Accepted manuscript of: Koszczał, G., Dobrosolski, J., Matuszek, M., Czarnul, P. (2024). Performance and Energy Aware Training of a Deep Neural Network in a Multi-GPU Environment with Power Capping. In: Zeinalipour, D., et al. Euro-Par 2023: Parallel Processing Workshops. Euro-Par 2023. Lecture Notes in Computer Science, vol 14352. Springer, Cham. https://doi.org/10.1007/978-3-031-48803-0_1

Performance and energy aware training of a deep neural network in a multi-GPU environment with power capping

Grzegorz Koszczał, Jan Dobrosolski, Mariusz Matuszek[0000−0001−7551−256X] and Paweł Czarnul[0000−0002−4918−9196]

Faculty of Electronics, Telecommunications and Informatics
Gdansk University of Technology, Narutowicza 11/12, 80-233 Poland
pczarnul@eti.pg.edu.pl

Abstract. In this paper we demonstrate that it is possible to obtain considerable improvement of performance and energy aware metrics for training of deep neural networks using a modern parallel multi-GPU system, by enforcing selected, non-default power caps on the GPUs. We measure the power and energy consumption of the whole node using a professional, certified hardware power meter. For a high performance workstation with 8 GPUs, we were able to find non-default GPU power cap settings within the range of 160–200W to improve the difference between percentage energy gain and performance loss by over 15.0%, EDP[1] by over 17.3%, EDS with k=1.5 by over 2.2%, EDS with k=2.0 by over 7.5% and pure energy by over 25%, compared to the default power cap setting of 260W per GPU. These findings demonstrate the potential of today's CPU+GPU systems for configuration improvement in the context of performance-energy consumption metrics.

Keywords: deep neural network training, power capping, multi GPU, performance-energy optimization

1 Introduction and motivation

Training of deep neural networks is a time consuming and a computationally demanding process. In recent years one could have observed a rapid development in the machine learning field, leading to a constant stream of innovation resulting in the emergence of new neural network topologies of growing complexity (e.g. GPT-3: 175 billion parameters, DLRM-2020: 100 billion parameters). Both models of such record breaking complexity, as well as other frequently used models (e.g. BERT: 340M parameters), require an immense amount of computing power in order to train on enormously large datasets (e.g. GPT-3: 45TB of data). This trend of creating more and more power hungry models and creating more extensive data corpse leads to time and energy consuming training processes (e.g. OpenAI Five training took 180 days), resulting in concerns regarding

[1] Abbreviations and terms used are described in main text.

the monetary cost and carbon footprint, correlated with growing significance of deep learning related research. One of the proposed solutions, that would not limit the progression of modern day artificial intelligence (AI), is to utilize the mechanism of power capping,[2] which has already been proven to be able to provide with the desired energy consumption – execution time trade-off in other applications [9].

The main goal of this study is to expand knowledge gained during research regarding power capping during deep neural network (DNN) training on a single GPU [10] by acquiring meaningful data for multi-GPU configurations. In this paper we investigate the degree of potential gains when optimizing performance-energy metrics by application of power caps when performing a deep neural network training in a *multi*-GPU hardware environment. Consideration of multiple GPUs is important as it is a straightforward and commonly applied technique to increase system performance, especially for AI workloads. Additionally, the novelty of our research lies in precise measurements using a professional certified hardware power meter and consideration of whole node's power and - consequently - energy consumption, contrary to power and energy of computing devices, as typically measured using Intel RAPL or NVIDIA NVML. We consider this approach much more practical in situations involving power-performance optimisation of whole clusters running AI workloads.

Additional motivation comes from the current state in the machine learning field. In case of natural language processing, the point where increasing the model size stops positively impacting its performance has not been, and it does not look like it will be reached anytime soon. With a growing network size and constant efforts being made to expand quality data corpse that fits the needs of NLP models, many concerns have been raised in regard to ecological consequences, as well as total cost of conducting work in the AI field (training of GPT-3 model from the ground up is estimated to cost 4.6 million USD, with energy usage as high as 936 MWh) [12]. When taking into consideration that the development process of a new model, or even fine tuning an existing model to a specific task using transfer learning takes multiple training cycles in an unavoidable process of hyper-parameter tuning, the estimated cost and carbon footprint of machine learning is a considerable problem [23] that needs to be tackled with, both for AI application feasibility as well as for environmental reasons. Those issues can be expected to only grow, when considering that models as big as 1.6 trillion parameters (Google's Switch-C model) have been announced.

The remaining part of the paper is structured as follows. Section "Related work" discusses similar research in this field, focusing on energy aware optimizations in AI. The "Methodology" section contains a description of the tested model, data, metrics and conditions necessary to obtain meaningful results. Next

[2] Power capping is a mechanism allowing limiting the power draw of a computing device such as a CPU or a GPU, available through Intel RAPL for Intel CPUs and NVIDIA NVML for NVIDIA GPUs, resulting in potentially lower performance but potential for optimization of energy consumption, even throughout extended application execution time [9–11].

comes the "Experiments" section, containing detailed descriptions of the testbed environment and experiments themselves. This description should help interested parties to reproduce our measurements. Finally, we conclude with a summary and an outline of future work.

2 Related work

Research regarding energy consumption – execution time trade-off is not a novelty, as the topic is constantly being explored, for example in context of multi- and many-core processors [6], or regarding AI workloads using specific frameworks on different platforms [14]. Optimization concerning energy consumption of AI workloads requires methods of estimation of energy and time (such as training or inference). While general frameworks and tools for optimization of energy efficiency of high performance computing (HPC) applications exist (e.g. [17]), a specific energy estimation method and tool has been proposed[3] for a deep neural network model based on its architecture, bitwidth and sparsity [22]. The authors concluded that the number of weights and multiplication-and-accumulation operations are not good metrics for energy consumption estimates and that data movement is more expensive than computation. Energy efficiency of using deep neural networks is considered in the context of using GPUs [24], FPGAs [18], hybrid GPU-FPGA [5] and severely energy and power constrained devices such as unmanned aerial vehicles (UAVs), smartphones [21], Internet of Things (IoT) [16] etc. For example, in [15] authors emphasize the need for energy reduction of machine learning and natural natural language processing (NLP), citing interesting trade-offs between decrease of energy usage and performance loss while training a transformer-based language model, thanks to power capping. Using NVIDIA V100 it was possible to save approx. 12.3% of energy at the cost of 8.5% performance loss for BERT, and to save approx. 15% energy at the cost of approx. 10% performance loss for DistilBERT. Energy efficiency of neural networks training is studied in paper [20], in particular stating correlation between CO_2 emissions and energy consumed and network architectures. Testing VGG16, VGG19 and ResNet50 architectures and CIFAR10, MNIST datasets in various locations, optimizing accuracy/energy led to either improvement of both energy efficiency and accuracy (MNIST) or improvement of energy efficiency with a loss of accuracy (CIFAR10). The authors also compared energy measurements from software-based CodeCarbon (based on hardware performance counters and I/O models), to those from a hardware meter, showing ratio of values from the former to the latter between 42% and 46% (generally software methods measure power of compute devices and memory only). This results in strong correlation but slightly different offsets for various configurations.

Multiple ways of searching for optimal configurations were proposed, one of them being GPOEO [19], an online energy optimization framework for machine learning applications run on GPUs. The framework's course of action can be divided into two phases. Firstly, during the offline phase, the framework collects

[3] https://energyestimation.mit.edu/

performance and energy logs for various frequency of streaming multiprocessors and memory, which is later used during the online phase, where the frequencies that exhibited best characteristics are applied during the training to optimize a function of energy and time.

Similar goals were achieved by using the widely recognised GPU DVFS technique [25], where a defined power limit was used to limit energy consumption while maintaining the optimal resource utilization leading to limiting the performance loss. It is worth mentioning that this technique is well grounded in energy preserving research regarding GPU computing, with a history of great results (14.4% energy savings with 3% performance loss) in non machine learning RODINIA and ISPASS benchmarks [13].

Visible results achieved in reduction of energy consumption by two exploration algorithms (56% energy consumption decrease, 12% execution time increase) showcase the possibilities of power capping (authors used DEPO, an automatic power capping software tool) when it comes to achieving considerable savings of energy while maintaining an acceptable performance loss, that hopefully can be transferred for other computationally complex tasks.

Another interesting mention is that not all approaches to limit the energy consumption during deep learning workloads revolve around power capping. A great example would be the effort made to minimize the energy expenditure of a computing cluster by shutting down unused nodes using a job allocation method using a MINLP formulation [7]. In paper [23] authors proposed a solution for reduction of a carbon footprint of DNN training. The method realized periodical adjustment of the GPU power limit in order to do so and utilizes a regression model for carbon intensity, taking into account historical data. They formulate a cost minimization problem that takes into account power, carbon intensity, throughput (inversely proportional to time-to-accuracy) and a parameter for definition of importance of carbon efficiency and performance of the training. For evaluation training ResNet50 on the ImageNet dataset, run on an NVIDIA A40 GPU was used, demonstrating reduction of carbon emission by 13.6% at the cost of 2.5% increase of training time compared to the default power configuration.

Even though our study revolves around optimizing the training process of a machine learning model, it is worth noting that it is not the only targeted part of AI by the energy consumption lowering efforts. In paper [1] authors managed to achieve high inference accuracy while keeping the process in the range of desired latency and obtaining gains in terms of energy saving. The proposed solution utilized multi-input-multi-output control framework OptimML, that flexibly adjusts the model size, effectively decreasing the required resources for inference depending on the server's set power cap.

In summary, while there exist works optimizing performance-energy metrics, in particular for AI oriented workloads, there is lack of research if and to what degree power capping can be beneficial in multi-GPU setups when optimizing performance-energy metrics.

3 Methodology

In order to create a way to compare different configurations, a convolutional neural network (CNN) using transfer learning technique utilizing the XCeption [2] model and a custom network top was implemented to solve a classification problem, which was based on a reduced version of well grounded dataset for 450 bird species from kaggle.com. The number of classes was reduced to 69 in order to shorten the test duration, while maintaining a considerable amount of time required to reach the assumed model quality metric (98%) accuracy over the test subset containing five entries for each of the classes kept. In order to keep the workload as stable as possible, a constant dataset was sub-sampled from the original dataset into a reduced version that contained 10000 training images, 345 validation images and 345 test images. The training length was established during several pilot training sessions using the early stopping technique, and was set for a constant five epochs for every configuration, as it was proved to converge for every configuration taking into account slight quality drops connected to multi-GPU training when compared to a single GPU approach. It shall be noted that for our purposes, we actually needed to generate a stable training workload for assessment of power capping impact. The reason behind this approach, compared to one using pretrained models without interfering into their top (e.g. Inception-V3, MobileNet, DenseNet [4]), is to be able to mimic a frequent situation, where a developer is using the transfer learning technique to solve a specific problem.

For the presented measurements we used a certified hardware meter Yokogawa WT310E (0.1% basic measurement accuracy according to the manufacturer), with the sampling frequency set to 10Hz. Hardware under test was located in a temperature-controlled server room, with ambient temperature set at 18 deg. Celsius. The ultimate goal was to establish best power capping settings, that provide the user with improved (versus the default power cap configuration) performance-energy conserving trade-offs, in 1/2/4/8 GPUs configurations, measured using the following metrics:

- percentage of energy gain,
- difference between percentage of energy gain and percentage of execution time loss,
- Energy Delay Product (EDP),
- Energy Delay Sum (EDS) – in this case we consider a weighted sum of energy and time $\alpha E + \beta t$, reference point of (t_{ref}, E_{ref}) as well as coefficient k which determines a potential performance loss ratio. This leads to determination of $\alpha = \frac{k-1}{k \cdot E_{ref}}$ and $\beta = \frac{1}{k \cdot t_{ref}}$ [11].

The first metric may be of interest if energy consumption shall be preferred over execution time (such as when the latter is not critical e.g. we are waiting for input data for subsequent computations that are taking much time). The second metric weighs percentage energy gain and percentage performance loss compared to the energy consumption and execution time of the default power

cap configuration. In this case positive values of the metric indicates optimized configurations. EDP and EDS are other frequently used single metrics finding trade-offs between energy consumption and execution time [11].

4 Experiments

4.1 Testbed environment

The tests were performed on an HPC server installed at Gdańsk University of Technology. The system is equipped with the following compute devices: 2x Intel(R) Xeon(R) Silver 4210 @ 2.20GHZ CPUs (10 physical cores, HT; for a total of 40 logical processors), 8x NVIDIA Quadro RTX 6000 GPUs, each with 24GB global memory, and 384 GB RAM. For these GPUs, the settable power cap values are in the range of 100 to 260W. The node is powered by four 1kW gold standard power supply units, for which we used custom adapter that would allow the Yokogawa meter to measure the power intake of the whole system.

4.2 Application

Due to the vast number of required tests to achieve assumed experiment coverage, an automated way of changing configurations and collecting data from tests has been developed. The application can be divided into four parts, depicted in Figure 1:

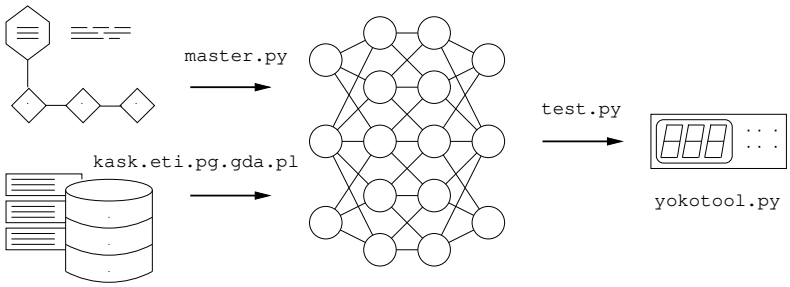

Fig. 1: Test application architecture

- master.py - application entry point containing the scheduler responsible for changing the test parameters (power cap values, the number of GPUs used),
- test.py - main part of the application containing code related to model compilation and the training process. Data is fetched from the user home directory for the needs of the training task. The code manages calling and terminating processes responsible for logging energy consumption data,
- yokotool.py - a Python script responsible for handling the yokotool daemon.

4.3 Efficient parallelization

In order to guarantee a high level of scalability and efficient parallelization of the training process, several steps have been taken in order to avoid performance drops related to loading data into multiple GPUs, and to ensure an optimized all-reduce step bound to distributed training.

A common problem in non-optimized training scripts is losing performance due to the lack or sub-optimal usage of caches present in GPU execution units. It was solved by utilizing prefetching with an autotuned buffer size provided by the tensorflow.data API. Such a solution makes for an efficient data pipeline, that provides training data batches, as well as optimizes data accesses of threads by moving future batches into L1/L3 caches, allowing for much faster data access.

Optimized distributed training has been achieved by using Horovod, a framework designed and built for highly efficient distributed training of artificial neural networks. Horovod has many advantages over Tensorflow, when it comes to implementation of multi-GPU training scirpts, with the most relevant to the goal of this paper being highly efficient communication that utilizes MPI [3] with multithreading and an efficient ring-all-reduce implementation that guarantees good scalability of this step.

4.4 Testing process and results

The assumption necessary for sufficient coverage of the power range is to traverse the search-space using a step of 10W, for each considered hardware configuration (1/2/4/8 GPUs). Regardless of the number of GPUs used in a particular configuration, the same power cap setting was applied to each GPU. The range of tested power cap settings per GPU is shown on the X axes of following charts. After conducting all of the required tests we obtained training results for all considered configurations.[4] For each configuration, i.e. the number of GPUs used for training and a given power cap, 10 runs were executed and results averaged. We measured average execution times, corresponding (low) time standard deviation values, average energy measured over training time (integrated power readings from the hardware power meter over time), corresponding (low) energy standard deviation values, as well as computed metrics for each configuration including: EDP, EDS for k=1.5 and k=2.0 (following [11]).

For 1, 2 and 4 GPUs, interesting results, in the context of optimization of the aforementioned metrics, can be seen in terms of energy gains, as shown in Figure 2. In those cases, where we are using only 1, 2 and 4 out of the total of 8 GPUs the unused GPUs are left idle. Idle GPUs still contribute to the total power consumption of the node. This does not allow to obtain interesting (notably improved) EDP, EDS nor percentage difference in energy gain and performance loss, compared to the default power cap. It is different, though, for

[4] Due to space constraints this data is available at `https://cdn.files.pg.edu.pl/eti/KASK/RAW2023-paper-supplementary-data/Supplementary_data_Performance_and_power_analysis_of_training_and_performance_quality.pdf`

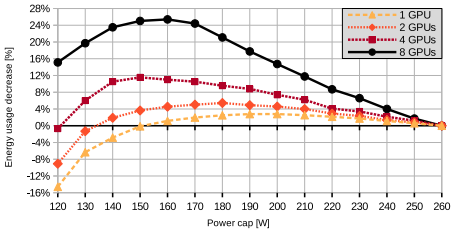

Fig. 2: Percentage energy decrease vs power cap – configurations with 1, 2, 4 and 8 GPUs, compared to default power cap

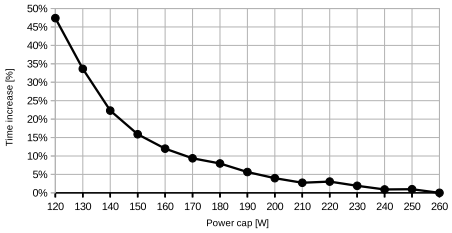

Fig. 3: Execution time increase vs power cap – 8 GPUs, compared to default power cap

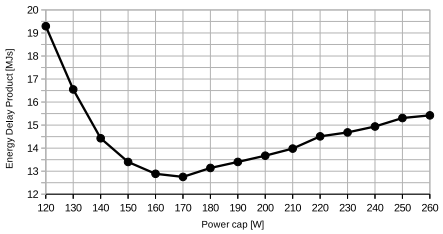

Fig. 4: EDP vs power cap – 8 GPUs

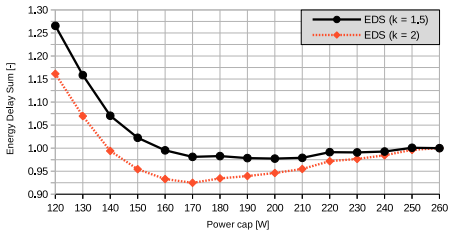

Fig. 5: EDS (k=1.5 and k=2.0) vs power cap – 8 GPUs

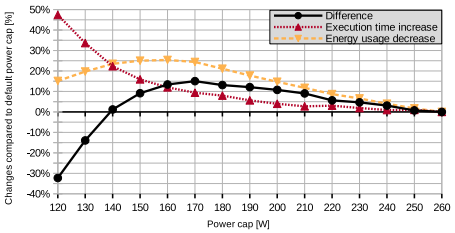

Fig. 6: Difference between percentage energy decrease and percentage time increase vs power cap – 8 GPUs, compared to default power cap

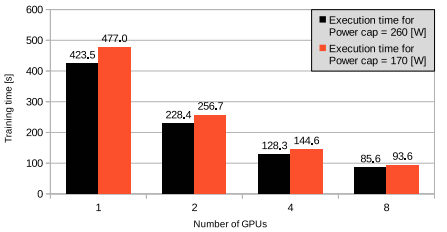

Fig. 7: Training times [s] versus the number of GPUs – for 260W and 170W power caps

the largest tested configuration, which is the one with 8 GPUs when we use the system to its full potential. In this case we definitely see other than the default power cap configurations, for which energy (Figure 2), EDP (Figure 4), EDS (Figure 5) and percentage difference between energy gain and performance loss (Figure 6) are significantly improved over those for the default power cap, at the cost of slightly increased running time (Figure 3). This is summarized in Table 1. While for EDS and k=1.5 there is a 2.2% improvement, for EDS k=2.0

we see the improvement of 7.5%, for EDP a notable 17.3%, for the difference of percentage energy gain and percentage performance loss over 15% and for pure energy the maximum gain is over 25%. Our conclusion is that, depending on the chosen goal, power capping can offer a significant improvement in DNN training using a high performance system with multiple (8 in this case) GPUs.

Table 1: Best power cap values and gains for particular metrics – 8 GPUs

Metric	**Gain for metric over default power cap configuration [%]**	**Obtained for power cap setting [W]**
Energy	25.39	160
Difference between energy gain [%] and performance loss [%]	15.03	170
EDP	17.32	170
EDS (k=1.5)	2.26	200
EDS (k=2.0)	7.52	170

Finally, we can observe scalability of the training process in the multi-GPU setup. In Figure 7 we present execution times for 1, 2, 4 and 8 GPUs for the two most important power caps: default 260W per GPU as well as 170W per GPU being best for optimization of the difference between percentage energy gain and performance loss compared to the default power cap configuration, EDP, EDS with k=2.0 and close to best for k=1.5. We can see that increasing the number of GPUs results in good scaling of the training. Specifically, for 4 GPUs we can observe a time speed-up of approx. 3.3 in both cases. For 8 GPUs and the power cap of 260W a speed-up of 4.95 can be seen while for 170W a speed-up of 5.1 can be observed.

5 Summary

In the paper we performed performance-energy analysis of training a deep neural network under various power caps in a multi-GPU enironment, using 1, 2, 4 and 8 NVIDIA Quadro RTX 6000 GPUs, within a high performance server with 2 Intel Xeon Silver 4210 CPUs. Power measurements were taken and considered for the whole system using a professional hardware power meter Yokogawa WT310E. We show that, depending on the number of GPUs used, enforcing an appropriate power cap allows to obtain percentage energy savings of approx. 3% when using 1 GPU out of 8 GPUs, 5.5% when using 2 GPUs, 11.5% for 4 GPUs and over 25% for 8 GPUs. These growing savings result from the fact that the total power of a smaller number of GPUs constitutes a smaller percentage of the whole machine power (the other GPUs remain idle). The main contribution of our paper is that for the most powerful configuration with 8 GPUs, we were able to determine that

enforcing selected power caps, different from the default one (260W per GPU in our case), allows to obtain significantly optimized values of various performance-energy metrics. In particular, we can improve the difference between percentage energy gain and performance loss by over 15.0% using the power cap of 170W, EDP by over 17.3% using the power cap of 170W, EDS with k=1.5 by over 2.2% using the power cap of 200W (similar gains within the range of 170-210W) and EDS with k=2.0 by over 7.5% using the power cap of 170W.

Future work will include, given a particular domain problem, optimization that involves, apart from execution time and energy, the neural network model and compute device models, taking into account the monetary cost of compute devices and the system. At a higher level of abstraction, such simulations will be used within a workload stream for which scheduling algorithms enhanced with power capping will be adopted in a supercomputing center, as identified in [8].

Acknowledgment

We would like to thank the administrator of the HPC server at Department of Computer Architecture at the GUT, dr Tomasz Boiński, for support regarding setting up the testbed environment. This work is supported by CERCIRAS COST Action CA19135 funded by the COST Association as well as statutory funds of Dept. of Computer Architecture, Faculty of Electronics, Telecommunications and Informatics, Gdańsk Tech.

References

1. Chen, G., Wang, X.: Performance optimization of machine learning inference under latency and server power constraints. In: 2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS). pp. 325–335 (2022). https://doi.org/10.1109/ICDCS54860.2022.00039
2. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1800–1807 (July 2017). https://doi.org/10.1109/CVPR.2017.195
3. Czarnul, P., Proficz, J., Drypczewski, K.: Survey of methodologies, approaches, and challenges in parallel programming using high-performance computing systems. Sci. Program. **2020**, 4176794:1–4176794:19 (2020). https://doi.org/10.1155/2020/4176794, https://doi.org/10.1155/2020/4176794
4. García-Martín, E., Rodrigues, C.F., Riley, G., Grahn, H.: Estimation of energy consumption in machine learning. Journal of Parallel and Distributed Computing **134**, 75–88 (2019). https://doi.org/https://doi.org/10.1016/j.jpdc.2019.07.007, https://www.sciencedirect.com/science/article/pii/S0743731518308773
5. He, X., Liu, J., Xie, Z., Chen, H., Chen, G., Zhang, W., Li, D.: Enabling energy-efficient dnn training on hybrid gpu-fpga accelerators. In: Proceedings of the ACM International Conference on Supercomputing. p. 227–241. ICS '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3447818.3460371, https://doi.org/10.1145/3447818.3460371

6. Jabłońska, K., Czarnul, P.: Benchmarking deep neural network training using multi- and many-core processors. In: Saeed, K., Dvorský, J. (eds.) Computer Information Systems and Industrial Management. pp. 230–242. Springer International Publishing, Cham (2020)
7. Kang, D.K., Lee, K.B., Kim, Y.C.: Cost efficient gpu cluster management for training and inference of deep learning. Energies **15**(2) (2022). https://doi.org/10.3390/en15020474, https://www.mdpi.com/1996-1073/15/2/474
8. Kocot, B., Czarnul, P., Proficz, J.: Energy-aware scheduling for high-performance computing systems: A survey. Energies **16**(2) (2023). https://doi.org/10.3390/en16020890, https://www.mdpi.com/1996-1073/16/2/890
9. Krzywaniak, A., Czarnul, P.: Performance/energy aware optimization of parallel applications on gpus under power capping. In: Wyrzykowski, R., Deelman, E., Dongarra, J., Karczewski, K. (eds.) Parallel Processing and Applied Mathematics. pp. 123–133. Springer International Publishing, Cham (2020)
10. Krzywaniak, A., Czarnul, P., Proficz, J.: Gpu power capping for energy-performance trade-offs in training of deep convolutional neural networks for image recognition. In: Groen, D., de Mulatier, C., Paszynski, M., Krzhizhanovskaya, V.V., Dongarra, J.J., Sloot, P.M.A. (eds.) Computational Science – ICCS 2022. pp. 667–681. Springer International Publishing, Cham (2022)
11. Krzywaniak, A., Czarnul, P., Proficz, J.: Depo: A dynamic energy-performance optimizer tool for automatic power capping for energy efficient high-performance computing. Software: Practice and Experience **52**(12), 2598–2634 (2022). https://doi.org/https://doi.org/10.1002/spe.3139, https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.3139
12. Lai, C., Ahmad, S., Dubinsky, D., Maver, C.: Ai is harming our planet: addressing ai's staggering energy cost (May 2022), https://www.numenta.com/blog/2022/05/24/ai-is-harming-our-planet/
13. Leng, J., Hetherington, T., ElTantawy, A., Gilani, S., Kim, N.S., Aamodt, T.M., Reddi, V.J.: Gpuwattch: Enabling energy optimizations in gpgpus. SIGARCH Comput. Archit. News **41**(3), 487–498 (jun 2013). https://doi.org/10.1145/2508148.2485964, https://doi.org/10.1145/2508148.2485964
14. Mazuecos Pérez, M.D., Seiler, N.G., Bederián, C.S., Wolovick, N., Vega, A.J.: Power efficiency analysis of a deep learning workload on an ibm "minsky" platform. In: Meneses, E., Castro, H., Barrios Hernández, C.J., Ramos-Pollan, R. (eds.) High Performance Computing. pp. 255–262. Springer International Publishing, Cham (2019)
15. McDonald, J., Li, B., Frey, N., Tiwari, D., Gadepally, V., Samsi, S.: Great power, great responsibility: Recommendations for reducing energy for training language models. In: Findings of the Association for Computational Linguistics: NAACL 2022. Association for Computational Linguistics (2022). https://doi.org/10.18653/v1/2022.findings-naacl.151, https://doi.org/10.18653\%2Fv1\%2F2022.findings-naacl.151
16. Rouhani, B.D., Mirhoseini, A., Koushanfar, F.: Delight: Adding energy dimension to deep neural networks. In: Proceedings of the 2016 International Symposium on Low Power Electronics and Design. p. 112–117. ISLPED '16, Association for Computing Machinery, New York, NY, USA (2016).

https://doi.org/10.1145/2934583.2934599, https://doi.org/10.1145/2934583.2934599

17. Schuchart, J., Gerndt, M., Kjeldsberg, P.G., Lysaght, M., Horák, D., źÍha, L., Gocht, A., Sourouri, M., Kumaraswamy, M., Chowdhury, A., Jahre, M., Diethelm, K., Bouizi, O., Mian, U.S., KruźÍk, J., Sojka, R., Beseda, M., Kannan, V., Bendifallah, Z., Hackenberg, D., Nagel, W.E.: The readex formalism for automatic tuning for energy efficiency. Computing **99**(8), 727–745 (aug 2017)
18. Tao, Y., Ma, R., Shyu, M.L., Chen, S.C.: Challenges in energy-efficient deep neural network training with fpga. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 1602–1611 (2020). https://doi.org/10.1109/CVPRW50498.2020.00208
19. Wang, F., Zhang, W., Lai, S., Hao, M., Wang, Z.: Dynamic gpu energy optimization for machine learning training workloads. IEEE Transactions on Parallel and Distributed Systems **33**(11), 2943–2954 (2022). https://doi.org/10.1109/TPDS.2021.3137867
20. Xu, Y., Martínez-Fernández, S., Martinez, M., Franch, X.: Energy efficiency of training neural network architectures: An empirical study (2023)
21. Yang, H., Zhu, Y., Liu, J.: Ecc: Platform-independent energy-constrained deep neural network compression via a bilinear regression model. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11198–11207. IEEE Computer Society, Los Alamitos, CA, USA (jun 2019). https://doi.org/10.1109/CVPR.2019.01146, https://doi.ieeecomputersociety.org/10.1109/CVPR.2019.01146
22. Yang, T.J., Chen, Y.H., Emer, J., Sze, V.: A method to estimate the energy consumption of deep neural networks. In: 2017 51st Asilomar Conference on Signals, Systems, and Computers. pp. 1916–1920 (2017). https://doi.org/10.1109/ACSSC.2017.8335698
23. Yang, Z., Meng, L., Chung, J.W., Chowdhury, M.: Chasing low-carbon electricity for practical and sustainable dnn training (2023)
24. You, J., Chung, J.W., Chowdhury, M.: Zeus: Understanding and optimizing GPU energy consumption of DNN training. In: 20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23). pp. 119–139. USENIX Association, Boston, MA (Apr 2023), https://www.usenix.org/conference/nsdi23/presentation/you
25. Zou, P., Li, A., Barker, K., Ge, R.: Indicator-directed dynamic power management for iterative workloads on gpu-accelerated systems. In: 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID). pp. 559–568 (2020). https://doi.org/10.1109/CCGrid49817.2020.00-37