International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 21 (2017) pp. 10655-10660

Hardware Processing Comparison in Convolutional Neural Networks Training

Robinson Jiménez Moreno[1], Ruben D. Hernández[2] and Javier Martinez Baquero[3]

[1,2,]*Professor, Faculty Engineering, Nueva Granada Military University, Bogotá, Colombia.*
[3]*Professor, Faculty of Engineering, University of Llanos, Meta, Colombia.*

[1]*Orcid: 0000-0002-4812-3734,* [2]*Orcid: 0000-0001-8152-2633,* [3]*Orcid: 0000-0003-4377-7867*

Abstract

This paper presents an analysis of the performance of three different computers when training 2 convolutional neural network architectures, based on the description of the characteristics of the processor and graphics card that each computer has. The architectures used have as input images of size 64x64 and 128x128, and their training is done with a dataset of 1400 images in total. After training in each computer, different parameters obtained from this training and the validation of the architectures will be compared, where the main parameter to be considered is the time each equipment uses to finish the training of 300 epochs, and in general the behavior that they had during the training, obtaining similarities in the accuracy reached, of about 85%, but with time differences taken to finish the training of between 1 to 9 hours, depending on the CNN architecture trained. With these comparisons, the user is given a selection criterion of portable computing equipment for effecting neural convolutional networks.

Keywords: Convolutional Neural Network, GPU, Computer Processing, Training Time, MATLAB.

INTRODUCTION

Artificial neural networks have been used on a large scale during the last decade [1][2], with the emergence of different types of neural networks, such as recurrent neural networks, deep belief neural networks, convolutional neural networks, and even combinations of kinds neural networks. Because they have been successful in their different applications, more robust developments have begun to be implemented, especially within the convolutional neural networks (CNN) [3].

CNN [4] is mainly oriented to the recognition of patterns in images, whose applications range from basic developments in handwriting recognition [5], hand gestures [6] or even object classification with simple patterns, such as pedestrians [7], to more complex and robust applications such as recognition of pathologies in medicine such as cancer [8], malaria [9] and atherosclerosis [10]. Its field of action even includes applications that leave the frame of the images, entering the voice recognition [11].

Because CNNs involve the use and processing of images, and that their training requires robust databases, usually of the order of hundreds of images per trained category, the computational cost for such training has been growing in recent years, taking long time to achieve high performance, which impacts on the convergence of results in the development of research or applications of very high performance in the short term. From this problem was derived in the use of the graphic cards to support the dense calculations and to reduce the times of the training. However, according to the application, long training times, ranging from hours to days, are still required, as described in [12], where to classify 1000 categories with 1.2 million images, it was necessary to use two GTX 580 graphic cards (GPU) of 3GB of memory, where CNN training took up to 6 days to get error rates of 36.7%.

To reduce the computational cost and to make faster the processing of any type of neural network, different techniques have been developed to optimize the calculation time that the GPU does [13][14][15], however, if these are applied to low-capacity GPUs or processors, the improvement in the times used in the calculations will not be relevant. On the other hand, the training of neural networks with robust databases are made in high performance computers that contain more than 3 GPUs, however, these computers have high costs and are not portable, so users, such as students or researchers, use laptops to perform network trainings. Such equipment has low processing capabilities compared to high performance computers. Considering this, it is important to know the characteristics of portable computers and their performance when training neural networks, for this reason, in this work the comparison of portable computers with different hardware characteristics is done with the objective of granting users a criterion of selection of the computer according to the requirements and needs for the training of neural networks, in this case, convolutional neural networks.

This paper presents the comparison of three different notebook computers with respect to time and behavior in the training of a convolutional neural network. The paper is divided into 3 sections, where section 2 describes the characteristics of both the computers and the CNN architectures used for the comparison, section 3 presents the results of the trainings under each computer architecture and finally, section 4 gives the conclusions obtained.

MATERIALS AND METHODS

This work focuses on analyzing the performance of the processing in 3 different notebook architectures at the time of the training of the same architecture of a convolutional neural network, under the same conditions, doing variations that show the capacities and benefits of a specific computer equipment. For comparison, different parameters obtained during the training are observed, which are: Capability to train the predetermined architectures, Training time and Accuracy obtained by the network. The programming environment is implemented in MATLAB® software and running on Intel® processors and NVIDIA® graphics cards.

A. Computer Characteristics

To perform the comparison of the processing in the training of the CNN, between the three different computers, it is necessary to know the characteristics of each one, both of its processor and of the GPU, because in their execution both are used in parallel during the training, both for mathematical calculations and in the processing of the images. Given this, 0and 0 shows the details of the processors and GPUs of each computer.

As it can be seen, each computer has different characteristics in both the processor and the GPU, where the computer with features of lower capability is the PC1 although it has a high capability RAM. For this equipment, a score of 3.0 is obtained in the computing capability that MATLAB gives to GPUs, which is the lowest rating the software allows to perform CNN trainings. On the other hand, PC2 and 3 have somewhat more equitable characteristics, differing notably in the size of the RAM within the general description of the computer, and the Clock rate of the GPU, giving an obvious advantage in the qualification obtained (see 0).

Table I: Computer Characteristics

	PC1	PC2		PC3
Processor	i7-4510U	i7-4710HQ		i7-7700HQ
Generation	4th Generation Intel® Core™	4th Generation Intel® Core™		7th Generation Intel® Core™
Num. Cores	2	4		4
Frequency	2,00 GHz	2,50 GHz		2,80 GHz
RAM	16 GB	8 GB		16 GB
OS	Win10 x64	Win10 x64		Win10 x64

Table II: GPU Characteristics

	GPU PC1	GPU PC2	GPU PC3
GPU	GeForce GT 750M	GeForce GTX 850M	GeForce GTX 1050 Ti
Total Memory	2048 MB GDDR5	4096 MB DDR3	4096 MB GDDR5
Multiprocessor Count	2	5	6
CUDA Cores	384	640	768
Clock Rate	941 MHz	901 MHz	1493 MHz
MATLAB Capability Score	3.0	5.0	6.1

B. Datasets and CNN Architectures

To do the training, a training and a validation database are built, which consist of 7 categories to recognize, which are different hand gestures labeled "Forward", "Backwards", "Stop", "Up", "Down", "Right" and "Left". Each training category consists of 200 images in JPG format in RGB color scale, with variable square sizes, obtaining a total of 1400 images that will form the database. For validation, each category consists of 20 images, for a total of 140. An example of each category is shown in 0. Because all images have different sizes, the input size defined for the architectures will be done with a resize function during training.

Figure 1: Examples of the Dataset categories.

For training on each computer, the same CNN architectures will be used to have an equivalent baseline. For this, two architectures with the same level of depth are built, but varying different parameters, as for example, one where the input will be of images of size 64x64 for architecture 1 (Arq1) and 128x128 for architecture 2 (Arq2). Given this, the two defined architectures, with their different parameters, are defined in 0

The kernel is defined by the size of the filter, the step with which the filter moves (S) and the addition of zeros at the borders of the image (P). In addition, it must be taken into account that each convolution layer has an additional ReLU activation layer and the first two fully-connected layers have a ReLU layer followed by a Dropout layer [16] with a 50% disconnection.

The parameterization of one architecture having a larger batch size than the other is due to the processing capacity of PC1, since with a larger batch, the CUDA, or parallel computing architecture of the GPU, tends to have execution errors for

Table III: Architectures Implemented and Their Training parameters

Architecture	Arq1			Arq2		
Type	**Kernel**		**Filters**	**Kernel**		**Filters**
Input	64 x 64		-	128 x 128		-
Convolution	4x4	S=1 P=2	32	8x8	S=1	32
Convolution	4x4	S=1 P=2	64	8x8	S=1	32
MaxPooling	2x2	S=2 P=2	-	2x2	S=2 P=1	-
Convolution	5x5	S=1	128	10x10	S=1	64
Convolution	5x5	S=1	128	10x10	S=1	64
MaxPooling	2x2	S=2	-	2x2	S=2	
Convolution	4x4	S=1	256	7x7	S=1	128
MaxPooling	2x2	S=2	-	2x2	S=2	-
Fully-Connected	1		512	1		256
Fully-Connected	1		1024	1		512
Fully-Connected	1		7	1		7
Softmax	7		-	7		-
Learning Rate	0.0001			0.0001		
Batch Size	100			56		
Training Epochs	300			300		

long waiting times when processing a certain amount of data, that is, it is not able to start or finish training by data saturation or memory overflow.

On the other hand, a slow learning rate is used with the objective that, in the 300 epochs in which the architectures are trained, these manage to achieve a training accuracy of at least 95%.

An important parameter that must be initialized to avoid very variable behaviors in the trainings performed in each computer and that all have the same initial parameters, are the initial values of the first convolution layer and fully-connected, since in order to do the initialization, random values are used and due to this, if this process were done again in each training, the initial values would change and this would affect the amount of calculations necessary for the neural network to converge to a high training accuracy, and therefore, a comparison would not be really under the same conditions on all equipment. Given this, an initialization of random values of these layers is done once for Arq1 and for Arq2, which will be loaded as initial parameters in each training.

Results and Discussions

Once the databases and the architectures are defined and initialized, the training of the networks in each computer is carried out. At the end of the training, different comparison parameters are obtained to verify the performance of each PC and make the respective comparison.

A. Training comparison for Arq1

The first parameter of comparison is the time that each computer takes to perform the respective training. 0 illustrates the time taken by the 3 computers to complete the respective training, where it can be observed that the PC1 uses much more time than the other two computers, in other words, while PC2 and PC3 were able to train the architectures in less than 120 minutes (2 hours), PC1 finished the training in a time greater than 4 hours, i.e. it took more than two times longer to complete the training, regardless of whether the PC1 GPU (941 MHz) has a higher speed than PC2 (901 MHz) and its RAM has also better capability. In addition, although the PC2 and PC3 characteristics of the GPU do not differ so drastically in their characteristics, only in speed, PC2 spent a little less than twice the time required by PC3 to perform the training.

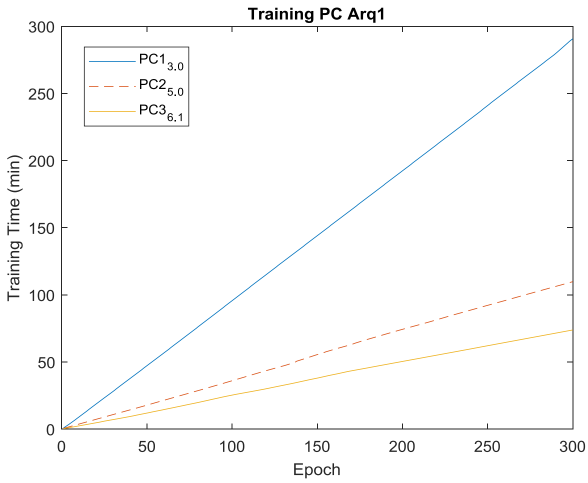

Figure 1: Training Time for Arq1 in each PC.

The second and third parameter is the accuracy that was reached in the training and training loss obtained, respectively. As can be seen in 0 and 0, the behavior of both

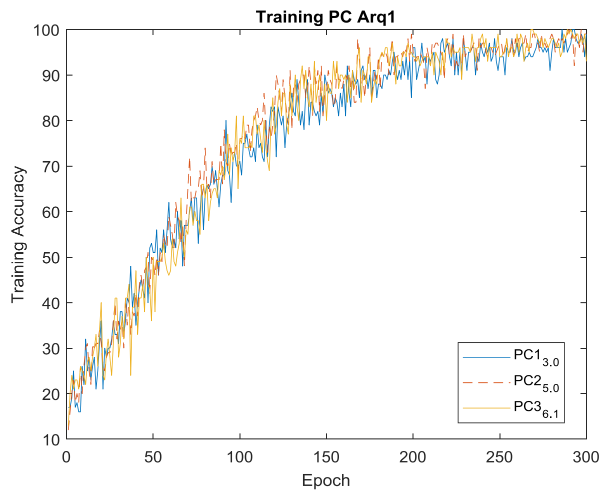

Figure 2: Training Accuracy for Arq1.

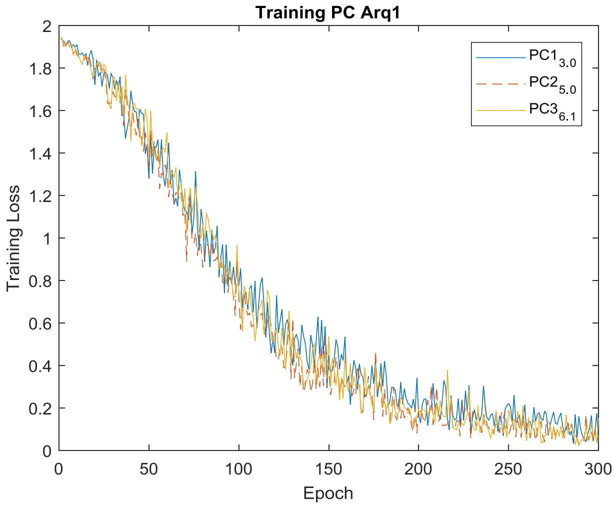

Figure 3: Training Loss for Arq1.

accuracy and loss was very similar in each of the computers, i.e. although one is delayed more than another, the behavior at the moment of calculations of both weights and bias will converge in the same way (it have to be taken into account that each architecture is initialized in the same way on all 3 computers). However, it can also be seen that, although the variation was small, PC1 started to obtain better results, but as epochs passed, it started to be slightly behind.

B. Training comparison for Arq2

For architecture 2 the same comparison parameters for the 3 computers are evaluated. With respect to the first parameter (Training Time), in 0 it is observed that the time used has the same growth relation as with Arq1 in terms of the size of the input, i.e., that the input image is twice as large. However, using half the number of filters in Arq1 caused the number of calculations to be reduced by half, compensating for the computational cost caused by the size of the filters, which were twice as large as those used in Arq1.

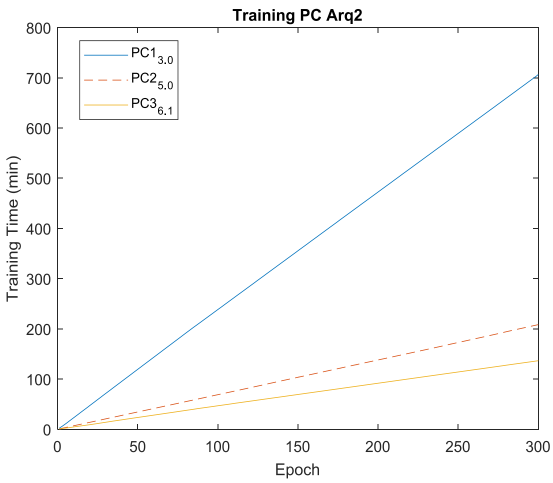

Figure 4: Training Time for Arq2 in each PC.

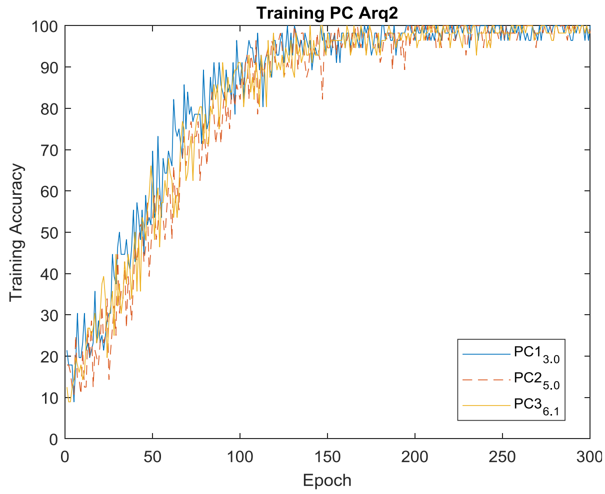

Figure 5: Training Accuracy for Arq2.

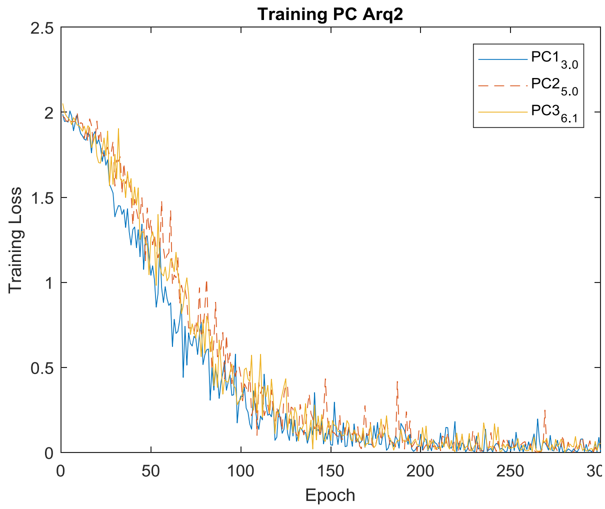

Figure 6: Training Loss for Arq2.

With regard to the following two parameters, a similar behavior was observed between the three computers, with the only difference that the training performed with the PC1 obtained a slightly better behavior, i.e., its accuracy grew faster than those obtained in PC2 and PC3, therefore, the training loss also decreased faster, as illustrated in 0 and 0.

C. Validation of the architectures trained

With the networks already trained, it is proceeded to obtain a last comparison parameter called validation accuracy which is obtained using the built test dataset, in order to verify which is the greater practical accuracy reached and how much it varies between one computer and another. To obtain this result, confusion matrices are used to not only observe the overall accuracy, but also how they recognized each of the categories.

0 shows the results obtained in both training and validation, where each computer can be compared in a better way.

In this table it can be seen the total execution time of the trainings for 300 epochs, evidencing the great time difference. Although MATLAB rates the processing power of the PC3 GPU twice as good as that of PC1, the results obtained give an overall capability of more than 5 times of processing when training the network with the Arq2, and approximately 4 times with the Arq1. For this reason, because MATLAB does not take into account the type of the processor, it cannot be deduced that the reduction of the execution time will be equivalent to the comparison of the GPU scores granted by MATLAB, since the processor can have a substantial influence on the execution of calculations that do not need to be performed by the GPU.

On the other hand, there are other results within the training that have some degree of incongruence. Since the architectures were initialized identically in the 3-equipment tested, the training result would converge to a similar validation result, however, although comparing the training of Arq1 between PC1 and PC2/3, resulting in an identical accuracy, in the training of Arq2, the PC1's accuracy was higher by about 3%, also obtaining this accuracy at an epoch well below other computers. In spite of this, the trend of recognition between categories was very similar between the best epochs obtained from the two architectures.

In general, the training and validations showed similar tendencies or results close to each other, and taking into account the processing time in which the training of the architectures was carried out, the best performance was obtained by PC3.

CONCLUSIONS

This work showed a comparison between three different computers with variable characteristics in both the processor and the GPU, performing a demonstration of computational capability through an important parameter when using neural networks, in this case CNN, which is the time in which a network is trained. With this, it is easier to have a criterion of choice of computational capability that a computer requires according to the needs of the user in both the processor and the graphics card, focused on the performance of these trainings.

Within the training of neural networks, it should be considered different computational aspects, such as the capability of the GPU to use and even multiprocesses that it can execute, since an algorithm with a very high computational cost can prevent training or even generate errors of data processing and possibly errors within the same operating system or computer drivers. In this context, it can be inferred that the amount of CUDA cores and mainly the amount of multiprocessors that a graphics card has may mean a drastic variation in processing capacity during CNN training, additionally, when the characteristics are similar in these two aspects, the GPU speed may play an important role in training times.

An important criterion for choosing a portable computer is the maximum capacity it has when performing neural network training, especially with respect to the GPU. An example of this occurred within the tests performed, where it was observed that PC1 has a maximum processing capability for a

Table IV: Overall Results

Architecture	**Arq1**			**Arq2**		
Computer	*PC1*	*PC2*	*PC3*	*PC1*	*PC2*	*PC3*
Training time (min)	290.75	109.80	73.83	706.51	208.39	136.53
Best Epoch	255	256	284	211	267	250
Epoch Training Accuracy	95.00%	95.00%	97.00%	96.43%	100.00%	98.21%
Epoch Training Loss	0.1240	0.1091	0.1010	0.1135	0.0056	0.0242
Epoch Validation Accuracy	85.00%	85.00%	85.00%	88.57%	85.00%	85.71%
Recognition (Max. 20 img)						
Forward	17	17	17	17	17	16
Backwards	11	12	11	11	12	12
Stop	15	15	15	17	15	16
Up	20	20	20	20	20	20
Down	18	18	19	19	18	19
Right	19	20	19	20	20	19
Left	19	17	18	20	17	18

CNN architecture like Arq2, since both a larger image size and the use of a larger batch would result in a GPU memory overflow error, i.e. the amount of calculations and data stored is greater than the storage capacity of the GPU.

ACKNOWLEDGEMENT

The authors are grateful to the Nueva Granada Military University, which, through its Vice chancellor for research, finances the present project with code IMP-ING-2290 and titled "Prototype of robot assistance for surgery", from which the present work is derived.

REFERENCES

[1] R. Ata, "Artificial neural networks applications in wind energy systems: a review", Renewable and Sustainable Energy Reviews, 2015, vol. 49, pp. 534-562.

[2] F. Amato, A. López, E. M. Peña-Méndez, P. Vaňhara, A. Hampl and J. Havel, "Artificial neural networks in medical diagnosis", Journal of Applied Biomedicine, 2013, vol. 11, pp. 47-58.

[3] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks For Large-Scale Image Recognition", 2015, arXiv preprint arXiv:1409.1556v6

[4] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks", 2013, arXiv preprint arXiv:1311.2901

[5] D. C. Ciresan, U. Meier, L. M. Gambardella and J. Schmidhuber, "Convolutional neural network committees for handwritten character classification. In International Conference on Document Analysis and Recognition (ICDAR), IEEE, 2011, pp. 1135-1139.

[6] P. Barros, S. Magg, C. Weber and S. Wermter, "A multichannel convolutional neural network for hand posture recognition", In International Conference on Artificial Neural Networks, Springer, Cham, 2014, pp. 403-410.

[7] I. Orozco, M. E. Buemi and J. J. Berlles, "A study on pedestrian detection using a deep convolutional neural network. In International Conference on Pattern Recognition Systems (ICPRS-16), IET, 2016. pp. 1-4.

[8] M. Halicek, et al. "Deep convolutional neural networks for classifying head and neck cancer using hyperspectral imaging", Journal of Biomedical Optics, 2017, vol. 22, no 6, pp. 060503.

[9] Y. Dong, et al. "Evaluations of deep convolutional neural networks for automatic identification of malaria infected cells", In IEEE EMBS International Conference on Biomedical & Health Informatics (BHI), IEEE, 2017, pp. 101-104.

[10] K. Chellamuthu, et al. "Atherosclerotic vascular calcification detection and segmentation on low dose computed tomography scans using convolutional neural networks", In 14th International Symposium on Biomedical Imaging (ISBI 2017), IEEE, 2017, pp. 388-391.

[11] O. Abdel-Hamid, A. R. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition", In IEEE/ACM Transactions on audio, speech, and language processing, 2014, Vol. 22, no 10, pp. 1533-1545.

[12] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet classification with deep convolutional neural networks", In Advances in neural information processing systems, 2012, pp. 1097-1105.

[13] V. Vanhoucke, A. Senior and M. Z. Mao, Improving the speed of neural networks on CPUs", In Proc. Deep Learning and Unsupervised Feature Learning NIPS Workshop, 2011, pp. 4.

[14] Z. Luo, H. Liu and X. Wu, "Artificial neural network computation on graphic process unit", In International Joint Conference on Neural Networks, IJCNN'05, Proceedings, IEEE, 2005, pp. 622-626.

[15] D. Strigl, K. Kofler and S. Podlipnig, "Performance and scalability of GPU-based convolutional neural networks", In 18th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), IEEE, 2010, pp. 317-324.

[16] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting", in Journal of Machine Learning Research 15, 2014, pp. 1929-1958