International Journal of Systems Signal Control and Engineering Application 10 (1-6): 61-76, 2017
ISSN: 1997-5422

Applications of Deep Neural Networks

Natalie Segura Velandia, Ruben D. Hernandez Beleno and Robinson Jimenez Moreno
Faculty of Engineering, Nueva Granada Military University, Bogota, Colombia

Abstract: This study presents a review of the computational methods implemented with the deep learning technique, highlighting the architecture of convolutional neural networks. Works that use these methods are addressed for image recognition and pattern recognition, each characterizing a specific task. With the technological and investigative momentum that has been generated today and with the development of systems with the capacity to process data with advanced intelligence equivalent to that of the human being, it has awakened an awareness of technological applications in all sectors from business to health where data processing takes place at times less than the human brain could do. In this way, deep learning is an approximation to human perception based on the hierarchical functioning of the neocortex, a fundamental part of the human brain. This technique relies on computational models composed of several layers of processing to learn the representations of data with several levels of abstraction, possessing the ability to modify its internal parameters which are used to calculate the representation in each layer from the previous layer.

Key words: Deep learning, convolutional neural network, recurrent neural networks, autoencoder, Boltzmann machine, deep belief network

INTRODUCTION

With the birth of Artificial Intelligence in 1943, persons like Warren McCulloch and Walter Pitts propose the first mathematical models of the functioning of a neuron where its activation depends on the value of the weights and the inputs, being this the method on which the current neural networks are based (LeCun *et al*., 2015; McCulloch and Pitts, 1943). Another important character in the advance of the artificial intelligence was Gordon Moore who revolutionized technology when he formulated Moore's law stating that the processing power of controllers would increase twice every two years which became a general rule for all processing system developers that compared their progress according to the capacity of their last processor (Espeso, 2017). This competition allowed the artificial intelligence to advance exponentially helped with the evolution of the Graphic Processing Unit (GPU), allowing higher processing capacities with large datasets reducing operating times.

Machine learning, inspired by neuroscience to create models such as perceptron and neural networks based on mathematical models of statistics, probability and optimization is divided into two categories according to their type of learning. The first is supervised learning which is characterized by being a learning process that performs training controlled by a teacher capable of determining the response that the network must generate from the input. The second is unsupervised learning which consists of a set of rules that will give the network the ability to learn the associations between the patterns.

Carreira-Perpinan and Hinton (2005) described the Restricted Boltzman Machine (RBM) algorithm as a stochastic neural network capable of learning a probability distribution associated with the input, implementing a training algorithm based on Contrast Divergence (CD). Following the above, there is the unsupervised learning algorithm known as Deep Belief Net (DBN) that is characterized by an architecture based on several (RBM's) stacked but its training may vary depending on the application that is to be given and if it is managed as an unsupervised network which would be used for the reconstruction of the inputs, unlike when it is managed as a supervised network that is mostly used for classification. This architecture is cataloged as the precursor to the development of deep learning (Hinton *et al*., 2006). In that same year, the deep Autoencoders (SAE) appear which take great importance as they perform the union of RBMs which allows them to obtain nonlinear relationships between the input values with the ability to form a neural network layer at the same time, i.e., the first hidden layer will be formed from the data entry information and the next hidden layer will be formed with the results of the first hidden layer and so on (Hinton and Salakhutdinov, 2006).

These deep learning algorithms that perform tasks such as representation of vision, language, images and signals (McCulloch and Pitts, 1943) are constituted by

Corresponding Author: Natalie Segura Velandia, Faculty of Engineering, Nueva Granada Military University, Bogota, Colombia

multiple levels of non-linear operations, neural network sets or propositional formulas of high processing complexity which is a fairly robust optimization task.

The deep neural networks have been applied successfully in numerical character classification tasks using the MNIST database (Bengio *et al*., 2007; Ranzato *et al*., 2006; Larochelle *et al*., 2007; Boureau and Cun, 2008; Vincent *et al*., 2008), gender classification, ethnicity and age range using the FRGC 2.0 database (Ahmed *et al*., 2008), face recognition with the Calltech-101 database (Lee *et al*., 2009), image recognition (Hubel and Wiesel, 1962), reduction of dimensionality of data from 3D-2D to simplify information analysis and processing (Hinton and Salakhutdinov, 2006), retrieval and prediction of information through the identification of text segments or keywords (Ranzato and Szummer, 2008; Salakhutdinov and Hinton, 2007, 2009) and in robotics for the planning of complex trajectories in unstructured terrains (Hadsell *et al*., 2008).

Deep learning models are one of the best solutions for extracting knowledge from a data set with enough information. The complexity of deep architectures are far more efficient than surface architectures. The deep multi-layer neural networks composed of several non-linear levels of abstraction allow to represent in a compact and diverse way high variability functions (Bengio *et al*., 2007) such as the mapping to the input and output of the data without depending on the characteristics of the human, since at the highest abstraction levels, humans often do not know how to specify the sensory inputs they are experiencing (Bengio *et al*., 2009). Thanks to this, powerful models that are developed under training algorithms as for example the stochastic gradient descent which looks for a number x_0 that initially is chosen in a random way that converges to the minimum or maximum the function f(x), however in a start it was considered impossible to apply in Deep Neural Networks (DNN) and Recurrent Neural Networks (RNN), until by Sutskever *et al*. (2013) they implement the stochastic gradient decrease with momentum using a random initialization designed under a program that slowly increases the momentum. Taking into account that at initialization, momentum values and learning rate are the variables that must be adjusted to obtain the best training results in the application of this algorithm successfully show a training capable of reducing initialization errors by adjusting the momentum values to such an extent that they were sufficient to eliminate the problems of curvature.

The ability to automatically learn powerful non-linear functions are increasingly important as these models are

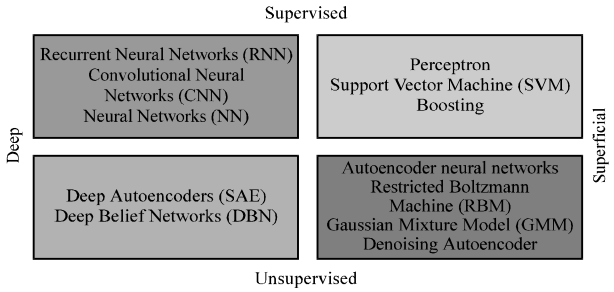

Fig. 1: Classification of neural networks

not only applied in engineering but have reached the financial market where they seek to make use of these models based on deep learning for the prediction of rise and fall in the market (buying and selling shares in the stock exchange) and manipulation of bank data (purchase and sale of securities). These are some of the applications that are being implemented with fast and accurate results.

Deep learning architectures: The term depth refers to the path to be traversed from the input to the output which determines the number of network layers which are composed of calculations of non-linear operations (Bengio *et al*., 2007). The depth of a model is studied not only in the computational field but also in the area of neuroscience, since the cerebral cortex can be seen as a deep architecture involving 5-10 layers during the processing of visual information (Serre *et al*., 2007) which is why it is the inspirational model of researchers to form deep neural networks multilayered (Bengio and LeCun, 2007; Utgoff and Stracuzzi, 2002). Figure 1 shows types of supervised and unsupervised deep learning.

The deep artificial neural network is composed of different layers that allow the system to recognize objects with different levels of abstraction, the lower layers are dedicated to recognizing simple features and the upper layers are dedicated to recognizing more specific characteristics such as the eyes and finally, the top or last level layers do the sum of all data. The data structures work in parallel to achieve a goal.

Architecture of the Convolutional Neural Network (CNN): Convolutional Neural Networks (CNN) or ConvNets were introduced by Fukushima and Miyake (1982) under the name "Neocognitron" and later inspired by the structure of the visual system of the models proposed by Hubel and Wiesel (1962) are structured in a functional architecture of the visual cortex implementing receptive fields of binocular interaction where the first terms of hierarchical connection of complex cells (Neuronal Network) based on the union of simple cells

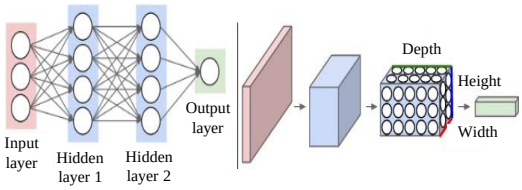

Fig. 2: Red neuronal simple y red neuronal convolucional

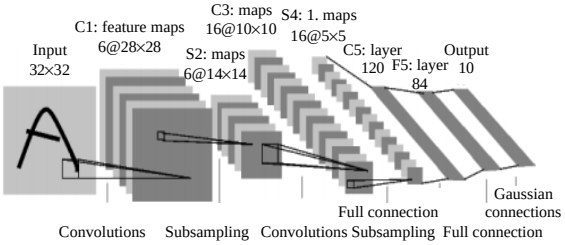

Fig. 3: Architecture of a convolutional neural network composed of three stages (Pusiol, 2014)

(Neuron) were introduced. After several years of studies but without any significant progress in mid-1998, CNNs again took a new path with the backpropagation algorithm that was able to train a model similar to digit recognition in text documents (LeCun *et al*., 1998, 2012) where each layer learns the characteristics of the images.

After that in 2012, Hinton proposed a model from CNNs that would be able to classify images into 1000 categories, revolutionizing the field of computer vision (Krizhevsky *et al*., 2012). Later, the multi-GPU implementation developed in 2014 by Alex Krizhevsky set a new way of implanting parallels between the formations of convolutional neural networks through multi-GPU. Finally, it is found the work done in 2015 by Simonyan and Zisserman, who implemented a very deep neural network of 19 layers for large scale image classification where they present the effect of a deep convolutional network in the recognition of these images, showing successful results in classification accuracy (Krizhevsky, 2014).

Convolutional neural networks are defined as trainable structures composed of several stages, capable of learning different characteristics of abstraction arranged in 3 dimensions: width, height and depth. On the left of Fig. 2 shows a neural network composed of 4 layers (input, two hidden layers and output) and on the right is a convolutional neural network projected in 3D, transforming its input volume into an output volume by means of neuronal activations (Anonymous, 2017). Each hidden layer consists of a set of neurons where each neuron is fully connected to all layers of the previous neuron and thus is how single-layer neurons function independently and do not share any of the connections.

A convolutional network for classification can be constructed from one or several stages where each stage might be composed of three layers: convolution layer, no-linear layer and sub-sampling layer as seen in Fig. 3 (Pusiol, 2014).

Architecture of a convolutional neural network

Input: Composed of an N-dimensional matrix where the image data is stored for example, a matrix for grayscale or RGB scale as seen in Fig. 4.

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Fig. 4: Input matrix representing the contour of a line (Deshpande, 2017)

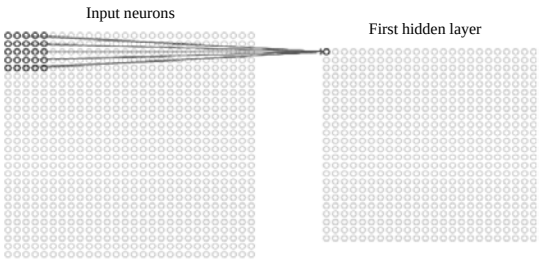

Fig. 5: Convolution layer (Deshpande, 2017)

Convolution: In this layer the input data contained in an array of pixel values is received and one or more filters (kernels) are applied which also use an array of numbers or weights which must have the same depth of the input layer. An example is shown in Fig. 5 where a 32×32×3 input layer to which a 5×5×3 filter is applied and this will slide through the input layer with a stride set, assuming a step of 1, this filter will be shifted to the right 1 unit and so on by extracting the specific characteristics of the input layer, resulting in an array of numbers of 28×28×1 called activation map or feature map.

If two or more filters were applied the result of the characteristics matrix would take other dimensions

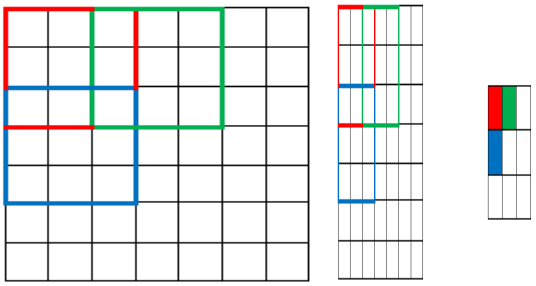

Fig. 6: Stride operation (Deshpande, 2017)

for example, if Fig. 5 were taken and two filters of 5×5×3 were applied, an array of characteristics with dimensions of 28×28×2 where the affected variable is the volume.

This layer is controlled under three parameters

Depth and size: This parameter corresponds to the number of filters and their sizes that are used to perform the convolution on which will depend how many maps of two-dimensional features are formed.

Stride: It corresponds to the step at which the filter is to be moved over the input matrix also known as the sliding step size which will help determine the size of the filter. It should be taken into account that this must be equal to or less than the size of the filter, otherwise important parameters or features will be lost. If this number is large, it generates maps of small characteristics and vice versa. For example in Fig. 6, an input matrix of 7×7 is observed to which a 3×3 filter with a stride of 2 is applied.

Padding: This parameter is very important when it is needed to acquire features from the edges or corners as padding helps to capture them. The 3×3 filter with a stride of 2 in its first position (red color) only passes once for the first 4 pixels of the input image, obtaining values of low level since it will not be possible to obtain more characteristics of these pixels. To solve thereof, this parameter is applied as shown in Fig. 7, making a fill of 0 at the edges of the input matrix capturing any type of features that it has at the corners or edges.

With Eq. 1, it can be determined the size of the fill that should be applied to the input matrix depending on the size and stride of the filter:

$$\text{Zero padding} = \frac{K\text{-}S}{2} \quad (1)$$

Where:

K = Filter size

S = Stride

Activation function: This non-linear function is applied after each convolutional layer in order to introduce nonlinearity to the system.

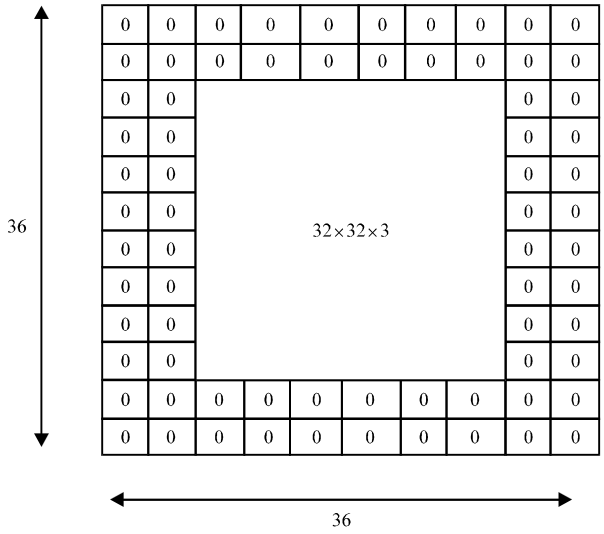

Fig. 7: Padding operation (Deshpande, 2017)

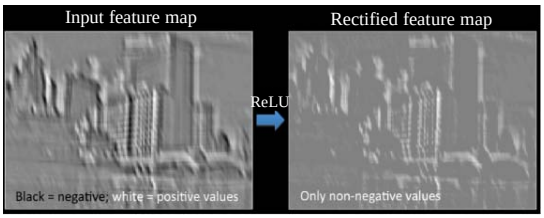

Fig. 8: ReLU operation (Fergus, 2017)

Fig. 9: Sigmoid Function (Boureau and Cun, 2008)

ReLU: This type of function or non-linear operation, known as Rectified Linear Unit and represented in Eq. 2, is applied to each pixel and replaces the negative values in the characteristic map by zero as shown in Fig. 8, increasing the nonlinear properties of the model and of the global network without affecting the negative fields of the convolutional layer:

$$F(x) = \max(0, x) \quad (2)$$

There are other nonlinear functions such as the sigmoid represented by Eq. 3 from which one can conclude that for large values of z, f(x) tends to 1, instead for small values, f(x) tends to 0. An example of how it is applied in an image is illustrated in Fig. 9.

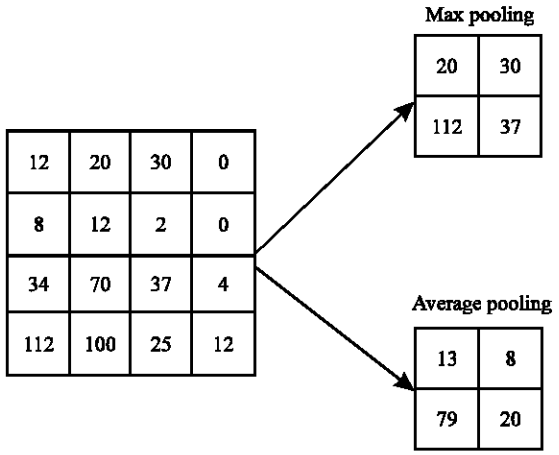

Fig. 10: Max-pooling and average pooling

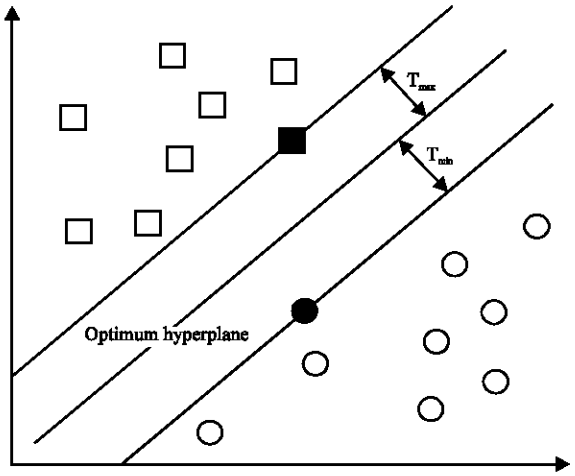

Fig. 11: Maximize margin with optimum hyperplane separation

$$F(x) = \frac{1}{1+e^{-x}} \quad (3)$$

The hyperbolic tangent function shown in Eq. 4 has been implemented time before the ReLU was created:

$$F(x) = Tanh(x)$$

Pooling: In this layer, a sampling operation is carried out along spatial dimensions, reducing the dimension of the convolutional responses to a degree of invariance by grouping the most important information and avoiding overtraining (Nielsen, 2017). Several methods can be used for this process where 2 are exposed.

MaxPooling: It is a sample-based discretization process which reduces the calculations by eliminating smaller values and retaining only the maximum value of each sub-division (Hijazi *et al*., 2015). This data set acts as an input value for the subsequent layers, maintaining the maximum characteristics within a sample size as shown in Fig. 10.

Average: This type of clustering works in such a way that it lowers the resolution by averaging the group of existing pixels (Fig. 10), generating a map of characteristics by each category of the classification task of the last convolution layer, instead of adding connected layers. One of its main advantages is that it prevents the network from learning image structures like borders and textures.

Full connection: The function of this layer is to analyze which high-level features are obtained from the last convolution layer and correlate strongly with the class. It is characterized by having a total connection with all the activations of the previous layer, capturing activation maps with high level characteristics, determining the characteristics that are most related to the class of study. In this layer as in the previous ones, an activation function must be applied between the last convolution layer and the total connection layer. These functions include.

Softmax: In charge of taking a vector of partial scores of real values taking it to a vector of values between zero and one.

Support Vector Machine (SVM): It is a set of supervised learning methods for classification derived from statistical learning where the mapping of the entry points is performed, transforming them into a space of characteristics of a larger dimension ($R^2 \rightarrow R^3$), generating a separation hyperplane margin in a two-dimensional space denoted as the minimum distance between the hyperplane and the closest sample of classes. When maximizing the hyperplane margin is considered to be optimal as shown in Fig. 11.

When these data cannot be separated linearly, a space change must be made by implementing a data transformation function to separate them linearly, finding functions such as Kernel, Polynomial Functions and Radial Base Functions (RBF) as seen in Fig. 12.

Maxout: Responsible for the maximum response over a mixture of linear models, allowing the convolutional network to easily model multiple data nodes.

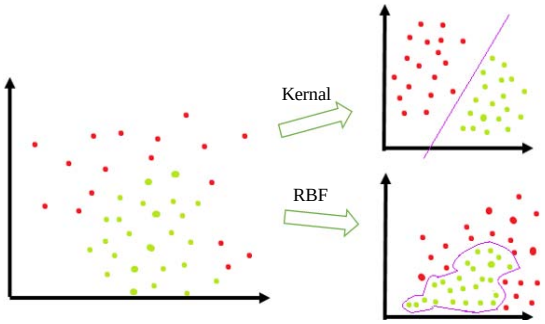

Fig. 12: Classification from the implementation of linear kernel and Radial Base Functions (RBF)

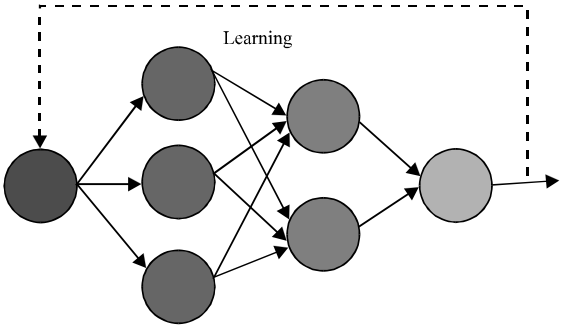

Fig. 13: Recurrent network architecture

Training: In order to carry out an adequate training it must be taken into account that during the first step, where the data entry matrix is taken and passed through the network which is constituted by weights and random filter values, a low level output matrix will be obtained which will be evaluated by a loss function, represented by Eq. 5 with which it will be observed the loss at this point, if it is very high, it is determined that there is an optimization problem where the weights of the filters must be recalculated and in this way to minimize the error:

$$P_{total} = \sum \frac{1}{2}(\text{Objective} - \text{Out})^2 \quad (5)$$

What mathematically can be expressed as in Eq. 6, where what will be done is to extract the weights that contributed to the loss of the information and find a new value for this to decrease, performing an updating of weights in the filters.

The learning rate is implemented to perform the updating of the weights but it must be taken into account that if this value is high, this process will take less time to converge to an optimal set of weights however, it should not be exceeded too high because it has problems reaching the optimum:

$$w = w_i - \eta \frac{dP}{dW} \quad (6)$$

Where:
W = Wheight
w_i = Initial weight
η = Learning rate

This process of updating weights is called epoch which can be repeated several times to a set of images called batch.

Architecture of the Recurrent Neural Network (RNN): This type of network is characterized by feedback which means that the output of one neuron can be used by another that is in an anterior layer as shown in Fig. 13. This type of architecture allows to simulate dynamic systems where the training is delayed since it does not have a single path forward (Atiya and Parlos, 2000). This architecture presents some limitations as to the difficulty that they have to stabilize, since it is governed under some architectures like the one of Hopfield which consists in reducing the energy of the states that the network must remember, converting it into an addressable memory system, i.e., it will be able to remember if it is given a part of the state. This turns out to be quite useful as it allows to rebuild a distorted input using the network status obtained during training. To achieve this type of recurrences are used.

Delay blocks that work with discrete times described by Eq. 7 where the output at time t is calculated as the input at an earlier time. It should be noted that when this class of blocks is performed for the first time, the condition $\alpha(0)$ is used as an initial condition. These types of blocks are implemented in the Haming and Hopfield networks:

$$\alpha(t) = u(t-1) \quad (7)$$

Where:
α = The output a at time t
T = The input vector at an earlier time

Integrator block works with continuous times as described in Eq. 8 where the output at time t is calculated as the integral from 0-t (moment in which it is) of the input u at a time Tao plus the initial conditional. The Grossberg network uses this type of block:

$$\alpha(t) = \int_0^t u(\tau)d\tau + \alpha(0) \quad (8)$$

Where:
α = The output a at time t
u = The input at a time Tao

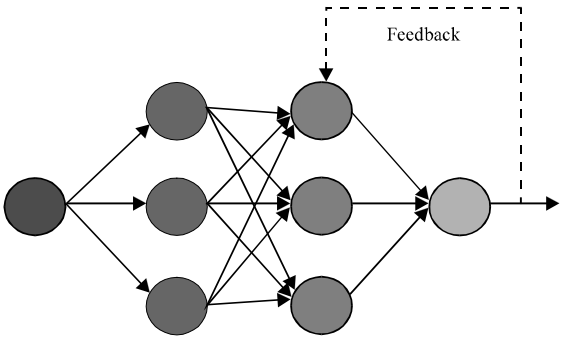

Fig. 14: Learning architecture by epochs

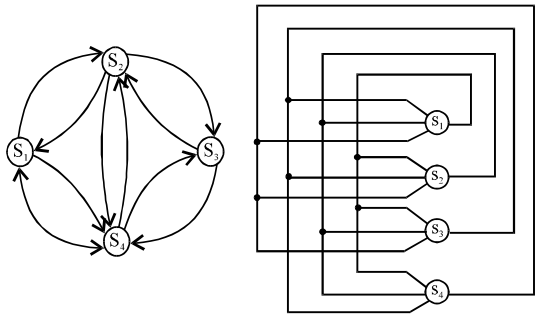

Fig. 15: Red hopfield with 4 neurons

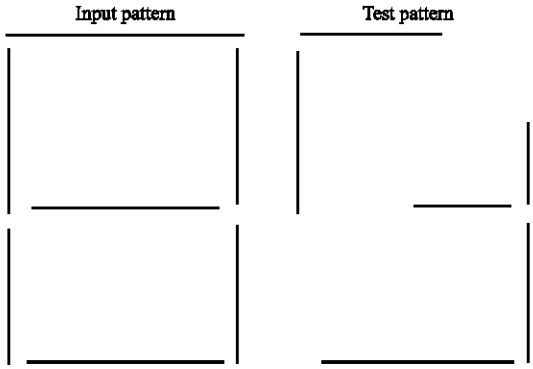

Fig. 16: Recovery phase example

Training: Recurrence techniques are used to correct network errors by automatically modifying the values used (weights and biases).

Training is perform by epochs, where the network evolves over a period of time (epoch) in which input parameters have been introduced that will give an adequate response. This means that when it reaches the end of the epoch, the network is restarted so that the initial state does not depend only on the final state of the previous epoch but feedback the information obtained by creating a new epoch to which the weight update will be carried out but it must be taken into account that this only happens at the end of each epoch as shown in Fig. 14.

For their training, a learning algorithm based on the gradient descent technique is used which can be grouped as follows (Atiya and Parlos, 2000):

- Back Propagation Through Time (BPTT)
- Forward propagation or Real Timer Recurrent Learning (RTRL)
- Fast forward propagation

The criterion of completion of training is only given as long as all the data were correctly classified, if it is exceeded with the number of iterations allowed in the training or if it reached the minimum error allowed because if training continues the network will have problems of overfitting (Bengio *et al*., 2013).

Hopfield network: It is a mono-layer neural network with unsupervised learning, where all its networks are interconnected as shown in Fig. 15.

In the research that is shown by Hopfield (1982), it defines that the Hopfield network is composed of symmetric connections that are easy to train because there is an energy function defined as the scalar value associated with the state of the network, described in Eq. 9. It aims to find the points that stabilize the neuron being the minimum values of the energy function.

$$E = -\frac{1}{2}\sum_{i,\,j} w_{ij} S_i S_j + \sum_i \theta_i S_i \qquad (9)$$

Where:
θ = Threshold or bias
w_i = Synaptic weights
S_i = Unit status

The use of associative memories corresponds to the energy minima in a network which allow to have an addressable memory that allows the recovery of the content as will be explained in the recovery phase.

Operation phases

Storage phase: In this phase the neuron is trained where patterns are introduced which will have weights assigned and then store a standard set.

Recovery phase: It is a mechanism to recover the information that was stored from incomplete information, i.e., in this phase the network is able to reconstruct the test pattern from a previously stored input pattern as shown in Fig. 16.

Applications: Elman (1991) an application for language processing is presented that is implemented by means of

a trained Recurrent Neural Network used for lexical retrieval. This network is trained in 4 different steps: the first set of training consists exclusively in learning simple sentences, succeeding in forming 10,000 words. In the second part the network was exposed to a second training with a sample of 10,000 sentences where 25% were complex sentences and 75% were simple sentences. From this, the training result was a sentence of minimum length of 3 words and maximum of 13 words. In the third phase, the number of complex sentences increased to 50%, obtaining more complex sentences with a length of 3 words and a maximum of 13 words. The fourth phase consisted of taking a sample of 10,000 sentences where 75% were complex sentences and 25% were simple sentences, obtaining a maximum of 16 words in a sentence from this learning strategy, it was observed the behavior of the network, where they had an error initially of 12.45%, reducing it to 0.177%.

Also by Fukushima and Miyake (1982) they use a system for the nonlinear prediction of signals applied to speech coding, proposing a Pipeline Recurrent Neural Network (PRNN) composed of separate but identical modules, each designed as a recurrent network with a single output neuron, seeking to perform a linear correlation from the input space to an intermediate space in order to linearize the input signal. The second part is done with a Tapped Delay Line filter (TDL) which makes the linear assignment from the new space to an output space. It was implemented a combination of pipeline and a robust linear recurrent neural network with an adaptive filter.

Architecture of deep autoencoders This type of unsupervised architecture is able to make a safe approach to network input parameters which seek to reach an optimal value used to learn compressed and distributed representations of data. This means that this type of architecture will not learn as a training data map but will learn from an internal structure with data characteristics because the number of hidden layers are much smaller than the input/output layers, forcing the network to learn the most important features (Arteaga, 2015).

The first development was the Fukushima neocognitron which is observed in Fig. 17 used for the recognition of manuscript characters and pattern recognition where the author proposes an architecture inspired by the Hubel and Wiesel model (Hubel and Wiesel, 1962) that is constituted by neurons in charge of extracting local characteristics and a set of neurons (neuronal network) where these characteristics are gradually integrated and classified in higher layers. Thanks to this architecture the performance of the recognition pattern was improved and it proposed an

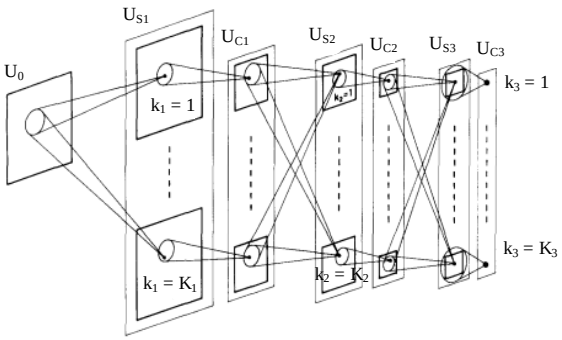

Fig. 17: Neocognitron Fukushima, interconnections between layers (LeCun *et al*., 2012)

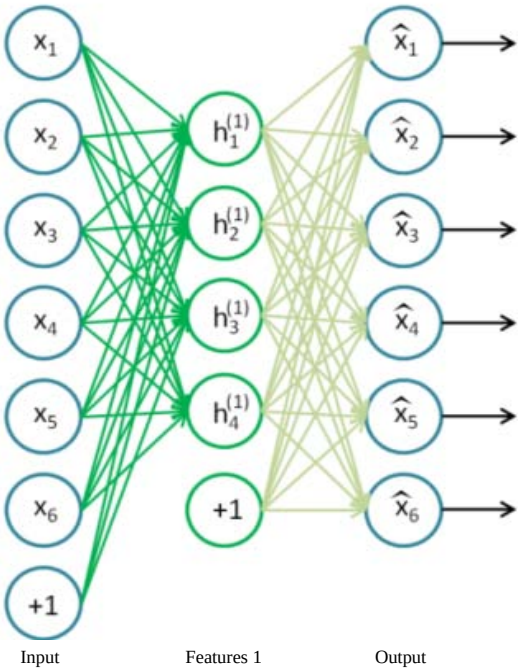

Fig. 18: Architecture of an Autoencoder

algorithm able to be applied in the processing of auditory information for speech recognition (Fukushima and Miyake, 1982).

Architecture: It consists of 3 layers (input, hidden and output) as can be seen in Fig. 18 where there is an input X which passes through a hidden layer that seeks to perform a replica of the input data on a space (output layer) of different dimension.

Compressive autoencoder: This type of encoder is characterized in that the size of its hidden layer is smaller than the size of the input layer, performing a

compression of the input information, especially used in application of dimensional data reduction (Theis *et al*., 2017).

Autoencoder of identity: This type of encoder has the same size in its three layers, demonstrating that the network tends to learn the identity matrix which is not something new, since what it performs is an exact replica of the input.

Denoising autoencoder: This type of encoder performs a mapping of the input data which pass through a stack of autoencoders which generate an output with a larger dimension space (Vincent *et al*., 2008).

Its functionality or process of autoencoder operation in general is in the hidden layer which describes a code to represent the output, is composed of two parts: an encoder described by Eq. 10 and a decoder described by Eq. 11:

$$h = f(x) \tag{10}$$

$$r = g(h) \tag{11}$$

Stacked Auto Encoders (SAE): A simple autoencoder only extracts simple features such as lines, edges and curves. From this, it was generated the need to extract more complex features such as face recognition where the technique of stacking auto-encoders was implemented, where the output of the first auto-encoder is the input of the second and so on, managing to extract much more complex features. After having a sufficiently robust network, will have a hierarchy of characteristics of greater complexity without the need for a supervisor to explain to the network what a face is like.

Stacked Denoising Autoencoders (SDAE): This type of architecture consists of stacking autoencoders resizing the output to a space of greater dimension. To do this, the following steps: first, the error is calculated which correlates with a dispersion index that restricts the number of combinations in the hidden layer. Second, noise is introduced into the input vector and leaves the output without any disturbance, forcing the hidden layer to focus on the common characteristics of all versions of the input data to which different noises were applied.

Architecture of the Restricted Boltzmann Machine (RBM): Scientists and researchers worked on much simpler methods to perform trainings using the model of a recurrent network by opting for simpler learning algorithms. Ackley *et al*. (1985) gave the name of

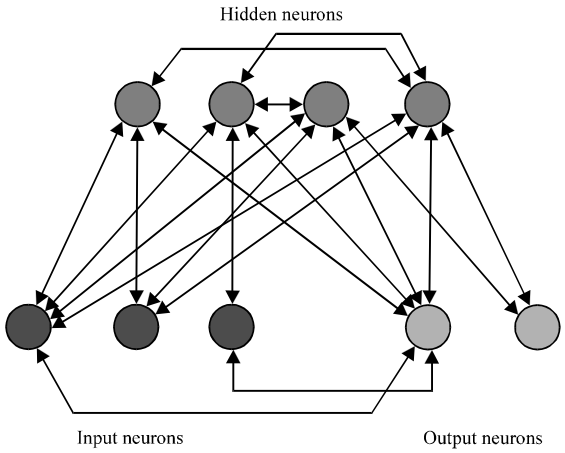

Fig. 19: Restricted Boltzmann machine

Boltzmann Machine where it defines that everything is in terms of energy, given by Eq. 12 where the first term relates to the summation of the binary states of the visible layer with bias, the second term refers to the summation of binary states of the hidden layer with its bias, the third term refers to the values of each non-identical pair of i and j, the fourth term relates the weights between the visible layer i, the hidden layer k and finally the term that relates the interactions between the hidden layers. From this equation then Eq. 13 is performed which relates the probability of the visible and hidden layers as a function of energy:

$$-E(v, h) = \sum_{i \in vis} v_i b_i + \sum_{k \in hid} h_k b_k + \sum_{i<j} v_i v_j w_{ij} + \sum_{i,k} v_i h_k w_{ik} \sum_{k<l} h_k h_i w_{kl} \tag{12}$$

Where:
V = Visible units
h = Hidden units
S_i = Unit status

$$p(v, h) = \frac{e^{-E(v, h)}}{\sum_{u, q} e^{-E(u, g)}} \tag{13}$$

Where:
v = Visible units
h = Hidden units

Its architecture can be observed in Fig. 19 where the input neurons serve as interface between the network and the environment in which it operates, during the training phase, the states of the input neurons are adjusted to specific states set by the training patterns where

the hidden neurons operate bidirectionally using the underlying constraints contained in the input patterns (Heisler, 2017).

The Contrast Divergence (CD) is a gradient ascent algorithm implemented in training of unsupervised networks, more specifically in RBM (Carreira-Perpinan and Hinton, 2005). This algorithm consists in propagating the visible input layer (V_0) in the RBM where a bidirectional sending is developed to obtain the data of the hidden layer (H_0) to which the reverse operation will be applied converting it in a layer with the same characteristics of V_0 p but with different values, denominated V_1. Then, the updating of the weights described in Eq. 14 is done:

$$w(t+1) = w(t) + \alpha(V_0H_0 - V_1H_1) \quad (14)$$

Where:
V = Visible layer
H = Hidden layer
α = Learning rate
w = Weights

Training: The synaptic weights of the network are initialized with random values between -1 and 1. Visible units are adjusted to the training patterns. The network is updated for each value, randomly choosing a hidden neuron assigning the value described by Eq. 15:

$$W(s_i \rightarrow -s_i) = \frac{1}{1+e^{\frac{-\Delta E}{T}}} \quad (15)$$

Where:
s_i = State of the neuron
ΔE = Changing energy when moving from one state to another
T = Temperature

The adjustments are made again but only the input neurons estimate the correlations. To perform the update of synaptic weights according to Boltzmann's learning rule described in Eq. 16:

$$\Delta w_{i,j} = \frac{\eta}{T}\left(\langle v_I h_J\rangle_{blocked} - \langle v_I h_J\rangle_{free}\right) \quad (16)$$

Where:
η = Learning rate
T = System temperature
v, h = Visible and Hidden layer

Finally, the previous steps are repeated in order that the learning converges and does not produce changes in the value of the synaptic weights.

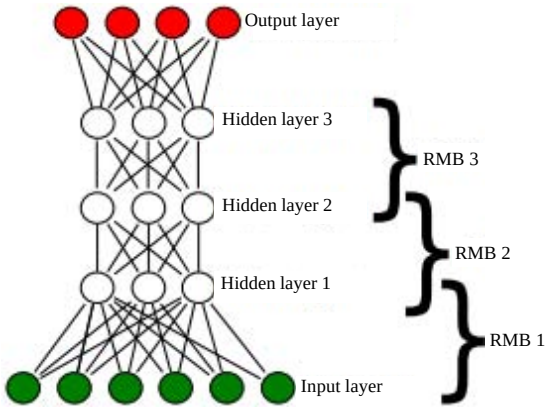

Fig. 20: Representation of a DBN (Hamel and Eck, 2010)

Deep belief network: This type of unsupervised network is created by stacking RBMs with connections between layers that learn probabilistically by reconstructing their inputs which means that the layers act as feature detectors because of their unsupervised training. These types of networks have the ability to have in their deeper layers a supervised layer to better classify their characteristics (Bengio, 2009; Arteaga, 2015).

Its limitation is derived from the complexity of the distributions only using a hidden layer detecting features which makes the model very rigid for higher order nonlinearities. Figure 20 illustrates a representation of a DBN network using a greater number of hidden layers where it is thought to consider this model as a RBM stack in which the first machine is trained with the data of the input to be modeled, taking the subsequent ones as input data for the activation function of the hidden layer of the previous machine (Hinton *et al*., 2006).

This algorithm is based on a probabilistic approach proposed by Geoffrey Hinton in charge of estimating the energy function gradient depending on the set of parameters that the model has and improving the calculation speed. Its main objective, being the fundamental basis of the DBN is to find the weights and bias that maximize the probability of reduction of training data.

Training: It is based on training the network by layers by grouping them in pairs, giving rise to a RBM and using the CD algorithm, described as follows: the RBM formed from the input layer and the hidden Layer 1 is trained. Once the first RBM has been trained, the data must be entered again to calculate the probability distribution generated by the hidden layer. The distribution was taken as the input data for the training of RBM formed by the hidden Layers 2 and 3. The process will be repeated until

all layers are trained. Finally, a non-optimal trained deep network is obtained that must be completed using the up-down algorithm which will perform the weight update.

State of the art: Krizhevsky *et al.* (2012) ImageNet classification with deep convolutional neural networks. In Proc. Advances in Neural Information Processing Systems 25 1090-1098 (2012). Krizhevsky *et al.* (2012) it is reported a deep convolutional neural network composed of 60 million parameters and 650 thousand neurons. This network consists of five convolutional layers followed by maxpooling layers and finally three fully-connected layers with an end consisting of the softmax technique. In order to avoid overfitting, it is used the dropout method with the purpose of classify 1.2 million high resolution images.

Agrawal analyzing the performance of multilayer neural networks for object recognition. Agrawal *et al.* (2014) aims to help researchers better understand convolutional neural networks where they performed tests on three sets of classification and detection data, demonstrating that the network can be trained with a small dataset but that the pre-training stage cannot be omitted as it improves performance. Using the guided gradient histograms (HOG) function allow an intuitive interpretation of border filter histograms in order to develop computer vision methods, working aspects of CNN's empirically through a lens.

Grishick , Rich feature hierarchies for accurate object detection and semantic segmentation. Girshick *et al.* (2014) it is proposed a simple and scalable detection algorithm that improves accuracy by >30% in relation to the previous one, obtaining a mean average precision of 53.3%. They called the R-CNN method which combines the high capacity of CNNs with region proposals in order to locate and segment the objects as far as the tagged training data are limited, the supervised pre-training acts by an auxiliary task followed by the specific task of tuning domain that produces an increase in performance.

Lerrel and Abhinav, Supersizing Self-supervision: Learning to Grasp from 50 K Tries and 700 robot hours (2015). Pinto and Gupta (2016) is shown how a robot collects a database for 700 h reaching a size of 50 K data points which allows it to form a Convolutional Neural Network (CNN) for the task of predicting the grip of the locations without overfitting. In the formulation they propose to perform a regression to a form of 18 binary classification paths on a patch image. They present a multi-stage learning approach where CNN is used to collect the negatives in the later stages, showing effectiveness in this method for the task of understanding data.

Perez-Carrasco *et al.* (2011) Red Neuronal Convolucional Rapida Sin Fotogramas Para Reconocimiento de Digitos (2011). They performed the implementation of six Convolutional Neural layers (ConvNet) where the system was trained with the backpropagation algorithm using 32×32 images stored in the database, encoding 10000 images in separate 50 ns events to test ConvNet. Results were obtained with real performance figures and a recognition rate of 93%.

Bermejo *et al.* "deteccion y clasificacion de enfisema pulmonar en imagenes de TAC mediante Redes Neuronales Convolucionales Multiescala". Pelaez *et al.* (2016) there is proposed a tool capable of recognizing pulmonary emphysema patterns, based on a neural convolutional multi-scale network designed for the detection and classification of 6 classes of lung tissue including 5 patterns of emphysema and normal tissue. This network consists of 4 convolutional and 3 pooling layers and the entrance corresponds to a multiscale representation of the pulmonary image to be classified.

Suarez *et al.* "clasificacion automatica de coberturas del suelo en imagenes satelitales utilizando redes neuronales convolucionales: un caso aplicado en parque nacionales naturales de Colombia". Suarez *et al.* is presented an automatic learning method based on convolutional neural networks trained from manual annotations by means of visual interpretation on the teleoperation images with which the experts generated a cover map for the automatic classification of ground coverings from Lanssat5 images.

Camacho and Labrandero (2016) neuronales convolucionales aplicadas a la traduccion del lenguaje verbal espanol al lenguaje de senas boliviano in 2016. Perez-Carrasco *et al.* (2011), it is presented the implementation of neural convolutional networks to interpret sounds emitted by people to be translated into Bolivian sign language. It was supported under the Fourier transform and the Mel scales to create training patterns with different sizes considering the loose words and phrases that are used in Spanish. The architecture of the convolutional neural network was LeNet-5 with a 32×32 input matrix with 3 convolutional layers of 16, 32 and 64 output maps, using 5×5 filters each, 2 layers of pooling with a factor of 1:2 between each of the convolutions to finally obtain a layer of fully connected neurons. The training technique that was implemented is the Asynchronous Stochastic Gradient Descent technique with momentum parameters of 0.02 and fixed learning factor of 0.01 reducing it by 2% every 10 epochs.

Roth *et al.* (2015) DeepOrgan: Multi-level Deep Convolutional Networks for Automated Pancreas

Segmentation in 2015. Roth *et al.* (2015) this study raises the problem that is in medicine which is to perform the analysis of medical images taking the pancreas as one of the organs with high anatomical variability, developing an investigative work based on probabilistic analysis with bottom-up approach using convolutional neural networks, they achieved the highest reported results of 71.8% accuracy in trials and 83.6% accuracy in training taking only a few minutes of computational cost.

RESULTS AND DISCUSSION

The development of the different methods of deep learning has given the opportunity to the people who do not associate directly with the model, to know and to interact with the different applications that have been developed under the architecture of deep learning which has been a success as in applications of speech recognition and image classifier. Here are presented some of them and important works applicable in this field.

Applications focused on the processing of sound signals
Speech recognition: Siri, the iOS virtual assistant is an application used by Apple devices, who in their laboratories and accompanied by their experts developed a physical structure which consists of a number of strategically placed microphones, adjusting their hardware so that, when performing audio processing is an incredible advantage compared to companies that are only focused on building the software in turn developed the firmware for the devices maximizing the performance of the deep convolutional neural network, stored in a brain of about 200 Megabytes (Levy, 2017).

Applications focused on text recognition
Handwriting recognition: First, there is the "Handwriting Recognition" which was a fundamental part to promote the use of convolutional networks where LeCun *et al.* (1998) obtained a 99.3% effectiveness in the use of LeNet-5 in 1988. After research by Hinton in 2012, the first network of convolutional neural network reached an accuracy of 99.77% (Ciregan *et al.*, 2012).

Similarly, there is QuickType application that was developed by Apple based on a convolutional neural network, whose function is to monitor the system while the user writes, speeding up the writing process by providing word suggestions according to the context in which the user operates. It should be emphasized that this application is not only responsible for predicting but is able to analyze texts and will give suggestions to give quick answers (Heisler, 2017).

Applications focused on image recognition: Face Recognition: "Facebook Face Recognition". In the research article "DeepFace: closing the gap to human-level performance in face verification" which is basis of the development of the application, it is presented from the alignment to the representation of the 3D model of the face in order to apply various pieces of image and derive the representation face to a deep neural network covering about 120 million parameters using several connected layers without shared weights and 4 million tagged images belonging to 4 million different identities. This is one of the most efficient applications, reaching a precision of 97.5% in the face tagging (Taigman *et al.*, 2014).

FaceNet: In the research article "FaceNet; A Unified Embedding for Face Recognition and Clustering", shows the implementation of neural networks for the recognition of faces with a learning rate of 0.05 initializing their models of random, reducing the error dramatically after 500 h of training. Results of a set of data the system achieves a new precision 99.63%. In YouTube Faces DB get the 95.12%. Reducing the error rate in comparison with the best (Schroff *et al.*, 2015).

Image search ("Traffic Signs"): In the research article "Traffic sign recognition with multi-scale Convolutional Networks" (Sermanet and LeCun, 2011), it is presented the application of convolutional neural networks for the task of classification of traffic signals where its traditional architecture was modified by the feeding of the first stage which includes the functions of the second stage for the classifier under this new architecture in which an accuracy of 98.97% was obtained, above human performance that is 98.81%. 32×32 input images were used, after which new experiments were performed where they obtained a recognition record of 99.17%. This application was tested in the German Traffic Sign Recognition Benchmark (GTSRB) competition.

"Pedestrian detection" The study "Histograms of Oriented Gradients for Human Detection" (Dalal and Triggs, 2005) which uses the INRIA Person Dataset, discusses the sets of functions for the recognition of solid visual objects, experimentally showing the grids of oriented gradient histograms which describe the set of existing functions for the pedestrian detection, additionally a study is made on each stage of the calculation of performance. The application contains images from different sources which are digital collections of people captured for a long time through Google Images with approximately 2592×1944 Pixels resolution is only able to recognize vertical people with a height greater than 100 cm.

Applications focused on the video area

Motion detection: These types of applications are seen in the UX/UI and videogames industry. Video analysis through convolutional neural networks where the challenge is to capture the complementary information about the appearance of the frames and the movement between them. For the development of this application it was proposed a ConvNet architecture that incorporates spatial and temporal networks, demonstrating that the convolutional network is able to handle a dense optical flow obtaining excellent results (Simonyan and Zisserman, 2014a, b).

Applications focused on video games

Motion prediction: Teaching to master the game Go has been a challenge which has been solved through convolutional neural networks trained to play and go through the training to predict movements made by expert players. For the development of this application used in total, three convolutional networks are trained of two different kinds: two policy networks provide guidance regarding which action to choose given the current state of the game, one of that was trained on 30 million positions from games played by human experts with an accuracy on a withheld test-set of 57% and the other one was trained well with an accuracy much lower 24.2% but much faster. And one value network. Both types of networks take as input the current game state, represented as an image and the output is a single number, representing the probability of a win (Silver *et al*., 2016).

CONCLUSION

Deep Learning has been designed under the hierarchical architecture, achieving an approach to the process developed by the human brain, implementing unsupervised learning to extract characteristics and supervised to classify these characteristics in such a way that the deep learning is able to form hybrid networks obtaining better properties for the operation of the network as observed in a DBN which is based on an unsupervised learning but a supervised phase is introduced to classify the characteristics.

The convolutional neural networks in which this research was focused and as it was observed in the works realized under this learning, the recognition of patterns, voice, images, videos and text are their specialty. At present there is no learning that develops this type of recognition more optimally and accurately thanks to its architecture and development.

The advantages of today's deep learning depend on the GPU which is responsible for supporting parallel workloads and accelerating the network process 20 times faster than normal, reducing its iterations from months to just days.

The deep learning in company of the most prestigious companies in the technology and networks have been able to reveal the versatility and importance that this has, making large investments in research in order to develop algorithms much more efficient and accurate than they have at the moment as was observed in the works carried out by Facebook, Apple and Google, where they developed algorithms that have impressed the world giving a perspective of what is being developed today.

ACKNOWLEDGEMENT

Researchers 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

Ackley, D.H., G.E. Hinton and T.J. Sejnowski, 1985. A learning algorithm for Boltzmann machines. Cognit. Sci., 9: 147-169.

Agrawal, P., R. Girshick and J. Malik, 2014. Analyzing the Performance of Multilayer Neural Networks for Object Recognition. In: Computer Vision, Fleet, D., T. Pajdla, B. Schiele and T. Tuytelaars (Eds.). Springer, Berlin, Germany, ISBN:978-3-319-10583-3, pp: 329-344.

Ahmed, A., K. Yu, W. Xu, Y. Gong and E. Xing, 2008. Training Hierarchical Feed-Forward Visual Recognition Models Using Transfer Learning from Pseudo-Tasks. In: Computer Vision, Forsyth, D., P. Torr and A. Zisserman (Eds.). Springer, Berlin, Germany, ISBN:978-3-540-88689-1, pp: 69-82.

Anonymous, 2017. CS231n: Convolutional neural networks for visual recognition. NVIDIA Auditorium, California.

Arteaga, G.J.P.R., 2015. Application of deep learning to the processing of digital signals. Bachelor's Thesis, Universidad Autonoma de Occidente, Cali, Colombia.

Atiya, A.F. and A.G. Parlos, 2000. New results on recurrent network training: Unifying the algorithms and accelerating convergence. IEEE Trans. Neural Networks, 11: 697-709.

Bengio, Y. and Y. LeCun, 2007. Scaling learning algorithms towards AI. Large Scale Kernel Mach., 34: 1-41.

Bengio, Y., 2009. Learning deep architectures for AI. Found. Trends Mach. Learn., 2: 1-127.

Bengio, Y., P. Lamblin, D. Popovici and H. Larochelle, 2007. Greedy Layer-Wise Training of Deep Networks. In: Advances in Neural Information Processing Systems, Scholkopf, B., J. Platt and T. Hofmann (Eds.). MIT Press, Cambridge, Massachusetts, ISBN-13:978-0-262-19568-3, pp: 153-160.

Bengio, Y., Y. Bengio, N. Boulanger-Lewandowski and R. Pascanu, 2013. Advances in optimizing recurrent networks. Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), May 26-31, 2013, IEEE, Vancouver, British Columbia, Canada, ISBN:978-1-4799-0356-6, pp: 8624-8628.

Boureau, Y.L. and Y.L. Cun, 2008. Sparse feature learning for deep belief networks. Proceedings of the 22nd Annual Conference on Neural Information Processing Systems (NIPS 2008), December 11, 2008, Hyatt Regency Vancouver, Canada, pp: 1185-1192.

Camacho, F. and J. Labrandero, 2016. Convolutional neural networks applied to the translation of Spanish verbal language into sign language Bolivian: Convolutionary neuronal networks applied to the translation of the verbal Spanish language to the Bolivian sign language. Sci. Technol. Innovation Mag., 12: 755-755.

Carreira-Perpinan, M.A. and G.E. Hinton, 2005. On contrastive divergence learning. Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics Vol. 10, January 6-8, 2005, Savannah Beach Hotel Barbados, pp: 33-40.

Ciregan, D., U. Meier and J. Schmidhuber, 2012. Multi-column deep neural networks for image classification. Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 16-21, 2012, IEEE, Providence, Rhode Island, ISBN:978-1-4673-1226-4, pp: 3642-3649.

Dalal, N. and B. Triggs, 2005. Histograms of oriented gradients for human detection. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognition, 1: 886-893.

Deshpande, A., 2017. A beginner's guide to understanding convolutional neural networks. Master Thesis, University of California, Los Angeles, California.

Elman, J.L., 1991. Distributed representations, simple recurrent networks and grammatical structure. Mach. Learn., 7: 195-225.

Espeso, P., 2017. 50 Years of Moore's law, perhaps the most misunderstood law of technology. Weblogs SL, Madrid, Spain. https://www.xataka.com/componentes/50-anos-de-la-ley-de-moore-la-quizas-ley-mas-incomprendida-de-la-tecnologia#sections.

Fergus, R., 2017. Neural networks. MS Thesis, Courant Institute of Mathematical Sciences, New York, USA.

Fukushima, K. and S. Miyake, 1982. Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Visual Pattern Recognition. In: Competition and Cooperation in Neural Nets, Amari, S. and M.A. Arbib (Eds.). Springer, Berlin, Germany, ISBN:978-3-540-11574-8, pp: 267-285.

Girshick, R., J. Donahue, T. Darrell and J. Malik, 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, June 23-28, 2014, ACM, Washington, DC, USA., ISBN:978-1-4799-5118-5, pp: 580-587.

Hadsell, R., A. Erkan, P. Sermanet, M. Scoffier and U. Muller *et al*., 2008. Deep belief net learning in a long-range vision system for autonomous off-road driving. Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2008), September 22-16, 2008, IEEE, Nice, France, ISBN:978-1-4244-2057-5, pp: 628-633.

Hamel, P. and D. Eck, 2010. Learning features from music audio with deep belief networks. Proceedings of the 11th International Conference on Society for Music Information Retrieval (ISMIR 2010), August 9-13, 2010, Utrecht University, Utrecht, Netherlands, pp: 339-344.

Heisler, Y., 2017. A look at IOS 8's new quicktype feature. Oath Inc., New York, USA. https://www.engadget.com/2014/09/17/a-look-at-ios-8s-new-quicktype-feature/.

Hijazi, S., R. Kumar and C. Rowen, 2015. Using convolutional neural networks for image recognition. Cadence Design Systems, Pennsylvania.

Hinton, G.E. and R.R. Salakhutdinov, 2006. Reducing the dimensionality of data with neural networks. Science, 313: 504-507.

Hinton, G.E., S. Osindero and Y.W. Teh, 2006. A fast learning algorithm for deep belief nets. Neural Comput., 18: 1527-1554.

Hopfield, J.J., 1982. Neural networks and physical systems with emergent collective computational abilities. Proc. Natl. Acad. Sci. USA., 79: 2554-2558.

Hubel, D.H. and T.N. Wiesel, 1962. Receptive fields, Binocular interaction and functional architecture in car's visual cortex. J. Physiol., 160: 106-155.

Krizhevsky, A., 2014. One weird trick for parallelizing convolutional neural networks. Neural Evol. Comput., 1: 1-7.

Krizhevsky, A., I. Sutskever and G.E. Hinton, 2012. Imagenet Classification with Deep Convolutional Neural Networks. In: Advances in Neural Information Processing Systems, Leen, T.K., G.D. Thomas and T. Volker (Eds.). MIT Press, Cambridge, Massachusetts, USA., ISBN:0-262-12241-3, pp: 1097-1105.

Larochelle, H., D. Erhan, A. Courville, J. Bergstra and Y. Bengio, 2007. An empirical evaluation of deep architectures on problems with many factors of variation. Proceedings of the 24th International Conference on Machine Learning, June 20-24, 2007, ACM, Corvalis, Oregon, USA., ISBN:978-1-59593-793-3, pp: 473-480.

LeCun, Y., L. Bottou, Y. Bengio and P. Haffner, 1998. Gradient-based learning applied to document recognition. Proc. IEEE, 86: 2278-2324.

LeCun, Y., Y. Bengio and G. Hinton, 2015. Deep learning. Nat., 521: 436-444.

LeCun, Y.A., L. Bottou, G.B. Orr and K.R. Muller, 2012. Efficient Backprop. In: Neural Networks: Tricks of the Trade, Montavon, G., G.B. Orr and K.R. Muller (Eds.). Springer, Berlin, Germany, ISBN:978-3-642-35288-1, pp: 9-48.

Lee, H., R. Grosse, R. Ranganath and A.Y. Ng, 2009. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. Proceedings of the 26th Annual International Conference on Machine Learning, June 14-18, 2009, ACM, Montreal, Quebec, Canada, ISBN:978-1-60558-516-1, pp: 609-616.

Levy, S., 2017. The iBrain is here and it's already inside your phone. Wired Inc., Buena Park, California. https://www.wired.com/2016/08/an-exclusive-look-at-how-ai-and-machine-learning-work-at-apple/.

McCulloch, W.S. and W. Pitts, 1943. A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys., 5: 115-133.

Nielsen, M.A., 2017. Neural networks and deep learning. NASDAQ, New York, USA.

Pelaez, D.B., R.S.J. Estepar and L.M.J. Carbayo, 2016. Detection and classification of pulmonary emphysema in CT images using multiscale convolutional neural networks. Proceedings of the 34th Annual Congress on Spanish Society of Biomedical Engineering, November 23-25, 2016, Polytechnic University of Valencia, Valencia, Spain, ISBN:978-84-9048-531-6, pp: 370-373.

Perez-Carrasco, J.A., C. Serrano, B. Acha, T. Serrano-Gotarredona and B. Linares-Barranco, 2011. Rapid convolutional neural network without frames for digit recognition. MSc Thesis, University of Seville, Institute of Microelectronics of Seville IMSE-CNM, Seville, Spain.

Pinto, L. and A. Gupta, 2016. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), May 16-21, 2016, IEEE, Stockholm, Sweden, ISBN:978-1-4673-8026-3, pp: 3406-3413.

Pusiol, P., 2014. [Convolutional networks in compression of scenes]. Master Thesis, National University of Cordoba, Argentina, South America. (In Spanish)

Ranzato, M.A. and M. Szummer, 2008. Semi-supervised learning of compact document representations with deep networks. Proceedings of the 25th International Conference on Machine Learning, July 5-9, 2008, ACM, Helsinki, Finland, ISBN:978-1-60558-205-4, pp: 792-799.

Ranzato, P.M.A., C. Poultney, S. Chopra and Y. LeCun, 2006. Efficient learning of sparse representations with an energy-based model. Proceedings of the 19th International Conference on Neural Information Processing Systems, December 4-7, 2006, MIT Press, Cambridge, Massachusetts, USA., pp: 1137-1144.

Roth, H.R., L. Lu, A. Farag, H.C. Shin and J. Liu *et al*., 2015. Deeporgan: Multi-Level Deep Convolutional Networks for Automated Pancreas Segmentation. In: Medical Image Computing and Computer-Assisted Intervention, Navab, N., J. Hornegger, W. Wells and A. Frangi (Eds.). Springer, Berlin, Germany, ISBN:978-3-319-24552-2, pp: 556-564.

Salakhutdinov, R. and G. Hinton, 2009. Semantic hashing. Intl. J. Approximate Reasoning, 50: 969-978.

Salakhutdinov, R. and G.E. Hinton, 2007. Learning a nonlinear embedding by preserving class neighbourhood structure. Proceedings of the 11th International Conference on Artificial Intelligence and Statistics, March 21-24, 2007, JMLR, Inc., San Juan, Puerto Rico, pp: 412-419.

Schroff, F., D. Kalenichenko and J. Philbin, 2015. Facenet: A unified embedding for face recognition and clustering. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, June 7-12, 2015, IEEE, Boston, Massachusetts, ISBN:978-1-4673-6965-7, pp: 815-823.

Sermanet, P. and Y. LeCun, 2011. Traffic sign recognition with multi-scale convolutional networks. Proceedings of the 2011 International Joint Conference on Neural Networks (IJCNN), July 31-August 5, 2011, IEEE, San Jose, California, USA., ISBN:978-1-4244-9635-8, pp: 2809-2813.

Serre, T., G. Kreiman, M. Kouh, C. Cadieu and U. Knoblich *et al*., 2007. A quantitative theory of immediate visual recognition. Prog. Brain Res., 165: 33-56.

Silver, D., A. Huang, C.J. Maddison, A. Guez and L. Sifre *et al*., 2016. Mastering the game of go with deep neural networks and tree search. Nat., 529: 484-489.

Simonyan, K. and A. Zisserman, 2014a. Two-stream convolutional networks for action recognition in videos. Proceedings of the 27th International Conference on Neural Information Processing Systems, December 8-13, 2014, MIT Press, Cambridge, Massachusetts, pp: 568-576.

Simonyan, K. and A. Zisserman, 2014b. Very deep convolutional networks for large-scale image recognition. Master Thesis, Cornell University, Ithaca, New York.

Sutskever, I., J. Martens, G. Dahl and G. Hinton, 2013. On the importance of initialization and momentum in deep learning. Intl. Conf. Mach. Learn., 28: 1139-1147.

Taigman, Y., M. Yang, M.A. Ranzato and L. Wolf, 2014. Deepface: Closing the gap to human-level performance in face verification. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, June 28, 2014, IEEE, New Jersey, USA., pp: 1701-1708.

Theis, L., W. Shi, A. Cunningham and F. Huszar, 2017. Lossy image compression with compressive autoencoders. Proceedings of the 19th International Conference on Learning Robots (ICLR'17), December 28-29, 2017, WASET, Paris, France, pp: 1-19.

Utgoff, P.E. and D.J. Stracuzzi, 2002. Many-layered learning. Neural Comput., 14: 2497-2529.

Vincent, P., H. Larochelle, Y. Bengio and P.A. Manzagol, 2008. Extracting and composing robust features with denoising autoencoders. Proceedings of the 25th International Conference on Machine Learning, July 05-09, 2008, ACM, Helsinki, Finland, ISBN:978-1-60558-205-4, pp: 1096-1103.