International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-3, September 2019

Identification of Non-linear Systems through Convolutional Neural Network

P. Rajendra, A. Subbarao, G. Ramu, Rahul Boadh

Abstract: ***The theory of control systems deals with the analysis and design of interacting components of a system in a configuration that provides the desired behavior. This paper deals with the problem of the identification of non-linear systems through Convolutional Neural Network (CNN). We propose a structure of a CNN and perform simulations with test data using unsupervised learning for the identification of nonlinear systems. Also, MLP is used to compare the results when there is noise in the training data, which allows us to see that the proposed CNN has better performance and can be used for cases where the noise is present. The proposed CNN is validated with test data. Tests are carried out with Gas oven data, comparing the proposed structure of CNN with a MLP. When there is noise in the data, CNN has better performance than MLP.***

Keywords: Convolutional neural network, Non-linear Systems, Unsupervised learning

I. INTRODUCTION

The identification of systems is the process of obtaining a mathematical model of a system using data observed from it [1]. The mathematical models with which this process is carried out range from a linear equation to more complicated models such as neural networks. One of the important properties of neural networks is their adaptive nature, which means that they obtain their knowledge of their environment through an adaptive process called learning [2]. There are three popular learning paradigms namely supervised, unsupervised and reinforcement learning, each corresponding to a particular learning task [3]. The theory of control systems deals with the analysis and design of interacting components of a system in a configuration that provides the desired behavior. A model is a simplified representation of a system. Most of the mathematical models used are linear, due to their ease of handling compared to non-linear systems, and can accurately represent the behavior of real systems in many useful cases [4].

The starting point in the analysis of a control system is its representation by a mathematical model, which is usually an operator between inputs and outputs of the system or as a set of differential equations [5].

The identification of systems is a tool that allows representing the real behavior of systems based on experimental data obtained from the system. This is an iterative process, and the success of the identification depends on the quality of the input signals.

The convolutional neural networks (CNNs) are an example of a specialized neural network architecture presented by LeCun et al [6] at the beginning of the nineties, which includes knowledge about the invariance of 2D forms using local connection patterns and with restrictions on weights [7]. Many publications have described greater efficiency in training convolutional neuronal networks using the Graphics Processing Units (GPU) [9, 10, 11, 12]. The proposed architecture of CNN is formed by the stacking of different layers. In 2012, Ciresan significantly improved performance for several image databases including NORB database, MNIST database, HWDB10 dataset, and the ImageNet dataset. This paper presents the use of CNNs for the modeling of systems applying unsupervised learning paradigm and making a comparison between MLPs and CNNs by changing the number of parameters of each.

II. CONVOLUTIONAL NEURAL NETWORK

The architecture of a convolutional neural network (CNN) is formed by the stacking of different layers which are described below. Tests are carried out with three different reference examples, comparing the proposed structure of CNN with a MLP. For each of them, three simulation cases are tested. The first case is to use the test data of each reference without noise. The second case using a noise generated by the function wgn in the interval$[-0.1,0.1]$. The last case is described using a random noise.

A. Architecture of CNN

The proposed structure of the CNN consists of a total of 5 layers for the modeling of non-linear systems is presented in Figure 2.1. The sequence of these layers is a convolutional layer, followed by a sub-sampling layer and another convolutional layer. The last two layers correspond to fully connected layers. The parameters of convolution layer consist of a set of learning filters, which has a very small receptive field. In the forward learning step, each filter is convoluted with the entire field of vision producing a map of characteristics.

Manuscript published on 30 September 2019
* Correspondence Author

P. Rajendra*, Department of Mathematics, CMR Institute of Technology, Bengaluru, India. Email: rajendra.padidhapu@gmail.com
A. Subbarao, Dept. of Mathematics, Madanapalle Institute of Technology and Science, Madanapalle, India. Email: subbaraoa@mits.ac.in
G. Ramu, Dept. of Mathematics, Madanapalle Institute of Technology and Science, Madanapalle, India. Email: ramug@mits.ac.in
Rahul Boadh, School of Basic and Applied Sciences, K.R. Mangalam University, Gurgaon, India. Email: rahulboadhmsc26@gmail.com

Each element of the characteristic map can be interpreted as the output of a neuron that extracts information from a small region of the input and shares parameters with neurons that are on the same map.

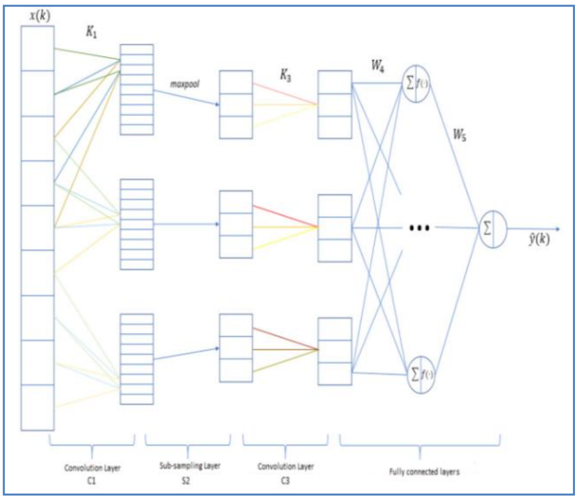

Figure 2.1: Structure of the Convolutional Neural Network for system modeling.

In general, the map of characteristics of a layer is given by [8]:

$$y_j^{(l)} = f\left(\sum_{i\in M_j} x_i^{(l-1)} k_{ji}^{(l-1)} + b_j\right) \qquad (2.1)$$

where M_j represents the set of selected inputs and the superscript l represents the current layer. Each output mapping has a different bias.

An important part of neural networks are the sub-sampling layers, which are applied after the convolutional layers. The main function of these layers is to reduce the size of the entrance, although carrying out this reduction in size does not lead to information losses, but it also brings benefits to the network. The most common operation to perform the reduction is max-pooling [16], which divides the input (image) into rectangles, and each of these rectangles retains the element of greatest value. Figure 2.2 shows the max-pooling operation reducing from a matrix 4x4 to a matrix 2x2 with the maximum values in each sub-region created in the input matrix. The most common is to use 2x2 dimension filters with a sub-sampling step of 2 units.

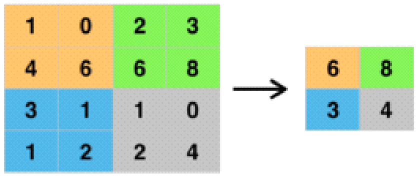

Figure 2.2: Max-pooling operation.

The output of this layer has the general form:

$$y_j^{(l)} = f\left(\beta_j down(x_i^{(l-1)}) + b_j\right) \qquad (2.2)$$

where $down(\cdot)$ represents a sub-sampling function. The combination between convolutional layers and sub-sampling is inspired by the work of Hubel and Wiesel on the notion of two types of cells in the visual cortex [17].

At the end of the combinations of the convolutional and sub-sampling layers, Fully connected layers are used where each element corresponds to a neuron and the total number of neurons in these layers corresponds to the number of classes to be predicted. The last layer is usually used for classification tasks. The softmax operator is the most used. Another method used is the SVM, which combined with the characteristics of the convolutional neuronal network can resolve different classification tasks [13, 14, 15].

B. UNSUPERVISED LEARNING

The input to the network is constituted by a vector $x^0(k) \in \Re^{n_u+n_v+1}$ with the following

form:

$$x^0(k) = [u(k-1)\dots u(k-n_u)\ v(k)v(k-1)\dots \quad u(k-n_v)]^T \qquad (2.3)$$

The inclusion of noise to the structure is shown in Figure 2.3, where it is observed that the new vector input to the network is defined as:

$$x^0_{noise}(k) = x^0(k) + \rho(k) \qquad (2.4)$$

Each convolutional layer consists of h filters $K_h \in \mathbb{R}^{f_l}$, which will be convoluted with the input vector to that layer. The map of characteristics generated when applying each filter, $x_h^{(l)} \in \mathbb{R}^{p_l}$ is given by:

$$x_{ih}^{l} = \sum_{a=0}^{f_l-1} K_{h,a} u_{h,i+a}^{(l-1)} + b_h^{(l)} \qquad (2.5)$$

with $i = 1,2,\dots p_l$, l indicates the index of the current layer and $b_h^{(l)}$ is the bias corresponding to each filter.

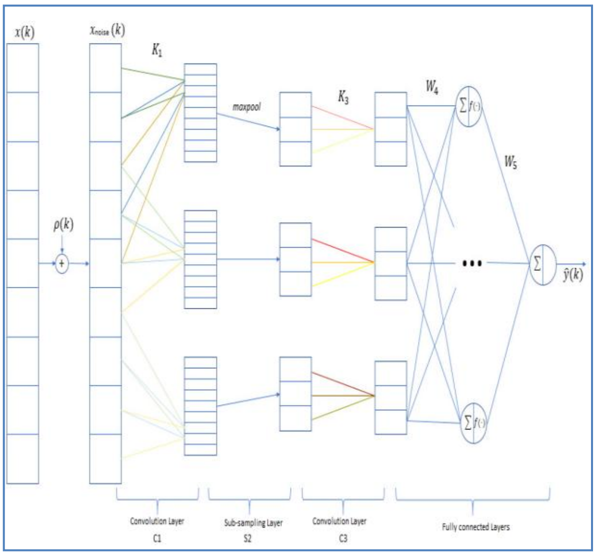

Figure 2.3: CNN structure for modeling with noise in the input data.

For the first layer, the input vector $x^0 = y^0$ and is common for all filters in this layer. The convolution operation is performed with a step $S = 1$ and a padding of zeroes $P = 1$. Therefore, the output dimension is $p_l = (W - f_l + 2P)/S + 1$, where W is the input vector dimension u_h^{l-1}. The output of the convolution layer is

$$y_h^{(l)} = f\left(x_h^{(l)}\right) \qquad (2.6)$$

with $f(.) = \tanh(.)$.

A sub-sampling layer is placed after the first convolutional layer in order to decrease the size of the data. The operation performed is max-pooling of dimension S_2 and the output of the sub-sampling layer is:

$$y_h^{(l)} = maxpool\left(u_h^{(l-1)}, S_2\right) \in \Re^{p_l} \qquad (2.7)$$

Two fully connected layers are used at the end of the network, with L-units in the hidden layer and one unit in the output layer. The input vector to the hidden layer is obtained by concatenating the outputs of the second convolutional layer in the following way:

$$X = \left[y_1^{(3)T} \dots\ y_h^{(3)T}\right]^T \in \Re^{h.p_3} \qquad (2.8)$$

The output of the network is obtained through:

$$\hat{y}(k) = W_5.f(W_4.X + b_4) + b_5 \qquad (2.9)$$

where $W_5 \in \Re^{1xL}$ are the weights in the output layer, $W_4 \in \Re^{Lxh.p_3}$ are the weights of the hidden layer, $b_4 \in \Re^L$ and $b_5 \in \Re^1$ are biases of their corresponding layers, and $f(.) = \tanh(.)$.

The total learning parameters of the network include the elements of the filters of the two convolutional layers, the bias of each of these filters and the weights with their biases in the two fully connected layers of the output. For the updating of network parameters, the use of randomized algorithms is proposed [18]. Therefore, the parameters of the convolutional layers and the hidden layer will remain fixed from the beginning, and only the parameters of the output layer will be updated in a supervised manner. The initialization of the parameters of the convolutional layers is done randomly within a range by:

$$K_h^l \in \left[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\right] \qquad (2.10)$$

where n represents the total inputs to that layer. The initialization of the weights of the fully connected layers is done randomly in the interval $[0,1]$. The two convolutional layers are used as filters to reduce the noise that may be present in the input data, because the convolution operation works as a linear filter in the spatial domain of the input data, being K_h^l the convolution kernels.

C. LEARNING BY LEAST SQUARES

The parameters of the output layer are updated by the generalized pseudo-reverse of Moore-Penrose [19] applied to linear systems as defined below.

Definition 2.1: The matrix $A^{-1} \in \Re^{nxm}$ is the Moore-Penrose inverse matrix of $A \in \Re^{nxm}$ if

$$AA^{-1}A = A, \quad A^{-1}AA^{-1} = A^{-1}, \quad AA^T,\ (AA^{-1})^T = (A^{-1}A)^T = A^{-1}A \qquad (2.11)$$

In particular, when A is full range by columns

$$A^{-1} = (A^TA)^{-1}A^T \qquad (2.12)$$

and when A is full range by rows

$$A^{-1} = A^T(A^TA)^{-1} \qquad (2.13)$$

Definition 2.2: $x_0 \in \Re^n$ is said to be the least standard solution for least squares of the linear system $y = Ax$ if $\|x_0\| \le \|x\|, \forall x \in \{x: \|Ax - y\| \le \|Az - y, \quad \forall z \in \Re n$
(2.14)

where $y \in\ \Re^m$ and $\|.\|$ is the Euclidean norm.

The solution by the least squares of a linear system $y = Ax, x_0$ is

$$\|Ax_0 - y\| = \min_x \|Ax - y\| \qquad (2.15)$$

Rewriting the whole model of CNN we have:

$$\hat{y}(k) = W_5.\Phi(k) \qquad (2.16)$$

where $\Phi(k) = f(W_4.X + b_4)$. This is a linear representation in the system parameters of the form $y = Ax$, where A may or may not be a square matrix, the solution x can be found by the generalized pseudo-reverse of Moore-Penrose. The objective of learning is to find the values of W_5 in such a way that the cost function J is minimized, being:

$$J = \sum_k \|y(k) - \hat{y}(k)\|^2 \qquad (2.17)$$

For this, consider the training data set $y(k)$ and $\Phi(k)$ with $k = 1,2, \dots, Q$, where Q is the total training data. Bringing together all the training set:

$$\begin{aligned} \hat{Y} &= [\hat{y}(1)\ \hat{y}(2) \dots\ \hat{y}(Q)] \\ &= [W_5.\Phi(1)\ W_5.\Phi(2)\ \dots\ W_5.\Phi(Q)] \\ &= W_5\Psi \qquad (2.18) \end{aligned}$$

where $\Psi = [\Phi(1)\ \Phi(2)\ \dots\ \Phi(Q)]$.

Considering the modeling error at each instant $e(k) = y(k) - \hat{y}(k)$.

We can rewrite (2.18) as:

$$\begin{aligned} Y &= [y(1)\ y(2) \dots\ y(Q)] \\ &= [W_5.\Phi(1) + e(1) \quad W_5.\Phi(2) + e(2) \dots W_5.\Phi(Q) + e(Q)] \end{aligned}$$

$$\therefore\ Y = W_5\Psi + \mathrm{E} \qquad (2.19)$$

where $E = [e(1) \quad e(2) \dots\ e(Q)]$. To minimize the cost function, we need $\frac{\partial J}{\partial W_5} = 0$. Using (2.13), we have that W_5^* can minimize J when:

$$W_5^* = Y\Psi^T(\Psi^T\Psi)^{-1} = Y\Psi^{-1} \qquad (2.20)$$

Since W_5^* is the minimum a least-squares solution of $Y = W_5^*\Psi + \mathrm{E}$, it achieves the minimum approach error within the entire training set.

III. SIMULATION AND RESULTS

Tests are carried out with Gas oven data, comparing the proposed structure of CNN with a MLP. Three simulation cases are tested. The first case is to use the test data of each reference without noise. The second case is the inclusion of a noise generated by the function wgn in the interval $[-0.1, 0.1]$. The last case is described using a random noise.

The Gas oven test data is obtained from G. Box et.al [23]. The data is the gas velocity $u(k)$ and the percentage of CO_2 present in the outlet gas

$y(k)$. The dataset has 296 data pairs $(u(k), y(k))$ in an interval of 9 seconds, of which 200 samples are used for learning and the rest for the generalization stage.

The input vector is chosen according to (2.3), with $n_u = 4$ and $n_v = 4$. The input data is normalized to the interval $[-1, 1]$. For convolutional layers, 3 filters are chosen per layer, $f_1 = f_3 = 3$ of dimensions $h_1 = h_3 = 3$. The dimension of the characteristic map of each filter in the corresponding layer is $p_1 = 9$ and $p_3 = 3$. In the sub-sampling layer, a maxpool operation is performed with $s_2 = 3$, having an output vector of three dimensions. The hidden layer has a total of $L = 18$ neurons. The Neural network with which the comparison is made consists of $L = 80$ neurons in the hidden layer and a total of 10 inputs, where these inputs correspond to the vector described in (2.3) with $n_u = 4$ and $n_v = 5$. The Figure 3.1 shows that the generalization part for CNN has good results.

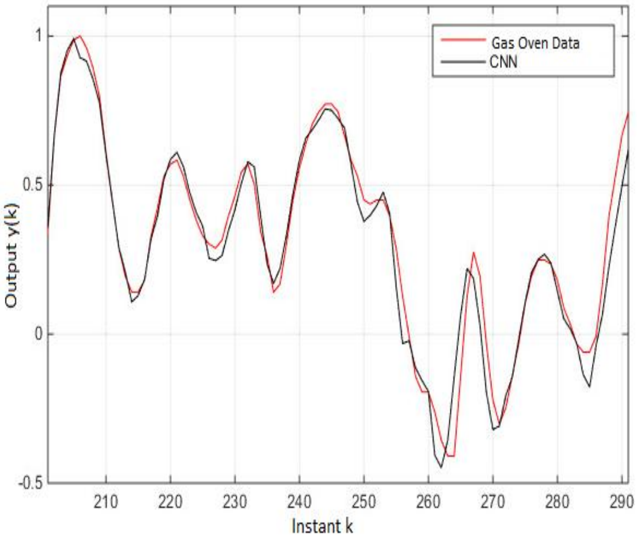

Figure 3.1: Generalization part of Convolutional Neural Network.

Figure 3.2 shows the result of the two-layer Neural Network, which behaves very much like CNN.

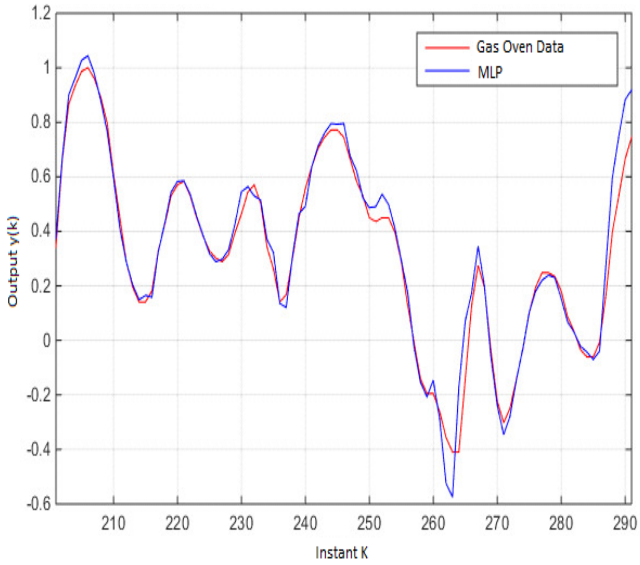

Figure 3.2: Output of Multilayer Neural Network.

With these results, it can be concluded that the two networks have very similar results and could improve or worsen according to the initial values of the parameters of both networks.

The wgn function generates Gaussian white noise, which is added to the input data of the network. The initializations of the parameters are the same as when there is no noise in the input data. The results are shown below in Figures 3.3, 3.4. As shown in Figure 3.3, the noise present in the input data does not affect too much in the generalization, the identification is still adequate. On the other hand, Figure 3.4 with the two-layer neural network is observed that it can no longer adequately identify the system.

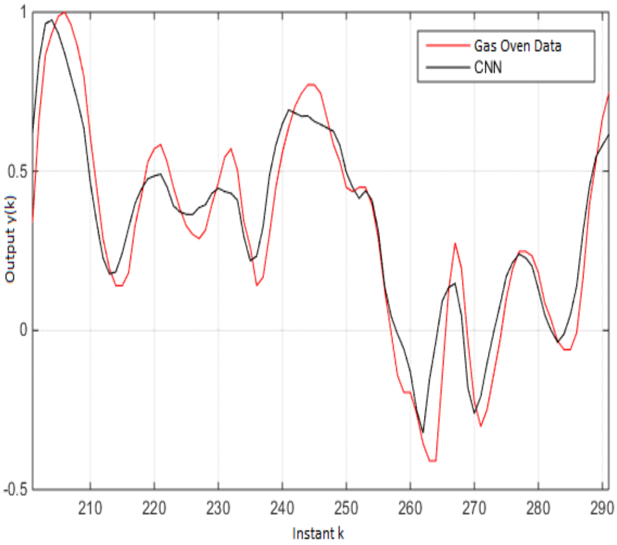

Figure 3.3: Convolutional Neural Network with Gaussian white noise

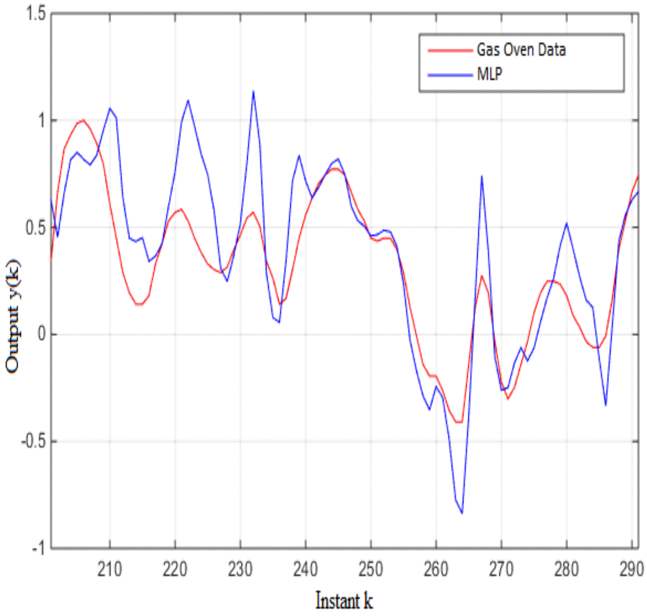

Figure 3.4: Multilayer Neural Network with Gaussian white noise

Now we present the results obtained when applying a random noise. The same initializations of the parameters are used as in the previous cases. Figure 3.5 shows the result of the network compared to the original test data.

Although noise is present in the input data, CNN maintains the general shape of the output, while the Multilayer Neural Network (Figure 3.6) completely loses the shape of the reference signal.

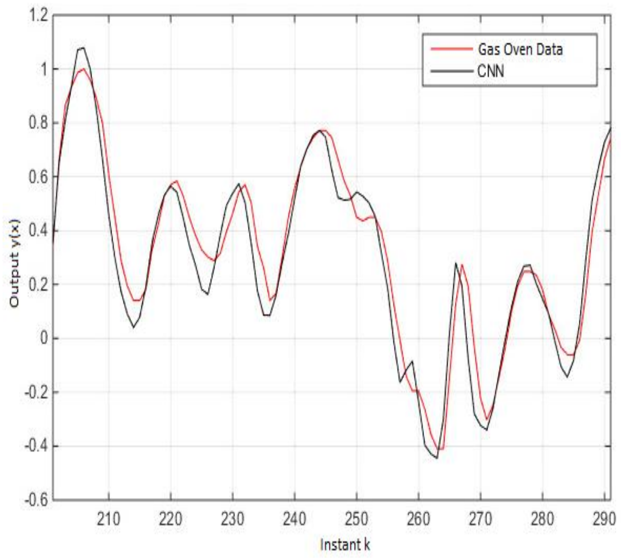

Figure 3.5: Convolutional Neural Network with noise.

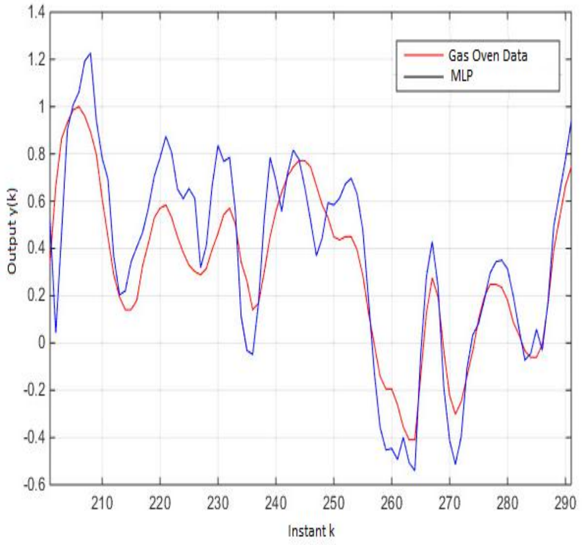

Figure 3.6: Multilayer Neural Network with noise.

In Table 3.1, the comparison of the quadratic mean errors for the results obtained using the gas oven data is shown. As it is observed, when there is no noise in the input data, both networks have very similar results and in the presence of noise, the convolutional neuronal network has the smallest errors. For this particular system, the results obtained using CNN are good, even in the presence of noise in the learning data, the network can identify the system.

Table 3.1: Gas oven Mean square error

	Without Noise	White Noise	Random Noise
CNN	0.0017	0.0025	0.0059
MLP	0.0014	0.0379	0.0111

VI. CONCLUSIONS

This work served to show the advantages of convolutional neural networks for the identification of systems when there is noise in the data. In the case of generalized pseudo-reverse of Moore-Penrose is used, both CNN and MLP results are very similar for the Gas oven systems. When there is noise in the data, CNN exceeds the MLP. As observed in this work, convolutional neuronal networks can be used for the identification of systems when there is noise in the data([20]-[27]).

REFERENCES

1. Y. LeCun, L. Bottou, Y. Bengio, P. Haffner. Gradient based learning applied to documentary recognition. Proceeding of the IEEE, 1998.
2. Rao, A., Sainath, S., Rajendra, P., et al. (2019). Mathematical Modelling of Hydromagnetic Casson non-Newtonian Nanofluid Convection Slip Flow from an Isothermal Sphere. *Nonlinear Engineering*, 8(1), pp. 645-660. Retrieved 16 Aug. 2019, from doi:10.1515/nleng-2018-0016
3. O. Nelles . Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy models, Science & Business Media, 2013.
4. Rajendra, P., Murthy, K.V.N., Subbarao, A. et al. Model. Earth Syst. Environ. (2019) 5: 1051. https://doi.org/10.1007/s40808-019-00590-2
5. H. Khalil. Nonlinear Systems. Second edition, Prentice Hall, NJ, 1996.
6. Y. LeCun, B. Boser, JS Denker, D. Henderson, RE Howard, W. Hubbard, LD Jackel. Backpropagation applied to handwritten zip code recognition. Neural Comput. 1(4), 541-551. 1989.
7. E. Busseti, I. Osband, and S. Wong., “Deep learning for time series modeling”, Technical report, Stanford University, 2012.
8. J. Bouvrie. Notes on Convolutional Neural Networks. MIT Press, 2006
9. Steinkraus, P. Simard. Using GPUs for Machine Learning Algorithms. 12th International Conference on Document Analysis and Recognition (ICDAR), pp. 115-119, 2005.
10. K. Chellapilla, S. Puri, P. Simard. High Performance Convolutional Neural Networks for Document Processing. Lorette, Guy. Tenth International Workshop on Frontiers in Handwriting Recognition, 2006.
11. GE Hinton, S. Osindero, YW Teh. A fast learning algorithm for deep belief nets, Neural Computation. Neural Computation, 18 (7), 2006.
12. J. Kruger, R. Westernmann. Linear Operators for GPU Implementations of numerical algorithms. Proceedings of SIGGRAPH, San Diego, pp. 908-916, 2013.
A. Krizhevsky, I. Sutskever, GE Hinton. ImageNet Classification with Deep Convolutional Neural Networks. in: NIPS. 2012.
13. Rajendra, P., Subbarao, A., Ramu, G. et al. Prediction of drug solubility on parallel computing architecture by support vector machines. Netw Model Anal Health Inform ioinforma, (2018) 7: 13.
14. Rajendra P, Kumar KS, Boadh R (2017) Design of a recognition system automatic vehicle license plate through a convolution neural network. Int J Comput Appl 177(3):47–54
15. Scherer, A. Muller, S. Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. in International Conference on Artificial Neural Network, 2010.
16. K. Fukushima, S. Miyake. Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position. Pattern Recognition, vol. 15, pp. 455-469, 1982.
17. de la Rosa, W. Yu. Randomized Algorithms for Nonlinear System Identification with Deep Learning Modification. Information Sciences, Vo. 364-365, pp. 197-212, 2016.
18. Box, G. Jenkins, G. Reinsel. Time Series Analysis: Forecasting and Control. 4th Ed,Wiley, 2008.
19. Murthy KVN, Saravana R, Rajendra P (2018a) Unobserved component modeling for seasonal rainfall patterns in Rayalaseema region, India 1951–2015. MAAP.
20. Murthy KVN, Saravana R, Rajendra P (2018b) Critical comparison of north east monsoon rainfall for different regions through analysis of means technique. MAUSAM 69(3):411–418

21. Padidhapu R, Bathul S, Brahmajirao V (2013) Least square and gauss jordan methods applied to fit the dielectric data vs. concentration of ionic liquids. Int J Sci Technol 2(1):60–70
22. Boadh, R., Satyanarayana, A.N.V., Ramakrishna et al .(2017), Numerical simulation of boundary layer flow parameters by using WRF-ARW Model over a tropical region. Jr. of Industrial Pollution Control. 33(1) : 1148-1154.

AUTHORS PROFILE

Dr. P. Rajendra, is working as Associate Professor, Department of Mathematics, CMR Institute of Technology, Bangalore. He completed his Ph.D in Mathematics from JNTUH, Hyderabad. He has 15 years of experience in teaching and research. His research interest includes mathematical modeling, machine learning, deep learning and their applications. He has more than 15 publications to his credit in various reputed international journals and conference proceedings. He has authored one book published by Lambert Academic Publishing, UK.

Dr. A. Subba Rao, is working as an Assistant Professor in the department of Mathematics, Madanapalle Institute of Technology & Science, Madanapalle since 2008. He completed his Ph.D. in 2014 under the supervision of Prof. N. Bhaskar Reddy and Prof. V. Ramachandra Prasad from Sri Venkateswara University, Tirupati, Andrapradesh. He has published 40 articles in reputed Journals. His research has also been presented at over 10 conferences. He is currently engaged in different non-Newtonian fluids over different curved bodies.

G. Ramu, is working as assistant professor in Madanapalle Institute of Technology & Science (MITS), India. His area of interest is operator theory, especially linear operators in Frame theory. He has good research articles in reputed national / international journals of mathematical sciences.

Dr. Rahul Boadh, working as Assistant Professor in School of Basic and Applied Sciences, K. R. Mangalam University, Gurgaon. He completed his Ph.D in Mathematics from IIT Kharagpur. After his PhD in August 2015, he joined as research associate in the project titled “Development of a real-time short-term ocean hindcast / forecast system with advanced initialization and DA capability for the Bay of Bengal (OTB)”. It was MHRD funded project. He has more than 15 publications to his credit in various reputed international journals and conference proceedings.