Estimation of pressure drop in a demister for multi-stage flash desalination process based on stacked long short-term memory (LSTM) neural networks

Jarakmol Dawkraja1, Weerawun Weerachapichasgul1, Wachira Daosud1 and Paisan Kittisupakorn1

1 Department of Chemical Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand
2 Division of Chemical Engineering, Department of Industrial Engineering, Faculty of Engineering, Naresuan University, Phitsanulok, Thailand
3 Department of Chemical Engineering, Faculty of Engineering, Burapha University, Chonburi, Thailand

E-mail: paisan.k@chula.ac.th

Keywords: LSTM, deep learning, estimation, multi-stage flash desalination process, pressure drop, demister

Abstract

The thermal desalination process such as a multi-stage flash desalination process (MSF) uses demisters as the separator between the flashed off vapor and the brine droplets in the flashing stage. The performance of the MSF plant depends on the quantity of fresh water produced. The separation efficiency depends on the pressure drops in the demister that influences the plant performance. This study proposes the application of Long Short-Term Memory (LSTM) neural networks for estimating pressure drop across demisters. The stacked LSTM algorithm is effective in estimating the pressure drop for experiment and real plant data. The superiority of stacked LSTM algorithm above reference benchmarks is evident. The Root Mean Square Error (RMSE) of the estimation from stacked LSTM algorithm is less than the estimation from the work of Al-Fulaij et al CFD model Al-Fulaij H, et al 2016, Desalination, 385: 148–157 by 40%.

1. Introduction

Demisters are commonly used in processes involving separation of micron scale droplets. They are mostly found in petroleum refining, pulp and paper, petrochemicals, dryers, scrubbers, flash drums, etc. Demister acts as a mist separator equipment. A porous blanket of wire separates the liquid droplets from vapor [1]. Wire materials such as stainless steel, monel, nickel, aluminium, copper, carbon steel, tantalum, polyethylene, fluoropolymer, and glass multifilament are common for being used as knitting meshes [2]. Even with the operation involving high pressure, the droplets sizing at least 10 μ are successfully removed by demisters [3]. When the mixing stream of liquid and vapor passes through demisters, the wire acts as an obstacle in the flow path. The smaller droplets and the vapor can move around the wire while the bigger droplets adhere to the wire by surface tension and coalesce with other droplets [4]. When the combined liquid droplet becomes heavier to overcome the surface tension and the drag force, it starts to detach from the wires [1].

With the fact that the packing density, void fraction and specific surface area are major characteristics of demisters. Vapor velocity is the major indicator for demisters’ performance [5]. When the vapor velocity is low, the momentum of liquid droplet is also low so that it drips through the wire and carries over with the vapor stream. With high vapor velocity, the momentum of liquid droplet is high and collides with the wire. Liquid droplet has no chance to flow through demister with vapor stream. Therefore, the efficiency of demister is increased. Originally, El-Dessouky et al [1] and Al-Dughaither et al [6] propose empirical models which are simple models for demister efficiency determination. Those empirical models relate to vapor and liquid physical properties, demister characteristics including wire diameter, demister height, and demister area [7]. From empirical models, the pressure drop positively depends on the vapor velocity and the packing density whereas it negatively depends on the wire diameter.
The experiments conducted by El-Dessouki et al. [1] base on vapor velocity $V_i$ (0.98–7.5 m s$^{-1}$), packing density (80.317–208.16 kg m$^{-3}$), demister thickness (100–200 cm), wire diameter $D_w$ (0.2–0.32 mm), and diameter of captured droplets $D_i$ (1–5 mm). El-Dessouky et al. correlation [1] for determining the separation efficiency $\eta_i$ is shown in equation (1) where $\rho_i$ is liquid density.

$$\eta_i = 17.5047(D_w)^{-0.28264}(\rho_i)^{0.099625}(V_i)^{0.106678}(D_i)^{-0.383197}$$

Even though, the empirical models are easy to use, the values of the empirical constants are varied from experiment to experiment. That is the reason why the empirical models are not well suited to describe the demister performance. The computational fluid dynamics models (CFD) is then introduced and proven to be effective estimating the demister performance. Rahimi and Abaspour simulated CFD model for wire mesh demisters range of 1–7 m s$^{-1}$, packing density of 200 kg m$^{-3}$, pad thickness 0.2 m and wire diameter of 0.31 mm. The estimation was closely to the experimental data and match the empirical model of El-Dessouky et al. at low vapor velocities [8]. Galletti et al. utilized either the Eulerian or Lagrangian approaches CFD model for wave-plate demister and the estimation was in close match to the experimental data [9]. For multi-stage flash (MSF) desalination process, Al-Fulaij et al. [10] using the Eulerian approach for vapor phase and the Lagrangian approach for the brine droplets modeled consistent estimation with experiment data. The operating conditions for Al-Fulaij et al. were velocity range (1.13–10.4 m s$^{-1}$), packing density (80.317–176.35 kg m$^{-3}$), thickness of demister pad (100–200 mm), wire diameter (0.2–0.32 mm), droplet diameter (10 micron-m), and volume fraction of inlet water droplets ($1 \times 10^{-5}$) [10]. The increased in vapor velocity makes the demister efficiency initially increased. However, the separation efficiency starts to drop when pasting the flooding limit. The demister efficiency varies inversely with pressure [11]. The value of pressure may estimate the efficiency of demister.

Both empirical model and CFD model are simple and yet provide reliable results, however, they are heavily relying on assumptions. Deep learning referred as several neural networks are tasked for handling non-linearity and it utilizes the plant data to generate the estimation without knowing plant characteristics. Recent research works show that the deep neural networks yield better performance for time series data [12]. The case study is the multi-stage flash desalination process utilizing wire mesh demisters to remove brine droplets from flashed off water vapor. In this work, stacked Long Short-Term Memory (LSTM) algorithm is proposed for estimating pressure drop from five distinct datasets [13]. The experimental data by El-Dessouky et al. [1] and the work of Al-Fulaij et al. [10] are used to evaluate models. The detail of LSTM-based neural network is discussed in section 2. The experimental results and discussion are presented in section 3. Conclusion is in section 4.

2. LSTM-based neural network

The Long Short-Term Memory (LSTM) network is recurrent neural network variant where the past data is in memory with three gates namely input, forget and output gates for controlling the flow of information [14]. The LSTM structured from recurrent neural network by Hochreiter and Schmidhuber in 1997 [15] is shown in figure 1.

![Figure 1. LSTM structure. (a) The forget gate; (b) The input gate; (c) The output gate.](image-url)
The sigmoid activation function, \( \sigma \), generates output at current time, \( f_t \), using previous hidden unit, \( H_{t-1} \), weight matrices, \( W_f, H \), bias, \( b_f \) and current input, \( x_t \) at forget gate as shown in equation (2). The output with the sigmoid activation function displays between zero and one. The gate is blocking when the value is zero while the gate is passing everything through when the value is one \[16\]. The forget gate determines which information will be used. Mostly, the key information is adopted while the useless information is dropped. The previous hidden unit, \( H_{t-1} \), weight matrices, \( W_i, H \), bias, \( b_i \) and current input, \( x_t \) are updated at input gate, \( i_t \) as shown in equation (3). The modulation gate \( \tilde{c}_t \) changes the previous state and the new input using a tanh activation function as shown in equation (4).

The current cell state, \( c_t \), is updated from the previous cell state, \( c_{t-1} \), and the current input cell, \( i_t \) as shown in equation (5). The previous cell state multiplies \( \otimes \) with the current output value from forget gate. That product adds to the updated current input cell with hyperbolic tangent (tanh) activating function to get the new current cell state value. The output gate, \( o_t \), uses sigmoid activation function as shown in equation (6). The hidden unit at time \( t \), \( H_t \), is calculated from equation (7). Sigmoid and hyperbolic tangent (tanh) activation functions are shown in equations (8) and (9) respectively.

\[
f_t = \sigma (W_f x_t + W_{Hf} H_{t-1} + b_f) \tag{2}
\]
\[
i_t = \sigma (W_i x_t + W_{Ht} H_{t-1} + b_i) \tag{3}
\]
\[
\tilde{c}_t = \tanh (W_c x_t + W_{Hc} H_{t-1} + b_c) \tag{4}
\]
\[
c_t = c_{t-1} \otimes (f_t) + (i_t) \otimes (\tilde{c}_t) \tag{5}
\]
\[
o_t = \sigma (W_o x_t + W_{Hc} H_{t-1} + b_o) \tag{6}
\]
\[
H_t = o_t \otimes \tanh (c_t) \tag{7}
\]
\[
\sigma = \frac{1}{1 + e^{-x}} \tag{8}
\]
\[
\tanh = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{9}
\]

Algorithm method is a combined several base models either in a parallel or a sequential manner \[12\]. In this study, only the stacking technique is mainly focused. When two LSTMs are trained simultaneously, these structures are called bidirectional Long Short-Term Memory (BiLSTM) \[17\]. Two or more LSTM trained in series are called stacked LSTM. For stacked LSTM, the output from the first LSTM layer is fed as the input for the second LSTM and so forth. The LSTM algorithm is shown as figure 2. The number of LSTM signifies the learning capacity. Therefore, the learning should be improved when adding number of LSTM to the network. The computation time is increased and the overfitting is detected \[18\].
Figure 3. BiLSTM architecture.

Table 1. The network architecture for LSTM, BiLSTM and stacked LSTM.

| LSTM | BiLSTM | Stacked LSTM |
|------|--------|--------------|
| Input | Input | Input |
| Batch Normalization Layer | Batch Normalization Layer | Batch Normalization Layer |
| LSTM Layer | LSTM Layer | LSTM Layer |
| Dropout Layer | Dropout Layer | Dropout Layer |
| Fully Connected Layer | Fully Connected Layer | Fully Connected Layer |
| Regression Layer | Batch Normalization Layer | Batch Normalization Layer |
| LSTM Layer | LSTM Layer | LSTM Layer |
| Dropout Layer | Dropout Layer | Dropout Layer |
| Fully Connected Layer | Fully Connected Layer | Fully Connected Layer |
| Regression Layer | Batch Normalization Layer | Batch Normalization Layer |

Figure 4. Stacked LSTM architecture.
Table 1 lists the network architecture for LSTM, BiLSTM and stacked LSTM. All inputs are preprocessed and are stabilized in minibatch before entering LSTM layer to accelerate the learning process. For LSTM algorithm as shown in figure 2, the network is trained using 80 hidden units, batch size of 25, adaptive moment estimation (Adam) optimizer, and learning rate of 0.0001. The output from the first LSTM layer is then fed into dropout layer (50%) and fully connected layer to avoid overfitting. Then, data extracted from the first LSTM layer pass through batch normalization layer to be stabilized. Thus, the learning process is accelerated before entering the second LSTM layer. The network is trained using 120 hidden units, batch size of 25, adaptive moment estimation (Adam) optimizer, and learning rate of 0.0001. The estimated output then is displayed after dropout layer (50%), fully connected layer, and regression layer.

For BiLSTM algorithm as shown in figure 3, all inputs are preprocessed and stabilized in minibatch before entering the first LSTM layer to accelerate the learning process. The network is trained using 80 hidden units, batch size of 25, adaptive moment estimation (Adam) optimizer, and learning rate of 0.0001. The output from the first LSTM layer is then fed into dropout layer (50%) and fully connected layer to avoid overfitting. Then, data extracted from the first LSTM layer pass through batch normalization layer to be stabilized. Thus, the learning process is accelerated before entering the second LSTM layer. The network is trained using 120 hidden units, batch size of 25, adaptive moment estimation (Adam) optimizer, and learning rate of 0.0001. The estimated output then is displayed after dropout layer (50%), fully connected layer, and regression layer.

The stacked LSTM algorithm as shown in figure 4, starts with preprocessed and stabilized input data to accelerate the learning process before entering the first LSTM training. Network is trained using 80 hidden units, batch size of 25, adaptive moment estimation (Adam) optimizer, and learning rate of 0.0001. The output from the first LSTM training enter dropout layer (50%) and fully connected layer to perform overfitting and data extraction. Before entering the second LSTM training, data are stabilized to accelerate the learning process. Network is trained using 100 hidden units, batch size of 25, adaptive moment estimation (Adam) optimizer, and learning rate of 0.0001. After training, data flow into dropout layer (50%) to avoid overfitting and fully connected layer to extract data. Then, before entering the third LSTM training, data are stabilized to accelerate the learning process. Network is trained using 130 hidden units, batch size of 25, adaptive moment estimation (Adam) optimizer, and learning rate of 0.0001. The estimated output is displayed after dropout layer (50%), fully connected layer, and regression layer.

Two common metrics used to measure accuracy are Mean absolute error (MAE) and Root mean squared error (RMSE). The lower value of these metrics means a model with higher performance. Mean absolute error (MAE) as shown in equation (10) where \( y_{i}^{\text{target}} \) and \( y_{i}^{\text{predicted}} \) are the target and estimated values, respectively with a total of \( N \) observations. The MAE measures how close calculated values are to the target values [18]. The average magnitudes of the errors are only measured for the MAE. Mean squared error (MSE) and root mean square error (RMSE) as shown in equations (11) and (12) average the squared errors and take a square root of the squared errors, respectively. The MSE and RMSE are closed to zero if the model is well estimated [19].

\[
\text{MAE} = \frac{1}{N} \sum_{i=1}^{N} |y_{i}^{\text{target}} - y_{i}^{\text{predicted}}| 
\]

\[
\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (y_{i}^{\text{target}} - y_{i}^{\text{predicted}})^2 
\]

\[
\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_{i}^{\text{target}} - y_{i}^{\text{predicted}})^2} 
\]

All proposed LSTM algorithms are performed using MATLAB and Deep Learning Toolbox Release 2021b, The MathWorks, Inc., Natick, Massachusetts, United States. All computations are calculated on a laptop PC with 1.80 GHz Intel Core i7 processor and 16 GB RAM. The input data is normalized for each batch before training to improve the generalization ability and convergence speed [19].

Each LSTM model is trained for a total of 300 epochs in which one epoch is equal to \( \frac{\text{[training set]}}{\text{[mini-batch size]}} = \frac{1000}{25} \approx 40 \). Consequently, 300 epochs equals to 12000 training iterations [20]. The mean (\( \mu \)) and variance (\( \sigma^2 \)) are first calculated as in equations (13) and (14), respectively. Then each sample is normalized as in equation (15) where \( \varepsilon \) is a constant for numerical stability. The output is updated with normalized input for each neural network layer as shown in equation (16) where \( \gamma \) and \( \beta \) are valued from optimization process [21].

\[
\mu = \frac{1}{m} \sum_{m} x_i 
\]

\[
\sigma^2 = \frac{1}{m} \sum_{m} (x_i - \mu)^2 
\]

\[
\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \varepsilon}} 
\]
After data have been normalized, data are split into 90% training and 10% testing. The weight and bias are adjusted accordingly so that the error from the model is minimized during training. The model is verified during testing. The reason behind this 90% training is that data used in this study are limited and highly imbalanced \[22\]. The process is also run 10 times to shape the ten-fold cross-validation so that the test error rate estimation is not suffered from high bias and high variance \[23\]. The hyperparameters such as the minibatch size \[24\], the number of hidden units, and the dropout percentage (50%) affect performance of a model. The gradient threshold is fixed at two to avoid the gradient exploding. All LSTM algorithms are trained with the adaptive

\[
\gamma_i = \gamma \hat{x}_i + \beta_i
\]

(16)
Figure 7. All LSTM algorithms estimation for MSF-Brine circulation plant operating at low temperature [13].

Figure 8. All LSTM algorithms estimation for MSF-Brine circulation plant operating at high temperature [13].

Figure 9. All LSTM algorithms estimation for MSF-Once Through plant operating at low temperature [13].
moment estimation (Adam) optimizer. The experiment results are compared to El-Dessouky empirical correlation [1] and Al-Fulaij et al CFD model [10]. The performance metrics are RMSE and MAE.

3. Experimental results and discussion

This section presents experimental results that examine the effectiveness of stacked LSTM algorithms to estimate pressure drop. In this study, we focus on the pressure drop estimation from five different sets of data. Two datasets from experiment contain vapor velocity at a specific packing density and a wire diameter [1]. The other three sets of data collected from real plant at two mode of operations which are a brine circulation at low and high temperature, and a once-through operation only at low temperature [13]. The different conditions of the El-Dessouky et al experiments [1] and real MSF plant data are employed to verify the effectiveness of our proposed models as shown in table 2.

All LSTM algorithms estimate closely identical to the real value comparing to the El-Dessouky empirical correlation [1] and the Al-Fulaij et al CFD model [10]. The estimations achieved by each model are shown in figure 5 for the packing density of 80.317 kg m\(^{-3}\), the vapor velocity in the range of 2.435–6.696 m s\(^{-1}\), and the wire diameter of 0.28 mm. The pressure drop estimation shown in figure 6 is for the packing density of 176.350 kg m\(^{-3}\), the vapor velocity in the range of 1.422–4.711 m s\(^{-1}\), and the wire diameter of 0.24 mm. The real plant operation estimations for Brine circulation plant operating at low temperature, at high temperature, and Once-through operating at low temperature achieved by each model are shown in figure 7–9, respectively.
The MAE performance comparison for all dataset is shown in figure 10. For dataset 1, estimation errors are less than the El-Dessouky empirical correlation [1] as much as 57%, 58% and 82% for LSTM, BiLSTM, and stacked LSTM, respectively. Estimation errors are less than the Al-Fulaij et al CFD model [10] as much as 2.62%, 5.03% and 5.9% for LSTM, BiLSTM, and stacked LSTM, respectively.

For dataset 2, estimation errors are less than the El-Dessouky empirical correlation [1] as much as 76%, 88.8% and 89% for LSTM, BiLSTM, and stacked LSTM, respectively. Estimation errors are less than the Al-Fulaij et al CFD model [10] as much as 38%, 70% and 71% for LSTM, BiLSTM, and stacked LSTM, respectively.

For dataset 3, estimation errors are less than the El-Dessouky empirical correlation [1] as much as 83%, 85% and 85.2% for LSTM, BiLSTM, and stacked LSTM, respectively. Estimation errors are less than the Al-Fulaij et al CFD model [10] as much as 28%, 36% and 36.4% for LSTM, BiLSTM, and stacked LSTM, respectively.

For dataset 4, estimation errors are less than the El-Dessouky empirical correlation [1] as much as 87%, 91% and 93% for LSTM, BiLSTM, and stacked LSTM, respectively. Estimation errors are less than the Al-Fulaij et al CFD model [10] as much as 43%, 63% and 71% for LSTM, BiLSTM, and stacked LSTM, respectively.

For dataset 5, estimation errors are less than the El-Dessouky empirical correlation [1] as much as 91%, 92% and 93% for LSTM, BiLSTM, and stacked LSTM, respectively. Estimation errors are less than the Al-Fulaij et al CFD model [10] as much as 14%, 24% and 29% for LSTM, BiLSTM, and stacked LSTM, respectively.

The RMSE performance metric for all dataset is shown in figure 11. For dataset 1, estimation errors are less than the experimental El-Dessouky et al [1] as much as 66%, 67% and 85% for LSTM, BiLSTM, and stacked LSTM, respectively. Estimation errors are less than the Al-Fulaij et al CFD model [10] as much as 2.24%, 5.79% and 5.7% for LSTM, BiLSTM, and stacked LSTM, respectively.

For dataset 2, estimation errors are less than the El-Dessouky empirical correlation [1] as much as 76%, 88% and 89% for LSTM, BiLSTM, and stacked LSTM, respectively. Estimation errors are less than the Al-Fulaij et al CFD model [10] as much as 38%, 67% and 71% for LSTM, BiLSTM, and stacked LSTM, respectively.

For dataset 3, estimation errors are less than the El-Dessouky empirical correlation [1] as much as 80%, 80.3% and 83% for LSTM, BiLSTM, and stacked LSTM, respectively. Estimation errors are less than the Al-Fulaij et al CFD model [10] as much as 45%, 45.9% and 54% for LSTM, BiLSTM, and stacked LSTM, respectively.

For dataset 4, estimation errors are less than the El-Dessouky empirical correlation [1] as much as 86%, 92% and 93% for LSTM, BiLSTM, and stacked LSTM, respectively. Estimation errors are less than the Al-Fulaij et al CFD model [10] as much as 62%, 78% and 82% for LSTM, BiLSTM, and stacked LSTM, respectively.

For dataset 5, estimation errors are less than the El-Dessouky empirical correlation [1] as much as 91.1%, 91.3% and 91.4% for LSTM, BiLSTM, and stacked LSTM, respectively. Estimation errors are less than the Al-Fulaij et al CFD model [10] as much as 2%, 5% and 6% for LSTM, BiLSTM, and stacked LSTM, respectively.

4. Conclusion

In this paper, the stacked Long Short-Term Memory (LSTM) algorithm estimating the pressure drop across demisters for the Multi-stage flash desalination process (MSF) is proposed. There are five datasets used in this study where two sets are from El-Dessouky et al experiments [1] and the rest are from real operating plant [13]. We select a high accuracy, reliable correlation, and a computational fluid dynamic model (CFD) as benchmarks to verify the model effectiveness. The algorithm of stacked Long Short Term Memory (LSTM) performs better than the Al-Fulaij et al CFD model [10] and El-Dessouky et al Correlation [1]. Since the structure of stacked LSTM where previous time steps influence current time step so that it is suited for dynamic estimating [25]. In addition, the data is precisely estimated with stacking LSTM algorithm. Moreover, the accuracy is generally improved as the size of LSTM training increases; however, overfitting issue required tends to be higher with a greater number of LSTM training layers [26]. The overfitting issue can be avoided by implementing the dropout layer into stacked LSTM architecture. The Root Mean Square Error (RMSE) of the estimation from stacked LSTM algorithm is less than the estimation from the work of Al-Fulaij et al CFD model [10] and the El-Dessouky et al Correlation [1] by 40% and 88%, respectively.

Acknowledgments

The authors would like to acknowledge the supports provided by Department of Chemical Engineering, Faculty of Engineering, Chulalongkorn University under the 60/40 scholarship and Intelligent Control Automation of Process System Laboratory co-funded by the Eramus + Programme of the European Union, project number: 609854-EPP-1-2019–1-FR-EPPKA2-CBHE-JP.
Data availability statement

The data that support the findings of this study are available upon reasonable request from the authors.

Credit authorship contribution statement

Jarukamol Dawkrajai: Conceptualization, Data processing, Methodology, Validation, Writing—original draft, Approving the submitted version. Weerawun Weerachapichasgul: Investigation, Writing—review, Approving the submitted version. Wachira Daosud: Resources, Writing—review, Approving the submitted version. Paisan Kittisupakorn: Data analysis, Conclusion statement discussion, Supervision, Funding acquisition, Manuscript review and correction, Approving the submitted version.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

ORCID iDs

Jarukamol Dawkrajai https://orcid.org/0000-0003-4014-8571

References

[1] El-Dessouky H T et al 2000 Performance of wire mesh mist eliminator Chem. Eng. Proces. Intensif. 39 129–39
[2] Kayode Coker A 2007 Ludwig s applied process design for chemical and petrochemical plants Chp. 6—Mechanical Separations. vol 1 4th (Amsterdam: Gulf Professional Publishing) 371–443 (https://www.sciencedirect.com/book/9780730677660/ludwigs-applied-process-design-for-chemical-and-petrochemical-plants)
[3] Calvert S 1970 Venturi and other atomizing scrubbers efficiency and pressure drop AIChE J. 16 392–6
[4] Poppele E W 1958 Correlation of maximum air velocity with liquid entrainment loading for wire mesh mist eliminators Thesis Submitted to Newark College of Engineering New Jersey Institute of Technology Newark, New Jersey, USA (http://archives.njit.edu/vol01/etd/1950s/1958/njit-etd1958-001/njit-etd1958-001.pdf)
[5] Kazi S, Umale S, Venkatesh M and Neve V 2021 Application of various correlations for design of demisters to study the effect of geometrical parameters on its performance Mater. Today Proc. 38 3035–40
[6] Al-Dughaiter A A, Ibrahim A A and Al-Masy w A 2011 Investigating pressure drop across wire mesh mist eliminators in bubble column J. Saudi Chem. Soc. 15 1–9
[7] Bruazzi E and Paglianti A 1988 Design of wire mesh mist eliminators AIChE J. 44 505–12
[8] Rahimi R and Abbaspour D 2008 Determination of pressure drop in wire mesh mist eliminator by CFD Chem. Eng. Proces. 47 1504–8
[9] Galletti C, Bruazzi E and Tognotti L 2008 A numerical model for gas flow and droplet motion in wave-plate mist eliminators with drainage channels Chem. Eng. Sci. 63 (23) 5639–52
[10] Al-Fulaij H et al 2016 Eulerian—Lagrangian modeling and computational fluid dynamics simulation of wire mesh demisters in MSF plants Desalination 385 148–57
[11] Kazi S et al 2021 Application of various correlations for design of demisters to study the effect of geometrical parameters on its performance Mater. Today Proc. 38 3035–40
[12] Farsi M 2021 Application of ensemble RNN Deep neural network to the fall detection through IoT environment Alex. Eng. J. 60 199–211
[13] Al-Fulaij H F 2011 Dynamic Modeling of Multi Stage Flash (MSF) Desalination Plant, PhD Dissertation Unpublished to Department of Chemical Engineering University College London London, UK (https://discovery.ucl.ac.uk/id/eprint/1324506/)
[14] Bao W, Yue J and Rao Y 2017 A deep learning framework for financial time series using stacked autoencoders and long-short term memory PloS One 12
[15] Hochreiter S and Schmidhuber J 1997 Long short-term memory Neural Comput. 9 (Cambridge, MA: MIT Press) 1735–80
[16] Chohan M et al 2020 Image captioning using deep learning: a systematic literature review Int. J. Adv. Comput. Sci. Appl. 11 5
[17] Dandil E and Karaca S 2021 Detection of pseudo brain tumors via stacked LSTM neural networks using MR spectroscopy signals Biocybern Biomed. Eng. 41 173–95
[18] Bianchi F M, Maiorino E, Kampffmeyer M C, Rizzi A and Jenessen R 2017 Recurrent Neural Networks for Short Term Load Forecasting An Overview and Comparative Analysis 1st edn (Berlin: Springer) (https://doi.org/10.1007/978-3-319-70338-1)
[19] Adhikari R 2015 A neural network based linear ensemble framework for time series forecasting Neurocomputing 157 231–42
[20] Joffe S and Szegedy C 2015 Batch normalization: accelerating deep network training by reducing internal covariate shift arXiv:1502.03167
[21] Borovkova S and Tsimas I 2019 An ensemble of LSTM neural networks for high-frequency stock market classification J. Forecast. 38 (6) 600–19
[22] Fang Z, Wang Y, Peng L et al 2021 Predicting flood susceptibility using LSTM neural networks J. Hydrol. 594 125734
[23] Karimipour H and Derakhshian F 2021 AI-enabled threat detection and security analysis for industrial IoT Springer Nature 151 (Berlin: Springer)
[24] James G, Witten D, Hastie T and Tibshirani R 2013 An introduction to statistical learning: with applications in R Springer Texts in Statistics 184 2nd (Berlin: Springer) (https://web.stanford.edu/~hastie/ISLR2/ISLRv2_website.pdf)
[25] Xia J et al 2019 A long short-term memory algorithm approach for improving the outcome estimation in intensive care unit Comput Math Methods Med. 2019 8152713
[26] Huang W et al 2007 Neural networks in finance and economics forecasting Int. J. Inf. Technol. Decis. Mak. 6 113–40