Int. J. Adv. Res. Sci. Technol. Volume 15, Issue 8, 2026, pp. 2516-2525.

IJARST

International Journal of Advanced Research in Science and Technology

https://doi.org/10.62226/ijarst20262765

ISSN 2319 – 1783 (Print)

ISSN 2320 – 1126 (Online)

Modeling of the Vacuum Distillation Unit of a Refinery Using Attention-Based and Conventional Neural Network Approaches

AKPAN Nseabasi M.[1] and EGEMBA Kingsley C.[2*]

[1,2]Department of Chemical Engineering, University of Uyo, Uyo. Nigeria

****Email:** kingsleyegemba@uniuyo.edu.ng*

ARTICLEINFO

Article history:

Received	16 Aug 2026
Accepted	27 Aug 2026
Available online	31 Aug 2026

Keywords:

Vacuum Distillation Unit, Neural Network Modeling, Attention-Based Mechanism, Vacuum Distillation P roduct Prediction, Model Interpretability.

Indexed in:

and in major libraries

ABSTRACT

The vacuum distillation unit (VDU) of a West African refinery was modeled using conventional Feedforward Artificial Neural Network (FF-ANN) and Attention-Based Artificial Neural Network (Attention-ANN) approaches, and the effectiveness of the models in predicting critical product variables were evaluated. Using preprocessed plant data, the models were trained with 14 input process variables and six process outputs. The performance of the models was evaluated using the coefficient of determination (R^2), root mean square error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), and Pearson's correlation coefficient (R), while paired samples t-test was performed over 20 repeated runs. Both models achieved high predictive accuracy, with the Attention-ANN demonstrating superior overall performance (R^2 = 0.9970, RMSE = 6.3080, MAE = 4.5341, and MAPE = 4.27%) compared with the FF-ANN (R^2 = 0.9853, RMSE = 14.0252, MAE = 5.2037, and MAPE = 7.92%), while also achieving a slightly better performance in predicting five of the six individual output variables. The observed differences between the predictions of the models were not statistically significant, though the attention mechanism enhanced model interpretability by identifying the column bottom pressure as the most influential process variable. The predictive performance achieved compared favorably with reported refinery modelling studies for VDU modeling. These findings indicate that attention-based neural networks represent an effective and interpretable framework for multi-output VDU modelling.

INTRODUCTION

Africa's population is estimated to double by 2050 with a growing energy demand that could more than double in that time ([1]; [2]). Fossil fuels are expected to power up to 60% of Africa's energy by 2040, and Africa's local crude oil refineries are crucial in addressing the increasing energy demands of the fast-growing developing continent [3]. In West Africa, new refineries are being built while existing refineries are being revamped to offset importation of crude oil products and meet rising demand of a rapidly growing populace. Recent projects in the region include; the Dangote Refinery (Nigeria) - the largest in Africa, and the Sentuo Oil Refinery in Ghana, which are now operational at capacities of 650 bbl/d/1k and 45 bbl/d/1k respectively [4].

Indeed globally, fossil-based fuels remain the leader in energy consumption despite the detrimental environmental impact and incentives in the form of carbon credits to reduce carbon emission [5]. Arguably, the solution to a sustainable and clean energy sufficiency lies on improving efficiencies in the production and use of non-renewable energy sources in the short to medium term, of which fossil fuels is the leader, while working to advance technologies, build infrastructure, establish supply chains, and fine-tune regulations and policies for expanding the production and use of renewable clean energy sources in the medium to long-term. In this scenario, local crude oil refineries are essential in meeting the energy needs of our growing continent, and it is important to operate them efficiently.

Refining crude oil relies heavily on crude oil distillation, a critical separation process that transforms crude oil into valuable finished products [6]. Among the distillation processes, vacuum distillation plays a vital role in the use of reduced pressure to separate heavier
crude oil fractions that would otherwise undergo thermal cracking at atmospheric conditions [7]. This process is particularly crucial for producing feedstocks for downstream conversion units like fluid catalytic cracking and hydrocracking [8]. Therefore, the vacuum distillation unit (VDU) is essential in refinery operations to maximize the recovery of valuable products in alignment with the refinery's profitability goals. However, the complex nature of vacuum distillation, characterized by nonlinear dynamic relationships between process variables and product yields, poses significant challenges for effective control and optimization ([6], [9]). Consequently, there has been a growing interest in advanced modeling techniques, such as machine learning (ML), that can accurately capture the intricacies of this process [10].

In the field of ML, artificial neural networks (ANNs), have emerged as a powerful tool for modeling complex industrial processes due to their ability to learn nonlinear relationships from data, and have demonstrated their ability to represent the complex dynamics of systems and to shorten computing times considerably [11]. Additionally, attention-based ANNs represent a significant advancement in their ability to dynamically focus on the most relevant parts of the input data unlike traditional ANNs that treat all input features equally during training. This is especially useful in complex processes like vacuum distillation, where certain operating parameters may have a stronger impact on outputs under specific conditions. By integrating attention into the ANN architecture, the model can learn to assign different weights to inputs based on their impact, leading to more accurate predictions. The present work applies both approaches to model the interactions between process variables and product outputs in a vacuum distillation unit, while comparing their performances.

METHODOLOGY

Overview

Two neural network architectures were developed to model the VDU. A systematic cleaning protocol was implemented to clean raw operational plant data, to remove out of stable operating conditions as well as observations with zero or missing values [12]. A total of about 78 cleaned plant data, were then used in the work. To ensure robust model evaluation and prevent data leakage, the cleaned datasets were split into three subsets using a two-stage random splitting approach [13]. The same sampling and splitting strategies were applied to both models. 70% of total data (54 samples) were used for training in model optimization and weight updates, 10% of total data (8 samples) were used for validation in hyperparameter monitoring and early stopping, while 20% of total data (16 samples) were used for final testing. The models had Fourteen (14) input process variables and six (6) target output variables (see Table 1). In both models, input features and target variables were scaled independently, using the Standard Scaler functions which compute the Z-score normalization as in Equations 1 and 2. Where X_{scaled} and Y_{scaled} are the scaled input and output values, and μ_X, σ_X and μ_Y, σ_Y represent the mean and standard deviation of the inputs *(X)* and outputs *(Y)* respectively. The models were developed in the Python programming (version 3.13) environment.

$$X_{scaled} = \frac{X-\mu_X}{\sigma_X} \quad \text{Equation 1}$$

$$Y_{scaled} = \frac{Y-\mu_Y}{\sigma_Y} \quad \text{Equation 2}$$

Description of the modeled VDU

The VDU studied was designed to operate under vacuum conditions to facilitate the recovery of LVGO and HVGO from atmospheric residue. The VDU furnace is operated to transfer feed (atmospheric residue) to the column at about a temperature of 380°C. This temperature ensures maximum vaporization of the feedstock while preventing the degradation of heavy hydrocarbons into coke and non-condensable gases. To achieve separation at this temperature, the column maintains an average bottom pressure of about 45 mmHg. The vacuum condition is sustained by a dedicated overhead ejector system, consisting of multiple stages of steam ejectors and condensers. The column employs a hybrid internal configuration consisting of nine trays integrated with structured packing in specific sections. With very little overhead liquid to provide reflux at the top of the column, the VDU relies heavily on multiple pump-around circuits along the height of the column to remove heat and generate internal reflux for separation. A schematic of the VDU column is shown in Figure 1.

Table 1:Variables used in modeling

Input variables	Descripotion
feed_flow	Feed flow rate
feed_temp	Feed temperature
flash_zone _temp	Flash zone temperature
bottom_pr	Bottom column pressure
overhead_temp	Overhead condenser temperature
tray_8_temp	Temperature at tray 8
tray_2_temp	Temperature at tray 2
tray_1_temp	Temperature at tray1
lvgo_pa_flow	Light vacuum gas oil pump-around flow rate
lvgo_pa temp	Light vacuum gas oil pump-around temperature
lvgo_split	Light vacuum gas oil split flow rate
hvgo_pa_flow	Heavy vacuum gas oil pump-around flow rate
hvgo_pa_temp	Heavy vacuum gas oil pump-around temperature
vr_cooling_stream	Vacuum residue cooling stream temperature
Output target Variables	**Description**
lvgo_flow	Light vacuum gas oil flow rate
hvgo_flow	Heavy vacuum gas oil flow rate
vr_flow	Vacuum residue flow rate
lvgo_temp	Light vacuum gas oil temperature
hvgo_temp	Heavy vacuum gas oil temperature
vr_temp	Vacuum residue temperature

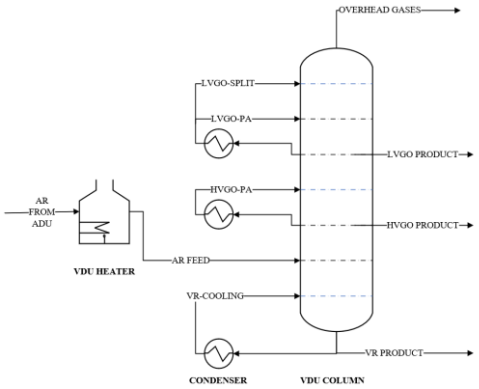

Figure 1: Schematic of the modeled VDU

Development of Models

FF-ANN Model Architecture

The baseline FF-ANN ([14]) employed a configurable architecture optimized via random search [15]. The optimal configuration obtained from the search and used in this study was 64 hidden dimension, 3 hidden layers, and dropout of 0.1. In each hidden layer, a sequence of transformations was implemented in the following order; Linear transformation, Layer normalization, Rectified Linear Unit ($ReLU$) activation function and dropout. The linear transformation is shown in Equation 3,

$$z_l = W_l h_{l-1} + b_l \quad \text{Equation 3}$$

where z_l is the pre-activation vector, of the input vector $h_{l-1}$from the previous layer, while W_l and b_l are the learnable weight matrix and bias vector for the layer l respectively. To stabilize the training, layer normalization was applied directly to the linear output and the normalized vector was computed as in Equation 4,

$$LayerNorm = \hat{z}_l = \gamma \odot \left(\frac{z_l - \mu}{\sqrt{\sigma^2 + \varepsilon}}\right) + \beta \quad \text{Equation 4}$$

where z_l is the linear transformation from the previous layer, μ and σ are the mean and standard deviation computed across all the hidden units for the specific sample, ε is a small constant (epsilon = 10^{-5}) for numerical stability, while γ and β are the learnable scaling and shift parameters.

This was followed by the activation function – ReLU, which introduces non-linearity into the network by setting all the negative pre-activation values to zero and leaving the positive values unchanged. Dropout was the final operation before the output layer; it acted as regularization by randomly zeroing-out 10% of active neurons during training to inhibit co-adaptation. The transformation in each layer i is represented by Equation 5,

$$h_i = Dropout\ (ReLU\ (LayerNorm(W_i h_{i-1} + b_i))) \quad \text{for } i = 1, \ldots, L \quad \text{Equation 5}$$

where h_i are the predicted output vectors of hidden layers, W_i are the learnable weights, b_i are the bias vectors, and L is the number of hidden layers.

In the output layer, the final hidden representation is transformed to the 6-dimensional output space, to produce the six (6) target variables of the predicted vector, using Equation 6.

$$\hat{y} = W_{i+1} h_L + b_{i+1} \quad \text{Equation 6}$$

Where $\hat{y}$ is the predicted vector, W_{i+1} and b_{i+1} represent the weights and biases of output layer, and h_L is the final transformation from the last hidden layer.

Attention-ANN Model Architecture

The Attention-ANN model ([16]) implements a flattened attention architecture comprising; feature embedding, multi-head self-attention, residual normalization, flattening, and forward pass regression layers. Each input variable is embedded as a feature token, allowing the self-attention mechanism learn complex interdependencies between process variables prior to regression. In the attention mechanism, each neuron's output is a weighted sum of the input values. The input layer vectors x_i were projected to the embedding layer and transformed using Equation 7,

$$e_i = W_e x_i + b_e \quad \text{Equation 7}$$

where e_i is the embedding vector for feature i, W_e is the learnable weight matrix, b_e is the bias vector, x_i is the scalar value of i-th input feature. The resulting output is the embedded tensor (E) (Equation 8), formed by applying this transformation to all the input features.

$$E = \begin{bmatrix} e_1 \\ e_2 \\ . \\ . \\ . \\ e_F \end{bmatrix}$$

The core of the attention mechanism lies in the multi-head self-attention layer, where each head learns the different relationships between the features. For each head (h), the embedded tensor E is projected into query (Q), key (K), and value (V) matrix representations through the independent linear transformations in Equations 9, 10 and 11.

$$Q_h = EW_Q^{(h)} \quad \text{Equation 9}$$

$$K_h = EW_K^{(h)} \quad \text{Equation 10}$$

$$V_h = EW_V^{(h)} \quad \text{Equation 11}$$

Where Q_h, K_h, V_h are the query, key, and value matrices for attention head h, $W_Q^{(h)}, W_K^{(h)}, W_V^{(h)}$ are the learnable projection matrices. Computed similarity score between each pair of embedded feature vectors were normalized via the Softmax probability function to produce attention weight combinations of the value vectors, The normalized attended weights were expressed as in Equation 12,

$$Attention_h\ (Q, K, V) = Softmax\left(\frac{Q_h {K_h}^T}{\sqrt{d_K}}\right) V_h \quad \text{Equation 12}$$

where $Attention_h\ (Q, K, V)$ is the attention-score weighted sum of value vectors, $\frac{Q_h {K_h}^T}{\sqrt{d_K}}$ are the attention scores, ${K_h}^T$is the transpose of the key matrix, and d_K is the dimension per head. The scaling factor $\sqrt{d_K}$ prevents the dot products from growing too large, maintaining stable gradients. The outputs from all the attention heads (H) were then concatenated and linearly projected using the output matrix in Equation 13,

$$MultiHead(E) = Concat(Attention_1, \ldots, Attention_H) * W_o \quad \text{Equation13}$$

where $Multihead(E)$ is the attention-enhanced feature embeddings that are subsequently passed through residual connection and layer normalization, flattening and specialized regression branches and W_O is the output projection matrix. To stabilize training and prevent information loss, residual connection and normalization were applied across the embedding dimension using Equations 14 and 15 respectively.

$$E_{residual} = E + Dropout\big(MultiHead(E)\big) \quad \text{Equation 14}$$

$$E_{norm} = LayerNorm = \gamma \odot \left(\frac{E_{residual} - \mu}{\sqrt{\sigma^2 + \varepsilon}}\right) + \beta \quad \text{Equation 15}$$

where μ and σ are the mean and standard deviation computed across the embedding dimension, ε is a small constant (epsilon = 10^{-5}) for numerical stability, γ is the learnable scaling parameter, and β is the learnable shift parameter. where μ and σ are the mean and standard deviation computed across the embedding dimension, ε is a small constant (epsilon = 10^{-5}) for numerical stability, γ is the learnable scaling parameter, and β is the learnable shift parameter.

The attention-weighted feature representation tensor was flattened into a single vector (x'_{flat}) as in Equation 16,

$$x'_{flat} = Flatten(E_{norm}) \qquad \text{Equation 16}$$

where x'_{flat} is the flattened vector representing the state of the real process, and E_{norm} is the normalized residual.

The flattened representation was then passed through the dense feed-forward layer with ReLU activation to produce predicted target outputs defined by Equation 17.

$$\hat{y} = W_2\,(Dropout\,(ReLU(W_1 * x'_{flatten} + b_1) + b_2$$

Equation 17

Where W_1, b_1 and W_2, b_2, are the weights and biases of the hidden layer and output layer respectively.

Models Training

To ensure fair comparison, between the models, the same dataset, train/validation/test splits, and random seed were used for the training of both models. Training was standardized across models using identical batch size, optimizer, early stopping criteria, and learning rate scheduler, following the recommendations of Masters and Luschi [17]. The normalization layers were initialized with unit weight and zero bias. For the feed forward linear layers, before the ReLU activation operation, the learnable weight vectors were initialized using Kaiming uniform initialization, as expressed in Equation 18

$$W \sim U\left(-\sqrt{\frac{6}{n_{in}}}, \sqrt{\frac{6}{n_{in}}}\right) \qquad \text{Equation18}$$

where W is the learnable weight sampled from a uniform distribution (U), n_{in} is the number of input features to the layer, and the associated biases were initialized to zero. For the attention mechanism, learnable weight matrices of the query, key, and value projection layers before the Softmax activation operation were initialized using the Xavier initialization, expressed in Equation 19.

$$W \sim U\left(-\sqrt{\frac{6}{n_{in}+n_{out}}}, \sqrt{\frac{6}{n_{in}+n_{out}}}\right) \qquad \text{Equation 19}$$

Where n_{in} is the number of input features to the layer, n_{out} is the number of output features from the layer, weight (W) is the learnable weight matrix sampled from a uniform distribution (U), and the corresponding biases were initialized to zero. After initialization, the network weights were iteratively optimized using the backpropagation algorithm in conjunction with the Adam optimization algorithm to minimize the Mean Squared Error (MSE) loss between the predicted and actual target outputs, defined by Equation 20,

$$\mathcal{L} = \frac{1}{Nm}\sum_{i=1}^{N}\sum_{j=0}^{m}(y_{ij} - \hat{y}_{ij})^2 \quad \text{Equation 20}$$

where $\mathcal{L}$ is the MSE loss, y_{ij} is the true value of output j for sample i, $\hat{y}_{ij}$ is the corresponding predicted value, N is the number of training samples in the mini-batch, and m is the number of output variables.

Backpropagation computed the gradient of the loss function for every learnable parameter by applying the chain rule shown in Equation 21 through the network.

$$g_t = \frac{\partial \mathcal{L}}{\partial W_t} = \frac{\partial \mathcal{L}}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial W_t} \qquad \text{Equation 21}$$

Where, g_t is the gradient of the MSE loss at iteration t, $\mathcal{L}$ is the MSE of the loss function, W_t is the learnable weight matrix in the embedding, attention, or feed-forward layers, and $\hat{y}$ is the network's predicted value.

The Adam optimizer uses the gradient descent with its first and second order moment estimate to calculate the weights. This process is repeated until the validation loss no longer improves or the maximum number of epochs is reached. The final weights were updated according to Equation 22.

$$W_{t+1} = W_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon} \qquad \text{Equation 22}$$

Where W_{t+1} is the next weight to be computed, W_t is the current weight from iteration t, $\hat{m}_t, \hat{v}_t$ are corrected first and second order moment estimate at iteration t, α is the training rate, and ε is a small positive number to maintain numerical stability during optimization. In both models, random search strategy ([18]) was employed for hyperparameter tuning using batch size 8. The random search explored 50 random combinations from a defined hyperparameter grid [19].

Evaluation Metrics

To assess the specific performance of each variable, metrics were calculated for each individual output (6 target variables). Overall aggregated metrics were also calculated for global assessment. All metrics were computed on inverse-transformed predictions to maintain interpretability in original units. The metrics used for evaluation and comparison were, coefficient of determination (R^2), root mean square error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), Pearson correlation coefficient (R), repeated measures ANOVA, and attention heat map to examine the interpretability of the Attention-ANN model. A key advantage of the Attention-ANN architecture is its inherent interpretability through attention mechanisms. The multi-head attention mechanism produces weights representing feature-to-feature relevance for each batch sample and head. These weights were averaged across batches and heads to obtain a consolidated attention matrix, which is visualized as a heatmap. The upper triangle of the heatmap is masked to avoid redundancy, and color intensity and numerical annotations indicated attention strength, revealing which process variables the model considers most interrelated when generating predictions. The metrics were computed as presented in Equations 23, 24, 25, 26 and 27.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \qquad \text{Equation 23}$$

Where y_i represents the actual target values, $\hat{y}_i$ represents the model's predicted target values, $\bar{y}$ represents the mean of the actual target values, and n represents the size of test

data

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad \text{Equation 24}$$

Where y_i represents the actual target values, $\hat{y}_i$ represents the model's predicted target values, and n represents the size of test data.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \quad \text{Equation 25}$$

Where y_i represents the actual target values, $\hat{y}_i$ represents the model's predicted target values, and n represents the size of test data.

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \quad \text{Equation 26}$$

Where y_i represents the actual target values, $\hat{y}_i$ represents the model's predicted target values, and n represents the size of test data.

$$r = \frac{\sum_{i=1}^{n}(y_i - \bar{y}_i;\hat{y}_i - \bar{\hat{y}}_i)}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2(\hat{y}_i - \bar{\hat{y}}_i)^2}} \quad \text{Equation 27}$$

Where y_i represents the actual target values, $\bar{y}$ represents the mean of the actual target values, $\hat{y}_i$ represents the model's predicted target values, $\bar{\hat{y}}_i$ represents the mean of the model's predicted target values, and n represents the size of test data.

RESULTS AND DISCUSSION

FF-ANN Model

The actual and predicted values of the FF-ANN model, showing the mean absolute errors and the mean absolute percentage errors, are presented in Table 2, while the scatter plots of actual and predicted values are shown in Figure 2. Tables 3 and 4 display the overall performance metrics and the per-output performance metrics respectively, while the learning curve is shown in Figure 3. It can be observed from Table 2 that the flow variables lvgo_flow, hvgo_flow, and vr_flow had mean percentage errors of 11.25%, 4.47% and 3.70% respectively, while their mean errors were between 1.6538 m^3/hr and 3.9681 m^3/hr . For the temperature variables, the lvgo_temp, hvgo_temp and vr_temp had mean percentage errors of 25.14%, 1.66% and 1.29% respectively, while their mean errors were below 10^OC, except for lvgo_temp. The deviation analysis suggest that the best predicted variable was the vr_temp variable, while the least was the lvgo-temp variable. A percentage error of 10% or less and an error of 10^OC or less indicate very good predictions. An inspection of Figure 2, suggests a good fit of the FF-ANN model for lvgo_flow and vr_temp (high values of regression coefficients), while it showed an average fit for vr_flow, lvgo_temp and hvgo_temp. It however shows a poor fit for the hvgo_flow.

The performance metrics in Table 3, shows that the FF-ANN model achieved an overall R^2 of 0.9853 suggesting that the model has excellent performance accuracy with approximately 98.3% of the variance in the combined target outputs explained by the model. Also, a high Pearson correlation of 0.9930 demonstrates linear agreement between predictions and actual values. The per-output performance metrics of the model (Table 4) reveals substantial variation across the six target variables. The vr-temp is the model's most accurately predicted target output (R^2=0.9450, R=0.9810), followed by lvgo-flow (R^2=0.8130, R=0.9243), demonstrating the model's capability for modelling these variables. In predicting lvgo-temp (R^2=0.5300, R=0.9824) and hvgo-temp (R^2=0.5042, R=0.9794), the model displayed discrepancies across metrics with average R^2 score and strong linear R trend. This suggests that the model captures changes in the direction, but moderately predicts the magnitude of change in the variable. For vr_flow (R^2=0.5978, R=0.7926), the model showed moderate predictive performance, suggesting that the model is acceptable for predicting relationships between input features and these variables. hvgo-flow was the worst-predicted output, exhibiting negative R^2 (-7.7518) and a poor correlation (R= 0.0940), indicating the model performs worse than simply predicting the mean of this variable [20].

The learning curve of the FF-ANN model in Figure 3, shows no indication of overfitting – which is usually characterized by validation loss increasing while training loss decreases. Instead, the validation loss consistently decreased, suggesting that the model generalizes well to the validation set. A key observation from training history of the FF-ANN model is the step-like downward progression of the training curve, whereas the validation curve is smooth. It suggests that the regularization regime provided averaged low-variance gradients, triggering early stopping at epoch 35. The parameters of the best model are restored from epoch 25 (validation loss ≈ 0.32 MSE) [14].

Attention-ANN Model

The actual and predicted values of the Attention-ANN model across the 16 observations of the test dataset, showing the mean absolute error and the mean absolute percentage error for the six output target variables are displayed in Table 5

Table 2: Predicted vs Actual target values for the FF-ANN model

lvgo_flow (m^3/hr)		hvgo_flow (m^3/hr)		vr_flow (m^3/hr)		lvgo_temp (^OC)		hvgo_temp (^OC)		vr_temp (^OC)	
Act.	Pred.	Act.	Pred.	Act.	Pred.	Act.	Pred.	Act.	Pred.	Act.	Pred.
21.26	25.27	91.40	91.71	48.00	45.59	216.60	228.16	307.10	309.97	306.60	308.35
19.73	25.97	96.70	94.87	46.30	45.59	214.80	224.29	302.50	307.03	305.20	300.40
29.85	27.05	89.70	89.52	46.90	46.46	221.50	227.36	309.90	308.42	299.50	305.74
23.03	25.16	91.20	91.73	48.50	47.72	216.40	222.54	305.30	298.65	305.40	310.13
25.97	27.76	90.70	90.42	48.30	45.41	219.20	225.87	308.30	306.82	307.00	308.00
16.97	20.35	88.30	133.21	57.20	60.22	33.80	152.87	262.00	231.90	229.20	235.27
26.14	27.37	92.00	91.12	49.30	49.22	219.70	231.24	309.40	313.68	309.40	307.16

17.28	21.53	94.90	95.95	46.00	47.64	212.10	221.54	302.70	303.93	306.10	302.91
39.17	35.06	82.70	85.42	37.00	44.31	224.30	229.65	316.20	318.29	300.60	304.68
37.82	31.77	92.30	90.62	49.10	48.88	220.50	223.22	307.80	308.47	300.40	303.05
32.28	34.91	93.20	92.76	47.60	46.85	212.70	226.27	306.00	310.70	297.80	305.27
28.25	28.33	94.50	91.48	44.60	45.70	210.70	220.34	304.90	306.57	307.40	306.30
43.98	42.24	83.00	85.45	47.10	45.95	223.60	218.19	313.40	318.77	302.50	306.84
24.22	23.82	92.60	94.88	46.70	44.24	223.00	223.68	310.00	305.05	296.90	303.75
26.15	23.38	94.10	94.34	47.60	48.43	217.90	226.88	308.20	311.54	305.80	305.18
33.82	32.58	87.40	86.72	46.20	45.53	224.40	225.70	312.70	313.98	307.70	311.23
MAE	2.8030		3.9681		1.6538		14.2146		4.7928		3.7898
MAPE (%)	11.25		4.47		3.70		25.14		1.66		1.29

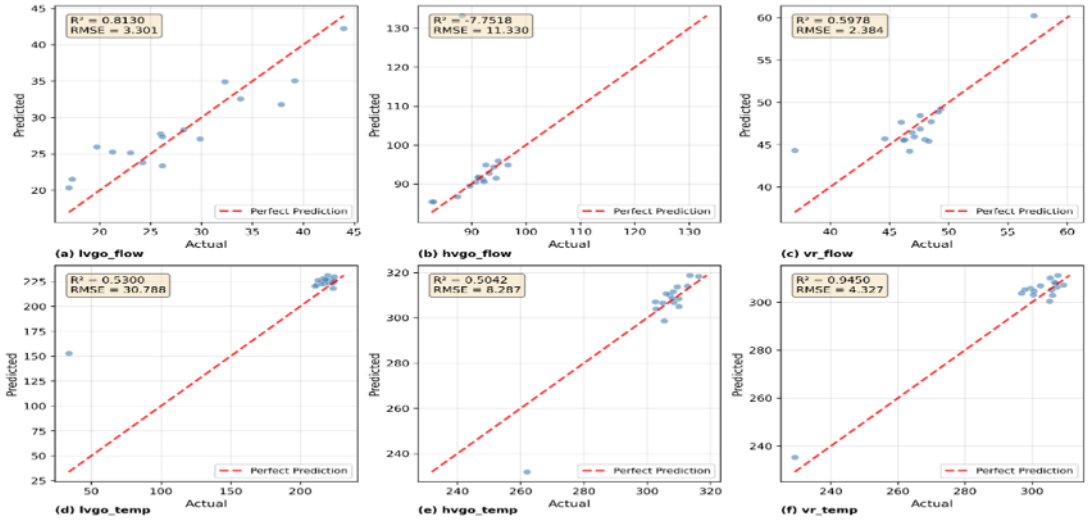

Figure 2: FF-ANN scatter plot of actual vs. predicted target values for the six target outputs (a) lvgo_flow, (b) hvgo_flow, (c) vr_flow, (d) lvgo_temp, (e) hvgo_temp, and (f) vr_temp.

The lvgo_flow, hvgo_flow and vr_flow variables had mean absolute percentage errors of 8.04%, 5.72% and 3.43% respectively, while their mean errors were between 1.6111 m^3/hr and 5.1490 m^3/hr. For the temperature variables, lvgo_temp, hvgo_temp, and vr_temp had mean absolute percentage errors of 4.02%, 1.57% and 2.83% respectively, while their mean errors were between 4.7784 ^{O}C and 8.2429 ^{O}C. Mean percentage errors for all the predicted variables were below 10%, while the mean error for all the temperature variables were less than 10 ^{O}C, suggesting that the model's predictions closely capture the changes in both the flow variables and the temperature variables. Overall, the model performed better – showed less variation, at predicting temperature variables than flow variables. The scatter plots of the actual and predicted values (Figure 4), showed good model fit for the variables (high regression correlation coefficient), except for hvgo_flow. The fit of the Attention-ANN model was better than those of the FF-ANN model for all the output variables, except the vr_temp.

Table 3: FF-ANN overall target metrics

Metric	Value
Overall R^2	0.9853
Overall RMSE	14.0252
Overall MAE	5.2037
Overall MAPE	7.92%
Overall Pearson R	0.9930

Table 4: FF-ANN per-output target metrics

Output	R^2	RMSE	MAE	MAPE (%)	R
lvgo_flow (m^3/hr)	0.8130	3.3013	2.8030	11.25	0.9243
hvgo_flow (m^3/hr)	-7.7518	11.3295	3.9681	4.47	0.0940
vr_flow (m^3/hr)	0.5978	2.3844	1.6538	3.70	0.7926
lvgo_temp (^{O}C)	0.5300	30.7879	14.2146	25.14	0.9624
hvgo_temp (^{O}C)	0.5042	8.2870	4.7928	1.66	0.9794
vr_temp (^{O}C)	0.9450	4.3272	3.7898	1.29	0.9810

From the overall performance metrics shown in Table 6, the Attention-ANN model achieved an overall R^2 of 0.9970 and a high Pearson correlation of (R) 0.9987. These metrics indicate that the model captures 99.6% of the total variance in the combined target variables while maintaining strong linear relationship between predictions and actual values. While the model showed excellent overall performance metrics, an analysis of the per-output performance metrics of the Attention-ANN model (Table 7) reveals varying performance across the output target variables. lvgo_temp was the model's most accurately predicted target output (R^2=0.9810, R=0.9915), followed by lvgo_flow (R^2=0.8656, R=0.9462) and hvgo_temp (R^2=0.7800, R=0.9468), demonstrating the model's high capability for modelling these variables. The model shows fairly accurate predictions for vr_temp (R^2=0.7507, R=0.9256) and vr_flow (R^2=0.6429, R=0.8654), indicating the model performs acceptably in capturing the operating range of these variables. hvgo_flow was the worst-predicted output with R^2 =-3.9251 (R = 0.4572), indicating that the model performs slightly worse than simply predicting the mean of the training data [20]. Although the Attention-ANN model was more accurate than the FF-ANN model in predicting the hvgo_flow variable, both models struggled to predict the variable, suggesting that the input features may not have provided sufficient information for the models to map the flow dynamics of this specific stream [21].

The training and validation loss trajectories of the Attention-ANN is presented in Figure 5. The plots do not show validation loss increasing while training loss decreased, which would indicate overfitting. Instead, validation loss was consistently lower than the training loss for most part of the training, suggesting that the model generalizes well to the validation set.

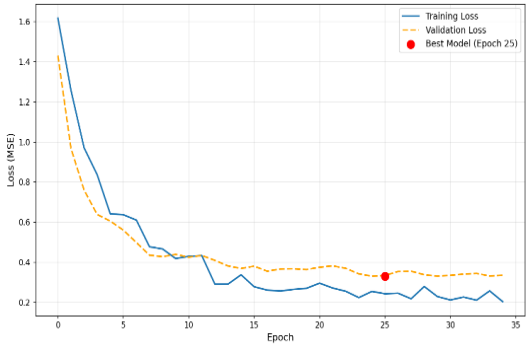

Figure 3: Learning curves for FF-ANN

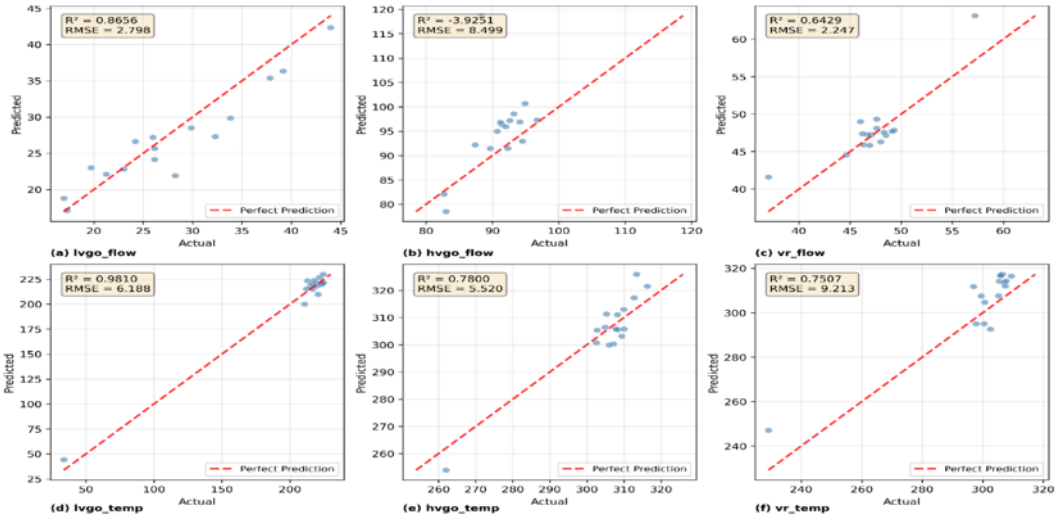

Figure 4: Attention-ANN scatter plot of actual vs. predicted target values for the six target outputs (a) lvgo_flow, (b) hvgo_flow, (c) vr_flow, (d) lvgo_temp, (e) hvgo_temp, and (f) vr_temp.

Table 5: Predicted vs Actual target values for the Attention-ANN model

lvgo-flow (m^3/hr)		hvgo-flow (m^3/hr)		vr-flow (m^3/hr)		lvgo-temp (^{O}C)		hvgo-temp (^{O}C)		vr-temp (^{O}C)	
Act.	**Pred.**	**Act.**	**Pred.**	**Act.**	**Pred.**	**Act.**	**Pred.**	**Act.**	**Pred.**	**Act.**	**Pred.**
21.26	22.12	91.40	96.35	48.00	46.30	216.60	215.04	300.28	307.10	306.60	317.35
19.73	23.00	96.70	97.31	46.30	45.91	214.80	218.22	300.70	302.50	305.20	307.67
29.85	28.52	89.70	91.43	46.90	45.84	221.50	226.60	313.01	309.90	299.50	307.62
23.03	22.82	91.20	96.83	48.50	47.16	216.40	220.34	311.28	305.30	305.40	314.18
25.97	27.20	90.70	94.98	48.30	47.58	219.20	220.44	305.48	308.30	307.00	313.68
16.97	18.80	88.30	118.71	57.20	63.13	33.80	44.29	253.95	262.00	229.20	247.09
26.14	25.67	92.00	95.96	49.30	47.88	219.70	218.17	303.22	309.40	309.40	316.47
17.28	17.13	94.90	100.67	46.00	49.01	212.10	215.28	305.31	302.70	306.10	317.13
39.17	36.34	82.70	82.09	37.00	41.60	224.30	229.92	321.47	316.20	300.60	304.73

37.82	35.38	92.30	91.48	49.10	47.69	220.50	209.79	305.82	307.80	300.40	295.12
32.28	27.33	93.20	98.57	47.60	48.11	212.70	223.30	299.91	306.00	297.80	295.06
28.25	21.91	94.50	92.93	44.60	44.55	210.70	200.03	306.49	304.90	307.40	312.06
43.98	42.35	83.00	78.51	47.10	47.27	223.60	220.91	325.91	313.40	302.50	292.64
24.22	26.63	92.60	97.17	46.70	47.28	223.00	219.73	305.79	310.00	296.90	311.85
26.15	24.16	94.10	96.92	47.60	49.32	217.90	223.66	311.04	308.20	305.80	316.67
33.82	29.85	87.40	92.21	46.20	47.38	224.40	221.34	317.29	312.70	307.70	314.29
MAE	2.2452		5.1490		1.6111		5.1780		4.7784		8.2429
MAPE %	8.04		5.72		3.43		4.02		1.57		2.83

Table 6: Attention-ANN overall target metrics

Metric	Value
Overall R^2	0.9970
Overall RMSE	6.3080
Overall MAE	4.5341
Overall MAPE	4.27%
Overall Pearson R	0.9987

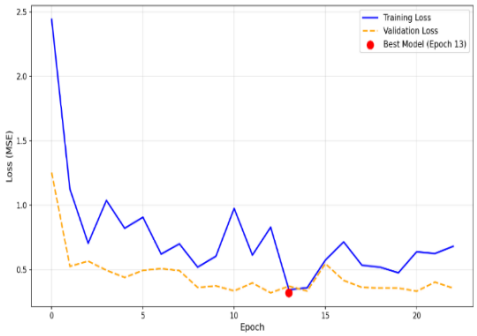

Figure 5: Learning curves for Attention-ANN

Table 7: Attention-ANN per-output targets

Output	R^2	RMSE	MAE	MAPE (%)	R
lvgo_flow (m^3/hr)	0.8656	2.7981	2.2452	8.04	0.9462
hvgo_flow (m^3/hr)	-3.9251	8.4990	5.1490	5.72	0.4572
vr_flow (m^3/hr)	0.6429	2.2468	1.6111	3.43	0.8654
lvgo_temp (^{O}C)	0.9810	6.1881	5.1780	4.02	0.9915
hvgo_temp (^{O}C)	0.7800	5.5201	4.7784	1.57	0.9468
vr_temp (^{O}C)	0.7507	9.2125	8.2429	2.83	0.9256

A key observation from training history of the Attention-ANN model are the sudden spikes in the progression of the training curve, whereas, the learning curve has a smoother trajectory. It suggests that the batch size chosen (size 8) and regularization regime prevent the model falling into suboptimal local minima ([22]), enabling it to maintain strong generalization despite active training volatility [23]. Early stopping was triggered at epoch 23, and the parameters of the best model were restored from epoch 13 (validation loss = 0.32 MSE).

Several patterns can be observed from the heatmap in Figure 6. The attention weights were selectively distributed across the process variables, with values ranging from 0.045 to 0.118. Of all the variables, bottom_pr (column bottom pressure) consistently received the highest attention from most variables, including lvgo_pa_flow (0.118), hvgo_flow (0.114), lvgo_split and overhead_temp (0.109), and lvgo_pa_temp (0.103). In contrast, feed_temp and the flash_zone_temp were the least attended to by the other variables, with weights approximately 0.045 to 0.065. This suggests that bottom_pr plays a major role in column dynamics and separation efficiency, aligning with foundational vacuum distillation theory such as phase transition, vapour liquid equilibrium (VLE), and relative volatility. This shows that the Attention-ANN model captures meaningful interactions among key VDU process variables rather than relying on isolated measurements [24].

Comparison of Models

To compare the predictive performance of the two optimized models, a paired statistical testing framework was employed over 20 independent runs using different random seeds. In each run, identical train-validation-test splits and training conditions were applied to both models, enabling fair paired comparisons. A summary of the results is presented in Table 8. The analysis was conducted separately for each performance metric.

The results reveal that the Attention-ANN and FF-ANN models achieve comparable overall predictive performance, with the Attention-ANN model dominating across most metrics. The Attention-ANN exhibited slightly higher mean R^2 (0.9847 vs. 0.9831), lower RMSE (14.01 vs. 14.97), lower MAPE (7.37% vs. 7.84%), and lower R(0.9917 vs 0.9926), while the FF-ANN demonstrated substantially lower MAE (4.47 vs. 5.48). However, except for MAE, these differences were not statistically significant at the $\alpha = 0.05$ level. For R^2, RMSE, MAPE, and R, the paired samples t-test yielded $t = -1.1534$, $t = 1.5561$, $t = 2.0347$, and $t = -1.2882$ respectively, with small effect sizes, and CIs containing zero, confirming that the observed advantage of the Attention-ANN over the FF-ANN on these metrics were not statistically significant [25].

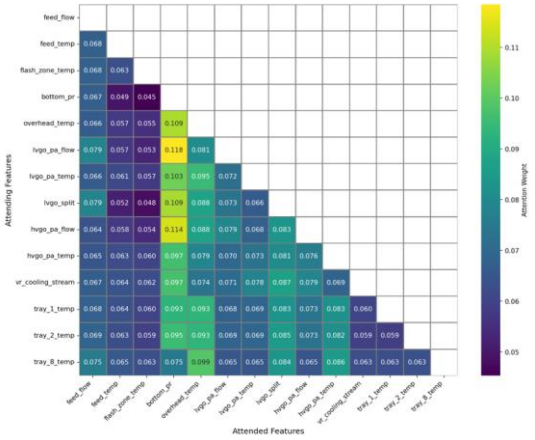

Figure 6: Attention heatmap of the Attention-ANN model showing the average attention weights across all test samples.

CONCLUSION

The Attention-ANN model slightly outperformed the FF-ANN model in predicting five out of the six output variables namely; the lvgo_flow, hvgo_flow, vr_flow, lvgo_temp and hvgo_temp. While the FF-ANN model predicted the vr_temp output variable better. Five out of six per-output performance metrics were also better for the Attention-ANN model. However, statistical analysis of both models using the paired samples t-test showed no statistical difference between the models, suggesting that the models were of comparable accuracy statistically. In terms of interpretability, the Attention-ANN model correctly attributed greater attention weights to the importance of the VDU column bottom pressure (bottom_pr) in the model's predictions, aligning with process engineering principles for column dynamics and separation efficiency. This achievement of predictive accuracy and physical interpretability is a core requirement for industrial adoption where understanding the predictions is as important as the prediction accuracy. This additional interpretability of the Attention-ANN model gives it advantage over the black-box nature of the FF-ANN model. That the FF-ANN model was better in predicting the vacuum residue temperature (vr_temp), suggests that a hybrid architecture – that assigns individual prediction tasks to the architecture demonstrating the best predictive capability, may provide a more effective solution

REFERENCES

1. Chandler, B. (2022). Addressing Africa's Energy Deficit: Gas, Renewables. Mo Ibrahim Foundation. https://mo.ibrahim.foundation/sites/default/files/2022-09/energy-transition.pdf. (Retreieved 9th August 2015)
2. Ozan, S. (2024)." Africa's Escalating Energy Demand: Fostering Opportunities for Sustainable Growth". ReportLinker, https://www.reportlinker.com/article/8460#:~:text=The%20Surge%20in%20Energy%20Needs,towards%20innovative%20solutions%20and%20collaborations (retrieved 9th August 2025)
3. Davies, D., A. Cobbinah, C. Sithole, and S. Williams. (2024) "60% of Africa's Energy Will Be Fossil Fuel-Based by 2040". Africa Briefing, October 2024. https://africabriefing.com/60-of-africas-energy-will-be-fossil-fuel-based-by-2040/. (Retrieved 9th August 2025).
4. Hawilti. (2024). Refineries Watch: Sub-Saharan Africa. https://hawilti.com/wp-content/uploads/2024/07/Refineries-Watch-2024-Report.pdf. (Retrieved 9th August 2025)
5. Holechek, J. L., H. M. Geli,M. N. Sawalhah, and R. Valdez. (2022). "A Global Assessment: Can Renewable Energy Replace Fossil Fuels by 2050?" Sustainability 14(8): 4792.
6. H'ng, S., L. Ng, D. Ng, and V. Andiappan. (2024). "Optimisation of Vacuum Distillation Units in Oil Refineries Using Surrogate Models". Process Integration and Optimization for Sustainability. 8: 1–23.
7. Elayane, J., R. Bchitou, and A. Bouhaouss. (2017)" Study of the Thermal Cracking during the Vacuum Distillation of Atmospheric Residue of Crude Oil". Scientific Study and Research: Chemistry and Chemical Engineering, Biotechnology, Food Industry. 18: 61–71.
8. Jin, Q., Z. Li, Z. Yan, B. Wang, and Z. Wang. (2022). "Optimization Study on Enhancing Deep-Cut Effect of the Vacuum Distillation Unit (VDU)". Processes. 10(2): 359.
9. Atta, M., Khan, H., Ali, M., Tariq, R., Yasir, A., Iqbal, M., Din, S., and Krzywanski, J. (2024). "Simulation of Vacuum Distillation Unit in Oil Refinery: Operational Strategies for Optimal Yield Efficiency". Energies. 17, 3806.
10. Okwonna, O., and A. Obuebite. (2021). "Artificial Intelligence Approach in Crude Distillation Unit Operation". Global Journal of Engineering and Technology Advances 09(02): 075–082.
11. Egemba, K., and Igbokwe P. (2024). Artificial Neural Network Model for the Prediction of the Products of a Crude Distillation Column. *Journal of Engineering Research and Reports*. 26(8): 1–9.
12. Oster, K., Güttel, S., Chen, L., Shapiro, J. L., and Jobson, M. (2021). "Machine learning-based soft sensors for vacuum distillation unit". arXiv. https://arxiv.org/abs/2111.11251
13. Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P.,Gramfort, A., Grobler, J., Layton, R., Vanderplas, J., Joly, A., Holt, B., and Varoquaux, G. (2013). "API Design for Machine Learning Software: Experiences from the Scikit-learn Project". (arXiv:1309.0238). arXiv. https://arxiv.org/abs/1309.0238.
14. Goodfellow, I., Y. Bengio, and A. Courville. (2016). Deep Learning. MIT Press.
15. Bergstra, J., and Bengio, Y. (2012). "Random search for hyper-parameter optimization". Journal of Machine Learning Research, 13(2): 281-305.
16. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). "Attention is all you need". In I. Guyon, U. von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. V. N.Vishwanathan, and R. Garnett (Editors), Advances in Neural Information Processing Systems. Volume 30. 7102p : 5998–6008
17. Masters, D. and Luschi, C., (2018). "Revisiting small batch training for deep neural networks". arXiv. arXiv:1804.07612
18. Prechelt, L. (1998). "Early Stopping - But When?". In G. B. Orr & K.-R. Müller (Editors.), Neural Networks: Tricks of the Trade. Springer. 55-69
19. Feurer, M. and Hutter, F. (2019). "Hyperparameter Optimization". In: Hutter, F., Kotthoff, L., Vanschoren, J.

(Editors). Automated Machine Learning. The Springer Series on Challenges in Machine Learning. Springer, Cham. https://doi.org/10.1007/978-3-030-05318-5_1

20. Chicco, D., Tötsch, N., and Jurman, G. (2021). "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE, and RMSE in regression analysis evaluation". PeerJ Computer Science, 7, Article e623. https://doi.org/10.7717/peerj-cs.623
21. Willard, J., Jia, X., Xu, S., Steinbach, M., and Kumar, V. (2023). Integrating scientific knowledge with machine learning for engineering and environmental systems. ACM Computing Surveys, 55(4): 1 – 37
22. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., (2014). "Dropout: a simple way to prevent neural networks from overfitting". The Journal of Machine Learning Research, 15(1): 1929–1958
23. Kleinberg, B., Li, Y. and Yuan, Y., (2018). "An alternative view: When does SGD Escape Local Minima?".International Conference on Machine Learning (ICML). 2698–2707
24. Kister, H. Z. Distillation Design. McGraw-Hill, 1992
25. Halsey, L. G. (2025). "The reign of the p-value is over: What alternative analyses could we employ to fill the power vacuum?" Nature Human Behaviour. 9(2): 141-152