A cost-benefit analysis of GPU-based EC2 instances for a deep learning algorithm

Eva M. Malta[1], **Charles B. Rodamilans**[1,2], **Sandra Avila**[1], **Edson Borin**[1]

[1]Center for Petroleum Studies (CEPETRO), University of Campinas (UNICAMP)
Campinas – SP – Brazil

[2]Computing and Informatics Department, Mackenzie Presbyterian University
São Paulo – SP – Brazil

emaiamalta@gmail.com, charles.rodamilans@mackenzie.br
{sandra,edson}@ic.unicamp.br

Abstract. *This paper analyzes the cost-benefit of using EC2 instances, specifically the p2 and p3 virtual machine types, which have GPU accelerators, to execute a machine learning algorithm. This analysis includes the runtime of a convolutional neural network executions, and it takes into consideration the necessary time to stabilize the accuracy value with different batch sizes. Also, we measure the cost of using each machine type, and we define a relation between this cost and the execution time for each virtual machine. The results show that, although the price per hour of the p3 instance is three times bigger, it is faster and costs almost the same as the p2 instance type to train the deep learning algorithm.*

1. Introduction

Machine Learning algorithms are tools used to solve diverse problems in academia and industry. An example of a Machine Learning algorithm class is Deep Neural Network, which is a common approach to deal with classification and regression problems. Image recognition, for instance, usually uses a specific neural network type called Convolutional Neural Networks (CNNs) [LeCun et al. 1990].

A common problem when we deal with CNNs is the execution time required to train the model. The training process may take days to achieve the expected accuracy, and it is often necessary to train them many times. Graphic Process Units (GPU) are a good option to make the training process faster [Ben-Nun and Hoefler 2018].

However, some people do not have this kind of resource, and even companies may have bureaucratic or financial difficulties to acquire the necessary amount of these machines. Because of that, cloud computing is an attractive solution for those who need one or more machines with an affordable cost at any time, since the user pays only for what they use.

Amazon Web Services (AWS) [Miller et al. 2010], for instance, offers a service called Elastic Compute Cloud (EC2) that provides virtual machines, which types differentiate themselves by hardware characteristics, such as CPU's quantity, storage, memory, and network performance. The instances with accelerated computing are: *p2* and *p3*, both for general problems; *g3* for graphic-intensive applications; and *f1*, which offer customized hardware acceleration. In this context, a common issue is deciding which

machine type is more adequate to train a deep learning model, given the hardware and price differences.

To address this problem, we analyze the cost-benefit of using *p2* and *p3* instances to train the ResNet [He et al. 2016] on the CIFAR-10 dataset [Krizhevsky et al. 2010]. Usually, to make this kind of analysis, the runtime is the only variable to be observed. In machine learning, however, the accuracy must also be taken into consideration, since this variable is crucial to measure the quality of a model. With this in mind, our analysis measures the time it takes for the accuracy model to become stabilized and also observes if the batch size influences the results.

In a similar way, [Kaplunovich and Yesha 2017] proposed an AWS virtual machine recommending platform based on a machine learning algorithm and dataset size. Besides, [Lee and Son 2017] introduced a system that uses AWS spot instances and checkpoint technology to reduce the cost of training machine learning algorithms.

2. Materials and methods

In our experiments, we used a benchmark provided by the TensorFlow community[1]. It implements a ResNet-50 network and was designed to deal specifically with the CIFAR-10 dataset. This dataset is popular among the machine learning community, and is composed of 60,000 32×32 images. 50,000 of these images are used in the training process, and the remaining in the test step. The images are equally distributed between 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck.

The EC2 instances types used in experiments are the p2.xlarge and p3.2xlarge. Both are optimized for accelerating computing and contain one GPU each. Table 1 shows the characteristics of each of these instance types.

Table 1. EC2 instances main features obtained in the AWS webpage.

Instance type	NVIDIA GPU	GPUs	vCPUs	Mem	GPU Mem	Price p/h
p2.xlarge	K80	1	4	61 (GiB)	12 (GiB)	0.90 (U$)
p3.2xlarge	Tesla V100	1	8	61 (GiB)	16 (GiB)	3.06 (U$)

Regarding the applied methodology during experiments, we executed the application in each instance, and we varied the batch size in 64, 128, 256, and 512 images per iteration. For each size, we ran it three times to measure the variance in accuracy, runtime, and the number of iterations, which indicates how many times the batch is sent to training.

First, we have started the experiments with p2.xlarge instance. During this step, we ran the program with 50,000 iterations for all batch sizes to discover in which iteration the model's accuracy became stabilized, which was determined by visual inspection. With a batch size of 64 images, we have repeated the process increasing the number of iterations to 100,000, given the accuracy instability.

Once we discovered this number for each batch size, we have defined this variable with the previous results, and then we have repeated the process with the p3.2xlarge instance. Finally, we have registered the mean runtime to achieve that number of iterations for each batch size, and we have calculated the cost per hour for using each instance type.

[1]https://github.com/tensorflow/models/tree/master/tutorials/image/cifar10_estimator

3. Results and discussions

The first set of experiments, when we have tested only the p2.xlarge, shows that as we increase the batch size, fewer steps are needed, as expected. Figure 1 shows the learning curve of the three runs with a batch size of 128. Notice that the model accuracy becomes stabilized after 30,000 iterations. We call the iteration in which the accuracy stabilizes the "stabilization point".

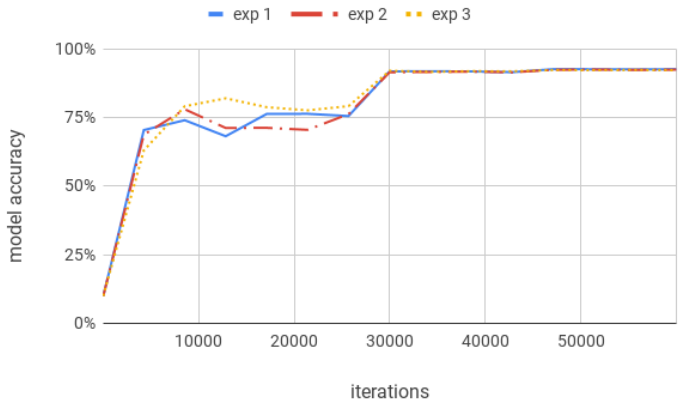

Figure 1. Accuracy evolution to a batch size with 128 images on the p2.xlarge.

Table 2 shows the stabilization points for all four batch values. It also shows the average time for each virtual machine to achieve its stabilization point, the average accuracy achieved until that point, and the average cost in dollars. It is worthy of note that in 75% of occasions the p3.2xlarge instance, despite costing per hour almost three times more, is less expensive than p2.xlarge. Figure 2 illustrates the cost *vs.* execution time.

Table 2. Stabilization point, average runtime, average cost of p2.xlarge, and p3.2xlarge instances.

Batch size	Instance type	Stabilization point	Average time (h)	Speedup	Average accuracy (%)	Average cost (U$)
64	p2.xlarge	60,000	$1.33 \pm 1.26\times10^{-3}$	2.70	91.20 ± 0.39	1.20
	p3.2xlarge		$0.49 \pm 1.13\times10^{-2}$			1.51
128	p2.xlarge	30,000	$1.17 \pm 1.16\times10^{-3}$	3.51	91.67 ± 0.30	1.05
	p3.2xlarge		$0.33 \pm 4.36\times10^{-4}$			1.02
256	p2.xlarge	17,000	$1.17 \pm 2.20\times10^{-3}$	3.68	91.03 ± 0.49	1.05
	p3.2xlarge		$0.32 \pm 6.07\times10^{-4}$			0.97
512	p2.xlarge	8,000	$1.00 \pm 5.92\times10^{-3}$	3.63	89.77 ± 0.21	0.90
	p3.2xlarge		$0.28 \pm 1.68\times10^{-3}$			0.85

We also observe that the model with a batch size of 512 images ran faster, and with 64 images we spent more time to achieve the stabilization point. This happens because, as the gradients are updated after an iteration, the learning after a single update with a small batch size tends to be smaller than with a bigger one, this implies in more iterations to become stabilized. Regarding the average accuracy, it was a few worse with a batch size of 512. Some researches already pinpointed that as bigger the batch size is, smaller the final accuracy tends to be [Ben-Nun and Hoefler 2018] [Keskar et al. 2016].

4. Conclusion

Our results showed that, as we expected, the p3.2xlarge instance executed the algorithm in less time than the p2.xlarge on all occasions. This result was not surprising, given the

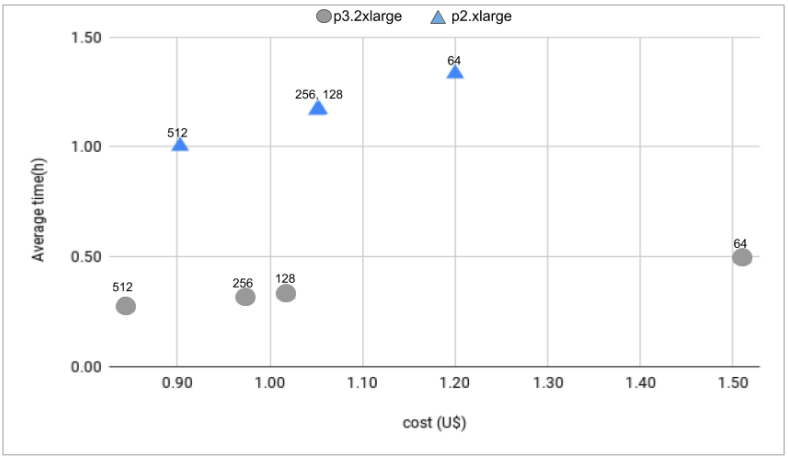

Figure 2. The correlation between cost (in US$) and runtime (per hour).

GPU type difference. The exciting discovery is that this performance improvement was significant enough to make the p3.2xlarge the less costly option to this kind of problem. Once that, if the cost of using the two machines was not so different, the execution was three times faster.

For future works, we intend to analyze a larger variety of instances and to verify other models, and other frameworks as well, to establish a cost-benefit relation that investigates more variables. Finally, we would like to thank Petrobras, CNPq (313012/2017-2), and FAPESP (2013/08293-7, 2017/16246-0) for their financial support, and High-Performance Geophysics (HPG) team for its technical support.

References

Ben-Nun, T. and Hoefler, T. (2018). Demystifying parallel and distributed deep learning: An in-depth concurrency analysis. *arXiv preprint arXiv:1802.09941*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*, pages 770–778. IEEE.

Kaplunovich, A. and Yesha, Y. (2017). Cloud big data decision support system for machine learning on AWS: Analytics of analytics. In *Big Data*, pages 3508–3516. IEEE.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. (2016). On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*.

Krizhevsky, A., Nair, V., and Hinton, G. (2010). CIFAR-10 (Canadian Institute for Advanced Research). `http://www.cs.toronto.edu/~kriz/cifar.html`.

LeCun, Y., Boser, B., Denker, J., Howard, R., Habbard, W., Jackel, L., and Henderson, D. (1990). Handwritten digit recognition with a back-propagation network. In *NIPS*, pages 396–404.

Lee, K. and Son, M. (2017). DeepSpotCloud: leveraging cross-region gpu spot instances for deep learning. In *CLOUD*, pages 98–105. IEEE.

Miller, F. P., Vandome, A. F., and McBrewster, J. (2010). *Amazon Web Services*. Alpha Press.