International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 19 (2018) pp. 14151-14157

Convolutional Neural Networks Training for Tools Recognition

Paula Catalina Useche-Murillo[1], Javier Orlando Pinzón-Arenas[2], Robinson Jiménez-Moreno[3]

Department of Mechatronics Engineering, Nueva Granada Military University, Bogotá, Colombia.

[1]u3900235@unimilitar.edu.co, [2]u3900231@unimilitar.edu.co, [3]robinson.jimenez@unimilitar.edu.co

Abstract

This paper presents the training of a convolutional neural network using MATLAB ® and the MatConvNet library, which is oriented to the recognition of three tools: "Scalpel", "Screwdriver" and "Scissors". The training algorithm has as input a series of images with the three categories established, and results in an individual classification, according to the tools set. The validation error and the level of accuracy were evaluated by confusion matrices, with the objective of establishing the best convolutional architecture, for which the training parameters were varied: depth, dimension and characteristics of the database, as well as the number of training epochs. The convolutional neural network training presents an accuracy of 97,3%.

Keywords. Machine vision, convolutional neural network, object recognition.

INTRODUCTION

Numerous investigations in the field of machine vision, such as those presented in [1-4], show that convolutional neural networks have acquired a strong importance in the development of applications in this field. Their applications in the performance of complex tasks such as the recognition and classification of objects in a real world environment show a higher performance than those achieved by conventional image processing algorithms, which at the same time is fundamental for the development of robots that interact autonomously in a given medium looking for identification objects, human machine interaction and similar applications. The parameters of a convolutional neural network that meet the objective of object classification depend on characteristics such as training images with which the network learns the objects to be classified, the number of classifications and the degree of similarity between the images of different categories, hence there is no general structure to implement a convolutional neural network, making each case unique and must be considered independently as mentioned in [5].

In [6] the training of a convolutional neural network is presented for the classification of 1000 object categories with 1.2 million high resolution images, reaching errors of up to 26.2% in the classification, which means that out of a total of 100 images to be classified, approximately 26 of them will be misclassified, and 74 will be classified correctly.

Another consideration is the high level of operations that the machine must perform during the training of a convolutional neural network makes the GPU (Graphics Processor Unit) an important factor for the rapid training of such networks, as evidenced in [6,7], where the GPU proved to be a useful tool for training with very deep convolutional neural networks or with very large databases. In [7], for example, a GPU was used for the training of a deep convolutional neural network for the detection of objects in the NORB, CIFAR10 and MNIST databases obtaining validation errors between 0.35% and 19.51% in the classification where, the lower the error, the greater the recognition capacity of the network.

The validation error is obtained by taking a certain group of "validation images", which is not found among the training images, and entering them into the network in order to observe how many of them are correctly recognized in the learning process. When said error is less than 20%, as in the case of [7], it means that about 80% of the validation images were classified correctly.

From another perspective, the database used to train a network strongly influences the results of training and classification, generating an increase or reduction in validation error. The larger the training database and the less different the images of a category from each other, the better the results of the network, as was observed in [4,8], where comparisons were made with databases of different sizes.

The convolutional neural networks have different fields of application in addition to the classification of objects in real environments, some examples of activities or tasks in which these networks are used are the recognition and tracking of hands [9], visual analysis of documents for character classification [8], detection of fatigue states [10], recognition and classification of tools [3], animal recognition [11,12], recognition of environments such as some cities [13], among others.

This article is organized as follows. In section 2 " Convolutional Neuronal Network Architecture ", a brief introduction of the different types of layers that can be used in a convolutional neural network is made, the base structure from which the network for object recognition was raised is shown and the main operation of each of the layers is explained. Then, section 2.1 "Training" clarifies details on the databases used for the training of the convolutional neural network to then train and observe the behavior of the validation error of the different trained networks. In section 3 "Results" the results obtained for the different types of validated network architectures are analyzed. In section 4 "Analysis", confusion matrices are used to determine the percentage accuracy of the network, and the error and accuracy of the best trained networks are compared. Finally, in section 5 "Conclusions", the conclusions derived from the training of convolutional networks for the discrimination of tools are established.

CONVOLUTIONAL NEURONAL NETWORK ARCHITECTURE

In order to establish the architecture of a deep convolutional neural network oriented to the recognition of objects, the first step is to set the classification of these and to structure the corresponding database, for the case they are taken as tools objects within three categories: "Scalpel", "Screwdriver" y "Scissors". The software used for the development is MATLAB®, using the MatConvNet library designed for use in this software [13], which allows to vary the structure and parameters of a network, train it, test it and observe the behavior of the error epoch by epoch in a simple and flexible way [14].

The next step is to set the network parameters. As it can be seen in Fig. 1 and as explained in [14] and [15], the network can be composed of layers with or without particular parameters such as the CONV and RELU layer, where CONV represents a convolutional layer that results in the convolution between the input volume and a bank of K filters with dimension F. RELU refers to the Rectified Linear Units layer, which behaves as an activation function eliminating all values less than zero, the POOL layer refers to a Max Pooling used to reduce the size of the input image (subsampling), the DROP layer allows to disconnect neurons randomly and avoid overtraining [16]. The FC or Full Connected layer obtains the membership values of the evaluated image to determine which category belongs and finally the last layer SF or Softmax that takes the results obtained in the FC and passes each value to a number between 0 and 1 to choose the winning category.

In Fig. 1, the input to the network is a grayscale image that contains the shape of the object to be recognized and has a volume equal to the height by the width by the depth of the image, where the depth is equal to 1 for grayscale images and 3 for color images. Then it is processed in the CONV layer which is characterized by the parameters F, K, P, S (see Table 1) where P or padding represents an optional addition of zeros to the edges of the image and S is how many pixels the convolution filter F is going to move over the input image (see Fig. 2). The resulting volume of the CONV layer enters a POOL layer with parameters F1 and S1 where the input volume is reduced, then it goes to the RELU layer which has no parameters and then to a DROP layer where only the percentage of disconnection is specified.

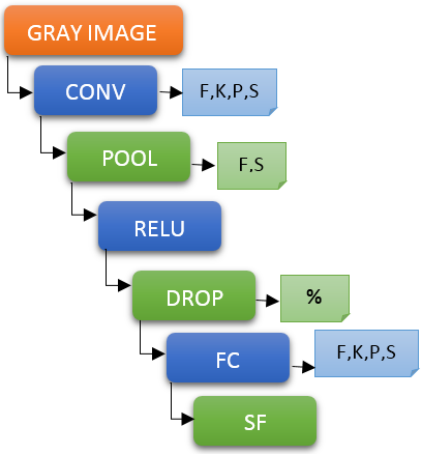


Figure 1. Structure of a two layer convolutional network.

Table I summarized the different abbreviations used in this article to refer to the different types of layers in a network and to each of its parameters.

Table 1. Abbreviations of layers and parameters

Abbreviation	Layer
DROP (D)	Dropout
CONV	Convolutional
RELU	Rectified Linear Units
POOL	Max Pooling
NORM	Normalization
FC	Full Connected
SF	Softmax
Abbreviation	**Parameter**
K	Number of filters
F	Size of square filters
S	Stride
P	Padding

When operating the volume of the input image (width x height x depth) with a CONV or POOL layer, changes in the output volumes of these layers are generated as shown in Fig. 2. Depending on the dimensions of the input volume to the mentioned layers and the existence or not of padding considered in the input, the dimensions of the output volume were calculated to be able to choose the parameters of the following layer without having errors of dimension, following the (1) to (3).

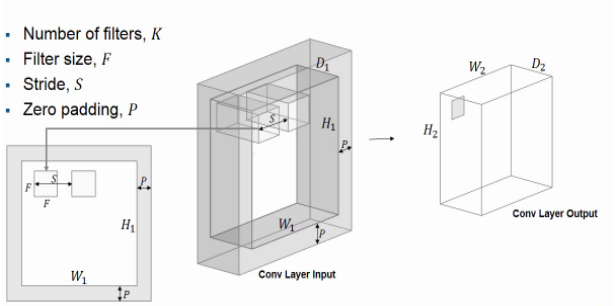


Figure 2. Input and output volumes of the convolution layer (Source: http://www.mathworks.com/)

Fig. 2 shows a conv input layer representing the input volume of the convolution layer with width W1, high H1, depth D1 and Padding P, and a square filter of dimension F with a pitch S, with the same depth of the input volume and K number of filters to be applied on the input image. The output volume has dimensions H2 and W2, which depend on the dimensions of the filter and the input volume and can be calculated by (1) to (3).

The choice of the layer type, the number of layers and the order of the layers were chosen from a base structure (Fig. 1), varying them manually according to the tasks of each layer and the training objective.

$$W_2 = \frac{W_1 - F + 2P}{S} + 1 \qquad (1)$$

$$H_2 = \frac{H_1 - F + 2P}{S} + 1 \qquad (2)$$

$$D_2 = K \quad (3)$$

For the training, two databases were used: a base A with 50 images and a base B with 300 images, both correspond to each of the three tools (scalpel, screwdriver, and scissors). In the larger database, there were used images that were very similar to each other, where each one was repeated several times but with different orientations, while in the other database, images of different geometries of each one of the tools were used, i.e. different shapes and sizes. This in order to vary the database to observe the behavior of the training and the network already trained.

Initially, each of the training and test images were taken and scaled to a smaller square size than the original and converted to grayscale as seen in Fig. 3 to simplify the calculations and to accelerate the network training process.

Figure 3. Training tools normalized to 128x128x1

An initial network structure was then fixed as shown in Fig.1, and the parameters were calculated from the dimensions and characteristics of the tools that were expected to be captured with the learning. Finally, about 50 tests were performed in order to adjust the parameters and dimensions of the network to achieve a network capable of classifying the three tools. The results of these tests are set out below.

RESULTS

From the scaling dimensions of the images and the distinguishing characteristics of each tool: such as the screwdriver handle, the sharp point of the scalpel and the handle of the scissors, different sizes of filters were proposed for the parameters of the network, in order to capture these characteristics and make the network learn them. Based on Fig. 3, it was estimated that the size of the filters should be about 1/10 the size of the input image to be able to encompass the aforementioned characteristics, and add other smaller filters to identify details such as the slim body of the scalpel and the screwdriver.

Initial tests were performed with 128x128x1 images such as those observed in Fig. 3, with 12x12 size filters for the first convolutional layer and a total of 45 filters, achieving a validation error of 64%, as observed in Fig. 4, where that error showed a constant behavior over the epochs, suffering minor sporadic variations. This behavior is inconvenient because it shows that the network is not really learning. When the network manages to learn some categories, it is expected that the behavior of its error will be as in Fig. 5, where the error (blue signal) becomes smaller as epochs goes by, which means that each time is able to correctly classify a greater number of images.

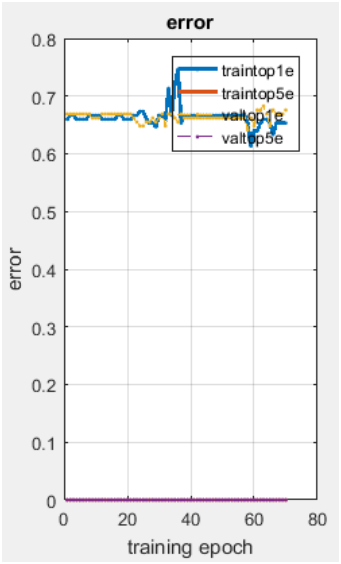


Figure 4. Error with shallow convolutional network

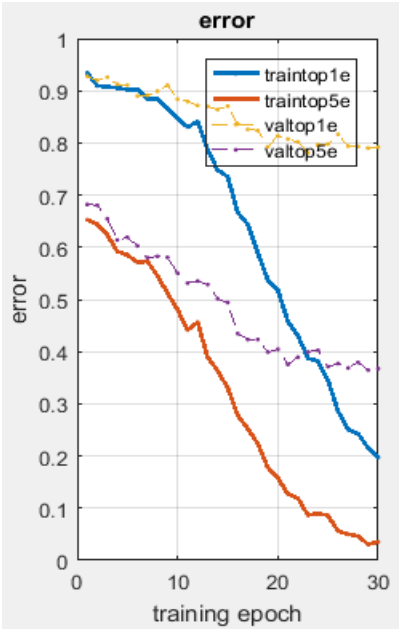


Figure 5. Expected error of training with the CIFAR-10 database

Subsequently, the training of the network was varied with the small database taking 50 images of each category, and the parameters were readjusted to try to generate more significant changes in the error, but it remained stable between 60% and 65% with behavior that is observed in Fig. 4, so that another layer of convolution was added to give greater depth to the network hoping that it would have more learning capacity.

In Fig. 6 it can be seen that, when adding the other convolution layer, a slight reduction in the error was achieved, however, the

minimum error that was obtained by varying the parameters of the layers was only 57% as illustrated in Fig. 7.

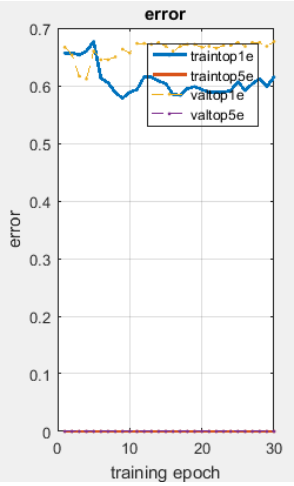


Figure 6. Small error reduction (59% error).

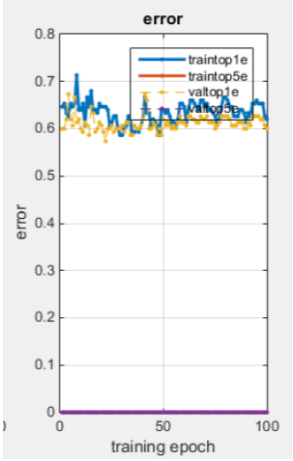


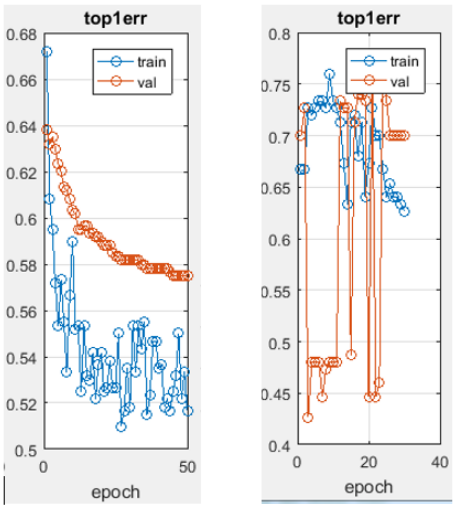
Figure 7. 57% Error.

Seeing that the error remained relatively constant, other tests were performed by increasing the depth, changing to the second database of the training images, normalizing the images to 224x224x1, varying the network layer types, using Dropout layers, and changing parameters such as the size and quantity of filters in all layers, achieving better results than in previous tests.

Fig. 8 shows a better error behavior when handling a network with three convolutional layers (Fig. 8-a), and in Fig. 8-b, an irregular behavior was observed but with a much lower error than the previous tests, also using a network of three convolutional layers but with different parameters.

Comparing the parameters of the different trained network structures that generated error rates less than 60%, it was observed that when using a series of filters of relatively large size, as seen in Fig. 9, and a number of filters K1 = 8, K2 = 16, K3 = 32 or K3 = 64, being each the same number of filters for each convolutional layer, networks with errors of less than 57% could be trained.

It was also observed that when using a batch size smaller than the number of training images, unstable error behaviors were obtained as in Fig. 8-b, whereas when using a batch size equal to the number of training images, the Error tended to fall regularly as in Fig. 8-a.



a b

Figure 8. 57% error (a, orange signal). 42% error (b, orange signal)

A reduction in the batch size caused the error to fall faster but with the risk that it would rise again a few epochs later, as observed in Fig. 8-b, where there were large error reductions at some epochs (epoch 3 with 42%) and then a sharp increase in error of about 20% (epoch 12 with 73%).

The structure of Fig. 9 was tested with more normalization layers before each convolution layer, achieving validation errors of about 60% with a behavior similar to that of Fig. 8-a. Subsequently, networks with deeper architectures were trained to seek a reduction in the error, as presented in the following section, where a comparison between the best results obtained is made.

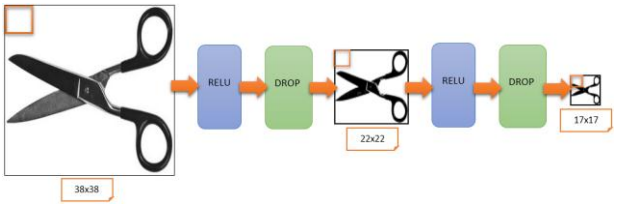


Figure 9. Final structure of the training network.

ANALYSIS

After obtaining several convolutional networks with error margins between 40% and 60%, the 3 best percentages were taken and their accuracy was evaluated for the classification of tools using confusion matrices, which serve to represent and analyze the degree of classification accuracy of an already trained network [10], Organizing the different cases presented during the classification in a square matrix as seen in Table 2.

The rows represent the actual classification of the tools, while the columns represent the classification assigned by the network [10].

Table 2. Confusion Matrix

		PREDICTED CLASS	
		NEGATIVE	POSITIVE
REAL CLASS	NEGATIVE	A	B
	POSITIVE	C	D

Because the network was trained to recognize three different types of tools, a confusion matrix for three classes was used, with the structure shown in Table 3. On the diagonal of the table it can be seen the number of correctly recognized tools (A, E, I). At the bottom of the table was added the accuracy of the network in percentage when sorting tools.

Table 3. Confusion matrix for 3 categories

		Classification		
	Error: %	SCALPEL	SCREWDRIVER	SCISSORS
Real Class	SCALPEL	A	B	C
	SCREWDRIVER	D	E	F
	SCISSORS	G	H	I
		Accuracy		
		%		

Accuracy was calculated as the sum of the number of correctly classified tools, divided by the sum total of images evaluated by the network as observed in (4).

$$Ac = \frac{A + E + I}{A + B + C + D + E + F + G + H + I} \qquad (4)$$

After training networks with three layers of convolution without obtaining error percentages below 40%, it was proceeded to train networks with four convolutional layers, where it was possible to establish an architecture capable of correctly recognizing more than 95% of the images evaluated. Table 4 summarize the values of accuracy obtained with the different trained networks, finding that networks with four layers of convolution present considerable improvements in accuracy with respect to architectures with only 3 convolutions.

In Table 5, it was calculated the percentage of success of each tool correctly classified with respect to the total of correctly classified tools, finding that the network with 2.66% error, whose architecture is presented in Fig. 11, presented a greater ease to recognize the three types of tools with similar percentages to each other, while the architectures with 3 layers of convolution tended to recognize more one tool than the other two, as shown in Table V.

Table 5. True Positive Recognition Rate

	Error 42.6%	Error 52%	Error 57.3%	Error 3%
Total True Positive	102	97	100	292
SCALPEL	35%	20%	58%	100%
SCREWDRIVER	12%	21%	40%	92%
SCISSORS	53%	55%	0%	100%

In Table 6 it is possible to detail the recognition behavior of the network 4 with respect to the three categories. This network, despite having a higher accuracy than the other three networks, has a tendency to confuse screwdrivers with scalpels, however, it is considered an acceptable error since it does not represent more than 10% of the total rated screwdrivers.

Table 6. Matrix of confusion for the network 4

		Classification		
	Error: 0.02666	**SCAL**	**SCREW**	**SCIS**
Real Class	**SCALPEL**	100	8	0
	SCREWDRIVER	0	92	0
	SCISSORS	0	0	100
			Accuracy	97.3333%

An example of positive recognition of the three tools is presented in Fig. 10, where none of them belong to the training or validation images.

Figure 10. Correct recognition of the tools.

Table 4. Accuracy of the best trained networks

CNN	Error Rate	Architecture	Parameters	Image Dataset	Number of training images	Accuracy
1	42.6%	2 CONV / 2 RELU / 2 DROP / 1 POOL	K1=10 / K2=30 / F1=38 / F2=20 / D1=0.8 / D2=0.5 / S1=2 / P1=2	A	50	34.00%
2	52%	2 CONV / 2 RELU / 2 DROP / 1 POOL	K1=30 / K2=90 / F1=38 / F2=20 / D1=0.8 / D2=0.5 / S1=2 / P1=2	A	50	32.33%
3	57.3%	3 CONV / 3 RELU / 2 DROP / 3 NORM	K1=8 / K2=16 / K3=32 / F1=38 / F2=22 / F3=17 / D1=0.8 / D2=0.5 / S1=2 / P1=2	B	100	33.00%
4	**2.66%**	4 CONV / 4 RELU / 2 POOL	K1=20 / K2=30 / K3=40 / K4=60 / F1=22 / F2=18 / F3=12 / F4=8 / $F1_{pool}$=2 / $F2_{pool}$=3 / $S1_{pool}$=2 / $S2_{pool}$=2	B	100	**97.33%**

The final tests were performed with a base structure as shown in Fig. 11, with four convolutions not counting the FC, two POOL and one RELU by convolution.

CONCLUSIONS

The training and correct classification of objects with convolutional neural networks depends on variables such as the depth of the network, the type of layers that compose it and the database used in training. It was observed that shallow networks may become insufficient when classifying objects from a small database.

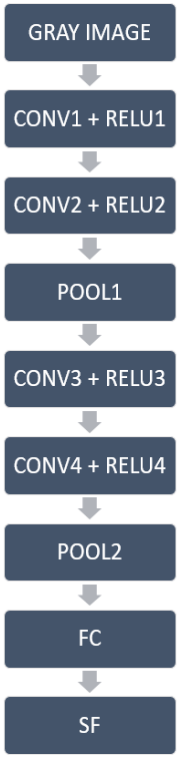


Figure 11. Architecture of the convolutional neural network with accuracy of 97.33%

Considering that the training databases for this type of network are usually greater than 1000 images per category [6], for classification groups of few classes or few discriminants between them, it is necessary to increase the database to observe the behavior in the training of the network using those that generated better results, And increase the number of epochs to allow a greater training process with a view to reducing the error to the maximum without overtraining the network [3].

In order to determine the recognition quality of a network, it is insufficient to rely on the validation error since it does not provide the correct recognition percentage for each category individually, but is based on how many of the validation images were correctly recognized at a given epoch of training. Regarding this, the confusion matrix allowed a detailed analysis of the relationship between the number of successful predictions in one category compared to the others and in order to assess how likely it is that the network is capable of recognizing all categories or recognizing only one of them.

The increase of a convolutional layer in the architecture of the network cause considerable improvements in the performance of recognition of the images, thanks to the fact that it has more possibilities to extract important characteristics for the classification of each element than with only three convolutions. In addition, the order in which the layers are added also influences the training of the network, since the resulting image changes with respect to each of the layers and generates new characteristics that determine the classification of the images

ACKNOWLEDGMENT

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 (2017-2018) and titled "Prototype of robot assistance for surgery", from which the present work is derived.

REFERENCES

[1] Maturana D, Scherer S. (2015) VoxNet: A 3D Convolutional Neural Network for real-time object recognition. In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) [Internet]. Institute of Electrical and Electronics Engineers (IEEE); 2015 Sep; DOI: http://dx.doi.org/10.1109/iros.2015.7353481.

[2] Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L. (2014) Large-Scale Video Classification with Convolutional Neural Networks. In IEEE Conference on Computer Vision and Pattern Recognition [Internet]. Institute of Electrical and Electronics Engineers (IEEE); 2014 Jun; DOI: http://dx.doi.org/10.1109/cvpr.2014.223.

[3] Wang Z, Li Z, Wang B, Liu H. (2016) Robot grasp detection using multimodal deep convolutional neural networks. Advances in Mechanical Engineering [Internet]. SAGE Publications; 2016 Sep 23;8(9). DOI: http://dx.doi.org/10.1177/1687814016668077.

[4] Vinyals O, Toshev A, Bengio S, Erhan D. (2015) Show and tell: A neural image caption generator. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) [Internet]. Institute of Electrical and Electronics Engineers (IEEE); 2015 Jun; DOI: http://dx.doi.org/10.1109/cvpr.2015.7298935.

[5] Ming Liang, Xiaolin Hu. (2015) Recurrent convolutional neural network for object recognition. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) [Internet]. Institute of Electrical and Electronics Engineers (IEEE); 2015 Jun; DOI: http://dx.doi.org/10.1109/cvpr.2015.7298958.

[6] Krizhevsky A, Sutskever I, Hinton G. (2012) ImageNet classification with deep convolutional neural networks. University of Toronto. Conference on Advances in neural information processing systems, 2012, pp 1097-1105.

[7] Dan C. Ciresan, Ueli Meier, Jonathan Masci, Luca M. Gambardella, Jurgen Schmidhuber. (2011) Flexible, High Performance Convolutional Neural Network for Image Classification. In Tweny-Second International Joint Conference on Artificial Intelligence, 2011 Proceedings. Manno-Lugano Switzerland.

[8] Simard PY, Steinkraus D, Platt JC. (2003) Best practices for convolutional neural networks applied to visual document analysis. Seventh International Conference on Document Analysis and Recognition, 2003 Proceedings [Internet]. Institute of Electrical and Electronics Engineers (IEEE); DOI: http://dx.doi.org/10.1109/icdar.2003.1227801.

[9] Nowlan S, Platt J. (1995) A Convolutional Neural Network Hand Tracker. Inproceedings: Neural Information Processing Systems Foundation.

[10] Jiménez R. (2011) Sistema de Detección de Nivel de Cansancio em Conductores Mediante Técnicas de Visión por Computador. National University of Colombia. Bogotá D.C., Colombia.

[11] Yao L, Miller J. (2016) Tiny ImageNet classification with convolutional neural networks. Stanford University.

[12] CS231n Convolutional Neural Networks for Visual Recognition. http://cs231n.github.io/classification/

[13] Vedaldi A, Lenc K. (2015) MatConvNet. Proceedings of the 23rd ACM international conference on Multimedia - MM'15 [Internet]. Association for Computing Machinery (ACM); 2015; DOI: http://dx.doi.org/10.1145/2733373.2807412.

[14] CS231n Convolutional Neural Networks for Visual Recognition. http://cs231n.github.io/linear-classify/

[15] Vedaldi A, Lenc K. (2016) MatConvNet, Convolutional neural networks for MATLAB. Proceedings of the 23rd ACM international conference on Multimedia - MM '15.

[16] Srivastava Nitish. (2013) Improving Neural Networks with Dropout. University of Toronto.