Power Efficiency Analysis of a Deep Learning Workload on an IBM "Minsky" Platform

Mauricio D. Mazuecos Pérez[1], Nahuel G. Seiler[1], Carlos Sergio Bederián[1,2], Nicolás Wolovick[1(✉)], and Augusto J. Vega[3]

[1] FaMAF, Universidad Nacional de Córdoba, Córdoba, Argentina
nicolasw@famaf.unc.edu.ar
[2] CONICET, Buenos Aires, Argentina
[3] IBM T. J. Watson Research Center, Yorktown Heights, USA

Abstract. The rise of Deep Learning techniques has attracted special attention to GPUs usage for better performance of model computation. Most frameworks for Cognitive Computing include support to offload model training and inferencing to graphics hardware, and this is so common that GPU designers are reserving die area for special function units tailored to accelerating Deep Learning computation. Measuring the capability of a hardware platform to run these workloads is a major concern for vendors and consumers of this exponentially growing market. In a previous work [9] we analyzed the execution times of the Fathom AI workloads [2] in CPUs and CPUs+GPUs. In this work we measure the Fathom workloads in the POWER8-based "Minsky" [15] platform, profiling power consumption and energy efficiency in GPUs. We explore alternative forms of execution via GPU power and frequency capping with the aim of reducing Energy-to-Solution (ETS) and Energy-Delay-Product (EDP). We show important ETS savings of up to 27% with half of the workloads decreasing the EDP. We also expose the advantages of frequency capping with respect to power capping in NVIDIA GPUs.

Keywords: Fathom · GPU · Power capping · Frequency capping · Energy-to-Solution · Energy-Delay-Product

1 Introduction

Machine learning and in particular neural networks was a well established subject back in 1992, but from 2012 onwards its attractiveness has grown exponentially both in academia and industry in the form of deep neural networks. The main reasons are advances in algorithms, datasets, benchmarks, and hardware [6]. ImageNet [8] and MNIST Database [18] are examples of benchmarks that drove the field of Computer Vision using Deep Learning (DL) techniques. The computational demand is so large that vendors are currently offering better than Moore's law improvements for specific Machine Learning workloads

E. Meneses et al. (Eds.): CARLA 2018, CCIS 979, pp. 255–262, 2019.
https://doi.org/10.1007/978-3-030-16205-4_19

through the use of special function units like NVIDIA's Tensor Cores, Google's Tensor Processing Units (TPU) and Intel Nervana Neural Network Processors (NNP). The trend in showing the prowess of hardware platform in very specific tests [5] makes choosing the best platform for general DL difficult. Fathom [2,3] is an attempt to cope with this need, providing eight different workloads in different areas of DL using the TensorFlow [1] framework: AlexNet [17], Variational Autoencoder [13,16], Deep Reinforcement Learning [20], End-to-End Memory Networks [24,27], Residual Networks [12], Sequence-to-Sequence Translation [25], Deep Speech [10] and VGG-19 [23].

The IBM Power System S822LC [15] or "Minsky" is a power-horse for DL workloads. Equipped with two IBM POWER8 processors and four NVIDIA Tesla P100 GPUs, the whole set exhibits an aggregated performance exceeding 40 TFLOPS in single precision and 80 TFLOPS in half-precision peak performance [4]. Besides those impressive computation numbers, communication bandwidth is also remarkable as shown in Fig. 1. The technology behind this system corresponds to one generation before Summit, the fastest supercomputer on earth at the time of writing [26].

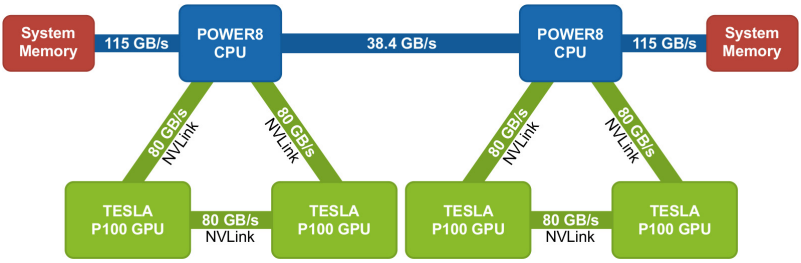

Fig. 1. Minsky platform data flow diagram

The energy consumed by Cognitive Computing workloads is not negligible. For example, servers like the IBM S822LC can draw up to 2.5 kW at full load [4], and a complex Deep Neural Network takes days to train [5]. This issue also affects computers on the other end of the spectrum, namely wearable devices and embedded computers powered by batteries, which have to operate under stringent power budgets.

The basic law guiding the power drawn by processors is $P \propto V^2 f$, where V is the voltage and f is the processor frequency. There are two common measures for the energy consumed. One is Energy-to-Solution (ETS), the area under the curve of power consumption $ETS = \int_0^T P(t)dt$, where T is the execution time and $P(t)$ is the instantaneous power drawn. One possible way to dynamically reduce P is to adjust f conveniently with the so-called frequency scalers. Current processors embed frequency scaler algorithms in silicon. Notice that in a naïve processor model, ETS should not decrease if f decreases, as halving frequency implies doubling computation time. However ETS gains are notable using frequency

scaling and this is due to the processors spending most of their time waiting for the memory and communication subsystems. The other measure to try minimize is Energy-delay-Product $EDP = ETS \times T$, that puts together two quantities that appear to be inverse of each other, since decreasing ETS implies lower frequency and therefore increasing computation time. The relation is not linear and the $EDP(f)$ has a global minimum in the interval of available frequencies. Voltage scaling is also a current research topic, both for hardware designers and software designers since the potential savings can be large [7].

In previous work [9], we showed how GPUs improve performance or Time-to-Solution (TTS) on the IBM "Minsky" Platform for the Fathom workloads. The contribution of this paper is the analysis of energy efficiency in the Fathom workloads executing on the same platform using power profiles, ETS and EDP. The ETS was improved 27% over standard execution via power and frequency capping in the GPU. We have also obtained improvements in EDP in half of the workloads.

The rest of the paper is organized as follows. Section 2 shows the direct and derived power measurements using power capping. Section 3 presents the energy consumption improvements using frequency capping. Finally, Sect. 4 summarizes findings and discusses future research.

2 Power Analysis

In this work we decided to upgrade TensorFlow 1.0 used in our previous work [9] to TensorFlow 1.1. Some modifications in Fathom were needed in order to run Seq2Seq in TensorFlow 1.1, and this patch has been accepted upstream [21]. The rest of the system software in the IBM Poughkeepsie WW Client Center [14] is Ubuntu Server 16.04 ppc64le, NVIDIA Driver 384.66 and GCC 5.4. In Fig. 2 we assess the execution time with respect to our former work. The execution time are the average of twenty samples. In five workloads (AlexNet, DeepQ, Residual, Seq2Seq, VGG) the results are within a 25% difference. The rest of the workloads (AutoEnc, MemNet) which are the shortest, taking around 16 s, suffer from increased setup time in the newer TensorFlow.

We run all the Fathom workloads using all 20 physical cores available in POWER8 chips and one P100 GPU. The measurements are the mean of 2 samples and the workloads have been tuned from the original Fathom to increase the number of steps from 10 to 100 in order to increase their duration. The power measurements were sampled with a 1-sec granularity using `nvidia-smi dmon`.

Using the power capping features of the NVIDIA Driver (`nvidia-smi -pl <power_limit>`) we tested unlimited (300 W), 250 W, 200 W and the minimum available (150 W) power caps. The power traces for each workload[1] are shown in Fig. 3. There are three groups with respect to workload duration: short (AutoEnc, MemNet), medium (AlexNet, Residual, VGG) and long (Seq2Seq). The P100 GPU idle power is around 27 W, making the base of all these curves.

[1] DeepQ measurements are missing due to system errors.

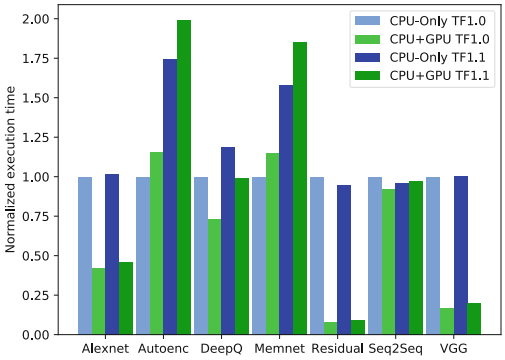

Fig. 2. Fathom using TensorFlow 1.0 vs. TensorFlow 1.1

Power consumptions are partitioned in two: low consumption (AutoEnc, MemNet, Seq2Seq) and high consumption (AlexNet, Residual, VGG). AlexNet and Residual includes high frequency oscillations in power consumption. In general the four power capping levels do not seem to produce significant differences in power nor in computation time. The best power profile in terms of ETS seems to be VGG: lower curves for 150 W power capping, while not increasing the computation time. Figure 4 presents the normalized ETS and EDP metrics for different power capping values. The marginal gains show that the power capping feature present in the NVIDIA driver does not adapt well to the workloads being tested, while other workloads like password cracking and cryptocurrency mining improve their energy efficiency [11,22]. Since the area and execution time have slight variation, ETS and EDP figures are similar but not equal.

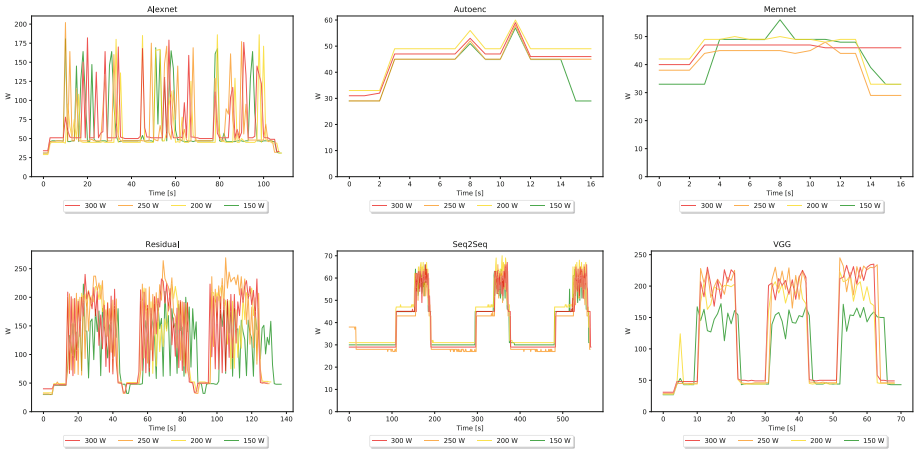

Fig. 3. Power traces using GPU power capping

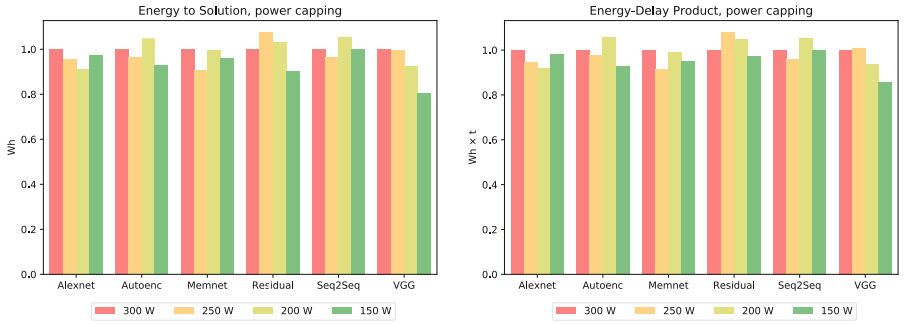

Fig. 4. ETS and EDP using GPU power capping

3 Improvements

The expression governing power dissipation in processors $P \propto V^2 f$ shows an alternative way of power capping: controlling the processor frequency directly. We try frequency capping using `nvidia-smi -ac 5004,<freq>`, avoiding the power capping mechanism used by the NVIDIA driver that controls the frequency to achieve the desired power. The power profile curves in Fig. 5 include two GPU frequencies: maximum 1488 MHz and minimum 544 MHz. The figures show a more interesting behavior. Capping frequency at 544 MHz, AlexNet shows lower power consumption with little change in execution time. AutoEnc, Seq2Seq and MemNet are also similar, and we expect lower ETS and EDP. For Residual and VGG we have lower power curves, but execution time has increased.

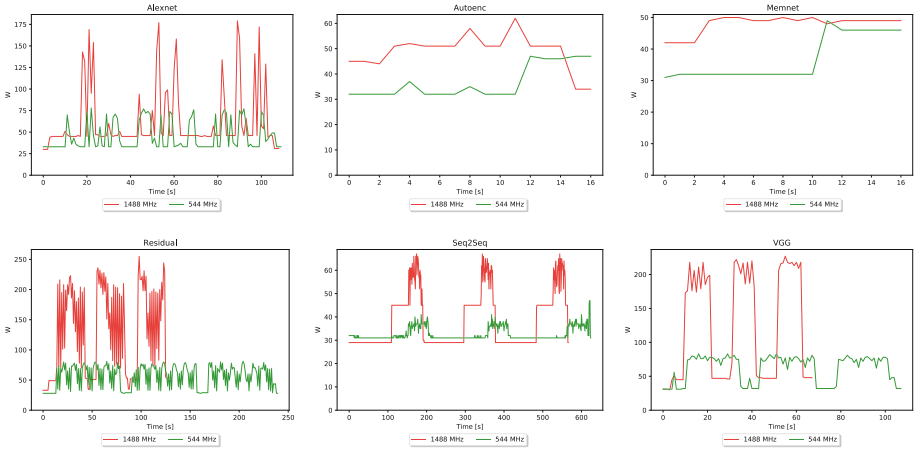

Fig. 5. Power profile of workloads using GPU frequency capping

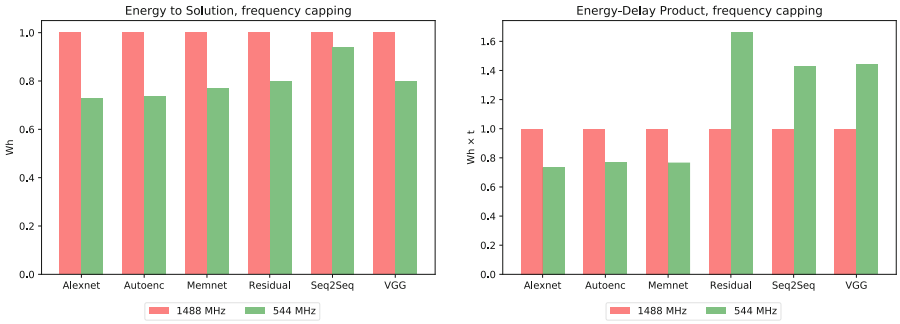

Fig. 6. ETS and EDP using GPU frequency capping

ETS and EDP for frequency capping in Fig. 6 show clear gains. All workloads improve energy efficiency, with AlexNet achieving 27% of ETS savings. For EDP the first three workloads analyzed improve the metric, while Residual, Seq2Seq and VGG increase EDP. For Seq2Seq there are energy gains with increased execution time, but the idle power drawn is comparatively high with respect to the load imposed by this test. Residual and VGG show worse EDP measurements due to much longer execution times that offset the power savings.

4 Conclusion

We obtained a working version of Fathom using a more recent version of TensorFlow. The comparison with the previous version exhibits performance degradation for the shortest workloads (AutoEnc, MemNet) probably due to higher setting up and tearing down costs in TensorFlow 1.1 with respect to TensorFlow 1.0. For the rest of the workloads there are clear gains in upgrading the library.

GPU power profiles for an uncapped NVIDIA P100 card show variable power consumption and performance profiles. AlexNet, Residual and VGG put stress on the GPU while the rest mildly activate the transistors inside the P100 chip.

We first tried to improve energy efficiency via the GPU driver power capping feature, but it was not successful enough. Power profiles did not exhibit good gains, and execution times did not change significantly. We also tried the GPU driver frequency capping feature and this significantly improved energy efficiency for all workloads. The reason for power capping not working seems to be the interplay between the high frequency power profile and the control loop of the driver. The software piece controlling the frequency to achieve the power capping is not prepared for these loads. Going back to the roots and doing frequency capping is the correct way of improving ETS and EDP for these deep learning workloads. For AlexNet we achieved 27% in energy savings with a slight increase in execution time, that is an overall 28% gain in Energy-Delay-Product.

It is worth remarking that among the cryptocurrency mining and password cracking community it is common knowledge to use power and frequency capping

to improve energy efficiency in GPUs and memory. In the case of HashCat there are reports of 54% [11], and 27% [22] energy efficiency improvements in Ethereum Mining (measured in MHashes/s/W).

Future work includes frequency capping not only in the GPU, but also in the memory subsystem of the GPU. We also plan to conduct similar benchmarking and analysis in the state-of-the-art machine learning benchmark MLBench [19].

Acknowledgments. This work was partially funded by SeCyT-UNC 2016 grant 30720150101248CB "Heterogeneous HPC" and 2016 IBM Faculty Award "Resilient Scale-Out for Deep Learning on Power Systems".

References

1. Abadi, M., et al.: TensorFlow: large-scale machine learning on heterogeneous systems (2015). software available: https://www.tensorflow.org/
2. Adolf, R., Rama, S., Reagen, B., Wei, G.Y., Brooks, D.M.: Fathom: reference workloads for modern deep learning methods. CoRR abs/1608.06581 (2016). http://arxiv.org/abs/1608.06581
3. Adolf, B.: Fathom, reference workloads for modern deep learning. https://fathom.readthedocs.io
4. Caldeira, A.B., Haug, V., Vetter, S.: IBM Power System S822LC for High Performance Computing Introduction and Technical Overview, 1st edn. IBM Redbooks, October 2016
5. Cho, M., Finkler, U., Kumar, S., Kung, D.S., Saxena, V., Sreedhar, D.: PowerAI DDL. CoRR abs/1708.02188 (2017)
6. Chollet, F.: Deep Learning with Python. Manning, Shelter Island (2018)
7. Deng, B., et al.: Extending Moore's law via computationally error-tolerant computing. ACM Trans. Archit. Code Optim. **15**(1), 8:1–8:27 (2018). https://doi.org/10.1145/3177837
8. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-fei, L.: Imagenet: a large-scale hierarchical image database. In: CVPR (2009)
9. Guignard, M., Schild, M., Bederián, C.S., Wolovick, N., Vega, A.J.: Performance characterization of state-of-the-art deep learning workloads on a "Minsky" platform. In: HICSS 2018 (2018)
10. Hannun, A.Y., et al.: Deep speech: scaling up end-to-end speech recognition. CoRR abs/1412.5567 (2014). http://arxiv.org/abs/1412.5567
11. @hashcat: GPU power efficiency (Hash/Watt) explained simple (2017). https://twitter.com/hashcat/status/893047795921416193
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR abs/1512.03385 (2015). http://arxiv.org/abs/1512.03385
13. Hinton, G., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
14. IBM: IBM Systems Client Centers (2018). https://www.ibm.com/it-infrastructure/services/client-centers
15. IBM Corporation: IBM POWER8 specification. https://www.ibm.com/us-en/marketplace/high-performance-computing
16. Kingma, D.P., Welling, M.: Stochastic gradient VB and the variational auto-encoder. In: Proceedings of the 2nd International Conference on Learning Representations, ICLR 2014 (2014)

17. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks, vol. 25, January 2012
18. LeCun, Y., Cortes, C., Burges, C.J.: The MNIST Dataset of Handwritten Digits (1999). http://yann.lecun.com/exdb/mnist/
19. MLPerf: A broad ML benchmark suite for measuring performance of ML software frameworks, ML hardware accelerators, and ml cloud platforms. https://mlperf.org/
20. Mnih, V., et al.: Playing Atari with deep reinforcement learning. CoRR abs/1312.5602 (2013). http://arxiv.org/abs/1312.5602
21. Seiler, N.G.: Changes to make seq2seq compatible with tensorflow versions later than 1.x (2017). https://github.com/rdadolf/fathom/pull/35
22. Mott, S.: Ethereum Mining with NVIDIA on Linux (2017). https://www.simonmott.co.uk/2017/07/ethereum-mining-nvidia-linux/
23. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2014). http://arxiv.org/abs/1409.1556
24. Sukhbaatar, S., Szlam, A., Weston, J., Fergus, R.: Weakly supervised memory networks. CoRR abs/1503.08895 (2015). http://arxiv.org/abs/1503.08895
25. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems, vol. 27, pp. 3104–3112. Curran Associates, Inc. (2014). http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf
26. TOP500: June 2018 List (2018). https://www.top500.org/lists/2018/06/
27. Weston, J., Chopra, S., Bordes, A.: Memory networks. CoRR abs/1410.3916 (2014). http://arxiv.org/abs/1410.3916