24th COBEM - 2017

24th ABCM International Congress of Mechanical Engineering
December 3-8, 2017, Curitiba, PR, Brazil

COBEM-2017-1471
DEEP LEARNING APPROACH BASED ON CONVOLUTIONAL NEURAL NETWORKS FOR IMAGE PROCESSING APPLICATIONS

João Guilherme Sauer
Marco Antonio Reichert Boaretto
Electrical Engineering Graduate Program (PPGEE), Federal University of Parana (UFPR), Curitiba, Brazil
joao.sauer@gmail.com , marco.boaretto@hotmail.com

Edson Gnatkovski Gruska
Industrial and Systems Engineering Graduate Program (PPGEPS), Pontifical Catholic University of Parana, Curitiba, Brazil
edson.gruska@gmail.com

Arthur Beltrame Canciglieri
Industrial and Systems Engineering Undergraduate Program, Pontifical Catholic University of Parana (PUCPR), Curitiba, Brazil
artcanci_98@hotmail.com

Gabriel Herman Bernardin Andrade
Electrical Engineering Graduate Program (PPGEE), Federal University of Parana (UFPR), Curitiba, Brazil
gabrielhbandrade@gmail.com

Leandro dos Santos Coelho
Industrial and Systems Engineering Graduate Program (PPGEPS), Pontifical Catholic University of Parana (PUCPR), and
Electrical Engineering Graduate Program (PPGEE), Federal University of Parana (UFPR), Curitiba, Brazil
leandro.coelho@pucpr.br

Abstract. *Machine learning has been successfully applied to a wide variety of knowledge fields ranging from information retrieval, data mining, and speech recognition, to computer graphics, visualization, and human-computer interaction. The general focus of machine learning is the representation of the input data and generalization of the learnt patterns for use on future unseen data. The goodness of the data representation has a large impact on the performance of machine learners on the data: a poor data representation is likely to reduce the performance of even an advanced, complex machine learner, while a good data representation can lead to high performance for a relatively simpler machine learner. In this context, deep learning has received significant attention recently as a promising solution to many complex problems in the Artificial Intelligence and signal processing fields related to processing of unstructured data. Among different types of deep artificial neural networks, convolutional neural networks (CNNs) have been most extensively studied. CNNs demonstrate superior performance when compared to other machine learning methods in the object detection and recognition applications. Each stage in a CNN is composed of a filter bank, some non-linearities, and feature pooling layers. With multiple stages, a CNN can learn multi-level hierarchies of features. The contribution of this paper is related to an overview of CNN applications and its key benefits related to image processing and computer vision. Furthermore, an image processing case study related to Mechatronics and Automation in the automotive field is evaluated using a CNN design approach.*

Keywords: *machine learning, deep learning, convolutional neural networks, image processing.*

1. INTRODUCTION

Computer vision has become ubiquitous in our society, with applications including search approaches, image understanding, apps, mapping, drones, and self-driving cars. The quality of visual features is crucial for a wide range of computer vision topics, e.g., scene classification, object recognition, and object detection, which are popular in recent computer vision venues. All these image classification tasks have traditionally relied on hand-crafted features to try to capture the essence of different visual patterns.

Fundamentally, a long-term goal in Artificial Intelligence (AI) is to build intelligent systems that can automatically learn meaningful feature representations from a massive amount of image data. On the other hand, machine learning (ML) paradigms have generated huge societal impacts in a wide range of applications.

Machine learning is a subfield of AI. It can be used in several domains and the advantage of this method is that a model can solve problems which are complex to be represented by explicit algorithms. A new breed of artificial neural networks has arisen in prominence within an emerging branch of ML, known as deep learning (DL), which is dedicated to the development of advanced pattern recognition systems modeled after human perception and cognitive processes. Mimicking the structure and function of the human neocortex, DL models comprise of a variety of advanced pattern recognition techniques that consider the hierarchical and temporal nature of human information processing.

DL is one promising avenue of research into the automated extraction of complex data representations (features) at high levels of abstraction. See illustration in Figure 1 of a classical ML information flow and DL flow. The main difference between traditional machine learning and deep learning algorithms is in the feature engineering. In traditional machine learning algorithms, we need to hand-craft the features. By contrast, in DL algorithms features engineering is done automatically by the algorithm. Feature engineering is difficult, time-consuming and requires domain expertise. The promise of deep learning is more accurate machine learning algorithms compared to traditional machine learning with less or no feature engineering. DL algorithms develop a layered, hierarchical architecture of learning and representing data, where higher-level (more abstract) features are defined in terms of lower-level (less abstract) features. The hierarchical learning architecture of DL algorithms is motivated by AI, which emulate the deep layered learning process of the primary sensorial areas of the neocortex in the human brain, extracting, automatically, features and abstractions from the underlying data (Bengio et al., 2013). DL algorithms are quite beneficial when dealing with learning from large amounts of unsupervised data, and typically learn data representations in a greedy layer-wise fashion (Schmidhuber, 2015; Liu et al., 2017). Significant performance gain of DL is mainly due to the increased availability of labeled data and processing power.

(a) traditional machine learning flow **(b)** deep learning flow

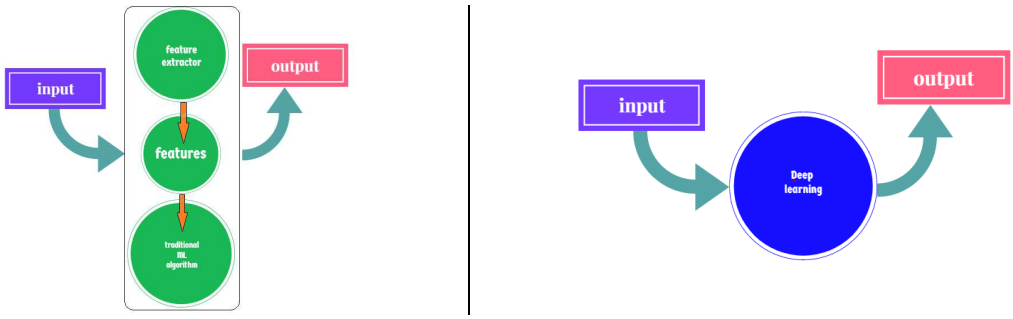

Figure 1. Flow data in a traditional machine learning flow and a deep learning flow.

In terms of DL approaches, convolutional neural networks (CNNs) have demonstrated remarkable performance and attracted considerable attention in recent years. Then applied in practice and across a wide range of computer vision tasks, such as image classification, object detection, semantic segmentation, object tracking and visual question answering (Guo et al., 2016; Zhao et al., 2017). Figure 2 illustrates the representation of a CNN applied to a classification problem related to image processing.

CNNs are biologically-inspired variants of artificial neural networks and composed of alternating convolutional layers and pooling layers. Considering the spatial structure of images, each convolutional layer supplies a particularly well-adapted architecture of local receptive fields and shared weights. Hence, neurons in each layer will only be connected to a small region of the lower layer, instead of all neurons in a fully-connected manner.

The contribution of this paper is related to an overview of CNN applications and its key benefits related to image processing and computer vision. The remainder of the paper is arranged as follows. In Section 2, the fundamentals of the artificial neural networks are presented. After, several applications of CNNs in image processing are introduced in Section 3. In Sections 4 and 5, a CNN design is validated to a classical case of image processing. Finally, concluding remarks are made in the last section.

2. FUNDAMENTALS OF ARTIFICIAL NEURAL NETWORK (ANN)

In this section, we will discuss about ANNs, its fundamentals, basic concepts and some architectures. A little review about CNNs is presented and discussed. Some layers of a classical CNN design are mentioned and commented.

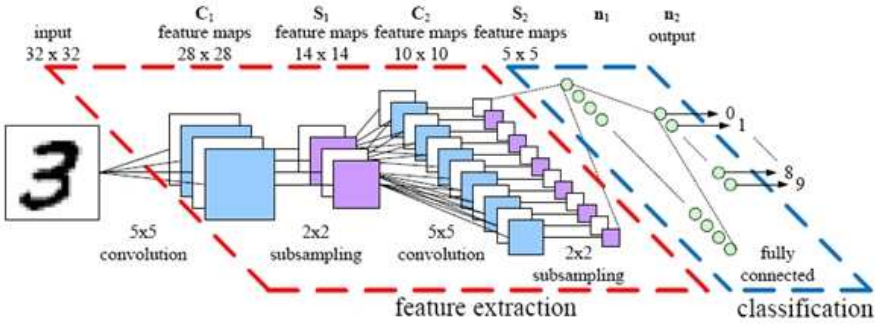

Figure 2. Representation of a CNN applied to a classification problem (Caraffi et al., 2012)

2.1 Fundamentals of DL

The term Deep Learning , in machine learning was introduced by Dechter (1986) and later as referring to ANN used by Aizenberg et al. (2013). One successful approach when working with a high dimensionality dataset, consist on reduce the dataset size by representing the original dataset's features in a smaller dimension by using feature-extraction techniques. However, feature-extraction is a complex and time-consuming operation that is highly application-dependent. DL, different from machine learning, has a unique automatic feature-extraction procedure, in which each hidden layer is responsible for training a unique set of features based on the output of the previous layer, as shown in Figure 3.

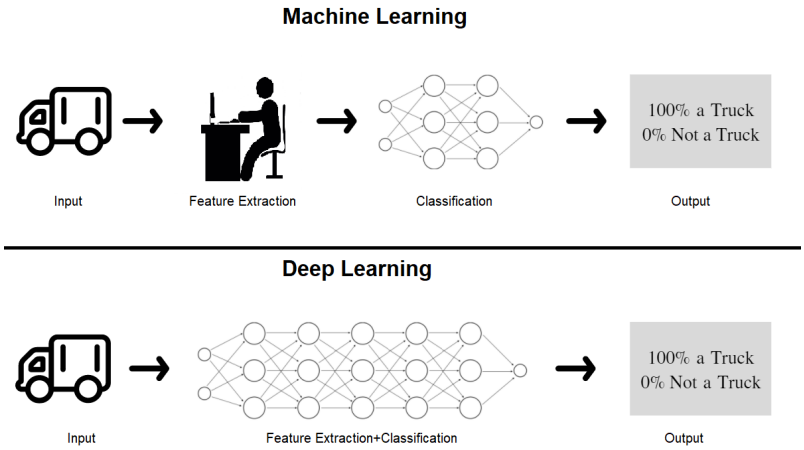

Figure 3. Classical Machine Learning approach versus Deep Learning

DL tries mimic the neocortex of the brain, which is responsible by many of cognitive abilities. The brain's neocortex propagate the sensory signals through a complex hierarchy of models (Lee and Mumford, 2003), that performs extraction of complex data representations (features) at high levels of abstraction (Najafabadi et al., 2015). DL develop a layered, hierarchical architecture of learning and representing data, where higher-level (more abstract) features are defined in terms of lower-level (less abstract) features (Najafabadi et al., 2015). DL algorithms use a huge amount of unsupervised data to automatically extract complex representation and have the capability to generalize in non-local and global ways, generating learning patterns and relationships beyond immediate neighbors in the data (Bengio and LeCun, 2007).

The main DL approaches are: Convolutional Neural Networks (CNNs), Deep Belief Networks (DBNs), Stacked Auto-Encoders, Hierarchical Temporal Memory (HTM) and Deep Spatiotemporal Inference Network (DESTIN). Details about CNN are presented in the next subsections.

2.2 CNN

The first general DL network, working learning algorithm for supervised, deep, feedforward, multilayer perceptron was published by Ivakhnenko and Lapa (1965). A paper described a deep network with eight layers trained by the group method of data handling algorithm (GMDH) (Ivaknenko, 1971).

The first CNN model, named as LeNet, was developed by LeCun et al. (1998) in order to classify handwritten digits using The MNIST (Modified National of Standards and Technology) dataset on the training step of the LeNet, as shown

in Figure 4. The authors proved that the use of the CNN convolutional nature eliminates the need for hand-crafted feature extractors (LeCun et al., 1998).

The CNN's inspiration bases on the studies of the human visual cortex, in this particular cortex there is a small region with neurons that are sensible to information of the visual range. In 1962, Hubel and Wiesel performed an experiment where they were able to prove that when the human visual cortex is exposed to images composed by oriented edges in a specific direction there are neurons which generate strong synapses to this exposure. In this experiment they realized that these neurons form an specific structure in which each neuron generate a synapse for each determined edge orientation, and together theses neurons are able to produce a visual perception (Hubel and Wiesel, 1962).

By performing this experiment the authors manage to conclude that each neuron has a determined task in order to find a specific feature, and by combining the neurons' task all feature detected by this neurons form the human visual perception (Hubel and Wiesel, 1962).

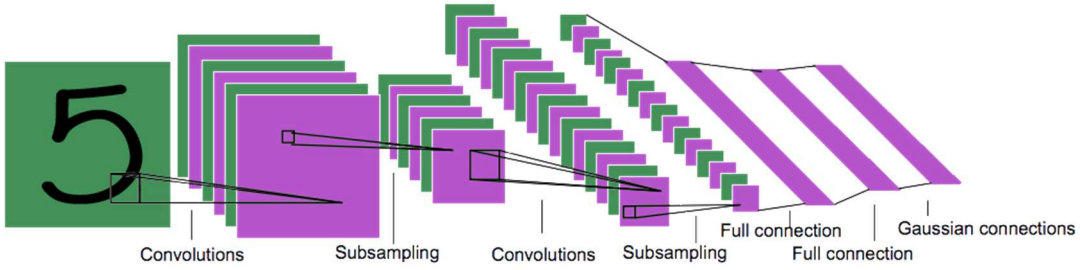

Figure 4. Illustration of the LeNet architecture

A classical CNN uses filters known as kernels, that are applied on the image through convolutions to extract features, hence mathematically performing the biological neuron function. The application of the convolution step on images, generate new images with determined features, and then these images successively generate new inputs for the deeper layers of the CNN, where the convolutions are applied again with different kernels sizes. By this dynamic, the superficial layers of the CNN can extract simpler features as vertical and horizontal edges, and as long as the image is advancing through deeper layers, more complex features are extracted as geometric shapes, until it reaches the layers where the combinations of these features (face and objects) are extracted. In general, the deepest the layer of the CNN more abstract is the extracted feature. The analysis of all the abstracted features generates the image classification procedure.

The CNN architecture is divided in layers, and in each layer a different kind of operation is performed. The CNN layers are divided as presented in the next subsections.

2.2.1 Input Layer

In this layer is located the input data, and the size of this layer vary on the image size of the dataset in case. Usually the input layer takes an order three tensor as input with an image of *M* rows and *N* columns, and 3 channels (Red, Green and Blue color channels), as shown by Fig. 5. The CNN can also take tensors of higher orders but the size of the input data can inflict directly on the processing time of the CNN.

2.2.2 Convolution Layer

In the convolution layer of a CNN the kernels used are generally matrices with small dimensions (3x3,5x5,7x7…), as in the ANN model weights are attributed for each value of the positions of these kernels. The kernels move over the input image performing small convolutions. The outcome of the convolution it's the product of the small convolutions performed by the kernel over the image, as shown in Figure 6. The discrete 2D convolution is given by

$$y(m,n) = x(m.n) * h(m,n) = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \times h[m-i, n-j] \quad (1)$$

where *y* is the output matrix, *x* is the kernel matrix and h is the input matrix. The variables *m*, *n*, *i* and *j* are the iterators used to represent each positions of the matrices' values.

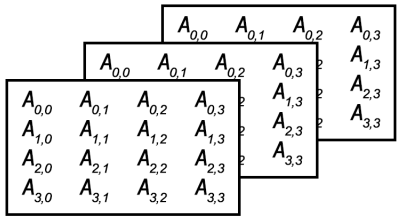

Figure 5. Example of a 4x4x3 input image

In this context, more than one convolution layer can be applied in the CNN to extract a higher number of features in different levels of abstraction.

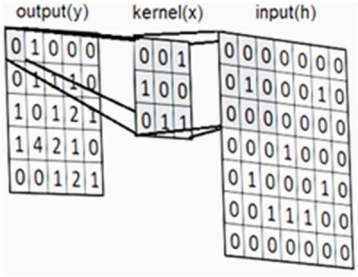

Figure 6. 2D image convolution

2.2.3 ReLU Layer

The Rectified Linear Unit (ReLU) layer does not change the size of the input that is applied in order to increase the non-linearity in the image, as given by Eq. 2. Since the semantic information of the input image is highly non-linear the purpose of the ReLU layer is to make the output of the convolutional layer non-linear as well (Wu, 2017). A possible representation is given by

$$y_{i,j,d} = \max\{0, x^{l}_{i,j,d}\} \tag{2}$$

where y and x are respective the output matrix and input matrix, the values of i, j and d are the values of the positions of the input matrix constrained between the ranges of $0 \leq i < M$, $0 \leq j < N$, and $0 \leq d < D$ (channels), and l is the layer.

2.2.4 Pooling Layer

In the Pooling layer, is performed transformations on the feature map prevenient of the previous layer, in order to increase the spatial invariance of the CNN (Scherer et al., 2010), which means that the CNN can distinguish features and recognize patterns regardless of the image position or texture. The max pooling operation, as shown in Figure 7, the maximum value of the feature map that is inside the pooling kernel goes to the pooled feature map.

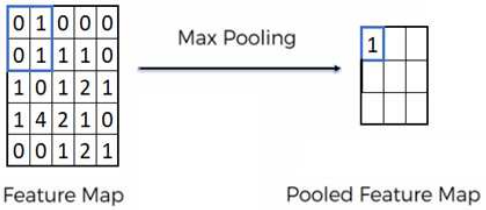

Figure 7. Example of max pooling operation

The max pooling operation is given by:

$$y_{j+1} = \max(x_j^{nxn} u(n,n)) \tag{3}$$

where x_j represents the feature map and y_{j+1} the pooled feature map, the index j represents the layer, n represents the pooling kernel size and $u(n,n)$ a window function that is applied on the input patch.

2.2.5 Flattening Layer

The flattening layer is a very simple step, consists on the vectorization of the previous layer, which is usually a pooled feature map, as shown in Figure 8. This step is usually added to the CNN when there are several pooling layers in the CNN architecture.

2.2.6 Fully connected (FC) Layer

The FC layer is commonly used at the end of the CNN, as well in other approaches used two or more consecutive FC layers. The FC layer is a Fully Connected ANN that perform the combination of the extracted features from the previous layers into more attributes to increase the prediction accuracy. When working with classification one important step to configure the FC layer is needed one output per class. The output of the FC layer is the probabilities referring to the predictions of each class.

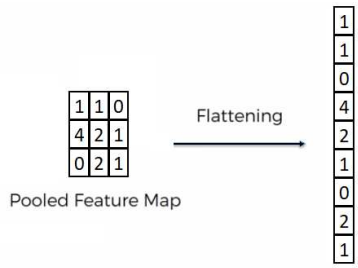

Figure 8. An example of flattening

3. CNN BACKGROUND IN IMAGE PROCESSING

In this section, we present the a very short background of CNN architectures used in image processing problems. We focused in two relevant research fields: (i) facial recognition and (ii) character recognition.

3.1 Facial Recognition

The analysis of facial images using DL has become more and more common mainly because of its performance. The CNNs allow us to strip the images in small pieces to increase the detection desirable.

Ascenso et al. (1999) explain that the complexity in do facial recognition is associated to the difficult in abstract the unique information that can make a facial image unique.

Faces have small differences between them and the same person can have different facial images, depending exclusively in its expressions, like happiness, sadness, joy and others. So, a robust system needs to focus only in the main relevance of the face, like eyes, nose or mouth. On the other hand, the forehead or chin have very few unique information.

Li et al. (2015) also say that face recognition is an area well used in the machine learning. While we have a robust system for frontal faces, there are new studies focusing in non-linear cases, where external factors can obstruct the data. Good examples would be extreme illumination, forced expressions or even different focus.

Accordingly to Zafeiriou et al. (2015), the training of a CNN for facial recognition it is a very slow task, once that it is necessary to analysis and test every interaction. So, to solve this problem its necessary to train various classifiers that matches the maximum false-positives characteristics, adding one by one to the characteristics selector, making then a scale contribution to increase the classification performance.

3.2 OCR Recognition

Per Seethalakshmi et al. (2005), the Optical Character Recognition (OCR) is responsible to recognize characters written by hand to characters in the computer, using an image that was digitalized. OCR is often used in the digitalization of documents, help in share it's information through the internet.

The most important principle in do an automatic recognition is making the machine learn the patterns classes that can occur and what they look like. The machine learning is made using different patterns and its possible variables, making then a unique classifier for the particular character. So, during the recognition, the digitalized characters can be compared with the different classes and being decided which one is the most probably one, using possible combinations (Eikvil, 1993).

Its complexity is lower than the facial recognition, but because one unique symbol can be represented by many ways that its recognition can be trick. Also, the handwriting can cause different between the same characters, what increases it's difficult. As per (Fu and Kara 2011), the symbol recognition is restricted to just the variants trained before, what makes possible to use CNNs specifics for the multilayer recognition.

4. THE TEST PROCEDURE

First off, the test was realized using MATLAB computational enviroment (MathWorks), using two scripts available from the platform. The first part was a simple check using the state of the art application, to be used as a comparative with the new approach.

The facial recognition was made using a simple CNN, with just few minutes used to train its network. The default images were also simple ones, with just simple cases, without containing any edge cases. This had caused the system to have false-positives, once that was not trained for more variance of faces.

After the facial recognition, the OCR tests were made. Once again, the plain settings were used, using only the default images that came with the script. This test image is presented in Figure 9.

Figure 9. OCR Test image

In this case, the results were perfect, mainly because the image used was very simple and without big variations. As it's possible to notice, the illumination and the quality of the characters were ideal for the recognition.

The important part about this part of the test was to verify that a CNN could be used to recognize characters in an image and translate then to real characters in a machine.

However, once that more edge cases were tested, it was noticeable that the default configuration were not enough to obtain satisfactory results, making then the use of a more robust network a necessity. In this case, the use of DL was adopted. With the use of DL, we could then show to the CNN much more images for training. However, for a more detailed and robust image recognition, would be necessary to have a very extensive quantity of images and configurations for the training period. This would then force the system to be more robust, what would be hard to make a valid comparison. So, for those reasons, it was decided that only numeric characters, from 0 to 9 would be used. For that, the database from MNIST (LeCun et al., 1998) was used. It's contain 70.000 images of 28x28 pixels of characters already being written and classified, what can then be used for the training of the Neural Network. A sample of this can be seen in Figure 5. In addition, a good approach for a CNN for this test can be made just with four hidden layers, what makes the entire system to be faster and with good results.

4.1 Description of the Case Study

The case study to be evaluated in the full paper using CNNs is called Toyota Motor Europe (TME) Motorway Dataset. The TME Motorway Dataset is composed by 28 clips for a total of approximately 27 minutes (30000+ frames) with vehicle annotation. Annotation was semi-automatically generated using laser-scanner data. Image sequences were selected from acquisition made in North Italian motorways in December 2011. This selection includes variable traffic situations, number of lanes, road curvature, and lighting, covering most of the conditions present in the complete acquisition (TME, 2017).

Vehicle detection and classification are important parts of Intelligent Transportation Systems. They aid traffic monitoring, counting, and surveillance, which are necessary for tracking the performance of traffic operations. In the full paper, the focus of CNN design is related to the vehicle detection and classification problem. The adopted CNN architecture is a remarkably versatile, yet conceptually simple paradigm that can be applied to a wide spectrum of perceptual tasks.

Despite the recent progress in DL, one of the major challenges of computer vision, ML, and AI in general in the next decade will be to devise methods that can automatically learn good features hierarchies from unlabeled and labeled data in an integrated fashion (LeCun et al., 2010).

Once that the parameters were configured for the CNN, the quantity of neurons in each layer were analyzed to get the best results with good performance. This analysis allowed a running time that could be used as base to be measured

for the other runs. So, in the end, it was measured and analyzed the time to run each period, the total time and the error signal of the CNN output.

4.2 Deeper variants

It was tested three different structures with two convolutional layers. The first, is found at literature – for convenience we denominated by *Default structure* – and it composed by 6 neurons in convolutional map with 5 neurons in pooled prop. in first layer, follow by 12 and 5 neurons in convolutional and pooled layers, respectively. For the second structure (*Proposed structure*$_1$), we change the number of neurons in each convolutional map, using 8 neurons in first layer and 15 in second layer. The las structure (*Proposed structure*$_2$), follow the same change, but with 15 and 12 neurons, for the first and the second convolutional map.

5. RESULTS ANALYSIS

After decided the study case, the scripts were ran in the same computer configuration: An Intel 3rd generation processor with 8 GB RAM (Random Access Memory). The three study cases were divided in different parts. The first one was running only in one period and its results are presented in the Table 1.

Table 1. Results obtained running one period

CNN	*Default structure*	*Proposed structure*$_1$	*Proposed structure*$_2$
Time (1 Period) in sec.	85.6	122.2	185.5
Error rate (%)	11.13	10.87	10.44

The curves obtained from the above running are presented in Figure 10. After this first test, the second one was made running 20 (twenty) periods at once. Table 2 shows the obtained results.

Table 2. Results obtained with 20 periods of training

CNN	*Default structure*	*Proposed structure*$_1$	*Proposed structure*$_2$
Median individual time (sec.)	84.6	124.3	190.7
Total time (min.)	28.21	41.44	63.58
Error Rate (%)	1.78	1.45	1.88

For the curves of the error rate, the results showed were similar, where the graphic generated between then becomes the same. For a detailed analysis, it is necessary to check direct the results table and not the graphic. For this reason, the Figure 11 only shows the result of one of the results, that is in this case, the model 2.

For the last test, where it was proposed to run 500 (five hundred) periods, it would take too long to run all the possible models, so only the *Proposed*$_1$ was used. The test took 17 (seventeen) hours with each period taken around 122 seconds to run. The results obtained showed an error rate of 0.86%. The graphic obtained for this run is presented in Figure 12.

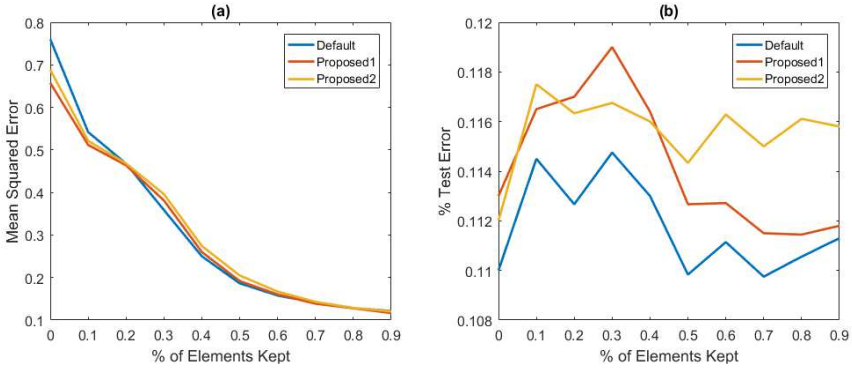

Figure 10. Results of CNN with 1 training epoch: (a) The MSE of training sample; (b) The error rate of test sample

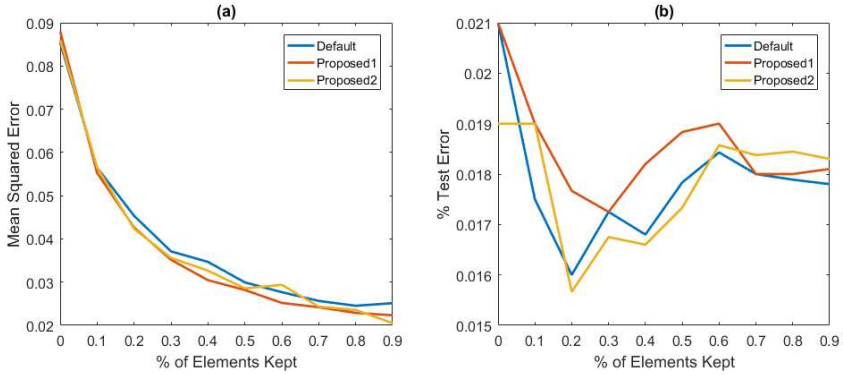

Figure 11. Results of CNN with 20 training epochs: (a) The MSE of training sample; (b) The error rate of test sample

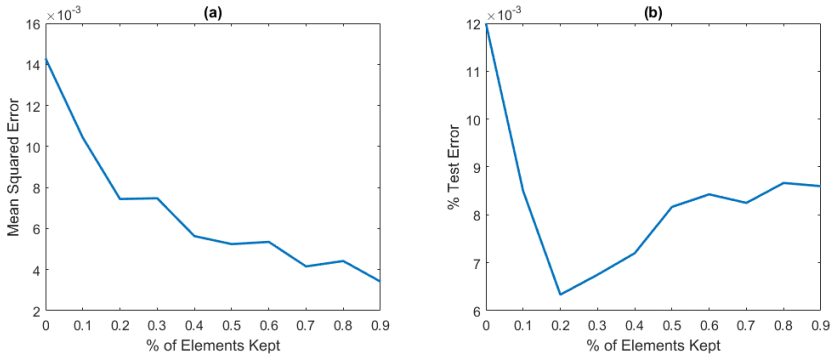

Figure 12. Results of CNN with 500 training epochs for *Proposed-structure*$_1$: (a) The MSE of training sample; (b) The error rate of test sample

6. CONCLUSION

It is possible to notice that the results presented in this paper were in general satisfactory. The usage of DL can be used to obtain the same or even better results than other approaches, including the human interaction. In Table 1, it is possible to notice that after a better configuration, the results increased, once that the error rate was decreasing. However, the time to process the layers increased as we increased them. In addition, it is possible to notice that the results obtained from the third network has being around just 3%, while that the time increased around 50%. Between the original configuration, the time increased was also around 50%, however, the results obtained increased in about 2%. Therefore, it is possible to notice that for the running of just one period, the results obtained were not increasing, even with the system running for more time.

In the Table 2, the processing time also increased between the different structures tested, very similar with the Table 1. However, it is possible to notice that the error after running 20 times has decreased dramatically proving that every new run can increase the results, as expected. In general, the results between the original structure and the first proposal structure have decrease around 18%. However, between the first and second proposed structured increased to 22%, being worse than the original CNN structure.

Finally, in the third and last test, it is possible to notice that the system has decreased the error in around 40%, but with a running period of so many hours that can cause the system to be unusable in practical terms. It is good to point out that the system used is a simple one that was not using GPU (Graphic Processing Unit) to do the calculations. A better system would be necessary to have better understand about the time usage of each of the structures. Finally, it is possible to notice that DL can be easily used for small applications that will not need high results and the processing time is not a main factor. Actually, the results reached in the tests are very similar with the results obtained by the humans. However, it is important to comment that DL can be used for applications that needs better performance in quality response, where it is necessary to just have a powerful processor and better analysis to define how many layers and neurons should be used.

7. ACKKNOWLEDGEMENTS

The authors would like to thank National Council of Scientific and Technologic Development of Brazil - CNPq (grants: 404659/2016-0 and 303908/2015-7-PQ), and “Fundação Araucária” (grant number: 117/2014) for the financial support of this work.

8. REFERENCES

Aizenberg, I., Aizenberg, N.N. and Vandewalle, J.P.L., 2013. *Multi-Valued and Universal Binary Neurons: Theory, Learning and Applications*, Springer Science & Business Media.

Ascenso, J., Valentim, J. and Pereira, F., 1999. "Reconhecimento automático de faces usando informação de textura e de geometria 3D". In 4º Encontro Nacional do Colégio de Engenharia Electrotécnica, Lisboa, Portugal (in Portuguese).

Bengio, Y., Courville, A. and Vincent, P., 2013. "Representation learning: a review and new perspectives". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 35, No. 8, pp. 1798–1828.

Bengio, Y. and LeCun, Y., 2007. "Scaling learning algorithms towards AI. *Large-scale kernel machines*", Vol. 34, No. 5, pp. 1–41.

Caraffi, C., Vojir, T., Trefny, J., Sochman, J. and Matas, J., 2012. "A system for real-time detection and tracking of vehicles from a single car-mounted camera". In *15th International IEEE Conference on Intelligent Transportation Systems*. Anchorage, AL, USA, pp. 975–982.

Dechter, R., 1986. *Learning while Searching in Constraint-Satisfaction Problems*, University of California, Computer Science Department, Cognitive Systems Laboratory, Los Angeles, CA, USA.

Eikvil, L., 1993. *OCR - Optical Character Recognition*, 20 Setp 2017 <http://www.citeseer.ist.psu.edu/142042.html>

Fu, L. and Kara, L. B., 2011. "Neural network-based symbol recognition using a few labeled samples". *Computers & Graphics*, Vol. 35, No. 5, pp.955–966.

Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S. and Lew, M.S., 2016. "Deep learning for visual understanding: a review". *Neurocomputing*, Vol. 187, pp. 27–48.

Hubel, D.H. and Wiesel, T.N., 1962. "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex". *The Journal of Physiology*, Vol. 160, No. 1, pp. 106–154.

Ivakhnenko, A. G. and Lapa, V. G. (1965). *Cybernetic Predicting Devices*. CCM Information Corporation, New York.

Ivakhnenko, A. G., 1971. "Polynomial theory of complex systems". *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 4, No. 1, pp. 364–378.

LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P., 1998. "Gradient-based learning applied to document recognition". *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324.

LeCun, Y., Cortes, C. and Burges, C., 1998. "The MNIST database of handwritten digits". 10 Aug. 2017 <http://yann.lecun.com/exdb/mnist/>

Lee, T.S. and Mumford, D., 2003. "Hierarchical Bayesian inference in the visual cortex". *Journal of the Optical Society of America*, Vol. 20, No. 7, pp. 1434–1448.

Li, H., Lin, Z., Shen, X., Brandt, J. and Hua, G., 2015. "A convolutional neural network cascade for face detection". In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 5325–5334. Boston, USA.

Li, H., Wang. Z., Liu, X.M., Zeng, N., Liu, Y. and Alsaadi, F. E., 2017. "A survey of deep neural network architectures and their applications". *Neurocomputing*, Vol. 234, pp. 11–26.

Najafabadi, M.M., Villanustre, F., Khoshgoftarr, T.M., Seliya, N., Wald, R. and Mhhaaremgic, E., 2015. "Deep learning applications and challenges in big data analytics". *Journal of Big Data*, Vol. 2, p. 1.

Scherer, D., Andreas, M. and Behnke, S., 2010. "Evaluation of pooling operations in convolutional architectures for object recognition". In *Proceedings of the 20th International Conference on Artificial Neural Networks: Part III*, pp. 92–101. Thessaloniki, Greece.

Schmidhuber, J., 2015. "Deep learning in neural networks: an overview". *Neural Networks*, Vol. 61, pp.85–117.

Seethalakshmi, R., Sreeranjani, T. R., Balachanfar, T., Singh, A., Singh, M., Ratan, R. and Kumar, S., 2005. "Optical character recognition for printed tamil text using unicode". *Journal of Zhejiang University-SCIENCE A*, Vol. 6, No. 11, pp. 1297–1305.

TME, 2017. 10 Aug. 2017 <*http://cmp.felk.cvut.cz/data/motorway/*>

Wu, J., 2017. *Introduction to Convolutional Neural Networks*, pp.1–31. 10 Aug. 2017 < https://cs.nju.edu.cn/wujx/paper/CNN.pdf>

Zafeiriou, S., Zhang, C. and Zhang, Z., 2015. "A survey on face detection in the wild: past, present and future". *Computer Vision and Image Understanding*, Vol. 138, pp. 1–24.

Zhao, J., Xie, X., Xu, X. and Sun, S., 2017. "Multi-view learning overview: recent progress and new challenges". *Information Fusion*, Vol. 38, pp. 43–54.

9. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.