of chip-to-chip variability. Our findings contribute to a deeper understanding of undervolting in AMD GPUs and its dependence on process variation, providing insights into practical power-saving strategies.**

***Index Terms*—GPU, design margins, voltage scaling, energy efficiency, chip characterization, process variation.**

I. Introduction

As semiconductor technology scales, process variation increasingly challenges CPU and GPU architectures. Fabrication variations cause transistor inconsistencies [1], resulting in differences in power, leakage, and performance among identical GPU models. Traditionally, conservative guardbands—voltage margins designed for worst-case process, voltage, and temperature (PVT) conditions—are employed. However, these guardbands often lead to unnecessary power consumption [2], [3], especially for chips capable of stable operation at lower voltages under typical conditions. Prior studies have revealed high voltage guardbands in modern CPUs [4]–[6], indicating significant potential for improving power consumption. This is even more important for GPUs, which typically consume more energy than general-purpose CPUs.

Power efficiency in GPU computing is critical, as high-performance GPUs often exceed 350 W, significantly surpassing CPUs, which typically operate under 150 W. Manufacturers and researchers have explored power-saving mechanisms, such as dynamic voltage and frequency scaling (DVFS), power gating, and clock gating [7], but these methods often overlook chip-to-chip variability. Reducing supply voltage beyond nominal conditions for a fixed clock frequency, can significantly improve GPU energy efficiency without sacrificing performance. Prior research has demonstrated substantial power savings by lowering voltages to the minimum stable operating point (V_{min}) [8]. However, existing studies predominantly target NVIDIA GPUs and CPUs, frequently employing predictive voltage models, leaving AMD GPUs relatively unexplored.

This study characterizes undervolting in three AMD NAVI GPUs (RX 7600 XT, 7700 XT, and 7800 XT). We empirically evaluate their energy efficiency under reduced voltage levels using conventional HPC [9] kernel and PyTorch-based ML workloads [10]. Additionally, we investigate chip-to-chip variability by analyzing multiple GPUs, quantifying how manufacturing differences impact undervolting effectiveness. Process variation significantly affects power consumption, thermal behavior, and voltage stability within the same GPU model [1], highlighting its importance for robust undervolting strategies. Undervolting efficiency differs across workloads due to varying computational intensities, memory access patterns, and power demands [8], [11]. Traditional scientific and graphics workloads exhibit different power characteristics than ML tasks, which primarily involve tensor operations and matrix multiplications. Effective undervolting strategies must therefore account for workload-specific power profiles.

Process variation introduces further complexity, affecting voltage and frequency scaling across GPUs, causing differences in leakage current, thermal properties, and overall efficiency even within a production batch. We examine these effects in AMD GPUs to quantify their impact on undervolting potential. Undervolting also influences thermal performance and hardware reliability [12]. Reduced voltage typically lowers power dissipation and temperatures, improving cooling efficiency. However, aggressive undervolting can increase vulnerability to transient voltage fluctuations, potentially affecting stability and hardware longevity. Understanding these trade-offs helps establish safe undervolting margins.

Our extensive evaluation measures power consumption and stability across multiple AMD GPUs, exploring how process variation and workload characteristics influence undervolting effectiveness. These findings offer valuable insights for optimizing energy efficiency in modern AMD GPU architectures.

II. Methodology

In this section, we outline the experimental workflow designed to accurately characterize the behavior of the target

TABLE I
GPUs CHARACTERIZED IN THIS WORK. ALL THREE GPU MODELS WHERE RELEASED IN Q3 2023 - Q1 2024

GPU Model	RX 7600 XT	RX 7700 XT	RX 7800 XT
Clock (MHz)	2860	2700	2700
Vnom (mV)	1160	1140	1140
Compute Units	32	54	60
AI Accelerators	64	108	120
VRAM (GB)	16	12	16
Technology	6 nm	5 nm	5 nm

GPUs on undervolted conditions. Our objective is to explore the trade-off between energy efficiency and reliability. While undervolting reduces power consumption, excessive voltage reduction can result in GPU driver crashes, application failures, and silent data corruptions (SDCs). To systematically assess GPU behavior under these conditions, we utilize a diverse set of workloads spanning both high-performance computing (HPC) kernels and machine learning (ML) applications. The characterization process is managed by a custom framework that automates undervolting, workload execution, logging, and result parsing. In the following subsections, we define the GPUs and workloads used in our study, provide a detailed explanation of the undervolting and data collection process, and describe how we interpret the workload execution results.

A. Hardware Setup

In this paper, we evaluate three commercial AMD GPU models based on the RDNA3 [13] architecture—the RX 7600 XT, 7700 XT, and 7800 XT. To account for potential effects of process variation, we test two boards from each model. Table I provides a detailed overview of these GPU models, including the clock speed we set for our experiments, the corresponding nominal voltage, the number of compute units and AI accelerators[1], the amount of VRAM, and the fabrication technology node for each model. To ensure consistency across workloads, we selected an operating frequency close to each card's boost frequency, ensuring that all workloads could run at this specific frequency without encountering performance throttling. This choice allows us to evaluate undervolting effects under stable conditions, as the GPUs remain within their thermal margins throughout the experiments.

B. Workloads

To accurately assess the behavior of the selected GPUs under undervolted conditions, we utilize a diverse set of workloads drawn from both high-performance computing (HPC) and machine learning (ML) applications. Specifically, we employ eight benchmarks from the Rodinia [9] benchmark suite, alongside twelve classification neural network models implemented in PyTorch for inference. A complete list of the selected benchmarks is provided in Table II, which details each workload's runtime (including CPU, GPU runtime and IO), as well as the datasets used for training and inference in the ML workloads. The selected workloads exert varying levels of stress on the GPUs, utilizing both memory and compute resources, ranging from moderate usage to full utilization.

[1]Specialized hardware dedicated explicitly to accelerating AI and machine-learning workloads, particularly matrix and tensor operations common in neural networks.

TABLE II
WORKLOADS EMPLOYED (RUNTIMES MEASURED ON 7600XT GPU)

Category	Benchmark	Dataset	Execution Time (s)
Rodinia [9]	cfd	N/A	1.34
	kmeans		2.40
	leukocyte		2.66
	lud		5.85
	nn		0.61
	nw		5.28
	srad		1.22
	streamcluster		1.20
Pytorch [10]	LeNet5 [14]	MNIST [15]	4.70
	ResNet50 [16]	CIFAR100 [17]	4.86
	VGG16 [18]		5.54
	Alexnet [19]	Imagenet [20]	6.41
	Mobilenet [21]		4.12
	Efficientnet [22]		4.29
	ViT [23]		6.01
	Squeezenet [24]		6.11
	Swin [25]		5.39
	Dense [26]		4.65
	Shufflenet [27]		6.17
	Googlenet [28]		6.78

The Rodinia benchmarks represent a collection of highly parallelizable HPC application kernels, commonly executed on GPUs. These benchmarks are well-suited for evaluating how undervolting impacts computational efficiency and stability under traditional GPU workloads. In contrast, the machine learning workloads cover a diverse range of neural network architectures, spanning different levels of complexity and dataset sizes. Unlike the Rodinia benchmarks, which typically involve a single computationally intensive execution, each ML workload performs a large number of inferences per run, ranging from 1,000 to 10,000 inferences per execution. This distinction allows us to examine how undervolting affects both HPC computations and repeated inference tasks in deep learning applications.

C. Experimental Flow

Fig. 1 illustrates our experimental flow. Our system is configured with Linux kernel version 5.15 and ROCm 6.2.4. To effectively apply our undervolting approach, certain features of the AMDGPU driver must first be enabled, and system parameters must be configured to automatically reset the GPU in case of a crash. The critical feature of the AMDGPU driver that facilitates undervolting is sysfs. Through sysfs, hardware configurations are represented as files, allowing parameters to be adjusted by writing values directly to these files or by reading them to obtain hardware metrics. We leverage this feature to set the GPU frequency to a fixed value throughout the experiment and to apply a gradually changing voltage offset relative to the nominal voltage. Additionally, we use sysfs to log GPU power consumption during each execution of the workloads. All experiments are performed in a climate-

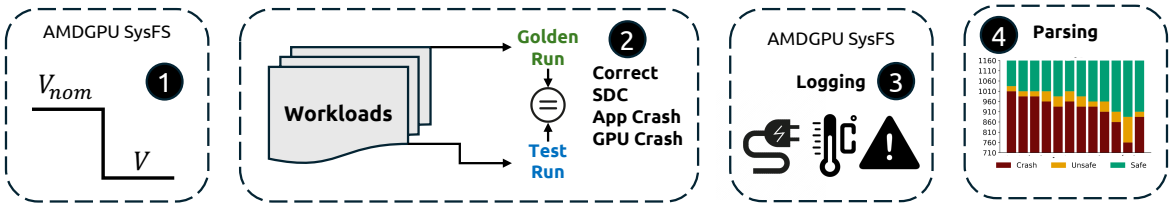

Fig. 1. Experimental flow.

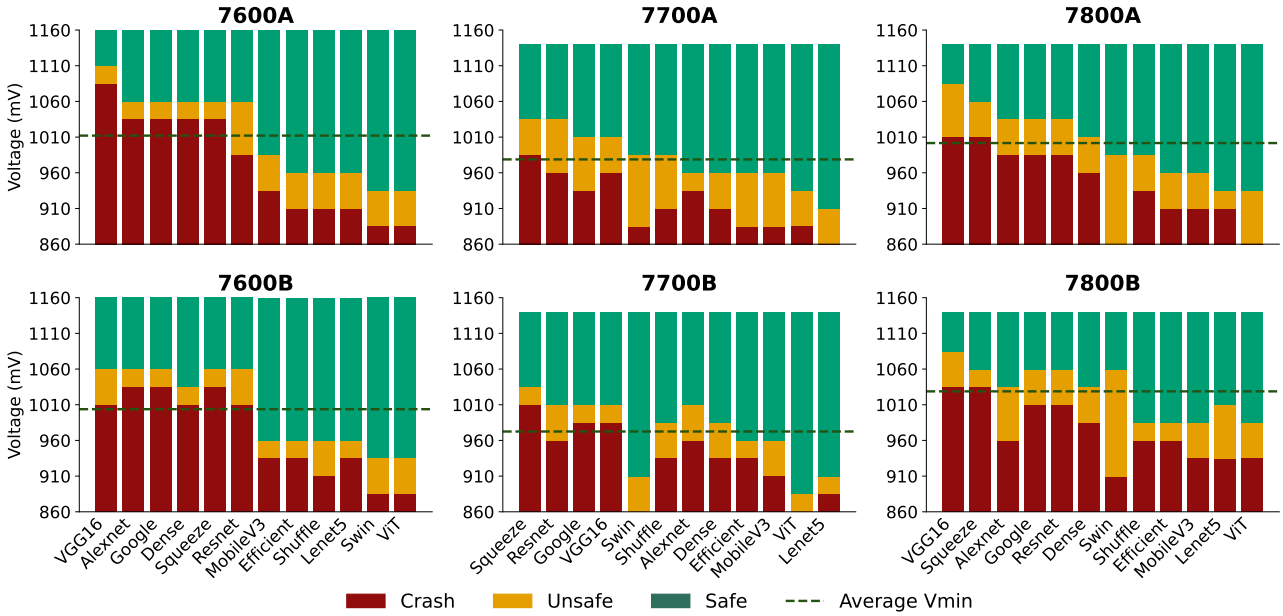

Fig. 2. Safe, unsafe, and crash regions for the six GPUs running ML workloads. The x-axis is shared among GPUs of the same model.

controlled environment to mitigate the impact of ambient temperature variations on characterization accuracy.

To systematically manage undervolting, we have created a custom framework that orchestrates the required steps. The framework accepts a configuration file specifying the workloads to execute, the number of iterations each workload should perform at each voltage level, and the number of inferences per run (for machine learning workloads). This configuration file also defines the initial voltage offset and the voltage step size, which is 10 mV for the Rodinia workloads and 25 mV for the machine learning workloads.

Initially, we perform golden runs for each workload to establish reference outputs. Following this, for each workload, we sequentially traverse voltage levels, decreasing the voltage ❶ incrementally until the GPU crashes. At each voltage step, the workload runs ❷ for the specified number of iterations, with each run generating logs for power consumption, as well as any system error messages encountered during execution ❸. The outcome of each execution is determined immediately. Runs are classified based on whether they result in an application crash, a timeout, or a successful run. Furthermore, by comparing each output against the corresponding golden reference, we identify whether a Silent Data Corruption (SDC) has occurred or if the run was successful. After all experiments have concluded, we parse the collected data to clearly define the Safe region, Unsafe region, and Crash region, and to evaluate the power savings achieved by undervolting ❹.

D. Undervolting Effects

To evaluate undervolting effects, we classify workload execution outcomes based on program outputs, error logs, and GPU driver logs into: **Success** (output matches golden reference); **GPU driver crash** (GPU enters an unrecoverable state, triggering an automatic reset via kernel configuration); **Application crash** (execution ends unsuccessfully due to undervolting instability); **Application timeout** (workload hangs indefinitely, terminated after a predefined period); and **Silent Data Corruption (SDC)** (workload completes but output deviates without system log errors).

SDCs in classification-based ML workloads vary in severity from insignificant (minor probability value changes) to highly significant (altered top-ranking predictions). To quantify SDC impacts, we introduce the following metrics: **Top-1 Accuracy** (percentage of correct top class predictions across all inferences performed) and **Top-5 Accuracy** (percentage of correct top five class predictions across all inferences performed). A successful run achieves both metrics at 1.0, whereas SDC-affected runs may maintain or reduce these accuracies.

We define three voltage regions based on these outcomes: *Crash region* (GPU crashes), *Unsafe region* (workloads exhibit unpredictable behaviors like crashes, timeouts, or SDCs), and *Safe region* (successful and correct workload execution, usable for energy savings). We systematically log power consumption and temperature during each workload to evaluate undervolting's impact on energy efficiency and thermal performance.

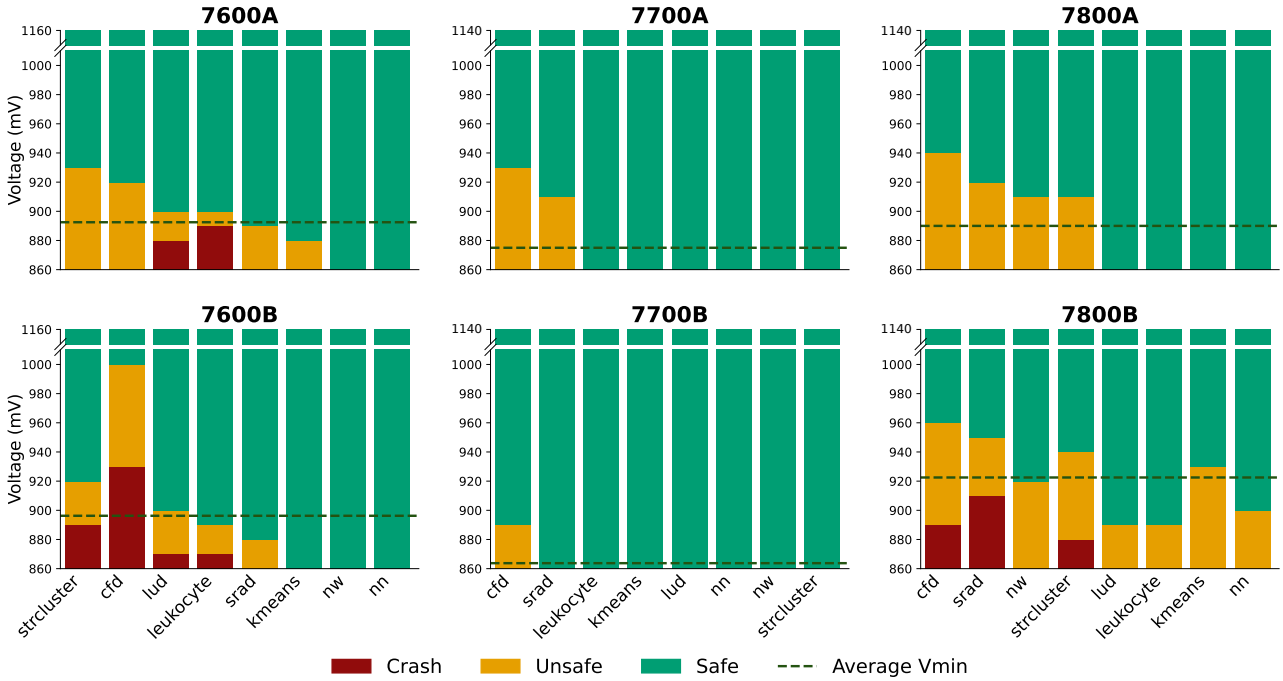

Fig. 3. Safe, unsafe, and crash regions for the six GPUs running Rodinia workloads. The x-axis is shared among GPUs of the same model.

III. Results

In this section we present the results derived using our experimental methodology. We first present the voltage margins of the GPUs in question for the Rodinia and ML workloads. Then we show the impact of undervolting on power consumption and accuracy of the ML workloads (we do not show these results for the HPC workloads due to limited space).

A. GPU Voltage Margins

Fig. 2 and Fig. 3 present the characterization results for the voltage margins obtained from the ML and HPC benchmarks across all six GPUs. These figures highlight the significant variability not only between different GPU models and workloads, but also among individual cards of the same model. For each workload and GPU, we illustrate the crash, unsafe, and safe voltage regions as defined in the methodology section. The lowest voltage within the safe region is referred to as V_{min}. This value ranges from 4% to 25% below the nominal voltage ($V_{nominal}$) and is generally lower for the Rodinia benchmarks. Rodinia benchmarks represent common HPC kernels which, even with the largest available input datasets, exhibit relatively short GPU runtimes. Their bursty nature results in less pronounced undervolting effects (some of these workloads can reach the full -300 mV offset allowed, without even experiencing a single SDC) compared to ML workloads, which place the GPU under sustained load for much longer periods. While we observe different behaviors between the two workload groups, the general trends between the GPU models remain consistent. We can see that the 7700 cards exhibit the largest voltage-margins followed by the 7600 and 7800-series cards. The differences observed between cards of the same model for a given workload are attributed to process variation. For example we can clearly see that the 7800A GPU has signficantly lower V_{min} than its B counterpart. Our results are broadly consistent with prior findings on NVIDIA architectures. For example, Leng et al. [8] directly measured a 20% voltage guardband on multiple NVIDIA GPU generations.

The unsafe region spans from 10 mV to 150 mV, indicating that some workloads experience gradual failure modes as voltage decreases—initially showing a few SDCs (Silent Data Corruptions), followed by SDCs combined with application crashes. In contrast, other workloads display a more binary behavior, running successfully until a sudden GPU crash occurs. Especially for the machine learning workloads, considering their general error tolerance [29], the unsafe region could be exploited for further power savings. To quantify the effect of operating in the unsafe region we present the Top-1 and Top-5 accuracy drop in the following subsection.

Focusing on the ML workloads, we observe that the less computationally intensive models, such as LeNet5 and MobileNetV3, are generally less affected by undervolting. This observation aligns with prior work on NVIDIA GPUs, where undervolting tolerance has been shown to correlate with workload intensity and structure [8], [30]. However, this trend is not fully generalizable, especially for higher-level workloads such as ML models, where additional factors—like intricate memory access patterns and model architecture—can significantly influence the effective V_{min}. For example, an interesting exception to this trend is observed with the two transformer-based models, ViT and Swin. Despite being among the most computationally intensive workloads, they exhibit greater resilience to undervolting. This may stem from two architectural traits of transformers. First, the attention

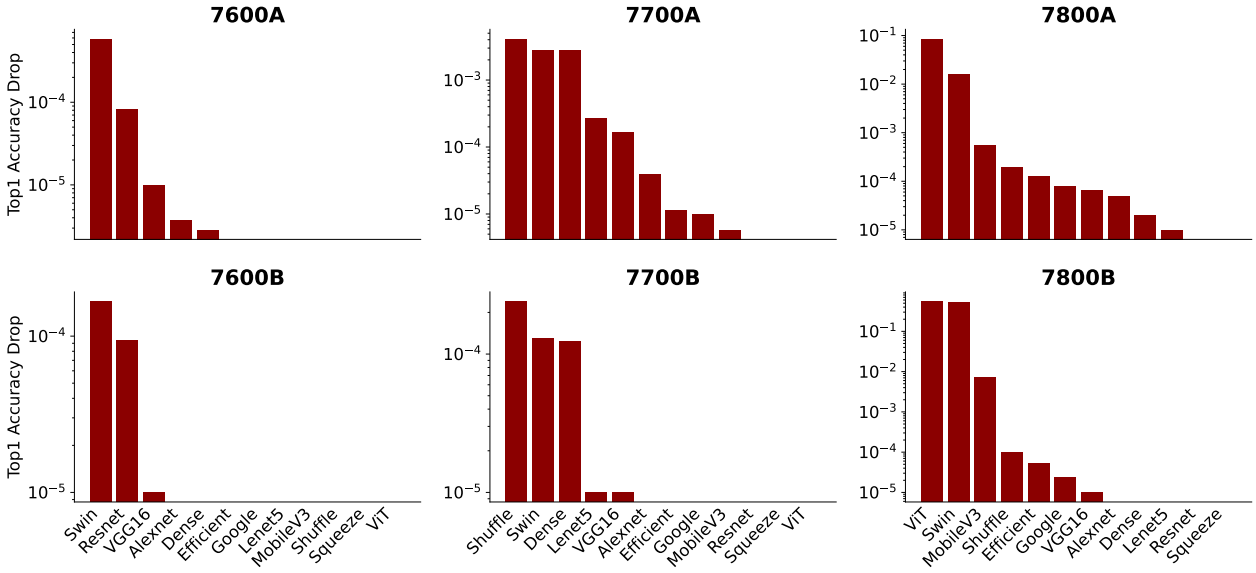

Fig. 4. Top1 accuracy drop in the Unsafe region for the six GPUs running ML workloads. The x-axis is shared among GPUs of the same model.

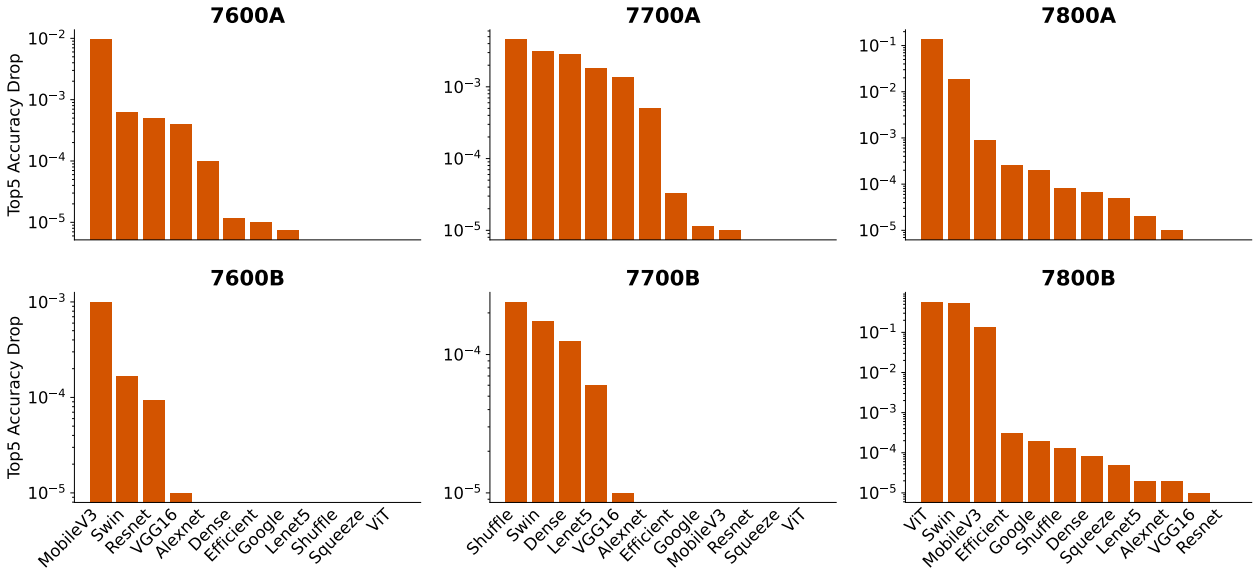

Fig. 5. Top5 accuracy drop in the Unsafe region for the six GPUs running ML workloads. The x-axis is shared among GPUs of the same model.

mechanism aggregates inputs using softmax normalization, which naturally tolerates small numerical errors—minor deviations may not significantly affect attention scores or outputs. Second, transformers leverage multi-head attention and deep-layer parallelism, offering redundancy that can mask localized computation errors. In contrast, Convolutional Neural Networks (CNNs)—particularly in early layers—may propagate even minor inaccuracies, resulting in greater sensitivity to undervolting.

B. ML Model Accuracy in the Unsafe Region

In ML workloads, we categorize SDCs based on their impact on classification accuracy, as described in the methodology section. Fig. 4 and Fig. 5 illustrate the average reduction in Top-1 and Top-5 accuracy across the entire unsafe region for each benchmark. The accuracy drop ranges from 0% to 55% for Top-1 accuracy and up to 57% for Top-5 accuracy. Since in the unsafe region aside from SDCs, application crashes and timeouts also occur, a low drop in Top-1 and Top-5 accuracy might be due to a combination of SDCs that do not lead to misclassifications and runs resulting in crashes or timeouts. In any case, these endpoints are considered benign because crashes and timeouts are easily detected, and voltages can be adaptively scaled up to avoid them. The most concerning outcomes are SDCs with high misclassification rates, as quantified in these figures. Notably, the 7700 and 7800 GPUs exhibit larger accuracy degradations across more workloads, which can be attributed to their wider unsafe voltage regions compared to the 7600. Overall, the impact on accuracy depends on both the neural network's architecture and the specific behavior of the GPU model.

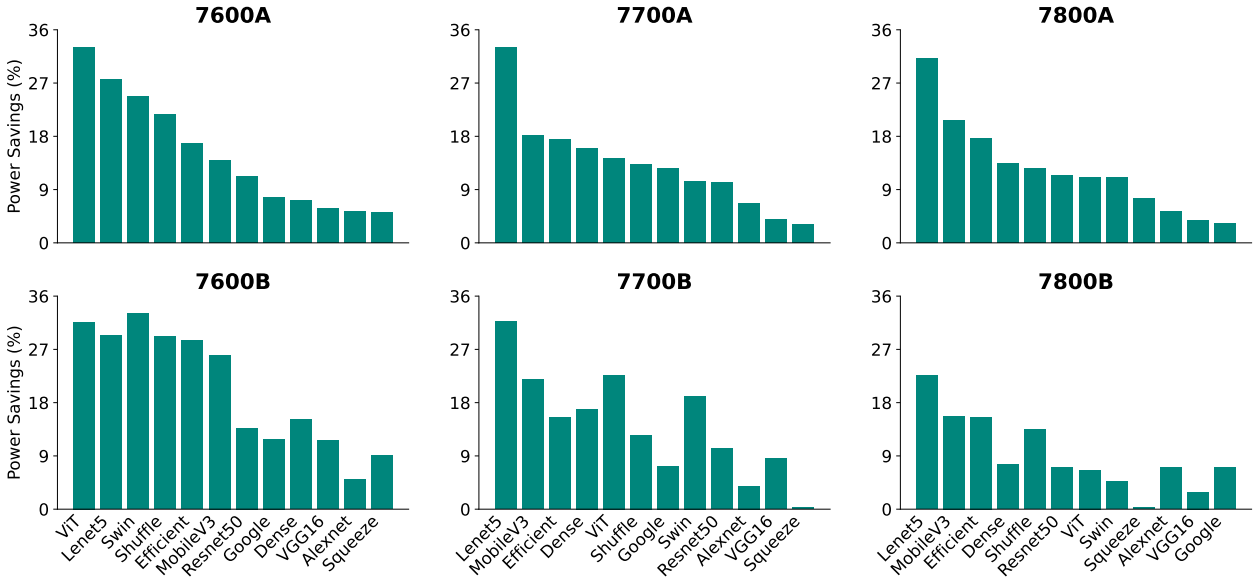

Fig. 6. Power savings for the six GPUs running ML workloads. The x-axis is shared among GPUs of the same model.

C. Power Savings due to Undervolting

The main goal of undervolting is to improve energy efficiency by reducing power consumption during workload execution. Fig. 6 shows the power savings achieved when running each workload at its V_{min} on each GPU. The observed power savings range from 0.24% to 33%, depending on the workload and GPU model. These savings are comparable to those achieved in the NVIDIA-focused studies, where voltage scaling strategies yielded savings up to 25% [8]. The resulting power savings naturally lead to a reduction in GPU temperature. The thermal benefits vary based on both the power reduction and the specific thermal characteristics of each GPU. Overall, undervolting provides noticeable energy and thermal improvements, especially for workloads with sustained GPU utilization. Considering an average power saving of 14% with a mean GPU power consumption of 300W, each GPU saves 42W (0.14 × 300W). For a datacenter with 10,000 GPUs, this amounts to a total saving of 42W × 10,000 = 420,000W, or 420 kW. Over a full day, that translates to 420 kW × 24 hours = 10,080 kWh, which is approximately 3.68 GWh annually.

IV. Related Work

Various techniques have been developed to relax CPU voltage guardbands and enhance energy efficiency. For example, [31] propose heuristics that dynamically reduce voltage margins by leveraging feedback from an on-chip error correction (ECC) mechanism integrated into the Itanium 9560. Meanwhile, [2] employ dedicated hardware protection mechanisms, incorporated at the design stage, to minimize voltage margins. Moreover, studies such as [32]–[38] conduct static offline analyses to quantify voltage margins in various commercial CPUs. The authors in [39] assess the impact of undervolting on soft-error rates. Recent studies on GPU undervolting focused on enhancing energy efficiency and reliability—predominantly on NVIDIA platforms. For instance, Leng et al. [8] examine safe voltage reduction limits using direct measurements, while Tan et al. [40] analyze instruction-level error characteristics under low supply voltages on NVIDIA GPUs. Additionally, Leng et al. [41] address spatial and temporal voltage noise, and predictive guardbanding techniques [30] dynamically adjust voltage margins. Finally, [12] shows a summary of voltage margins on CPUs, NVIDIA GPUs and FPGAs. In contrast, our work extends undervolting evaluation to AMD GPUs, a currently unexplored area.

V. Conclusion

In this work, we presented the first comprehensive study of undervolting (voltage scaling) on three modern AMD NAVI GPUs, evaluating power savings, thermal effects, and execution stability (resilience) across both HPC and machine learning workloads. Our results demonstrate that undervolting can yield significant energy and thermal benefits without compromising correctness, but the extent of these benefits is highly dependent on both the workload characteristics and the chip-to-chip process variation. Overall, our findings provide valuable insights into the practical feasibility of undervolting as an energy-saving technique for AMD GPUs, offering a foundation for future research and optimization in power-efficient GPU computing.

Acknowledgments

This work was supported by the European Union's Horizon Europe research and innovation programme under grant agreements No 101097224 (REBECCA) and No 101093062 (Vitamin-V). Views and opinions expressed are however, those of the authors only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them. This work was also supported by the Hellenic Foundation Research and Innovation (HFRI) project VEMER.

REFERENCES

[1] W. Schemmert and G. Zimmer, "Threshold-voltage sensitivity of ion-implanted m.o.s. transistors due to process variations," *Electronics Letters*, vol. 10, no. 9, p. 151, 1974.

[2] Y. Zu, C. R. Lefurgy, J. Leng, M. Halpern, M. S. Floyd, and V. J. Reddi, "Adaptive guardband scheduling to improve system-level efficiency of the power7+," in *2015 48th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2015, pp. 308–321.

[3] G. Papadimitriou, A. Chatzidimitriou, and D. Gizopoulos, "Adaptive voltage/frequency scaling and core allocation for balanced energy and performance on multicore cpus," in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2019, pp. 133–146.

[4] G. Papadimitriou, M. Kaliorakis, A. Chatzidimitriou, D. Gizopoulos, P. Lawthers, and S. Das, "Harnessing voltage margins for energy efficiency in multicore cpus," in *2017 50th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2017, pp. 503–516.

[5] G. Papadimitriou, M. Kaliorakis, A. Chatzidimitriou, C. Magdalinos, and D. Gizopoulos, "Voltage margins identification on commercial x86-64 multicore microprocessors," in *2017 IEEE 23rd International Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2017, pp. 51–56.

[6] Y. Zu, D. Richins, C. Lefurgy, and V. Reddi, "Fine-tuning the active timing margin (atm) control loop for maximizing multi-core efficiency on an ibm power server," in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2019, pp. 106–119.

[7] F. Mendes, P. Tomás, and N. Roma, "Exploiting non-conventional dvfs on gpus: Application to deep learning," in *2020 IEEE 32nd International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*, 2020, pp. 1–9.

[8] J. Leng, A. Buyuktosunoglu, R. Bertran, P. Bose, and V. J. Reddi, "Safe limits on voltage reduction efficiency in gpus: A direct measurement approach," in *2015 48th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2015, pp. 294–307.

[9] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheafer, S.-H. Lee, and K. Skadron, "Rodinia: A benchmark suite for heterogeneous computing," in *Proceedings of the 2009 IEEE International Symposium on Workload Characterization (IISWC)*. IEEE, 2009, pp. 44–54.

[10] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, *PyTorch: an imperative style, high-performance deep learning library*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[11] D. Gizopoulos, G. Papadimitriou, A. Chatzidimitriou, V. J. Reddi, B. Salami, O. S. Unsal, A. C. Kestelman, and J. Leng, "Modern hardware margins: Cpus, gpus, fpgas recent system-level studies," in *2019 IEEE 25th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2019, pp. 129–134.

[12] G. Papadimitriou, A. Chatzidimitriou, D. Gizopoulos, V. J. Reddi, J. Leng, B. Salami, O. S. Unsal, and A. C. Kestelman, "Exceeding conservative limits: A consolidated analysis on modern hardware margins," *IEEE Transactions on Device and Materials Reliability*, vol. 20, no. 2, pp. 341–350, 2020.

[13] AMD, "Introducing rdna architecture: The all-new radeon gaming architecture powering "navi"," 2019, accessed: [2025-03-10]. [Online]. Available: https://www.amd.com/system/files/documents/rdna-whitepaper.pdf

[14] Y. LeCun *et al.*, "Lenet-5, convolutional neural networks," *URL: http://yann. lecun. com/exdb/lenet*, vol. 20, no. 5, p. 14, 2015.

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[17] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Technical Report*, 2009, available at https://www.cs.toronto.edu/ kriz/cifar.html.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2012, pp. 1097–1105.

[20] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large-scale visual recognition challenge," in *International Journal of Computer Vision*, vol. 115, no. 3, 2015, pp. 211–252.

[21] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for mobilenetv3," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 1314–1324.

[22] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2019, pp. 6105–6114.

[23] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations (ICLR)*, 2021.

[24] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and ¡1mb model size," in *arXiv preprint arXiv:1602.07360*, 2016.

[25] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021, pp. 10 012–10 022.

[26] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4700–4708.

[27] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 6848–6856.

[28] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[29] C. Torres-Huitzil and B. Girau, "Fault and error tolerance in neural networks: A review," *IEEE access*, vol. 5, pp. 17 322–17 341, 2017.

[30] J. Leng, A. Buyuktosunoglu, R. Bertran, P. Bose, Y. Zu, and V. J. Reddi, "Predictive guardbanding: Program-driven timing margin reduction for gpus," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 1, pp. 171–184, 2021.

[31] A. Bacha and R. Teodorescu, "Using ecc feedback to guide voltage speculation in low-voltage processors," in *2014 47th Annual IEEE/ACM International Symposium on Microarchitecture*, 2014, pp. 306–318.

[32] K. Parasyris, P. Koutsovasilis, V. Vassiliadis, C. D. Antonopoulos, N. Bellas, and S. Lalis, "A framework for evaluating software on reduced margins hardware," in *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2018, pp. 330–337.

[33] G. Papadimitriou, M. Kaliorakis, A. Chatzidimitriou, D. Gizopoulos, P. Lawthers, and S. Das, "Harnessing voltage margins for energy efficiency in multicore cpus," in *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO-50 '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 503–516. [Online]. Available: https://doi.org/10.1145/3123939.3124537

[34] G. Papadimitriou, M. Kaliorakis, A. Chatzidimitriou, D. Gizopoulos, G. Favor, K. Sankaran, and S. Das, "A system-level voltage/frequency scaling characterization framework for multicore cpus," in *IEEE Silicon Errors in Logic – System Effects (SELSE 2017)*, 2017.

[35] K. Tovletoglou, L. Mukhanov, G. Karakonstantis, A. Chatzidimitriou, G. Papadimitriou, M. Kaliorakis, D. Gizopoulos, Z. Hadjilambrou, Y. Sazeides, A. Lampropulos, S. Das, and P. Vo, "Measuring and exploiting guardbands of server-grade armv8 cpu cores and drams," in *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, 2018, pp. 6–9.

[36] G. Papadimitriou, A. Chatzidimitriou, M. Kaliorakis, Y. Vastakis, and D. Gizopoulos, "Micro-viruses for fast system-level voltage margins characterization in multicore cpus," in *2018 IEEE International Sym-*

posium on Performance Analysis of Systems and Software (ISPASS), 2018, pp. 54–63.

[37] G. Papadimitriou, M. Kaliorakis, A. Chatzidimitriou, C. Magdalinos, and D. Gizopoulos, "Voltage margins identification on commercial x86-64 multicore microprocessors," in *2017 IEEE 23rd international symposium on on-line testing and robust system design (IOLTS)*. IEEE, 2017, pp. 51–56.

[38] G. Karakonstantis, K. Tovletoglou, L. Mukhanov, H. Vandierendonck, D. S. Nikolopoulos, P. Lawthers, P. Koutsovasilis, M. Maroudas, C. D. Antonopoulos, C. Kalogirou, N. Bellas, S. Lalis, S. Venugopal, A. Prat-Pérez, A. Lampropulos, M. Kleanthous, A. Diavastos, Z. Hadjilambrou, P. Nikolaou, Y. Sazeides, P. Trancoso, G. Papadimitriou, M. Kaliorakis, A. Chatzidimitriou, D. Gizopoulos, and S. Das, "An energy-efficient and error-resilient server ecosystem exceeding conservative scaling limits," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2018, pp. 1099–1104.

[39] D. Agiakatsikas, G. Papadimitriou, V. Karakostas, D. Gizopoulos, M. Psarakis, C. Belanger-Champagne, and E. Blackmore, "Impact of voltage scaling on soft errors susceptibility of multicore server cpus," in *Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 957–971. [Online]. Available: https://doi.org/10.1145/3613424.3614304

[40] J. Tan, J. Wang, K. Yan, X. Wei, and X. Fu, "Evaluating gpu's instruction-level error characteristics under low supply voltages," *IEEE Transactions on Computers*, vol. 74, no. 2, pp. 555–568, 2025.

[41] J. Leng, Y. Zu, and V. J. Reddi, "Gpu voltage noise: Characterization and hierarchical smoothing of spatial and temporal voltage noise interference in gpu architectures," in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, 2015, pp. 161–173.