UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Visión Electrónica
Más que un estado sólido

http: revistas.udistrital.edu.co/ojs/index.php/visele/index

VISIÓN ELECTRÓNICA

A RESEARCH VISION

Evaluation of hyperparameters in CNN for detecting patterns in images

Evaluación de hiperparámetros en CNN para detección de patrones de imágenes

Robinson Jiménez Moreno.[1]***, Oscar Avilés.***[2]***, Diana Marcela Ovalle.***[3]

INFORMACIÓN DEL ARTÍCULO

Historia del artículo:
Enviado: 12/01/2017
Recibido: 22/02/2017
Aceptado: 10/04/2017

Keywords:
Convolutional neural network
Deep learning
Image recognition
Pattern recognition

Palabras clave:
Red neuronal convolucional
Aprendizaje profundo
Reconocimiento de imagen
Reconocimiento de patrones

ABSTRACT

Deep learning techniques have emerged as an effective solution to the problems of current pattern recognition techniques, such as neural networks. Within these new techniques, the convolutional neural networks (CNN) offer an integration to the recognition of patterns in images, given by the traditional set of images processing plus neuronal networks. This article presents the analysis of the different hyper parameters that imply the training of a CNN, which allows to validate the effects on the accuracy of the network. It is used as a base the recognition of electric energy meters, obtaining a network with an accuracy of 96.32 %.

RESUMEN

Las técnicas de aprendizaje profundo han surgido como una solución eficaz a los problemas de las actuales técnicas de reconocimiento de patrones, como las redes neuronales. Dentro de estas nuevas técnicas, las redes neuronales convolucionales (CNN) ofrecen una integración al reconocimiento de patrones en imágenes, dados por el conjunto tradicional de procesamiento de imagen más redes neuronales. El presente artículo expone el análisis de los diferentes hiperparámetros que implican el entrenamiento de una CNN, que permite validar los efectos en la precisión de la red. Se emplea como imágenes de la base de pruebas, el reconocimiento de medidores de energía eléctrica, logrando obtener una red con una exactitud del 96,32 %.

[1]BSc. In Electronic Engineering, Universidad Distrital Francisco. José de Caldas, Colombia. MSc. In Industrial Automatization, Universidad Nacional de Colombia. Current position: Professor, Universidad Militar Nueva Granada, Colombia. E-mail: robinson.jimenez@unimilitar.edu.co.
[2]BSc. In Electronic Engineering, Universidad Antonio Nariño, Colombia. MSc. In automatics systems of production, Universidad Tecnológica de Pereira, Colombia. PhD. In Mechanics engineering, Universidade Estadual de Campinas, Brasil. Current position: professor Universidad Militar Nueva Granada, Colombia. E-mail: oscar.aviles@unimilitar.edu.co.
[3]BSc. In Electronic Engineering, Universidad Distrital Francisco José de Caldas, Colombia. MSc. In Electric Engineering, Universidad de los Andes, Colombia. PhD. In Industrial technology, Universidad Politécnica de Cartagena, España. Current position: professor Universidad Distrital Francisco José de Caldas, Colombia. E-mail: dmovalle@udistrital.edu.co.

Citar este artículo como: R. Jiménez, O. Avilés, D. M. Ovalle. "Evaluation of hyperparameters in CNN for detecting patterns in images". Visión Electrónica, algo más que un estado sólido, Vol. 11, No. 2, 140-145, july-december 2017.
https://doi.org/10.14483/22484728.14092

1. Introduction

The convolutional neural networks (CNN) are part of the recent deep learning techniques [1], which are oriented to applications of object recognition in images, evidencing a high performance in this task. The applications covered are oriented to the recognition of objects in various fields, for instance, detection in radar images [2], or taken from great heights, as in airplanes or satellites [3].

Additionally, CNNs present applications in robotics, oriented in the detection and grip of objects [4, 5], in medicine for recognition and diagnosis of pathologies [6–8] and anthropometric characterizations by pattern recognition [9–11]. Within the framework of this last application, it becomes relevant developments such as the one presented in [12], where the discrimination of states of fatigue in a driver, has potential contribution to the prevention of loss of life by vehicular accident.

The obvious benefits of convolutional networks, as discussed in the state of the art, include learning a large number of information patterns [13], even in speech recognition [14]. Although in the present development their fundamentals training are exposed, a greater understanding of these can be evidenced in [15].

The training of this type of networks involves determining multiple parameters, typically set in a heuristic way, although they are explained the theory related to the subject, it have not been exposed in the state of art consulted in a comparative way in the performance of a particular learning under variations thereof and the hardware used, where this will be the contribution of this work, which in turn seeks to guide through the tests exposed, criteria to quickly converge in an efficiently trained network.

The document is divided into three sections. The first section, corresponds to a brief explanation of the training of convolutional neural networks to show the parameters that are going to be determined. The second section, corresponds to the results of the variation of these parameters in the training for tool recognition. Finally, the last section presents the conclusions reached.

2. CNN Training parameters

The convolutional neural networks operate based on the convolution function, described by (1), between a pattern to be learned (w) and an input volume (h). The first volume that the network faces is the image that contains the basic training patterns. This image, either in grayscale or color, is composed of matrix by channels, 1 for grayscale or a binarized image and 3 for each channel of a color image, according to RGB standard (Red, Green and Blue) or even the YCbCr color space.

$$h_j^n = max\left(0, \sum_{k=1}^{K} h_k^{n-1} * w_{kj}^n\right) \tag{1}$$

The convolution operation involves a fixed input volume and a fixed filter size, which means that for the training of the network a database of images containing the objects to be recognized must be built, sorted by the categories that each object will belong to. Each category may have a different number of images and each image a different size, however, the input images must be resized uniformly in order to be entered to the network. This resizing operation may involve variations of the learning characteristics, for example, if the image is of considerable size in pixels (greater than a Megapixel), resizing will lose image resolution, as seen in Figure 1. This effect might get worse if the image is not square. So that the dimensions of the input image of the network already with the respective resizing are part of the parameters to be determined and, therefore, affect the computational cost.

Figure 1: Loss of resolution.

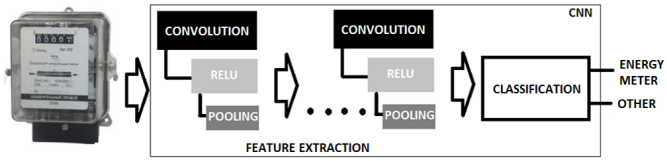
Source: own.

Another relevant aspect of the database is the number of images, the larger this is, the better the convolution filters can be set. This database must be divided into a training group and a validation group, in average ratio of 60/40 to 80/20. On the other hand, the training time will be directly affected by the size of the database, since the larger it would be, the longer the network takes to learn. During the training of the neural network, there is a configurable parameter called mini-Batch, which consists in determining how many samples of the training database are to be taken in each iteration, i.e., to take a fixed amount of images that the network will analyze to perform the weight update, and additionally, obtain the gradient of the loss function of the network in said analysis. Once all the elements of the database have been taken, the last iteration is called Epoch, and then the

process starts again.

The architecture of the network is made up of two fundamental sections, the learning of characteristics and the classification, so that the convolution filters learn in the first section, as shown in Figure 2. This section may be constituted of basic form by groupings of layers of convolution, pooling and rectification (ReLU). The ReLU layer simply overrides the negative values, while the pooling layer reduces the input volume of the next layer by means of the maximum or average methods, using (1) and (2), of a group of neighboring pixels whose size is another parameter to be determined, together with the number of convolution layers - ReLU – pooling.

Figure 2: CNN Architecture.



Source: own.

$$h_j^n(x,y) = max_{x\in N(x),y\in N(y)} h_j^{n-1}(\bar{x}\bar{y}) \tag{2}$$

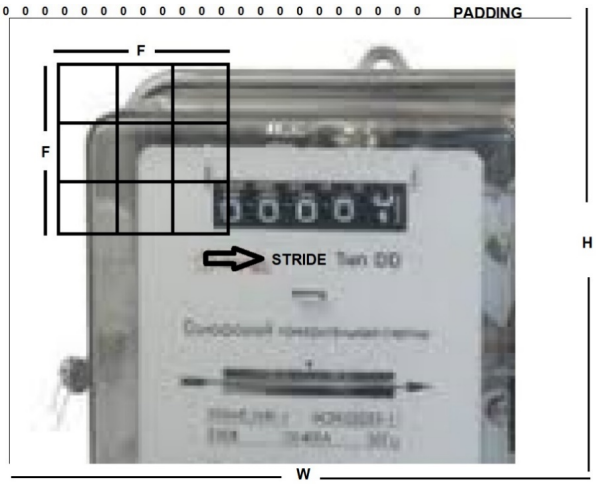
The convolution process is also determined by other setup parameters for network training. The stride (s) corresponds to the displacement of the convolution filter with respect to the image, which must be less than or equal to the size of the filter (F) to reach all the pixels. However, when there is relevant information at the borders of the image, a problem arises, which is that the filters can only pass once through that region, causing that it does not take sufficient relevance to the information that is in the borders, for that reason, an additional parameter is added, which is called padding, which increases the size of the edges without adding new relevant information, normally zeros, allowing the filter to pass a greater number of times around the borders to better acquire the information. The number of columns or rows that are added must be less than the size of the filter. Both parameters in conjunction with the input volume (Hn, Wn), determine the size of the output volume by (3) and (4) and their respective depth by (5). Figure 3 below, illustrates this network characteristics.

$$W_{n+1} = \frac{W_n - F + 2P}{S} + 1 \tag{3}$$

$$H_{n+1} = \frac{H_n - F + 2P}{S} + 1 \tag{4}$$

$$D_{n+1} = K_n \tag{5}$$

Figure 3: CNN Convolution Parameters.



Source: own.

3. Analisys and results of parameter variation

Due to the multiplicity of combinations that can be obtained from the different variations of the training parameters, only some of them will be taken to be able to appreciate its incidence in the determination of an optimal classification architecture. Next, these variations are subjected to the training by two computers of similar characteristics but one using CPU processing and the other by GPU, both computers are characterized by having a Seventh generation Core i7 processor and 16 GB of RAM, and the GPU is an NVIDIA 1050 4GB memory card.

In the following tables, the effect of using two image databases resized at different scales can be validated, subject to two types of networks with variations of their hyperparameters.

In Tables 1 and 2 the same network combinations are observed, subject to the same input databases, but validating training variations based on changes in processing equipment. The main difference lies in the time spent, where the CPU consumption becomes noticeable slowing down additional training processes and as evidenced by prolonged times, implying a high computational cost. The small differences in the efficiency of the trained network are given by the variations belonging to a particular training under ran-

Table 1: Training using CPU, Learning Rate 0.001.

	INPUT IMAGE 64X64				**INPUT IMAGE 128X128**			
	NETWORK 1		**NETWORK 2**		**NETWORK 1**		**NETWORK 2**	
Type	**Kernel**	**Filters**	**Kernel**	**Filters**	**Kernel**	**Filters**	**Kernel**	**Filters**
Convolution/ ReLU	5x5	30	5x5	30	5x5	30	5x5	30
MaxPooling	3x3	-	3x3	-	3x3	-	3x3	-
Convolution/ ReLU	3x3	50	3x3	50	3x3	50	3x3	50
MaxPooling	2x2	-	2x2	-	2x2	-	2x2	-
Convolution/ ReLU	N-I	N-I	2x2	10	N-I	N-I	2x2	10
MaxPooling	N-I	N-I	2x2	-	N-I	N-I	2x2	-
Fully-Connected	4	-	4	-	4	-	4	-
Softmax	4	-	4	-	4	-	4	-
Training Time	6,15 Hours		14,5 Hours		9,45 Hours		16,12 Hours	
Accuracy	78 %		76 %		81,2 %		82,6 %	

Source: own.

Table 2: Training using GPU, Learning Rate 0.001.

	INPUT IMAGE 64X64				**INPUT IMAGE 128X128**			
	NETWORK 1		**NETWORK 2**		**NETWORK 1**		**NETWORK 2**	
Type	**Kernel**	**Filters**	**Kernel**	**Filters**	**Kernel**	**Filters**	**Kernel**	**Filters**
Convolution/ ReLU	5x5	30	5x5	30	5x5	30	5x5	30
MaxPooling	3x3	-	3x3	-	3x3	-	3x3	-
Convolution/ ReLU	3x3	50	3x3	50	3x3	50	3x3	50
MaxPooling	2x2	-	2x2	-	2x2	-	2x2	-
Convolution/ ReLU	N-I	N-I	2x2	10	N-I	N-I	2x2	10
MaxPooling	N-I	N-I	2x2	-	N-I	N-I	2x2	-
Fully-Connected	4	-	4	-	4	-	4	-
Softmax	4	-	4	-	4	-	4	-
Training Time	0,15 Hours		0,21 Hours		0,22 Hours		0,35 Hours	
Accuracy	78,42 %		77,03 %		81,6 %		84,87 %	

Source: own.

Table 3: Training using GPU, Learning Rate 0.01.

	INPUT IMAGE 64X64				**INPUT IMAGE 128X128**			
	NETWORK 1		**NETWORK 2**		**NETWORK 1**		**NETWORK 2**	
Type	**Kernel**	**Filters**	**Kernel**	**Filters**	**Kernel**	**Filters**	**Kernel**	**Filters**
Convolution/ ReLU	5x5	30	5x5	30	5x5	30	5x5	30
MaxPooling	3x3	-	3x3	-	3x3	-	3x3	-
Convolution/ ReLU	3x3	50	3x3	50	3x3	50	3x3	50
MaxPooling	2x2	-	2x2	-	2x2	-	2x2	-
Convolution/ ReLU	N-I	N-I	2x2	10	N-I	N-I	2x2	10
MaxPooling	N-I	N-I	2x2	-	N-I	N-I	2x2	-
Fully-Connected	4	-	4	-	4	-	4	-
Softmax	4	-	4	-	4	-	4	-
Training Time	0,134 Hours		0,202 Hours		0,2092 Hours		0,312 Hours	
Accuracy	74,2 %		71,15 %		79,32 %		80,51 %	

Source: own.

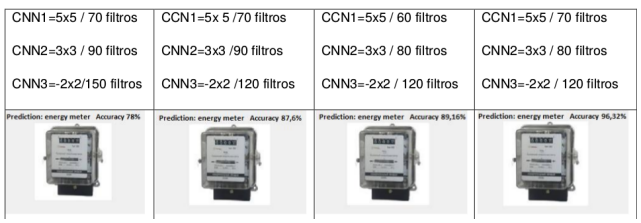
dom images that it uses in the separation of the group of training and validation. The network architecture 1 consists of 2 sets of Convolution-ReLU-Pooling layers, while network 2 consists of 3 of these sets, where the notation N-I in the tables refers to Not Implemented, denoting the difference between the two networks.

It can be seen at the same time how the training with larger images deliver more learning information to the network, however for the case used an increase in the depth of the network is not significantly better.

Table 3 is articulated with Table 2, showing the variations of the hyperparameters and depth of the network, varying the Learning rate to observe its effect. The first noticeable change is a reduction of learning time, but the efficiency of category discrimination decreases, even a test with this value in 0.005, searching for a mean between the two cases, evidence times and intermediate efficiencies between the tabulated ones. Where compared to computational cost, it is preferable to devote more time to training under this training parameter.

From the tests carried out, the network 2 is set as the base architecture with an input size of 128x128 and an accuracy of 84.87 % according to Table 2, due to the fact that it is the one that presents better performance in relation to the computational cost and thus more flexibility in the number of tests. Figure 4 illustrates the change in the number of filters for each of the three convolution layers, for this architecture, showing an improvement in accuracy up to 96.32 %. Such change must obey to a balance of information to learn from the previous layer, it is observed that an excessive use of filters, degrade the final performance.

Figure 4: Tuning setup of the CNN Architecture.



Source: own.

Once the most efficient network architecture has been determined, we proceed to use it as a classification tool, in order to evaluate its eficient in times terms of classes prediction in both, GPU and CPU equipment. Table 4 illustrates the results obtained, it is evident that there is no strong incidence in the operating time of the network trained under the architecture change, which was to be expected, although the GPU equipment responds about 10 milliseconds faster, for object recognition applications, it is an insignificant difference.

Table 4: Classification times.

	CPU Prediction	**GPU Prediction**
Time	0,112 seg	0,102 seg

Source: own.

4. Conclusions

It was observed that the incidence of the computer equipment becomes relevant for the training of the network and not for its use in the classification work, being thus indifferent to an application whether or not it has a GPU. However, it is necessary to iterate the training parameters of the network to optimize the architecture to be used, even when starting from a right database, according to the considerations established here, where the convergence times demonstrate the obvious advantage of using the GPU to do these iterations.

It was possible to observe that variations in the depth of the network does not imply better a performance or a better learning of characteristics. So that if the categories are highly discriminable, very deep architectures are not required. Trying to introduce a volume of input as large as possible, allows a better learning of filters for different categories, although the computational cost is considerable, taking into account that the work of the network is the prediction, the use of equipment high performance becomes an intermediate step, where the final application can even run on low-cost embedded systems, such as a raspberry pi.

To validate the presented tests, it is necessary to make gradual changes of a single parameter at a time, allowing the correct analysis of the effect that causes its variation in the training. Multiple iterations show the final relationships that allow to converge more quickly in an efficient network, but emphasizes the importance of an adequate database, i.e., that generates clear patterns for the learning. It is important to bear in mind that the exposed tests start from the same initial training conditions of each network, since the filters initialize randomly, the same parameters are adjusted in each test performed.

References

[1] P. Poonia, V. K. Jain, A. Kumar, "Deep Learning: Review", *International Journal of Computer & Mathematical Sciences IJCMS*, vol 5, Issue 12, december 2016, pp. 43- 47.

[2] J. Li, C. S. Wang, H. Wang and B. Zhang, "Classification of very high resolution SAR image based on convolutional neural network" International Workshop on Remote Sensing with Intelligent Processing (RSIP), Shanghai, 2017, pp. 1-4, https://doi.org/10.1109/RSIP.2017.7958811

[3] Z. Deng, L. Lei, H. Sun, H. Zou, S. Zhou and J. Zhao, "An enhanced deep convolutional neural network for densely packed objects detection in remote sensing images" International Workshop on Remote Sensing with Intelligent Processing (RSIP), Shanghai, 2017, pp. 1-4, https://doi.org/10.1109/RSIP.2017.7958800

[4] Z. Wang, Z. Li, B. Wang, H. Liu. "Robot grasp detection using multimodal deep convolutional neural networks". *Advances in Mechanical Engineering*, vol 8, Issue 9: September 2016, https://doi.org/10.1177/1687814016668077

[5] J. Redmon and A. Angelova, "Real-time grasp detection using convolutional neural networks" IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, 2015, pp. 1316-1322, https://doi.org/10.1109/ICRA.2015.7139361

[6] M. Halicek, G. Lu, J. V. Little "Deep convolutional neural networks for classifying head and neck cancer using hyperspectral imaging". *J. Biomedic Opt*, vol 22, no. 6, March 2017; https://doi.org/10.1117/12.2255562

[7] Z. Xiaolong, J. Beth, T. Chung "Self-Recalibrating Surface EMG Pattern Recognition for Neuroprosthesis Control Based on Convolutional Neural Network". *Frontiers in Neuroscience*. Vol 11, 2017, p 379, https://doi.org/10.3389/fnins.2017.00379

[8] S. Vieira, W. Pinaya, A. Mechelli, "Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and applications", *Neuroscience & Biobehavioral Reviews*, vol 74, 2017, pp. 58-75, https://doi.org/10.1016/j.neubiorev.2017.01.002

[9] I. P. Marras and I. Patras, "Deep Refinement Convolutional Networks for Human Pose Estimation" 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition, Washington, DC, 2017, pp. 446-453, https://doi.org/10.1109/FG.2017.148

[10] J. Gan, L. Lichen, Z. Yikui, L. Yinhua "Deep self-taught learning for facial beauty prediction", *Neurocomputing*, vol 144, november 2014, pp. 295-303, https://doi.org/10.1016/j.neucom.2014.05.028

[11] I. Song, K. Hyun-Jun and P. B. Jeon. "Deep learning for real-time robust facial expression recognition on a smartphone" Consumer Electronics (ICCE), IEEE International Conference on, 2014.

[12] K. Dwivedi, K. Biswaranjan, A. Sethi, "Drowsy driver detection using representation learning" Advance Computing Conference (IACC), IEEE International, 2014, pp.995, 999, https://doi.org/10.1109/IAdCC.2014.6779459

[13] 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.

[14] W. N. Hsu, Y. Zhang, A. Lee, and J. Glass, "Exploiting Depth and Highway Connections in Convolutional Recurrent Deep Neural Networks for Speech Recognition", In Interspeech, 2016, pp. 395-399.

[15] M. D. Zeiler, and R. Fergus. "Visualizing and Understanding Convolutional Networks". Springer : In Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) Computer Vision – ECCV, 2014.