Željka Ujević Andrijić[1], Nikola Rimac[1], Ines Martić[2]

Development of a Soft Sensor in a Propylene Production Plant

[1]Department of Measurements and Process Control, University of Zagreb Faculty of Chemical Engineering and Technology, 10000 Zagreb, Croatia
[2]Department of Digital transformation, INA - Industrija nafte, d.d., Avenija Većeslava Holjevca 10, 10020 Zagreb, Croatia

Abstract

Propylene is a crucial intermediate in petrochemical synthesis, which requires high purity. This study details the creation of a soft sensor model for continuous monitoring of propylene levels in a propane/propylene splitter refinery facility. The data obtained from the plant of the input variables are processed using different pre-processing methods. Soft sensor models of neural networks with multilayer perception (MLP) and neural networks with long short-term memory (LSTM) were created using the Python programming language. During the development of the MLP model, various hyperparameters were tested, including the number of neurons in the hidden layer and the impact of the activation function type on the model's quality. Similarly, when developing the LSTM model, the number of LSTM units and the number of time steps into the past were also examined. A statistical analysis of the findings was performed, which revealed that both model types give strong correlation values between model data and real data of propylene content and that both neural network model types may be used in the refinery information system. The use of developed soft sensors guarantees that propylene content information is always up to date and continuous, allowing for fast responses to changes in propylene content for enhanced process management. The soft sensors improve the end product's quality and can result in considerable cost savings.

Key words: *soft sensor, neural networks, multi-layer perceptron, long short-term memory networks, propylene*

1. Introduction

Soft sensors

Many processes and plants struggle to measure crucial process variables on a continuous and reliable basis. The process control of such processes is based on laboratory analysis, which is frequently uncommon and time-consuming, and the cost of using and maintaining online analyzers can be exceedingly expensive. For these reasons, there is a need for the development and use of intelligent software that can continuously monitor various physicochemical properties and process variables. Using soft sensors, it is feasible to determine the state of difficult-to-measure variables from easily measurable secondary variables like temperature, pressure, and flow by identifying their functional correlations. [1]. The creation of soft sensors in the field of process engineering necessitates expertise from several scientific fields, as well as a mix of scientific research and plant experience. Soft sensors enable process engineers to analyze process variables and circumstances that cannot be monitored in real time and utilize them to improve process control. The problem of not being able to measure product properties in real time is due to several reasons, such as the fact that refineries do not have enough process analyzers installed in the plant itself because they are very costly to install and maintain, or that for some variables, such as chemical reaction conversions, there is no possibility of direct measurement at all. In process engineering, it is therefore highly desirable to use such software estimators in the areas of process control and management, fault detection and process diagnostics, instrumentation and measurement, where the need for measuring devices is reduced and measuring devices are replaced. [2] Given the frequency of failures and maintenance requirements as well as the high cost of online analyzers used for continuous measurement of propylene content, it is necessary to develop a model for continuous assessment of propylene product content in a propane/propylene splitter production plant.

Soft sensor models are divided into three basic categories: white-box models, also referred to as "mechanistic", "analytical" and "fundamental" models, gray-box models, also referred to as "semi-analytical" and "hybrid" models, and black-box models, also referred to as "empirical" models. [3] Black-box models depend exclusively on collected data, whereas white-box models utilize theoretical knowledge of the process. [4] Gray box models combine black and white box models to address issues; that is, their construction uses theoretical knowledge of the process in conjunction with data obtained from the process. [5] For the purpose of this study, black-box models were created, i.e., models of neural networks with multilayer perceptions and neural networks with long-term short-term memory. The term 'black box' refers to a model in which it is not necessary to know the nature of the process itself. Instead, the model is constructed solely by identifying the functional connection between the input and the output of the process. [6]

The general process of developing soft sensor models includes data collection, pre-processing and model development. During data collection, the availability and quality of the data is important as it forms the basis for all subsequent steps. In the model development phase, theoretical knowledge of the process, such as physical principles, reaction equations, etc., can facilitate the selection of input variables and data pre-processing. [2] Additional information from operators and maintenance specialists can aid in creating a soft sensor. After collecting data, it is necessary to preprocess it, which includes procedures such as

data visualization, identification of extreme and missing values, and data filtering. Outliers were identified by applying the 3σ rule when preparing this paper. It is a fundamental concept in statistics that is used to describe the distribution of data and the probability of finding a data point within a certain range. It is based on the concept of standard deviation (σ) and is particularly important in the context of normal distributions. According to this rule, all data points that lie outside the range $\mu \pm 3\sigma$, where μ is the mean of the data, can be considered extreme values or anomalies.

When preprocessing the data, influencing variables are selected; these are the input variables that have a specific impact on the output variable, i.e. on the variable whose value must be predicted. The type of model for sensor development is also defined. In this paper, Pearson›s correlation coefficient was employed to select the influencing variables. This statistical measure quantifies the linear relationship between two continuous variables, with values ranging from -1 to +1. These values indicate both the intensity and the direction of the relationship between the variables by estimating how well the data points of two variables displayed in the coordinate system are aligned along a straight line. [7]

Artificial neural networks

Artificial neural networks are computer models employed in machine learning, processing data based on the workings of the human brain, albeit on a much smaller scale. These networks are instrumental in solving various complex problems, including pattern recognition, classification, regression, and optimization tasks [8]. The initial type of neural network models developed were those with multilayer perceptrons, featuring multiple hidden layers and activation functions. Information flows bidirectionally through such networks, with input information propagating forward, and weight coefficients updated backward using the error gradient.

The basic architecture of such a neural network, shown in Figure 1, consists of three types of layers composed of nodes: an input layer, an arbitrary number of hidden layers, and an output layer, where the output of one layer is the input to the next layer. The input layer receives the raw data, and no computations are performed in this step, but the input neurons simply pass the input data to the hidden layer. Activation functions are introduced in the hidden layer, and most of the time all hidden layers use the same activation function. The output layer is the last layer of the network, which receives the information learned via the hidden layers and converts it into the final result. An activation function can also be used in the output layer, which is usually different from the one used in the hidden layers. [9] Activation functions are used to introduce non-linearity into the models, which is achieved by introducing non-linear activation functions. An important feature of nonlinear activation functions is their derivability, so that a backpropagation algorithm can be applied to calculate errors with respect to the weight coefficients and accordingly optimize the weights using one of the optimization techniques. The nonlinear activation functions most commonly used in neural networks, including sigmoid, ReLU, ELU, and tanh, were also employed in the preparation of this paper. [10]

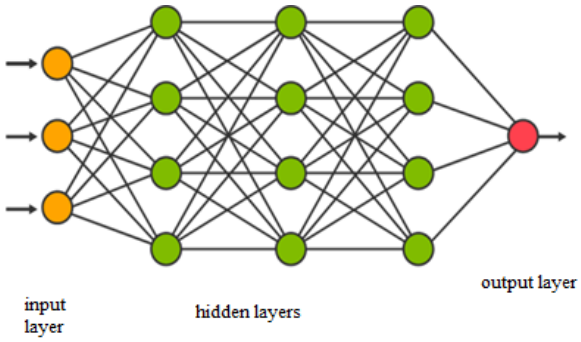

Fig. 1. *The basic structure of a multi-layer perceptrons neural network*

Each node has its own input information, through which it receives communication from other nodes and the environment, and its own output information, through which it communicates with other nodes and the environment. In addition, each node has its own activation function f, through which the input information is transformed into output information. The connections between the neurons in an artificial neural network are weighted by a weight coefficient w_i that represents the strength of the connection between neurons. The inputs are multiplied by the weight coefficients and then summed in the hidden layer using the *net* (Eq. 1) summation function, and their sum is compared to the neuron's threshold . If the sum of the weighted inputs exceeds the threshold of the neuron , the activation function *f* generates the output of the neuron *y* (Eq. 2). [11,12]

$$net = \sum_{i=0}^{n} w_i x_i \quad (1)$$

$$y = f(net - \theta) \quad (2)$$

Another type of neural network model that has recently been developed is the long short-term memory (LSTM) network. This variant of a feedback neural network addresses the challenge of retaining long-term dependencies present in conventional feedback neural networks. This is achieved by installing a "memory cell" that acts as a container and can store information over a longer period of time. The structure of a memory cell is shown in Figure 2. The memory cell of the LSTM model is managed by three so-called gates: input gate, forget gate and output gate. An input gate, mathematically represented by the formula (4), controls what information is added to the memory cell, a forget gate, mathematically represented by the formula (3), controls what data is removed from the memory cell, and an output gate, mathematically represented by the formula (5), controls what information is output from the memory cell.

$$f_t = \sigma((w \times x_t + w \times h_{t-1}) + b) \quad (3)$$

$$c_t = C_{t-1} \times f_t + i_t \times c'_t \quad (4)$$

$$h_t = o_t + tanh(c_t) \quad (5)$$

In this way, LSTM networks can selectively retain or discard information, allowing them to learn long-term dependencies. The cell state (represents long-term memory and is updated using a multiplication or summation operation, and this state is not directly affected by the weights. The hidden state () represents short-term memory and is affected by the weight values.[13]

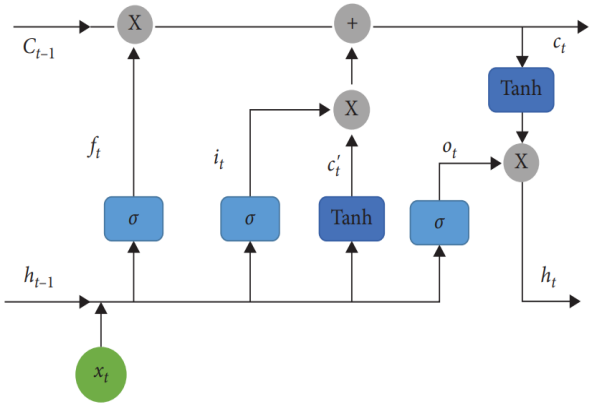

Fig. 2. *The structure of a memory cell of an LSTM network [13]*

With the backpropagation algorithm the gradient is calculated, that is, the amount by which the weights w_i must change in the positive or negative direction in order to minimize the loss function. After the network calculates the output, it is compared to the actual value of the output. The square of the difference between these two values, or the mean squared error, represents the loss function for regression problems. MSE can be mathematically represented by the formula [14]:

$$\text{MSE} = \frac{\Sigma_{i=1}^{n}(y_i - \bar{y}_i)^2}{n} \quad (6)$$

where n is the number of samples, is the actual output value, and is the corresponding predicted output value. The initial values of the weights are chosen randomly and passing through the network gives the first value which is compared with y_i.

After calculating the gradient, the weights are optimized layer by layer, starting from the last hidden layer and working backwards towards the input layer until the error no longer changes. At this point, learning is stopped, and it is assumed that the loss function has reached a global minimum. Problems that can occur during this learning process include reaching a local minimum instead of a global minimum and overfitting the network. An overtrained network has a very small error on the training data, but on the test data the error is significantly higher, i.e., the network loses its ability to generalize and becomes specific only to a particular data set. Although it is desirable that the error is as small as possible, it is also important that it is consistent on the training and validation dataset of the model so that the model has reasonable efficiency when introducing new data.[15]

2. Materials and methods

Propane/propylene splitter

The propane/propylene splitter (PPS) plant consists of two sections, which are shown in Figure 3. Section 1 is the section for the feed vessel of the raw material and its purification, while section 2 is for the separation of the propane-propylene mixture. For the purposes of this paper, more attention was paid to section 2, which consists of two distillation columns where the separation of propylene and propane takes place. The aim of this unit is to achieve maximum purity of propylene and minimize the propylene content in propane. The propylene produced in the process is a colorless, odorless, tasteless, flammable gas with a required minimum purity of 99.6% by volume.

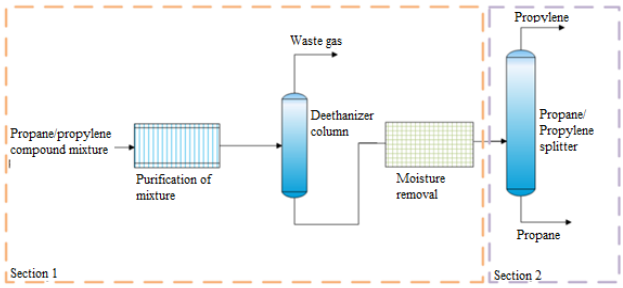

Fig. 3. *Propane/propylene splitter plant section*

The models were developed using Python, a programming language that contains numerous packages and modules specialized for machine learning and data analysis. First, NumPy and Pandas packages were utilized for data preprocessing and analysis. Matplotlib was employed for data visualization, allowing for the creation of insightful plots and graphs to better understand the data distribution and relationships. Neural network models were developed using the Keras and Scikit-learn tools.

Data from the refinery database was used to assess the content of propylene products in the PPS-producing facility. The imported data consists of input and output variables measured over two months. For all input and output variables, approximately 75,000 data points were collected with a sampling time of 1 minute. The data points for the output variable (propylene content) were taken from the AI201A/B gas chromatograph with a sampling time of 7 minutes.

After data acquisition in the system, data pre-processing followed. Data preprocessing for the development of the MLP model included: shifting the input variables data for a certain number of time steps based on the dead time, i.e., the time it takes for the input variable to begin impacting the output, followed by detecting and deleting extreme values. The detection of extreme values was carried out using the 3σ method and by visual inspection, as the values recognized as extreme by 3σ can be part of the dynamic behavior of the process, which is important for the creation of the model.

Data preprocessing during the creation of the LSTM model comprised moving the input data by a particular number of time steps owing to input dead time, and extreme values were not deleted because such models employ data sequences in which the time sequence of the data is significant.

The correlation between the input variables and the output variable was determined using the Pearson correlation coefficient. When selecting the number of input variables, care was taken to ensure that the number of influencing variables was large enough to capture as much important information from the process as possible, but not too large due to the need to simplify the model. Because of the connections discovered, several of the input variables were not included in the model's further development. Seven of the 16 available input variables were chosen based on their association with the output for further model building.

When developing the MLP model, the data set was divided into three parts: the training set, the test set, and the validation set. The training set comprised 80% of the data, while the test set comprised 20% of the data. The validation set consisted of 20% of the training set. When developing the networks, the number of hidden layers was set to 1, while the activation functions and the number of neurons in the hidden layer were changed. The activation functions used were sigmoid, tanh, ReLU and ELU, and the number of neurons in the hidden layer varied in the range from 1 to 20. A network with the highest correlation coefficient and lowest mean square error was chosen for each activation function. The Adam algorithm was used as the optimization algorithm.

When developing the LSTM network model, the data set was split in the same way as when developing the MLP model. The number of LSTM units ranged from 1 to 35, representing the number of time steps in the past, and the activation functions were modified. The activation functions used were the previously mentioned sigmoid, tanh, ReLU and ELU functions. For each of the activation functions used, the best network was selected, i.e., the network with the maximum correlation coefficient and the minimum mean square error. The Adam (Adaptive Moment Estimation) algorithm was used as the optimization algorithm.

In addition to the graphical comparison of the model results and the experimental measurements, the created models within a specific set of models were compared on the basis of the mean square error calculated on the training set, the test set and the validation set, as well as on the basis of the Pearson correlation coefficient between the results calculated by the model and the real data. The best models from a given set of models were additionally evaluated by analyzing the model's error trend (residuals) and using error histograms.

3. Results and discussion

The plant provided process information, including 16 input variables sampled every 1 minute and a single output variable, propylene content, which was similarly sampled every 1 minute. Approximately 75,000 data points were collected for each input and output variable. During the pre-processing of the data for the development of the MLP and LSTM models, the values of the input variables were temporally shifted by 7-time steps, equivalent to 7 minutes into the past concerning the output variable (representing the time required for sampling and calculating by the online analyzer).

When creating the MLP model, the dead time was taken into consideration, which represents the time it takes for the input variable to affect the output variable. Pearson correlation coefficient values were computed with a 7-time step shift, incorporating input variable lag times of 30, 60, 90, and 120 minutes. Following the comparison of all correlation values, data with a 7-time step shift plus a lag of 90 minutes were chosen for further analysis. Initially, influential variables were identified as those with a correlation greater than ± 0.10 with the output variable. Subsequently, mutual correlations among different input variables were examined. Considering the obtained correlation results, the input variables were selected for further development of the model: TI214 (exit temperature from the C-202 column (bottom of the column)), FIC204 and FIC203 (reflux of the propylene product at the top of the C-202 column), PI203A (pressure at the top of the C-202 column), PI202 (product pressure at the outlet of the C-201 column) and FIC201 (product flow of the C-201 product at the top of the V-102 column). The variables

Table 1. *Pearson correlation coefficients for the input variables on the training data set for the MLP model*

	Correlation with AI201A
TI214	-0.099
FIC208	0.497
FIC204	0.799
FIC203	0.504
PI203A	0.137
PI202	0.554
FIC201	-0.158

Table 2. *Pearson correlation coefficients for the input variables on the training data set for the LSTM model*

	Correlation with AI201A
TI214	-0.103
FIC208	0.488
FIC204	0.742
FIC203	0.489
PI203A	0.137
PI202	0.554
FIC201	-0.116

PI202 and PI203B with a correlation greater than ± 0.10 were selected as influencing variables. The next step was to analyze the relationships between the various input variables to determine their mutual correlations. The input variables for the further development of the model were selected based on the correlation results obtained: TI214 (outlet temperature from the C-202 column (bottom of the column)), FIC208 (reflux of the product from the bottom of the C-202 column). The variables PI202 and PI203B showed a similar correlation value with AI201A/B, of 95.5%. Consequently, PI203B was excluded from further model development. Similarly, the variables FIC202 and FIC208 exhibited a close correlation of 99.5% with AI201A/B, leading to the exclusion of FIC202 from additional model development.

The 3σ rule was used to identify extreme values, and missing values and related data from other variables were eliminated during the development of the MLP model.

When developing the LSTM model, the dead time of the input variables was not considered when pre-processing the data, as the LSTM shifts the input variables into the past by a certain number of time steps during its calculation. The dead time of the process's input variables can be determined by adjusting the number of previous time steps considered by the LSTM model during development. As with the development of the MLP model, all input variables whose correlation with the output variable was greater than ± 0.10 were first taken as influencing variables and then the mutual correlations between different input variables were observed. The results were the same as the MLP model, i.e., the variables PI203B and FIC202 were not considered in the further development of the model. Extreme values were not removed during data preprocessing, which is problematic for LSTM models that rely on temporal sequences of data. During data pre-processing for both model types, resampling was carried out with a time step of 3 minutes in order to reduce the amount of data and thus the calculation and overall model development time. The Pearson correlation coefficient was then recalculated, which increased slightly, i.e., all variables considered in the previous step still had a satisfactory Pearson coefficient value for continued model development.

The models were developed using the Python programming language (Python version 3.9.7) and its Anaconda distribution (Anaconda Navigator version 2.1.1). 80 MLP models were developed in advance, whose structure differed in terms of the hyperparameters of the model, such as the number of neurons in the hidden layers and the activation functions used. In addition, 140 LSTM models were preliminarily developed, which differed in terms of the hyperparameters of the LSTM model such as activation functions, number of LSTM units and number of steps in the past.

The entire data is divided into a training set and a test set in a ratio of 0.8:0.2. 20% of the data from the training set was used as the validation set. The Pearson coefficient values were then calculated only for the training data set to avoid possible data loss. Indeed, if the entire data is used when developing a model, there is a possibility of data leakage, where data that does not belong to the training dataset is used for some steps in the creation of the model. The freshly acquired information may assist the model in learning or discovering something it would not have known otherwise, resulting in a falsely satisfied appraisal of the model's performance. The results of the correlations of the selected input variables with the output variable are shown in Table 1 for the MLP model and in Table 2 for the LSTM model.

MLP model results

During the creation of the MLP model, a decision was made to have only one hidden layer, but adjustments were made to the activation functions and the number of neurons within that hidden layer. Of the activation functions, the sigmoid, tanh, ReLU and ELU functions were used, and the number of neurons in the hidden layer varied between 1 and 20. A computer experiment was conducted for each of the chosen activation functions to ascertain the optimal number of neurons in the hidden layer that yields the highest correlation factor and lowest mean square error on the training dataset. In addition to the correlation factor and the mean square error, the success of the model is also influenced by the mutual compatibility of the correlation factors of the training and the test and validation dataset as well as the overall data. An optimal model should have high and uniform correlation factors for all data sets and a minimum squared error (as close to 0 as possible). The results of the influence of the number of neurons in the hidden layer on the accuracy of the model are shown graphically in Figure 4.

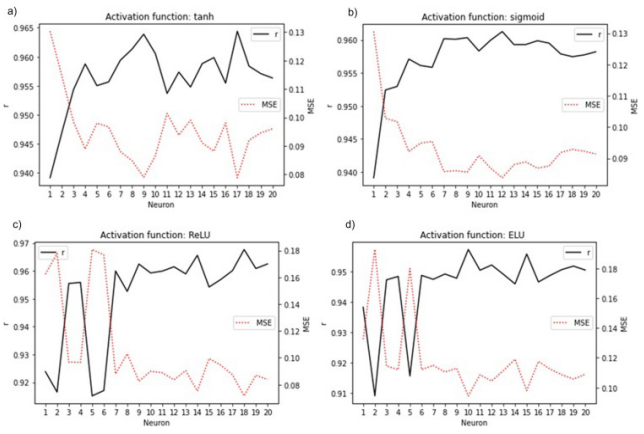

Fig. 4. *The influence of the number of model neurons in the hidden layer on the correlation factor and the MSE for the activation functions a) tanh (top left), b) sigmoid (top right), c) ReLU (bottom left) and d) ELU (bottom right)*

The findings presented in Figure 4 demonstrate that the models utilizing the tanh and sigmoid activation functions exhibit the highest level of stability in terms of correlation coefficient and the lowest squared error on the training set. Upon conducting a thorough examination of the acquired models, considering the numerical data presented in Table 3 and the visual representations in Figure 4, it

was determined that all models exhibit a satisfactory level of accuracy for implementation in the plant, attributed to the elevated correlation coefficients. The smallest mean square error on the test set is shown by the ReLU model with 18 neurons in the hidden layer, and slightly higher by the tanh model with 17 neurons. Finally, the model with the activation function tanh and 17 neurons in the hidden layer was selected as the best model in terms of correlation coefficients and minimum square error. The validation set (Fig. 5) and the entire dataset (Fig. 6) display the outcomes of a graphical analysis, illustrating the comparison between the model and the actual data for this network.

Table 3. *Comparison of MLP models with respect to correlation factors and MSE for different activation functions*

Activation function	tanh	sigmoid	ReLU	ELU
Number of neurons in hidden layer	17	12	18	10
Correlation - entire dataset	0.964	0.961	0.967	0.956
Correlation - training set	0.964	0.961	0.968	0.957
Correlation - test set	0.962	0.959	0.967	0.953
Correlation -validation set	0.961	0.959	0.967	0.955
MSE - training set	0.078	0.084	0.072	0.094
MSE - test set	0.077	0.082	0.070	0.096
MSE - validation set	0.076	0.084	0.069	0.093

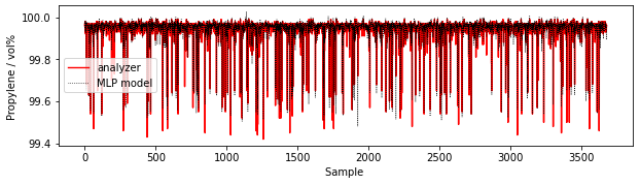

Fig. 5. *Comparison between real propylene content and MLP model on validation data for the model with tanh and 17 neurons in the hidden layer*

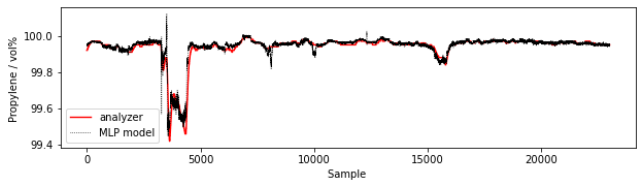

Fig. 6. *Comparison between real propylene content and the MLP model on the entire data for the model with tanh and 17 neurons in the hidden layer*

Figure 7 illustrates the histogram depicting the distribution of error values across the entire data of the MLP model. Conversely, Figure 8 displays the histogram representing the distribution of error values specifically on the validation data of the MLP model. The histograms illustrate that the error range lies within ± 0.15% in both cases, with the majority of errors falling within ± 0.1%. This level of accuracy is deemed satisfactory for the plant's application.

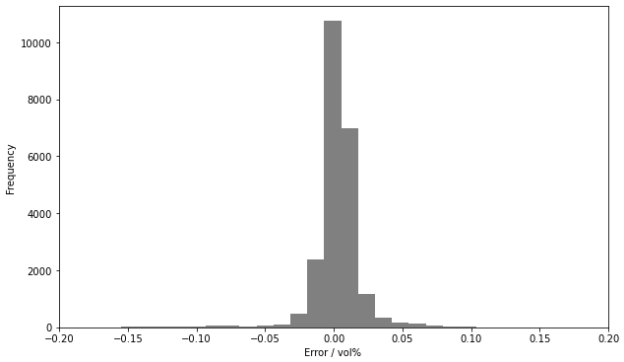

Fig. 7. *Histogram of the distribution of error values on the entire data for the MLP model tanh with 17 neurons in the hidden layer*

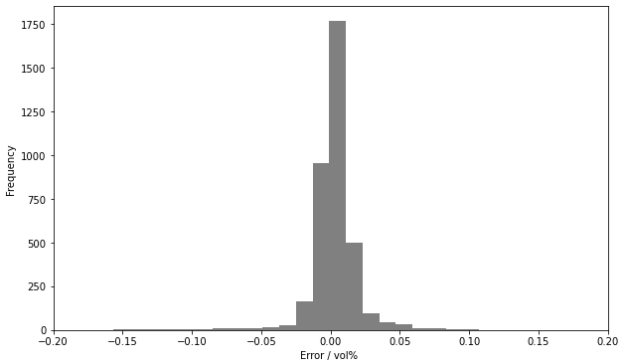

Fig. 8. *Histogram of the distribution of error values on the validation data for the MLP model with tanh and 17 neurons in the hidden layer*

LSTM model results

During the development of the LSTM networks, the number of LSTM units varied from 1 to 35, which also represents the number of time steps, and the activation functions were changed. The previously mentioned sigmoid, tanh, ReLU and ELU functions were used as activation functions. An extensive computer experiment was conducted to assess the optimal number of LSTM units in the hidden layer for each selected activation function. The experiment aimed to identify the configuration that yields the highest correlation factor and the lowest mean square error on the learning dataset. The success of the model depends on more than just the correlation factor and mean square error; it is also influenced by the mutual compatibility of correlation factors across the learning dataset, test dataset, and evaluation dataset, as well as the entire dataset. An optimal model is characterized by strong and uniform correlation factors across all data sets, with the squared error minimized to nearly zero. Figure 9 visually presents the correlation between the number of steps into the past and the accuracy of the model.

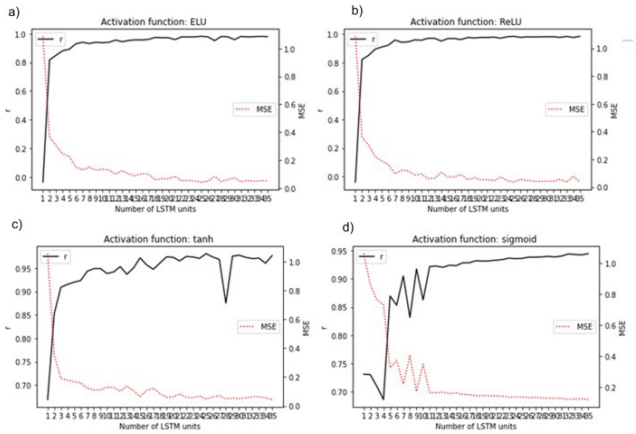

Fig. 9. *The influence of the number of model LSTM units in the hidden layer on the correlation factor and the MSE for the activation functions a) tanh (top left), b) sigmoid (top right c) ReLU (bottom left) and d) ELU (bottom right)*

The results shown in Figure 9 indicate that the models with the activation function ReLU and ELU have the most stable correlation coefficient and the lowest squared error in the training set.

Following an in-depth analysis of the obtained models, with reference to the numerical findings in Table 4 and the visual representations in Figure 9, it was established that all models demonstrate sufficient accuracy for practical use at the plant, given the high correlation coefficients.

Table 4. *Comparison of LSTM models with respect to correlation factors and MSE for different activation functions*

Activation function	tanh	sigmoid	ReLU	ELU
Number of LSTM units	25	31	33	31
Number of LSTM time steps	25	31	33	31
Correlation - entire dataset	0.981	0.940	0.985	0.981
Correlation -training dataset	0.982	0.938	0.985	0.981
Correlation - test dataset	0.979	0.948	0.986	0.982
Correlation -validation dataset	0.981	0.951	0.981	0.983
MSE - training dataset	0.047	0.129	0.036	0.045
MSE - test dataset	0.045	0.110	0.031	0.039
MSE - validation dataset	0.045	0.114	0.037	0.041

The validation set (Fig. 10) and the complete dataset (Fig. 11) exhibit the graphical comparison results between the model and the real data for this network. The smallest mean square error on the test set was found for the model with the ReLU function and 33-time steps and 33 LSTM units in the hidden layer, and the slightly larger ELU model with 31-time steps and 31 LSTM units in the hidden layer. Finally, the model with the activation function ReLU and 33-time steps and 33 LSTM units in the hidden layer was selected as the best model in terms of the magnitude of the correlation coefficients and the minimum square error.

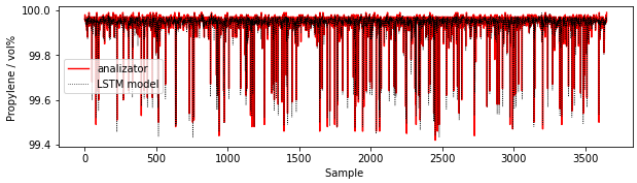

Fig. 10. *Comparison between actual propylene content and LSTM model on validation data for the model with ReLU, 33 LSTM units and 33-time steps*

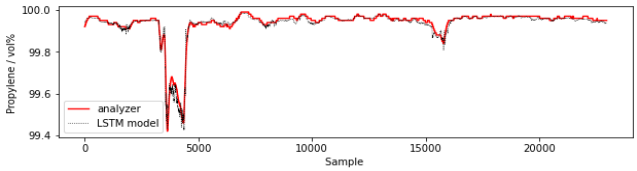

Fig. 11. *Comparison between actual propylene content and LSTM model on the entire data for the model with ReLU, 33 LSTM units and 33-time steps*

In Figure 12, the histogram visualizes the distribution of error values on the entire dataset of the LSTM model, while Figure 13 presents the distribution of error values specifically on the validation data of the LSTM model. By examining both histograms, it becomes evident that the errors predominantly lie within the ± 0.05% range. This finding serves as strong evidence for the exceptional reliability of the developed LSTM model.

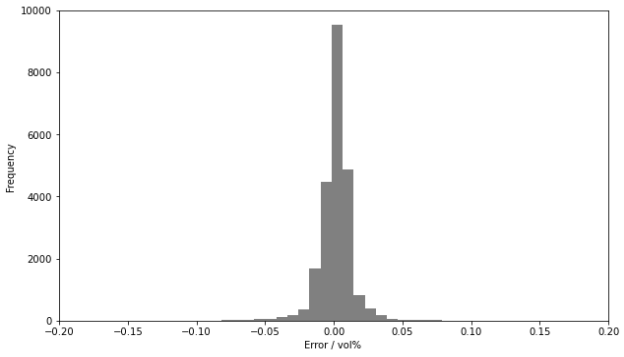

Fig. 12. *Histogram of the error values on the entire data for the LSTM model with ReLU, 33 LSTM units and 33-time steps*

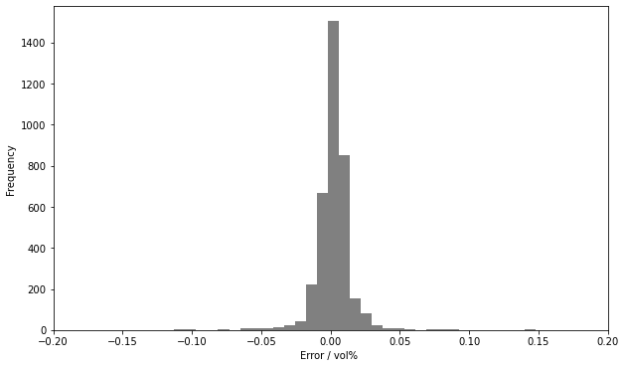

Fig. 13. *Histogram of error values on validation data for LSTM model with ReLU, 33 LSTM units and 33-time steps*

4. Conclusion

In the refinery's propane/propylene splitter plant, it is imperative to consistently monitor the propylene content in the product to enhance process efficiency. The propylene and propane separation process aims to achieve a minimum purity of 99.6 vol% in the propylene, making it advantageous for use in polymer production.

The deployment of a soft sensor for online product quality monitoring can improve process control and reduce expenses associated with the operation and maintenance of online analyzers.

The focus of this study is on the creation of models that incorporate neural networks, such as multi-layered perceptrons and long-short term memory networks. These models are utilized to monitor the propylene content in the product of the PPS plant. The models were created using the Python programming language in the integrated open-source development environment, Spyder.

Among the extensive range of models developed, two models were singled out as the most promising. One model proved to be highly effective in neural networks with multilayer perceptrons, while the other exhibited remarkable capabilities in neural networks with long-term memory. The model utilizing the tanh activation function and consisting of 17 neurons in the hidden layer was chosen as the optimal model among the MLP models because of its superior error stability. The model with the activation function ReLU and 33-time steps and 33 LSTM units in the hidden layer was selected as the best model from the group of LSTM models. Based on both graphical and numerical outcomes, the LSTM model demonstrates superiority as a result of its stronger correlations, reduced errors, and enhanced performance stability when adjusting the hyperparameters.

The most effective models of soft sensors in both categories delivered results that were more than satisfactory, showing similar correlation coefficients and errors in the model output data.

These models prove to be effective for application within the refinery information system. By incorporating these innovative soft sensors, the anticipated results include better control over processes and higher quality final products, ultimately resulting in substantial savings in production costs.

5. References

[1] Ž. Ujević Andrijić, N. Bolf, Primjena softverskih senzora za procjenu kvalitete produkata atmosferske destilacije, Goriva maziva, 50 (2011) 187-200.

[2] N. Bolf, Softverski senzori - alat suvremenog kemijskog inženjerstva, KUI, 60 (2011) 193-199.

[3] M. E. Khan, F. Khan, A comparative study of white box, black box and grey box testing techniques, Int. J. Adv. Comput. Sci. Appl., 3 (2012) 12-15.

[4] I. Ahmad, A. Ayub, M. Kano, I. I. Cheema, Gray-box Soft Sensors in Process Industry: Current Practice, and Future Prospects in Era of Big Data, Processes, 8 (2020) 243.

[5] S. Zendehboudi, N. Rezaei, A. Lohi, Applications of hybrid models in chemical, petroleum, and energy systems: A systematic review, Appl. Energy 228 (2018) 2539–2566.

[6] L. Fortuna, S. Graziani, A. Rizzo, M. G. Xibilia (2007), Soft Sensors for Monitoring and Control of Industrial Processes, London: Springer, 2007.

[7] V.A. Profillidis, G.N. Botzoris (2019) "Chapter 5 - Statistical Methods for Transport Demand Modeling", in: V.A. Profillidis, G.N. Botzoris (Ed.) Modeling of Transport Demand. Analyzing, Calculating, Forecasting Transport Demand, Oxford, Cambridge: Elsevier, 163-224.

[8] Ž. Ujević Andrijić, N. Bolf, Osvježimo znanje: Umjetne neuronske mreže, KUI, 68 (2019) 219-220.

[9] M. Uzair, N. Jamil, Effects of Hidden Layers on the Efficiency of Neural networks, in IEEE 23rd International Multitopic Conference (INMIC), Bahawalpur, Pakistan, 2020 1-6.

[10] S. Sharma, A. Athaiya, Activation functions in neural networks, Int. J. Eng. Appl. Sci. Tech. 4 (2020) 310-316

[11] E. Grossi, M. Buscema, Introduction to artificial neural networks, Eur. J. Gastroenterol. Hepatol. 19 (2008) 1046.-1054.

[12] D. R. Baughman, Y. A. Liu (1995), Neural networks in Bioproceesing and Chemical Engineering, San Diego: Academic Press, 1995.

[13] YL. Cong, LT. Hou, YC. Wu, YZ. Ma, Development of a Coupled EnergyPlus-MATLAB Simulation Based on LSTM for Predictive Control of HVAC System, Math. Probl, Eng., 2022 (2022) 5912967.

[14] S. M. Basha, D. S. Rajput, (2019) "Chapter 9 - Survey on Evaluating the Performance of Machine Learning Algorithms: Past Contributions and Future Roadmap", in: A. K. Sangaiah (Ed.) Deep Learning and Parallel Computing Environment for Bioengineering Systems, St. Louis: Academic Press, Elsevier, 153–164.

[15] A. Krogh, What are artificial neural networks?, Nat Biotechnol. 26 (2008) 195–197.