25th ABCM International Congress of Mechanical Engineering
October 20-25, 2019, Uberlândia, MG, Brazil

COB-2019-0225
DEEP LEARNING BASED INFERENCE SYSTEM FOR REAL-TIME ESTIMATION OF LPG CONTAMINANTS' MOLAR FRACTION

Jean Mario Moreira de Lima
Fabio Meneghetti Ugulino de Araujo
Federal University of Rio Grande do Norte, DCA, Control Process Laboratory. Natal/RN - Brazil, Zip Code 59078-970
jean@dca.ufrn.br
meneghet@dca.ufrn.br

Abstract. *The Liquefied Petroleum Gas (LPG) is one of the most lucrative final products of a Natural-Gas Processing Plant. Basically, LPG is composed of propane (C3) and butane (C4), with ethane (C2) and pentane (C5) as contaminants. Measuring and controlling the molar fraction of those contaminants are crucial for product quality monitoring. The molar fractions of the LPG contaminants are measured by gas chromatography or through laboratory analysis, which are high-cost and require large intervals. In this work, a inference system is built to estimate the molar fraction of LPG contaminants in real time. Deep learning models were implemented and their ability for estimation were tested. The results have showed that the deep learning based inference system estimates the molar fraction of LPG contaminants satisfactorily on a minute-by-minute basis. The mean percentage errors of estimation was* 0.36% *for ethane and* 0% *for pentane.*

Keywords: *LPG, Inference System, Deep Learning, Convolutional Neural Network*

1. INTRODUCTION

In face of an increasingly competitive market, efficient production is fundamental to obtain positive economic balance. Reducing costs, optimizing processes and offering high-quality products are factors that influence the economy of any industry. Consequently, techniques which can optimize production, such as monitoring of product quality or advanced and intelligent control become crucial to the industry.

In Natural Gas Processing Units (NGPPs), the monitoring of product quality is intrinsic to achieve a satisfactory production. As in most of the chemical processes, the quality control has been done through the chemical composition of the products. Chemical compositions are the primary variables of the process and also are hard-measure variables (Sotelo *et al.*, 2017). Usually done through laboratory analysis, the measurements of the chemical composition demand long time intervals for obtaining the required results (Fortuna *et al.*, 2005). Online measurement instruments are even available such as gas chromatographs and Near-InfraRed (NIR) analyzers. However, they both are high-cost equipment and have long measurement intervals (Shang *et al.*, 2014).

When technical limitations, high cost, and unavailability limit the use of real-time primary variable sensors, inference systems present a compelling manner to measure the process variables (Shokry *et al.*, 2018). The objective of developing an inference system is modelling a relationship between the primary variables, difficult to evaluate (e.g., chemical compositions), and secondary variables, which are easy and fast to measure (e.g., pressure, temperature, flow) (Yan *et al.*, 2017).

The most economically profitable product in an NGPP is the LPG. Ideally, it is formed by propane and butane (C3 and C4). However, the LPG has contaminants such as ethane and pentane (C2 and C5). The proposed inference system is built to estimate the molar fractions of the LPG contaminants (primary variables), from several easily measured as temperature and flow (secondary variables). The inference system may work as a gas chromatograph, generating molar fractions of pentane and ethane, but it does in real time. The quality monitoring of the LPG becomes possible since the large interval measurements issue is overcame. Thus, the product's quality and its profitability are improved.

The machine learning technique known as deep learning is used to implement the proposed inference system. Deep learning with multilayer architectures has excellent performance in complex inference problems. Theoretical studies indicated that deep architectures may be needed to efficiently model complex distributions and achieve better generalized performance on challenging learning (Yan *et al.*, 2017) (Lin and Yan, 2015). A deep neural network is capable of associating the nonlinear dynamic relationships between the primary and secondary variables of a process with no required prior knowledge about the system (Graziani and Xibilia, 2017).

In this work, a deep learning convolutional neural network (CNN) based inference system was implemented. Basically,

the easy-to-measure variables such as temperature, flow and pressure are the CNN inputs, while the CNN outputs is given by the molar fraction of the LPG contaminants: ethane and pentane.

A simulated NGPP on software HYSYS is used. The simulated NGPP is based on the Petrobras UPGNII GMR, located in Guamare City, RN, Brazil. This NGPP is formed by a deethanizing column in series with a debutanizing column.

2. THEORETICAL FUNDAMENTALS

2.1 Convolutional Neural Network

Convolutional neural network (CNN) was first proposed to process data that comes in the form of multiple arrays (LeCun *et al.* (1989), Zhou and Chellappa (1988)). Since then, it has made great success in object detection and recognition in the computer-vision domain, and also it has been utilized for time-series prediction as well. Basically, feature extraction and classification are the general functions of a CNN. In the feature extraction operation, convolutional layers and pooling layers are connected to process the input (raw data) into a representation. Then, dense layers or fully connected layers are adopted to classify/predict the obtained representation into a class/interval. The training phase is based on the backpropagation method and labeled data are required. Stochastic gradient descent (SGD) is the basic method for training deep neural networks such as a convolutional neural network. The SGD updates the weights and biases of neurons after each batch computation, and the main goal of which is to minimize the training error.

2.1.1 Convolutional Layer

In a convolutional layer, the output is composed of feature maps where each unit is related to a local patch in the input feature map through a kernel composed of a set of weights. The units in the output feature map share the same kernel (filter). Different feature maps in a layer use different filters. A example of a convolutional layer is shown in Fig. 1.

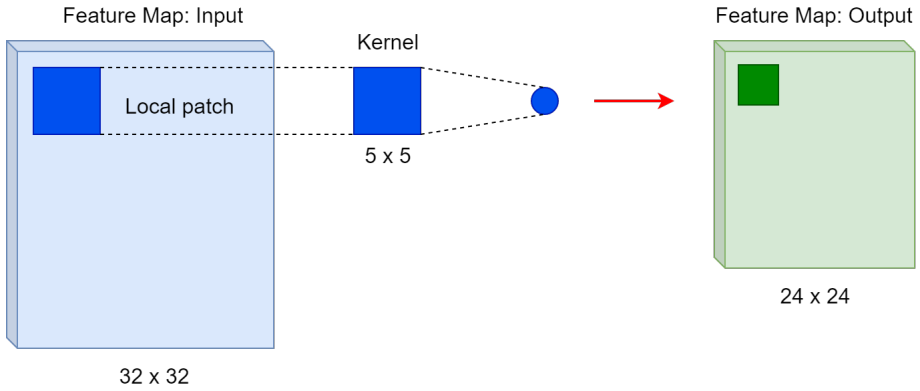

Figure 1. Convolutional layer.

In a convolutional layer, suppose that exists M feature maps as the input and N filters. Generally, Eq. (1) can be used to calculate the output feature maps of the lth layer:

$$x_j^l = f\left(\sum_{i=1,\ldots,M} x_i^{l-1} * k_{ij}^l + b_j^l\right), j = 1, \cdots, N, \tag{1}$$

where k_{ij}^l corresponds to the kernel of the jth filter connected to the ith input map, x_i^{l1} represents the ith input map and x_j^l is the jth output map, b_j^l corresponds to the bias corresponding to jth filter, f represents the activation function, and âĹŮ represents the convolutional operation. In this way, we can obtain N feature maps as the output. Assuming that the kernel size is $s \times s$, Eq. (2) can be used to calculate the number of the parameters of a convolutional layer:

$$P = N \times (s \times s \times M + 1) \tag{2}$$

For example, the convolutional operation is shown in Fig. 2. In this case, the size of the input map is 4×4, the kernel size is 2×2, and the stride is 1.

After the convolutional operation and the addition of the corresponding bias, the activation function is applied for computing the output feature maps. According to Wu et al. 2018 the common activation functions for neural networks include logistic function, hyperbolic tangent function and rectified linear unit (ReLU) function as shown in Eq. (3) - (5):

$$f(x) = \left(1 + e^{-x}\right)^{-1} \tag{3}$$

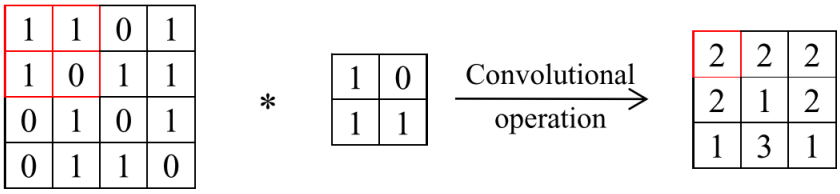

Figure 2. Convolutional operation.

$$f(x) = tanh(x) \tag{4}$$

$$f(x) = max(0, x) \tag{5}$$

According to Hoang et al. 2019, CNN with ReLUs can be trained several times faster than their equivalents with the other functions. Since then, ReLU has become the main activation function when a CNN architecture is designed.

2.1.2 Pooling Layer

A pooling layer is responsible for generating down-sampled versions of the input feature maps. The main goal of a pooling layer is to merge related local features into one. Basically, there are three advantages for the use of pooling layers. First, because the relative positions of the features forming a local pattern may vary slightly, it is more reliable for detection to merge similar features in local positions (Lecun *et al.* (2015)). Secondly, pooling layers usually do not have parameters and can reduce the dimension of the feature representation. The use of pooling layers can also greatly reduce the computation time and the parameters of the whole network. Thirdly, pooling layers are beneficial for preventing the "overfitting" (Wu and Zhao (2018)).

The computation procedure in a pooling layer is similar with that in a convolutional layer. In a pooling layer, assuming that there are M feature maps as the input, there must be M output feature maps. Generally, we can use Eq. (6) to calculate the output feature maps of the lth layer (Gunther *et al.* (2014)):

$$x_j^l = f\left(\beta_j^l down\left(x_j^{l-1}\right) + b_j^l\right), j = 1, \cdots, M, \tag{6}$$

where x_j^{l-1} is the jth input map and x_j^l corresponds the jth output map, b_j^l and β_j^l correspond the additive bias corresponding to the jth filter respectively, f is the activation function, and $down$ corresponds to the sub-sampling function. Then, it is possible to obtain M feature maps as the output.

2.1.3 Fully connected layer

Convolutional layers and pooling layers establish the feature extractor. Consecutive to the feature extractor, the purpose of fully connected (FC) layers is to classify the features extracted from the input. Consider that the lengths of the input and output are M and N, respectively. Each value of the input vector is connected to each value of the output vector through one neuron. Then we can use Eq. (7) to calculate the output vector of the lth layer (Wu and Zhao (2018)):

$$x_j^l = f\left(\sum_{i=1,\ldots,M} x_i^{l-1} \times w_{ij}^l + b_j^l\right), j = 1, \cdots, N, \tag{7}$$

where w_{ij}^l is the weight of the jth output value connected to the ith input value, x_i^{l1} performes the ith input value and x_j^l is the jth output value, b_j^l corresponds the bias corresponding to jth output value, f is the activation function. A regular CNN architecture consist of various FC layers in the end. The parameters of FC layers represents more than 80% of parameters of the whole CNN (Hoang and Kang (2019)).

2.2 Identification Using Neural Models

Basically, modeling structures based on neural networks with characteristics to identify nonlinear systems are generalizations of those that present linear modeling. Examples of linear modeling structures are: ARX (AutoRegressive eXogenous input), ARMAX (AutoRegressive, Moving Avegare, eXogenous input), OE (Output Error). These models are characterized by a vector that stores past values of the input variables. This vector is called the regression vector. If the regression vector chosen is equivalent to that used in the ARX model, the neural model structure is called Neural Network ARX (NNARX).

Each neural modeling structure can be described according to its regression vector. In the NNARX model, the regression vector, besides storing the past values of the input variables, also stores the past values of the output variables. Norgaard (2001), this model is stable since purely algebraic relations among its variables is present. Because of this, the neural modeling structure chosen as the basis to be used in the virtual sensor proposed here is the NNARX model illustrated in Fig 3.

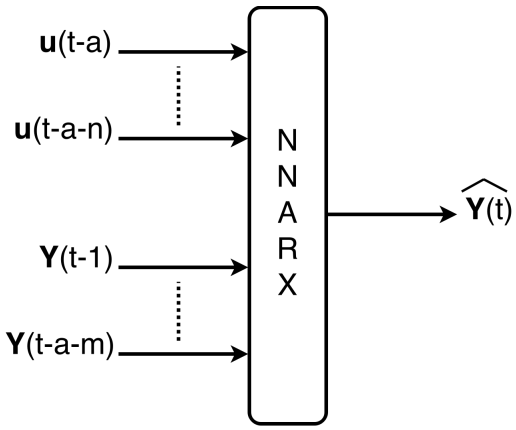

Figure 3. NNARX modeling structure.

Figure 3 shows the schematic diagram of an NNARX modeling. In the structure, $\widehat{Y}$ is the output of the network. The term a is the transport delay of the plant, n is the model entry order, m is the model output order, Y is the system output and u the input. It is also possible to observe regressors. The design of the structure is done to identify the dynamics of a physical system, the use of regressors is intrinsic, since they make the relationship between the output of the neural network and the past values of input and output of the system. The training model is the NNARX model, given by the equation:

$$\hat{y}(t) = f(y(t-1), ..., y(t-m), u(t-d), ..., u(t-d-n)) \tag{8}$$

where $\hat{y}$ is the output estimated by the network, f is the nonlinear function representing the mapping performed by the network, *d* the transport delay, *m* the order of the outputs and, finally, *n* represents the order of the entries.

In the tests, the performace of the inference system is evaluated. The actual values of output y will not be available to feed the virtual sensor, so it is fed back by its own estimates $\hat{y}$. Thus, the equation which rules the sensor implemented at validation or testing is described below:

$$\hat{y}(t) = f(\hat{y}(t-1), ..., \hat{y}(t-m), u(t-d), ..., u(t-d-n)) \tag{9}$$

where $\hat{y}$ is the output of the plant estimated by the convolutional neural network, f is the nonlinear function representing the mapping performed by the network, *d* the transport delay, of the outputs, and finally *n* represents the order of the inputs. Notice the output that feeds the network is its own inference.

3. PROPOSED INFERENCE SYSTEM

3.1 CNN Module

3.1.1 Outputs

Inferring difficult-to-measure primary variables from easy-to-measure secondary variables for monitoring is the purpose of this paper. The molar fractions of the main contaminants of the LPG are the primary variables, since the measurement of them through gas chromatographs is slow. Thus, the outputs of the proposed neural model are illustrated in Table 1:

Table 1. Primary variables of the process

p	**Primary Variable** (PV_i)
1	Ethane Molar Fraction (C2)
2	Pentane Molar Fraction (C5)

3.1.2 Inputs

The inputs are the secondary variables that will be used for the inference of the process variables, that is, the primary variables. The number of variables used will influence the degree of quality and complexity of the inference system. In distillation columns, temperatures of the different perforated plates, reflux flow, column pressure, temperature of the boiler and condenser, and volume of condensate may be secondary variables.

As the simulated system keeps real characteristics of the UPGN-II GMR plant, only the values of secondary variables that can be obtained through real physical sensors are taken into account.

Tray-16 temperature and refluxing flux of the debutanizer column are chosen for pentane inference and tray-40 temperature and the percent condensate volume of the deethenizer column for ethane inference. The choice is made based on the dynamic influence that the secondary variables imposes on the primaries..

Table 2. Secondary variables of the process

p	**Secondary Variable** (SV_i)	**Distillation Column**
1	Tray-40 temperature	Deetanizer Column
2	Condensate Volume	Deetanizer Column
3	Tray-16 temperature	Debutanizer Column
4	Reflux Flow	Debutanizer Column

Table 2 outlines the secondary variables chosen to compose the input of the CNN module. The controller column informs the PID controller associated with the secondary variable in the simulated plant.

3.1.3 Architecture

Mapping the dynamic relationships between primary variables and secondary variables of the process under study is the purpose of the neural network of the proposed inference system. The architecture to be used is the CNN. The modeling structure used is the NNARX for the network training, that is, during training, it will be fed back to the output values of the plant. In the validations and tests of the neural network of the system, it will be fed by its own estimates of the values of the molar fractions of C2 and C5. Then, we can combine CNN with de struture NNARX and call it CNNARX.

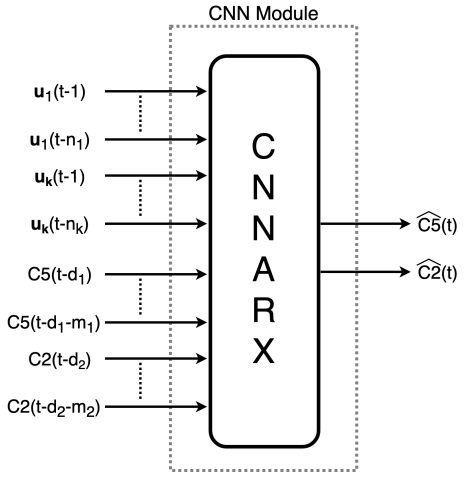

Figure 4. CNNARX structure.

Figure 4 illustrates the model adopted for training. The vector u represents each of the secondary variables used in this work. C2 and C5 represent the regression vector of the primary variables. They are values collected from the simulated system itself. $\widehat{C2}$ and $\widehat{C5}$ are the primary variables inferred in network training. The term **n** next to an index according to a respective secondary variable represents the order of the regression vector. Likewise, the term **m** is the vector regression order made up of the primary variables feeding the model. The term **d** is the delay of each of the feedbacks. Recalling that the schematic of the CNNARX structure is used for model training. For model tests, instead of the values of primary variables collected from the system for feedback, the values that are being inferred by the network are used. Thus, it is analyzed the potential of the system to operate without the necessity of analyzing the composition of the product in a short interval of time.

3.1.4 CNN Model

As the input are not multidimensional as a image, a one-dimensional CNN is built. Basically, a CNN model that has a convolutional hidden layer that operates over a 1D sequence is one-dimensional CNN. Kernel size was set in 2 and the chosen activation function was the ReLU. Also, a pooling layer whose job it is to distill the output of the convolutional layer to the most salient elements was built. In this model, the max pooling 1-D operation is chosen, with pooling size of 2. A max pooling unit computes the maximum of a local patch of units in a feature map.

The convolutional and pooling layers are followed by a dense layer (fully connected layer) that interprets the features extracted by the convolutional part of the model. A flatten layer is used between the convolutional layers and the dense layer to reduce the feature maps to a single one-dimensional vector. In the dense layer, the activation function is the ReLU.

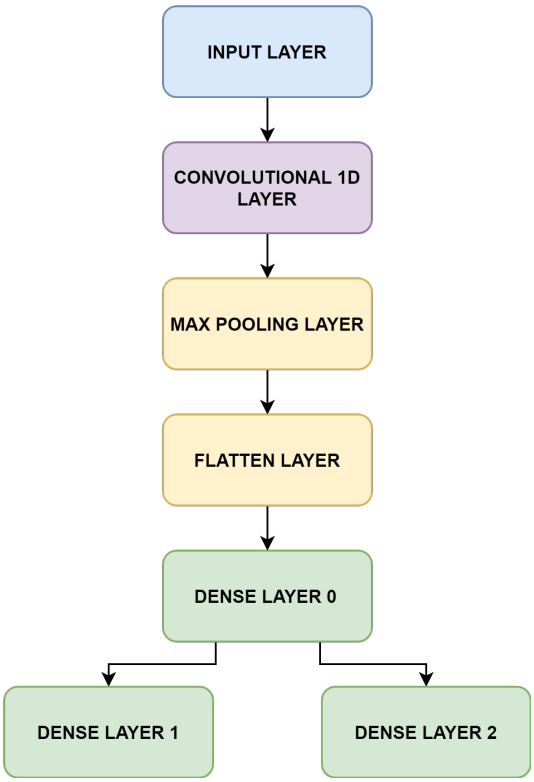

Figure 5. CNN Model.

Figure 5 represents the adopted CNN Model. The dense layers 1 and 2 give the results for inference of ethane and pentane, respectively. In the tests, it was built several CNNs with different numbers of neurons in the convolutional layer. The Mean Square Error (MSE) was used to evaluate the performance of the built structures CNN.

3.2 Implements

The methodology that was used in the development of the proposal of this work is briefly described. It presents the steps taken to implement the inference system and, consequently, to obtain results.

1. **Data Collection**: All necessary data were obtained for training and validation of the CNN. Pseudo-Random Signals (PRS) were applied to vary the setpoints of the controllers related to the sencodary variables listed in Table 2. Three sets of data were collected to adequately represent the system.

2. **CNN inference structure:** Once the data has been collected, the analysis is performed for several configurations of convolutional neural structures. The CNN is trained according to the configuration under evaluation: Second order inference structures with several neuron values in the hiddens layers. In this way, it is intended to observe how the architecture of the network can influence its ability to infer the results. After the validation of the structures, the best one is chosen to compose inference system. The comparison between the structures can be performed by analyzing the complexity of the CNN and the optimal inference capacity of the molar fraction of the GLP contaminants. In this work, this analysis can be done by the mean quadratic error of inference. For training and validation of the CNN structures, Keras API running on top of TensorFlow was used. The chosen optimizer algorithm is ADAM.

3. **Application in the system**: The behavior of the system is verified when applied to the simulated system to infer values of C5 and C2 for 800 minutes.

4. RESULTS AND DISCUSSION

4.1 Training and Validation of CNNs

For training of the convolutional neural networks, KERAS was used by running on TensorFlow. The networks were trained using the optimization algorithm: ADAM and cross validation and early stop were applied. It is also worth mentioning that before training the networks, the input and output data were normalized.

Table 3. Training and validation statistics.

Number of CNNs	**Trainings**	**Validations**	T_t	T_v
4	8	2	32	64

The number of convolutional layers was set 1 and several trainings were performed for various neuron values in the convolutional layer. Soon after each training, the convolutional networks were submitted to 2 sets of data, aiming to analyze their generalization capacity. Thus, the Table 3 with the statistics of training and validation is presented. T_t represents the total training, as well as T_v the total validations.

4.2 Analysis of Convolutional Neural Structures

The MSE of each of the structures are shown on Tebla 4. The term 'Model' represents the net where the layer and the numbers of neurons are showed, respectively. According to the MSE, the best structure is indicated to compose the inference system.

Table 4. Pentane and ethane inference validation statistics.

Model	**MSE pentane**	**MSE ethane**
Conv(32)-Pool-Flatten-Dense(50)-Dense1(1)-Dense2(1)	5.1732e-06	3.9915e-05
Conv(64)-Pool-Flatten-Dense(50)-Dense1(1)-Dense2(1)	2.9241e-06	5.035e-05
Conv(128)-Pool-Flatten-Dense(50)-Dense1(1)-Dense2(1)	1.3991e-06	1.0891e-06
Conv(256)-Pool-Flatten-Dense(50)-Dense1(1)-Dense2(1)	9.0721e-06	2.5563e-05

According to the Table 4, the best CNN model has 128 neurons in its convolutional layer. That CNN model was the selected structure to be applied to the simulated plant in real time. Pentane and ethane values were inferred over a range of 800 samples to test performance of the selected CNN.

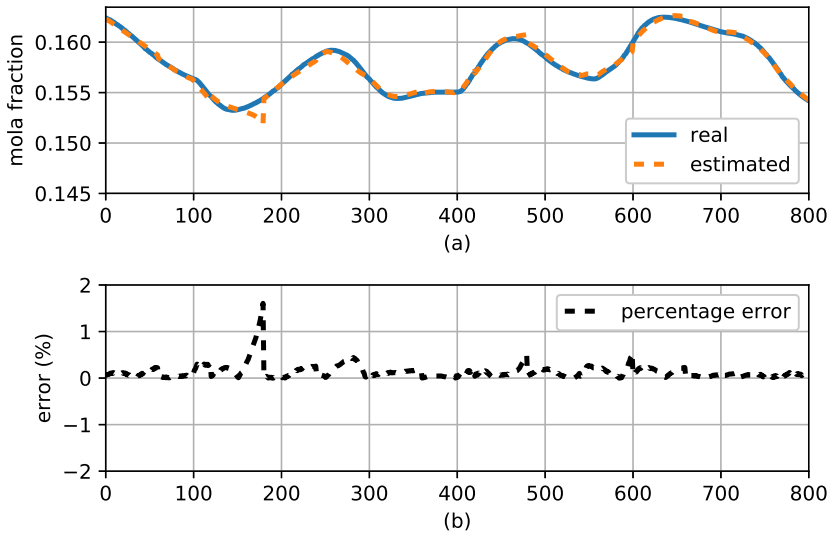

Figure 6. Real and estimated values of C2 molar fraction (a). Percentage error between real and estimated values (b).

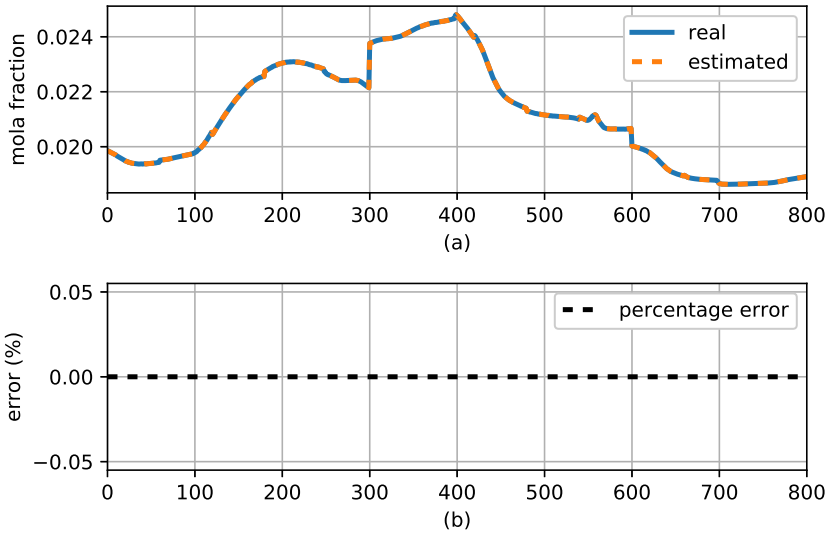

Figure 7. Real and estimated values of C5 molar fraction (a). Percentage error between real and estimated values (b).

In Figures 6, graph (a) presents the real (obtained by a gas chromatograph) and estimated molar fraction of C2, and graphs (b) shows the mean percentage error between real and estimated values of C2. In Figures 7, graph (a) presents the real and estimated molar fraction of C5, and graph (b) shows the mean percentage error between real and estimated values of C5.

According to the estimated values shown in Figures 6 and 7 convolutional neural network based inference system can estimate the molar fractions successfully. The mean percentage error of C2 inference is 0.36% and 0% for C5 inference.

5. CONCLUSION

According to the presented results, an efficient inference system for the estimation of LPG contaminants is possible to be implemented. The virtual sensor estimated molar fractions of pentane and ethane with an satisfactory approximation. The best obtained neural structure has shown MSE $1.3991e^{-06}$ for pentane inference and MSE $1.0891e^{-06}$ for ethane inference. The best CNN obtained structure was applied in the simulated plant to infer values of C5 and C2 minute-by-minute. Results have shown that the deep learning convolutional neural network based inference system can estimate the molar fractions successfully. The mean percentage error of C2 inference is 0.36% and 0% for C5 inference. The applicability of inference system based on deep learning in the processing of natural gas improves the LPG processing and enables quality monitoring.

6. REFERENCES

Fortuna, L., Graziani, S. and Xibilia, M.G., 2005. "Soft sensors for product quality monitoring in debutanizer distillation columns". Vol. 13, No. 4, pp. 499–508. ISSN 09670661. doi:10.1016/j.conengprac.2004.04.013.

Graziani, S. and Xibilia, M.G., 2017. "A deep learning based soft sensor for a sour water stripping plant". *I2MTC 2017 - 2017 IEEE International Instrumentation and Measurement Technology Conference, Proceedings*, pp. 1–6. doi:10.1109/I2MTC.2017.7969924.

Gunther, J., Pilarski, P.M., Helfrich, G., Shen, H. and Diepold, K., 2014. "First steps towards an intelligent laser welding architecture using deep neural networks and reinforcement learning". *Procedia Technology*, Vol. 15, pp. 474 – 483. ISSN 2212-0173. doi:https://doi.org/10.1016/j.protcy.2014.09.007. URL `http://www.sciencedirect.com/science/article/pii/S2212017314001224`. 2nd International Conference on System-Integrated Intelligence: Challenges for Product and Production Engineering.

Hoang, D.T. and Kang, H.J., 2019. "A survey on Deep Learning based bearing fault diagnosis". *Neurocomputing*, Vol. 335, pp. 327–335. ISSN 18728286. doi:10.1016/j.neucom.2018.06.078. URL `https://doi.org/10.1016/j.neucom.2018.06.078`.

LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W. and Jackel, L.D., 1989. "Backpropagation applied to handwritten zip code recognition". *Neural Comput.*, Vol. 1, No. 4, pp. 541–551. ISSN 0899-7667. doi:10.1162/neco.1989.1.4.541. URL `http://dx.doi.org/10.1162/neco.1989.1.4.541`.

Lecun, Y., Bengio, Y. and Hinton, G., 2015. "Deep learning". *Nature*, Vol. 521, No. 7553, pp. 436–444. ISSN 14764687. doi:10.1038/nature14539.

Lin, Y. and Yan, W., 2015. "Study of soft sensor modeling based on deep learning". *Proceedings of the American Control Conference*, Vol. 2015-July, pp. 5830–5835. ISSN 07431619. doi:10.1109/ACC.2015.7172253.

Norgaard, M., Ravn, O. and Poulsen, N.K., 2001. "Nnsysid and nnctrl tools for system identification and control with neural networks". *Computing Control Engineering Journal*, Vol. 12, No. 1, pp. 29–36. ISSN 0956-3385. doi: 10.1049/cce:20010105.

Shang, C., Yang, F., Huang, D. and Lyu, W., 2014. "Data-driven soft sensor development based on deep learning technique". *Journal of Process Control*, Vol. 24, No. 3, pp. 223–233. ISSN 09591524. doi:10.1016/j.jprocont.2014.01.012. URL `http://dx.doi.org/10.1016/j.jprocont.2014.01.012`.

Shokry, A., Vicente, P., Escudero, G., Pérez-Moya, M., Graells, M. and Espuña, A., 2018. "Data-driven soft-sensors for online monitoring of batch processes with different initial conditions". *Computers and Chemical Engineering*, Vol. 118, pp. 159–179. ISSN 00981354. doi:10.1016/j.compchemeng.2018.07.014. URL `https://doi.org/10.1016/j.compchemeng.2018.07.014`.

Sotelo, D., Favela-contreras, A., Sotelo, C., Jiménez, G. and Gallegos-canales, L., 2017. "Design and implementation of a control structure for quality products in a crude oil atmospheric distillation column". *ISA Transactions*, Vol. 71, pp. 573–584. ISSN 0019-0578. doi:10.1016/j.isatra.2017.08.005. URL `http://dx.doi.org/10.1016/j.isatra.2017.08.005`.

Wu, H. and Zhao, J., 2018. "Deep convolutional neural network model based chemical process fault diagnosis". *Computers and Chemical Engineering*, Vol. 115, pp. 185–197. ISSN 00981354. doi:10.1016/j.compchemeng.2018.04.009. URL `https://doi.org/10.1016/j.compchemeng.2018.04.009`.

Yan, W., Tang, D. and Lin, Y., 2017. "A data-driven soft sensor modeling method based on deep learning and its application". *IEEE Transactions on Industrial Electronics*, Vol. 64, No. 5, pp. 4237–4245. ISSN 02780046. doi: 10.1109/TIE.2016.2622668.

Zhou and Chellappa, 1988. "Computation of optical flow using a neural network". In *IEEE 1988 International Conference on Neural Networks*. pp. 71–78 vol.2. doi:10.1109/ICNN.1988.23914.

7. RESPONSIBILITY NOTICE

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