Machine learning techniques for modeling chemical absorption in CO_2 capture process

Huilan Zheng[a], Gaurav Mirlekar [a,*], Lars O. Nord [a]

[a] *NTNU - Norwegian University of Science and Technology, Dept. of Energy and Process Engineering, Trondheim, Norway*
gaurav.mirlekar@ntnu.no

Abstract

Post-combustion carbon capture (PCC) technologies play an important role in the reduction of CO_2 emissions to address climate challenges. This process is usually simulated in process simulation software based on first-principle models, which calculate physical properties directly from basic physical quantities such as mass and temperature. Using first-principle models usually requires a long computation time, which makes optimization and control difficult. In this study, machine learning algorithms, such as eXtreme Gradient Boosting (XGBoost) and Support Vector Regression (SVR), are investigated as potential alternative modeling approaches. XGBoost is an ensemble algorithm that is based on the decision tree and optimized by gradient boosting. SVR fits the best line within a predefined or threshold error value. These two algorithms are used to build models to predict the CO_2 capture rate (CR) and specific reboiler duty (SRD) in a monoethanolamine-based PCC process. By using the XGBoost, the verification result shows R^2 (a statistical measure that represents the fitness of the model) in predicting CR is 91.7% and in predicting SRD is 80.8%, while by using SVR the R^2 in predicting CR and SRD is 87.9% and 87.2% individually. In addition, XGBoost and SVR take 0.022 seconds and 0.317 seconds respectively to predict CR and SRD of 1318 cases, while the first-principal process simulation model needs 3.15 seconds to calculate 1 case. The data-driven models built using the XGBoost algorithm are employed for further optimization, which aims to find an operating point to have a higher CR and lower SRD. Particle swarm optimization (PSO), a stochastic optimization technique based on the movement and intelligence of swarms, is implemented for the optimization. The CR and SRD for optimal operating conditions are 72.2% and 4.3 MJ/kg each. The computations are faster with the data-driven models incorporated in the optimization technique. Thus, the application of machine learning techniques in carbon capture technologies is demonstrated successfully.

Keywords: Post-combustion carbon capture; Machine learning; Optimization

1. Introduction

The world has experienced global warming due to greenhouse gas emissions. The temperature difference between global mean surface temperature in 2020 and the pre-industrial baseline (1850-1900) reaches 1.2°C [1]. *Net zero emissions* is proposed to avoid the worst climate impacts. It refers to the balance between the amount of produced greenhouse gas and the amount removed from the atmosphere. Carbon capture and storage (CCS) is one of the technologies to achieve net zero emissions. CCS is the process of capturing CO_2 before it enters the atmosphere, transporting and storing it for centuries or millennia. Three technology routes are usually discussed in the CO_2 capture: pre-combustion capture, oxy-fuel combustion capture, and post-combustion capture(PCC).

PCC is a process to capture CO_2 from flue gas generated after burning the coal, oil, and gas before transportation, and it is the most commonly used carbon capture technology. Therefore, this study focused on the PCC process. The most advanced technology used in PCC technologies is chemical absorption followed by the thermal-stripping route. In the absorption process, CO_2 from the gas stream is captured by an absorbent solvent. While in the stripping process, pure CO_2 is released and the absorbent solvent is regenerated [2]. The typical solvents for absorption processes are amines, such as monoethanolamine (MEA) and diethanolamine (DEA).

In the literature, the PCC process is widely studied with the assistance of simulations by researchers to analyze the process behavior. In particular, several methods have been applied to improve prediction performance. Xiaobo Luo et al. developed an accurate rate-based steady-state model for the MEA-based carbon capture process and validated it against thermodynamic and physical properties calculations over a wide range of pressures, temperatures, and CO_2 loadings [3]. A case study was then performed to capture CO_2 from a 250 MWe combined cycle gas turbine (CCGT) power plant to achieve shorter packing height and lower specific duty. Rohan Dutta et al. used reduced stage efficiencies in an equilibrium-based absorber model to predict operating conditions within an accepted range [4]. This modification reduces the computational time for simulation. Typically, these processes are simulated by using mass and energy balance equations to calculate physical properties such as mass flow rate and temperature in a computer software environment, e.g., Aspen HYSYS®, Aspen Plus® or gPROMS® [5][6][7]. However, the process simulation models based on the first principle require considerable computational time for solving complex equations and thus pose challenges in the implementation of advanced optimization and control techniques. To overcome the challenges brought by the first-principle models, the application of machine learning techniques has also been investigated for various CO_2 capture processes. For example, the effectiveness of the machine learning techniques in recognizing high-performing metal-organic framework materials

for CO_2 capture has been proved in the past [8]. In addition, the applicability of these advanced methods can also be seen in ionic liquid-based CO_2 capture for prediction of structure-property relationships between molecular structures of cations and anions and their CO_2 solubilities in comparison with the quantum chemistry based COSMOtherm predictions [9]. Furthermore, improvement in absorption and regeneration-based carbon capture processes and opportunities for machine learning methods are investigated in the literature [10]. Specifically, the opportunities associated with reinforcement learning to get the optimal operational parameters by using data from software simulation and pilot plants are highlighted. Abdelhamid Shalaby, AliElkamel et al. developed machine learning approaches to predict the outputs of the PCC process simulation in gPROMS® using Matérn Gaussian process regression (GPR), rational quadratic GPR, squared exponential GPR models, and feed-forward artificial neural network model [11]. These approaches were able to forecast the system's energy requirement, capture rate, and the purity of the condenser outlet stream with artificial neural network showing higher accuracy. Fei Li, Jie Zhang et al. used gPROMS® to simulate the PCC process and collect data and applied the bootstrap aggregated extreme learning machine and bootstrap aggregated neural networks to predict capture rate [12]. In this case, the BA-ELM was demonstrated as a powerful tool due to smaller mean-square error (MSE) and less computational time.

The challenge of long simulation times for first-principles models remains. During the optimization, the optimal value is usually settled after many searches, if the search time can be reduced, the optimization efficiency could be improved. Therefore, a time-efficient model is highly desirable. Although Abdelhamid Shalaby, AliElkamel et al., Fei Li, Jie Zhang et al. have demonstrated the application of machine learning in the CO_2 capture process, the application of eXtreme Gradient Boosting (XGBoost) and support vector regression (SVR) in steady-state PCC process simulation is not studied yet. This study aims to fill the research gap.

In the past, the application of data-driven models for the optimization of the energy system is studied [13][14]. In particular, the Autoregressive model with exogenous inputs (ARX) method is used for deriving the simplified dynamic model and employed in the biologically inspired optimal control strategy (BIOCS) for implementation on a subsystem of a CO_2 capture process associated with an integrated gasification combined cycle (IGCC) power plant [14]. In this study, a systematic methodology is proposed to implement computationally efficient machine learning techniques in CO_2 capture process. XGBoost and SVR techniques are employed using data samples generated by process simulation to summarize the characteristics of the data sets and establish data-driven models. The main tasks are to predict CO_2 capture rate (CR) and specific reboiler duty (SRD) in the PCC process. Consequently, the developed data-driven models are incorporated into an optimization. In the present paper, particle swarm optimization (PSO) which iteratively improves an alternative solution for a given measure of quality is used as an optimization routine. It is a heuristic global optimization method extensively employed in mathematics and computer science for solving problems more quickly when classic methods are too slow or for finding an approximate solution when classic methods fail to find any exact solution. It has been used in engineering optimization such as, CCS cost and revenue optimization and the optimization of CO_2 solubility predicting model [15][16]. The optimization goal for this study is to find operating conditions with a relatively high CR and low SRD, which would be beneficial for improving energy efficiency during the process operation.

The paper is organized as follows: the methodology is described in section 2; section 3 discusses the results; and in section 4, the summary and conclusions are presented. The development of the PCC process simulation model, the theory of XGBoost and SVR algorithms, and the principle of PSO are introduced in the next section.

2. Methodology

In this section, the developed methodology for the implementation of machine learning techniques in the PCC process is described. The proposed framework of the work is shown in Figure 1. A steady-state process model is built in Aspen HYSYS® to simulate the PCC process. The data collected from this model is used as the raw data to build a data-driven model. After the data collection, the XGBoost and SVR algorithms are applied to build data-driven models. Coefficient of determination (R^2) and MSE are assessment indicators to measure model performance. Besides, a new data set is created to validate the model. The proposed data-driven models are developed by the process illustrated in Figure 1. Then the PSO is adopted to search for the optimal operating conditions in optimization and control.

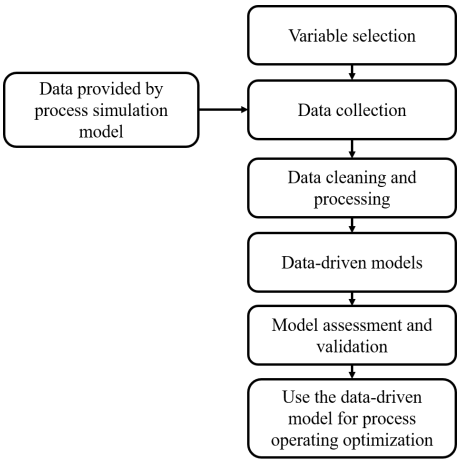


Figure 1: Methodology to implement machine learning technique in PCC process.

2.1. PCC process simulation model

The process flow diagram of the PCC steady-state simulation used for demonstration purposes is shown in Figure 2. This process is a two-step regenerative process, one is the absorption chemical process with solvent, and the other one is the desorption of CO_2 from the solvent and generation of the lean solvent. The flue gas enters the absorber from the bottom and encounters the lean amine which is going down in the column. Depleted gas and rich amine leave the absorber from the top and bottom. Rich amine is pumped, heated up by lean amine from the stripper, then enters the regenerator. In the regenerator, the stripping vapor goes up and condenses at the cooler and condenser, which are on the top of the column. The condensate containing the regenerated solvent is recycled back to the regenerator column. The CO_2 separated is sent

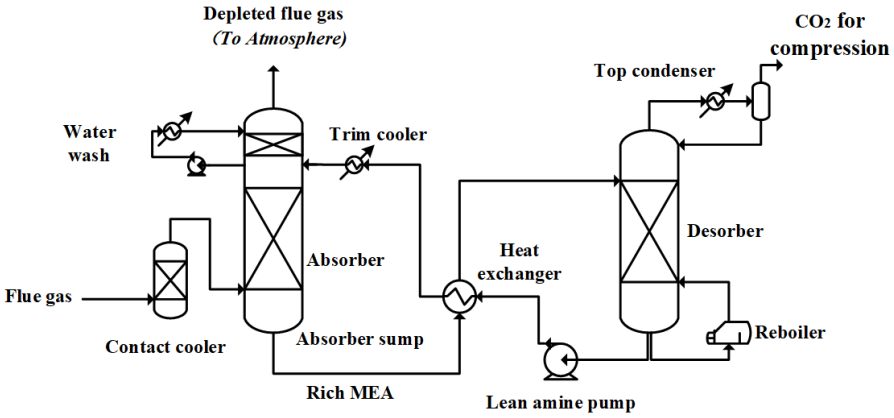


Figure 2: Schematic of chemical absorption process for post-combustion CO_2 capture

to the compressor and further processed for transportation or storage. Lean amine is recycled back to the absorber via a heat exchanger and a cooler.

Based on the selected independent variables, the control variable's value is changed in a step-wise manner while the rest variables are held constant. The corresponding output data information is collected and stored. A base case is defined with main parameters which are listed as Table 1 and Table 2 show.

Table 1: Main parameters in base case

Stream	Flow rate (kg/s)	Temperature (K)	Pressure (bar)
Lean amine	0.642	313.7	1.703
Flue gas	0.158	332.4	1.033

Table 2: Main parameters in base case

Stream	Value
CO_2 molar fraction in flue gas (-)	0.1666
Lean amine loading (-)	0.2814
Rich amine loading (-)	0.4879
Capture rate (-)	0.7627
Reboiler duty (MJ/kg)	5.9580

For the lean amine stream, the compositions are H_2O, CO_2 and MEA. The main compositions in flue gas are H_2O, CO_2 and N_2. Mass fractions of lean amine stream and molar fraction of compositions in the flue gas are listed in Table 3.

Table 3: Lean amine composition in base case

Compositions	Lean amine mass fraction	Flue gas molar fraction
H_2O	0.6334	0.0325
CO_2	0.0618	0.1666
MEA	0.3048	-
N_2	-	0.8009

A set of ranges for the selected variables is defined to create the data sets. As the system encounters the converged problem when the flue gas flow rate increases to 0.2623 kg/s or the lean amine flow rate decrease to 0.4520 kg/s, these two values are set to be the upper limit of the flue gas flow rate and the lower limit of lean amine flow rate. The initial objective is to collect around 1000 data samples when each variable changes, therefore, the step is set at 0.0001 for flue gas flow rate and 0.001 for lean amine flow rate. The numerical changes of flue gas and lean amine flow rate are done automatically via the script written in Python 3.8 in connection to the Aspen HYSYS® process simulation file.

Table 4: Variables range

	Unit	Lower limit	Upper limit
Flue gas flow rate	kg/s	0.1081	0.2623
Lean amine flow rate	kg/s	0.4520	2.0040
CO_2 molar fraction in flue gas	-	0.1666	0.2264
Lean loading	-	0.0278	0.5242

The steps and number of collected data samples for each variable are listed in Table 5.

Table 5: Number of collected samples

	Step	Total sample
Flue gas flow rate	0.0001	1543
Lean amine flow rate	0.01	1553
CO_2 molar fraction in flue gas	around 0.0004	150
Lean loading	around 0.0005	1147
Total	-	4393

Challenges encountered during data collection:

1. The values in the composition worksheet can not be changed automatically by the Python script that is connected to Aspen HYSYS®, no other scripts are found to realize the automatic filling of composition parameters.
2. While changing the CO_2 molar fraction, the total molar fractions of the flue gas stream do not sum up to 1.
3. When the molar fractions of different compositions are changed, Aspen HYSYS® would perform normalization, the step size of CO_2 molar fraction would not be exact 0.0004.

Proposed solutions to overcome these challenges:

1. Use the Python script to control the mouse and keyboard to achieve automatic filling. The mouse is set to click on a fixed position on the desktop, so the value of CO_2 and N_2 molar fraction is naturally written into the Aspen HYSYS® worksheet. But the worksheet menu of Aspen HYSYS® is not always in a fixed position when it pops up, this program fails sometimes.
2. The sum of CO_2 and N_2 molar fraction is assumed to be constant. When CO_2 molar fraction is changed, the N_2 molar fraction is changed to a value to make the total molar fraction of the flue gas stream to be 1.

3. The step size of CO_2 molar fraction is set to be approximately 0.0004.

Due to these challenges and technical limits, only 150 cases are generated for the change of CO_2 molar fraction in the flue gas. Lean amine loading is manipulated by changing the CO_2 molar flow rate in the lean amine stream. The method is the same as the mouse and keyboard control in changing CO_2 molar fraction in the flue gas stream. Consequently, the step is also an approximation. By controlling and changing values of different variables, 4393 cases are simulated to produce the raw data.

2.2. Machine learning techniques

2.2.1. eXtreme Gradient Boosting (XGBoost)

Ensemble machine learning technique is used to combine several base models for building one optimal predictive model. XGBoost stands for "eXtreme Gradient Boosting", an ensemble algorithm that is based on the decision tree and optimized by gradient boosting [17].

The following introduction starts from the decision tree. The evolution route is shown in Figure 3.

Figure 3: Schematic of evolution from decision tree to XGBoost.

Decision tree mainly consists of the root node, decision nodes, and leaf nodes. The root node is the start point. The decision node is a judgment condition, a question like "lean amine loading is larger than 0.2 or not", "flue gas flow rate is larger than 0.18kg/s or smaller than 0.1kg/s", different answers lead to the different nodes in the next level. It may enter another decision node to cross a new judgment, or go to the leaf node, which is the end of the prediction process. The leaf node is the final prediction result.

Bagging, stands for "Bootstrap aggregating", is an ensemble learning method. When bagging is used, a certain amount of samples can be randomly selected from data sets with replacement. The average or most voted result is the final result. A sample can be chosen more than one time in bagging. It is a good way to reduce variance within noisy data sets.

Random forest is a bagging-based algorithm. In a decision tree, some characteristics are chosen to be the judgment condition in a decision node. However, in a random forest, characteristics can be randomly picked to form different decision trees. All trees gather together to become a random forest, the prediction result of the random forest comes from the average or most voted of all tree results.

Boosting is a set of ensemble algorithms that can help convert weak learners (refer to models) to strong learners. The boosting would train a basic learner from training data, then focus more on the wrong prediction samples, and correct errors from the first learner to generate the second learner. Repeat this "correct" work step by step, until a strong learner is trained.

Gradient boosting is one kind of boosting. As the loss function represents the unreliability of the model, gradient boosting adjusts the model towards a gradient descent direction of the loss function from the previous model.

XGBoost is a decision-tree-based ensemble machine learning algorithm that uses a gradient boosting framework. It takes the bootstrap sample 1 to build model 1, then takes the bootstrap sample 2 to build model 2, which is more advanced than model 1. Then iterate to get the final model and result. It minimizes a regularized objective function:

$$\mathrm{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{1}$$

where $\Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2$.

$L(\phi)$ is the objective in this minimization problem. It is the summation of convex loss function represented as $\sum_i l(\hat{y}_i, y_i)$ and regularization item denoted as $\sum_k \Omega(f_k)$. Here, y_i and $\hat{y}_i$ stands for targeted value and predicted value respectively. λ and γ are hyperparameter constants. T is the number of leaf nodes. w is the predicted value of the leaf node.

2.2.2. Support vector regression (SVR)

An SVR model gives users the freedom to decide how much error is accepted. A line or a hyperplane (when the data has higher dimensions) is searched to fit the data. The goal is to minimize the coefficients, the l_2-norm of the coefficient vector, rather than the squared error which is usually used in linear regression [18]. The constraint would be:

$$|Y_i - \alpha X_i| \leq \varepsilon \tag{2}$$

The goal is to minimize:

$$\mathrm{MIN}\frac{1}{2}\|\alpha\|^2 \tag{3}$$

Here, $\mathrm{Y_i}$ stands for the targeted value, $\mathrm{X_i}$ represents the feature value, α denotes coefficients. In SVR models, kernel functions can be used to transform input data to the required form of processing data. There are different kinds of kernels: Gaussian Kernel, Radial Basis Function, Sigmoid Kernel, and Polynomial Kernel.

2.3. Optimization algorithm

2.3.1. Particle swarm optimization (PSO)

PSO is a search optimization technique inspired by the migration behavior of birds. Assume there are N particles in a swarm. These particles are subject to random uniform initialization, and they have random positions and velocities in a D-dimensional search space. These particles move at a certain speed to find the best position inside the whole space. For each particle, the new velocity is updated based on its own historical experience and the group experience. Assume the D-dimensional position vector of the i-th particle is:

$$S_i = (S_{i1}, S_{i2}, S_{i3}, ..., S_{iN}), i = 1, 2, ..., N \tag{4}$$

The velocity vector of i-th particle is:

$$V_i = (V_{i1}, V_{i2}, V_{i3}, ..., V_{iN}), i = 1, 2, ..., N \tag{5}$$

The equation to update the position and velocity of each particle is:

$$v_{id}^{k+1} = m v_{id}^k + c_1 r_1 (p_{id}^k - s_{id}^k) + c_2 r_2 (p_{gd}^k - s_{id}^k) \tag{6}$$

$$s_{id}^{k+1} = s_{id}^{k} + r v_{id}^{k+1} \quad (7)$$

This speed update equation of the PSO algorithm is the sum of three parts. The v_{id} and s_{id} are the velocity and position of the i-th particle in the d-th dimension, k and k+1 represent the current and next iterations. First part mv_{id}^{k} is the exploration, m is an inertia factor of fixed value. Second part $c_1 r_1 (p_{id}^{k} - s_{id}^{k})$ is self-learning, and the third part $c_2 r_2 (p_{gd}^{k} - s_{gd}^{k})$ is group learning. c_1 and c_2 are learning factors, r_1 and r_2 are random numbers within the range [0, 1]. p_{id} and p_{gd} are the best positions searched by the i-th particle and the whole group so far.

In the next section, the data analysis result, the prediction performance of built models, and the optimization result are shown and discussed.

3. Results

3.1. Data analysis and model development

The purpose of data analysis is to initially have an overview understanding of the data sets by observing the connections between variables. The Pearson correlation coefficient, which is a measure of linear correlation between two sets of data, is used in data analysis. Given paired data set (Z_{11},Z_{21}),(Z_{12},Z_{22}),...,(Z_{1n},Z_{2n}), the formula of Pearson correlation coefficient R_{Z_1,Z_2} is:

$$R_{Z_1,Z_2} = \frac{\sum_{i=1}^{n}(Z_{1i} - \overline{Z}_1)(Z_{2i} - \overline{Z}_2)}{\sqrt{\sum_{i=1}^{n}(Z_{1i} - \overline{Z}_1)^2}\sqrt{\sum_{i=1}^{n}(Z_{2i} - \overline{Z}_2)^2}} \quad (8)$$

where $\overline{Z}_1$ and $\overline{Z}_2$ are the average values of variables Z_1 and Z_2. Figure 4 shows Pearson correlation coefficients between different variables in this study.

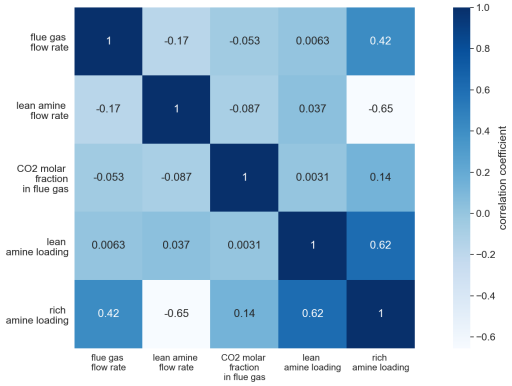


Figure 4: Plot of data correlation coefficients

The number on the grid is the correlation coefficient of the corresponding horizontal and vertical axis variable. The darkness or grid color shade is directly proportional to the correlation coefficient. For example, the grids which are intersected by lean amine flow rate and the flue gas flow rate are light blue, and the value of the correlation coefficient is -0.17. This indicates that they are slightly inverse related. The correlation coefficient between rich amine loading and lean amine loading is 0.62, which shows they might have positive relationships. Thus, rich amine loading is not included as an input variable. There may be some non-linear relations between different variables that are not included in the correlation coefficient. The purpose of the data analysis is to get a general understanding of data sets, thus, non-linear relations are not considered here.

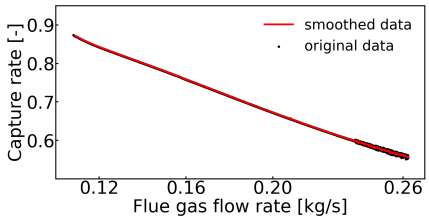
The relation between dependent variables and independent variables can be further investigated in Figure 5. In the following section, *To atmosphere*, *CO_2 for compression*, *Rich MEA* refer to gas stream out from absorber, captured CO_2 stream out from desorber, loaded amine stream out from absorber in Aspen HYSYS® simulation model as shown in Figure 2.

Figure 5-(a) shows the relationship between the capture rate and the flue gas flow rate of 1543 data samples. The highest point represents a case in which the flue gas flow rate is 0.1081kg/s and the capture rate is 0.8738, the lowest point represents a case the flue gas flow rate is 0.2623kg/s and the capture rate is 0.5557. Capture rate decreases as flue gas flow rate increases. As the flue gas mass flow rate increases, more CO_2 enter the absorber in a time unit. Due to the absorbing solvent limitations and increased inflow of CO_2, the capture rate is decreased and more CO_2 is released into the atmosphere through *To atmosphere* stream. Consequently, the capture rate becomes smaller. When the flue gas flow rate is close to 0.2623, the capture rate value starts fluctuating, this could be caused by the model instability. In Figure 5-(b), the SRD decreases as the flue gas flow rate increases. The reasons behind this are the increase of the CO_2 in *CO_2 for compression* stream and the decrease of the reboiler duty in the regenerator. SRD is calculated by the reboiler duty divide mass of CO_2 in *CO_2 for compression*, therefore, SRD decreases. When the lean amine flow rate increases, the CO_2 enter the rich stream increases. Thus, as Figure 5-(c) shows, the capture rate increases with the lean amine flow rate. In Figure 5-(d), when the lean amine flow rate is increased, the CO_2 in the *To atmosphere* stream becomes less and CO_2 in the *rich amine* stream increased, which increases the CR. Reboiler duty increases since the stream entering the regenerator include more MEA. Accordingly, the SRD increases. The increase of CO_2 molar fraction in flue gas results in the increase in CO_2 in the stream that is released into the atmosphere and *rich amine* stream, but the increase in *To atmosphere* is larger while the increase in *rich amine* stream is smaller, consequently, the CR is lower. The reboiler duty of the regenerator is lower, resulting in a lower SRD. Lean amine loading is changed by adjusting the mass flow rate of CO_2 in lean amine. When increases lean amine loading, CO_2 in the *To atmosphere* stream increases, and the capture rate decreases. Reboiler duty also decreases and accordingly SRD is lower.

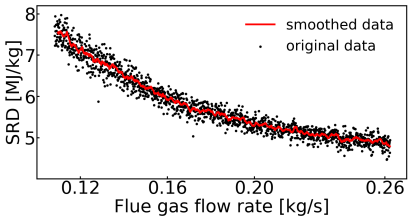
It is observed that SRD has fluctuations quite often. It is caused by the unstable reboiler duty. Data smoothing is suggested to deal with these instabilities. There are many ways to smooth data, such as simple exponential, moving average, exponential moving average, and Holt-Winters smoothing. The models developed by using smoothed didn't show an expected improvement. Therefore, to keep the original information of data, all the following models are developed based on original data without smoothing.

3.2. Modeling and validation

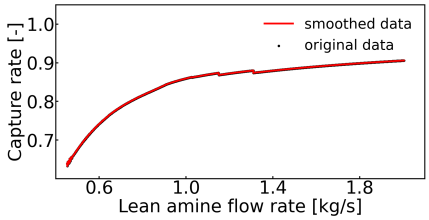
All data sets are divided into training data sets and test data sets with a division ratio of 7:3. After building models based on training data sets, the predicted values based on independent variables in test data sets are compared with the value of dependent variables in test data sets. Cross-validation score, MSE, RMSE, R^2 are calculated to assess model performance. Using XGBoost to predict capture rate had a cross-validation average score of 0.9995. The R^2 was 0.9996 and the MSE was 0.0000. The difference



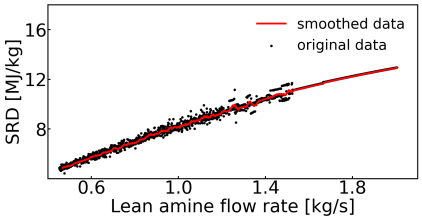
(a) Flue gas flow rate vs Capture rate



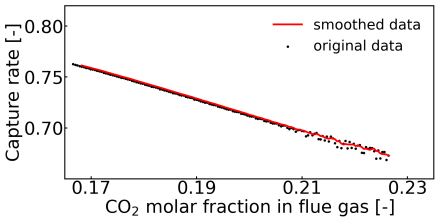
(b) Flue gas flow rate vs SRD



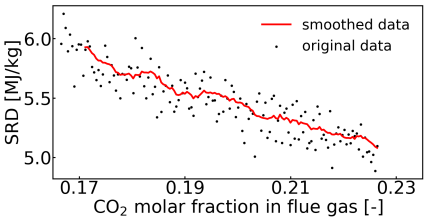
(c) Lean amine flow rate vs Capture rate



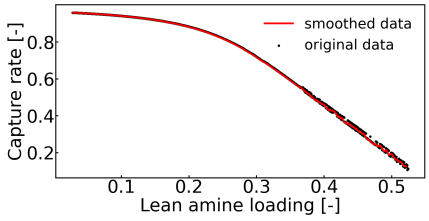
(d) Lean amine flow rate vs SRD



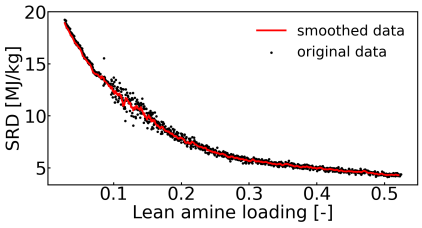
(e) CO_2 molar fraction vs Capture rate



(f) CO_2 molar fraction vs SRD



(g) Lean amine loading vs Capture rate



(h) Lean amine loading vs SRD

Figure 5: Plots of relationship between each independent variable and dependent variable

between predicted CR and tested CR is shown in Figure 6-(a). Most points were distributed around line y=0, which indicated that there are no huge deviations in prediction, and the model has a good prediction accuracy. It is shown that the accuracy of the XGBoost model is quite high. To validate this accuracy, cross-validation was used to avoid overfitting. And the SVR algorithm was applied to build the other model. If SVR shows a different accuracy level, the qualities of the models can be compared. The XGBoost and SVR algorithms are employed to predict capture rate and SRD respectively, therefore, 4 models are built. The performances of each model can be seen from Figure 6.

In each subplot is the scatter plot of the difference between the predicted value and the real value. It can be observed that when using XGBoost to predict SRD, the errors are smaller. SVR may not be a suitable method to predict CR. The error-represented points are randomly distributed in the different positions instead of scattering around line y=0.

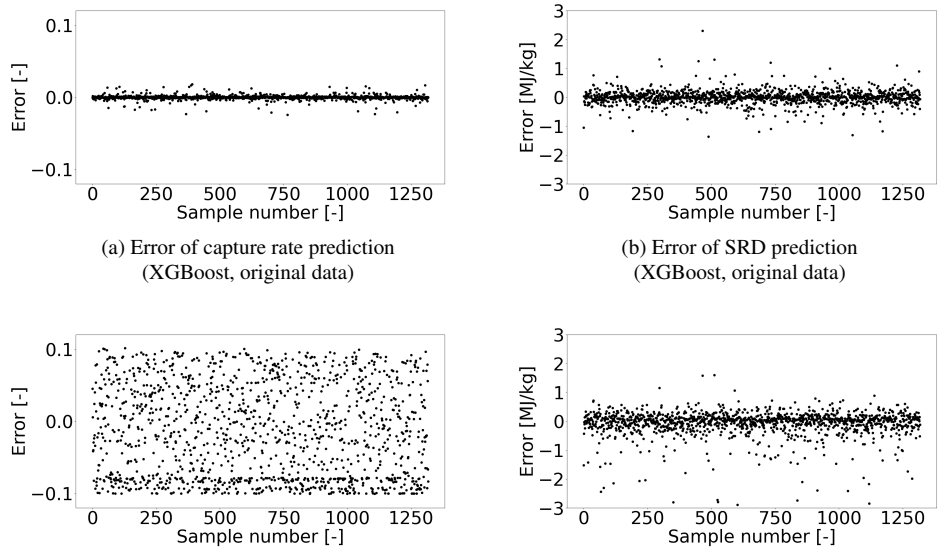
3.3. Results and verification

Table 6 summarize the performance of different models:

Table 6: Summary table

XGBoost		
assessment	**predict CR**	**predict SRD**
cross-validation score	0.9995	0.9934
R-squared(R^2)	0.9996	0.9930
MSE	0.0000	0.0595
RMSE	0.0032	0.2439
SVR		
assessment	**predict CR**	**predict SRD**
cross-validation score	0.8573	0.9620
R-squared(R^2)	0.8418	0.9622
MSE	0.0041	0.3231
RMSE	0.0643	0.5684

A verification data set of 78 data samples are generated to verify the model. This data set is different from training data sets and test data sets but within the same range as shown in Table 4. Note that the verification of the developed model can be improved with higher number



(a) Error of capture rate prediction (XGBoost, original data)

(b) Error of SRD prediction (XGBoost, original data)

(c) error of capture rate prediction (SVR, original data)

(d) Error of SRD prediction (SVR, original data)

Figure 6: Plots of errors of model prediction: (a)XGBoost model to predict CR; (b)XGBoost model to predict SRD; (c)SVR model to predict CR; (d)SVR model to predict SRD.

of data samples. However, the number of data samples mentioned above are restricted due to the operational range in which this model is developed. The verification results of models are shown in Table 7 :

Table 7: Verification result

	XGBoost	
assessment	**predict CR**	**predict SRD**
R-squared(R^2)	0.9170	0.8077
MSE	0.0029	0.4043
RMSE	0.0541	0.6358
	SVR	
assessment	**predict CR**	**predict SRD**
R-squared(R^2)	0.8793	0.8716
MSE	0.0042	0.2700
RMSE	0.0653	0.5196

XGBoost models are efficient as they showed verification accuracies of 91.7% to predict CR and 82.82% to predict SRD. These high values of R-squared indicated that models have generalization ability. The accuracy of the developed model is close to the accuracy of the simulation within the range of process variables in which this model is developed. The precision of the developed model in all cases or outside the range is a subject of investigation. The model is available when the input variables are within the ranges shown in Table 4. For the values of the variables outside the ranges, the model's availability is not investigated. Some observations can be discussed:
(1) In a general machine learning modeling process, the variables selection is a step after the data collection. During the data correlation and visualization step, data should be visualized to explore the trends or relations between various variables. Then the variables selection is implemented. In this order variables that are completely irrelevant or replaceable can be avoided. However, no data are pre-provided in this study, and it is hard to generate large data sets as all data are generated by Aspen HYSYS®. Thus the variables are chosen mainly based on some research results and previous experience, then data collection is executed.
(2) Some curves in Figure 5 have fluctuations (Figure 5-(b)) and breakpoints(Figure 5-(c)). That is because the steady-state simulation is calculated based on first-principle models which often include several differential and algebraic equations. The model is unstable sometimes, consequently, the same inputs may lead to a bit different outputs.
(3) Although the number of raw data samples is more than 4000, the steps between samples are quite small, which may not differentiate much. More data samples could help increase the performance of the model or lead to higher prediction accuracy in verification.
(4) XGBoost algorithm has been shown to have high accuracies in a lot of applications. In this study, training and test data sets are synthetic data generated by the steady-state simulation. This simulation has a certain mathematic expression, making the R-squared of the model 99% possible. The other reasons for this high accuracy lie in the complexity of the prediction task, the amount of training data sets, and the possibility of overfitting.

3.4. Optimization

In this section, optimization of the process operation conditions is discussed. From the operational point of view, the goal is to maximize the CR and minimize the SRD as much as possible. However, the general rule is that the smaller SRD is usually associated with lower CR. Therefore, it's important to make a trade-off. To show this in mathematical form, the objective goal can be viewed as:

$$f(x) = \frac{SRD(x)}{SRD_{upperlimit}} - \frac{CR(x)}{CR_{upperlimit}} \quad (9)$$

where the SRD(x) and CR(x) are the SRD and CR predicted by the XGBoost model based on the vector x, which represents the operation conditions. These two outputs are then scaled, as the CR and SRD are in different ranges and could not be evaluated at the same level without scaling.

The process simulation model needs 3.15s to predict CR and SVR for 1 case. However, it only takes 0.022s and

0.317s for XGBoost and SVR model to predict CR and SVR of 1318 cases. The computation time is drastically reduced, which would be beneficial for optimization implementation. With a swarm of 10 particles, after 1500 times iteration, the minimal value of fitness *f(x)* is -0.5299. The operating conditions are: flue gas flow rate of 0.18 kg/s, the lean amine flow rate of 0.46 kg/s, CO_2 molar fraction in the flue gas of 0.2002, lean amine loading of 0.3085, and corresponding CR and SRD is 72.2% and 4.3 MJ/kg respectively. Compared with the base case which has a CR of 76.3% and SRD of 5.9 MJ/kg, although the CR was 4.1% lower, the SRD decreased by 1.7 MJ/kg, which accounts for 28.2% of 5.9 MJ/kg.

4. Conclusions

In this paper, the application of machine learning techniques such as XGBoost and SVR for the PCC process model simulation was demonstrated successfully. The energy efficiency indicators and parameters associated with the PCC process model were identified, then machine learning algorithms were applied to build models to predict CR and SRD. The models were used in optimization, and adequate operation conditions are characterized. The data-driven models showed high accuracy in predicting the capture rate and energy requirement in the reboiler of the PCC model. The XGBoost model had the accuracy of 91.7% and 80.8% for predicting CR and SRD based on the validation data sets. The SVR model showed 87.9% and 87.2% in CR and SRD prediction. And the calculation time of 1318 cases for the XGBoost model and SVR model was 0.022 seconds and 0.317 seconds. Compared with the first-principle-based process model, which needed 3.15 seconds to calculate the parameters of 1 case, the data-driven models showed improved performance in the time-efficient aspect. The goal of developing time-efficient models by machine learning techniques was achieved. Integrated the data-driven model within optimization, the ideal operating condition was flue gas flow rate as 0.18 kg/s, lean amine flow rate as 0.46 kg/s, CO_2 molar fraction in flue gas as 0.20, lean amine loading as 0.31. The corresponding CR and SRD were 72.2% and 4.3 MJ/kg individually. The CR decreases 4.1% lower and SRD dwindles 1.7 MJ/kg compared with the base case. Thus, the machine learning techniques were demonstrated useful in process optimization and advanced control methods where faster model predictions are necessary.

Acknowledgment

The authors gratefully acknowledge the Department of Energy and Process Engineering at NTNU for funding support and facilitating with computer resources and software package. The authors would also like to thank NTNU Ph.D. candidate Valentin Formont for help during the work.

References

[1] W. M. Organization, "2020 was one of three warmest years on record."

[2] H. F. Svendsen, E. T. Hessen, and T. Mejdell, "Carbon dioxide capture by absorption, challenges and possibilities," *Chemical Engineering Journal*, vol. 171, no. 3, pp. 718–724, 2011. doi:10.1016/j.cej.2011.01.014.

[3] X. Luo and M. Wang, "Improving prediction accuracy of a rate-based model of an MEA-based carbon capture process for large-scale commercial deployment," *Engineering*, vol. 3, no. 2, pp. 232–243, 2017. doi:10.1016/J.ENG.2017.02.001.

[4] R. Dutta, L. O. Nord, and O. Bolland, "Applicability and validation of use of equilibrium-based absorber models with reduced stage efficiency for dynamic simulation of post-combustion CO_2 capture processes," *Energy Procedia*, vol. 114, pp. 1424–1433, 2017. doi:10.1016/j.egypro.2017.05.048.

[5] "Aspen HYSYS®(version 10) [software]," *https://www.aspentech.com/en*, 2021.

[6] "Aspen Plus®(version 12) [software]," *https://www.aspentech.com/en*, 2021.

[7] "gPROMS®(version 2) [software]," *https://www.psenterprise.com/products/gproms*, 2021.

[8] M. Fernandez, P. G. Boyd, T. D. Daff, M. Z. Aghaji, and T. K. Woo, "Rapid and accurate machine learning recognition of high performing metal organic frameworks for CO_2 capture," *The journal of physical chemistry letters*, vol. 5, no. 17, pp. 3056–3060, 2014. doi:10.1021/jz501331m.

[9] V. Venkatraman and B. K. Alsberg, "Predicting CO_2 capture of ionic liquids using machine learning," *Journal of CO_2 Utilization*, vol. 21, pp. 162–168, 2017. doi:10.1016/j.jcou.2017.06.012.

[10] M. Rahimi, S. M. Moosavi, B. Smit, and T. A. Hatton, "Toward smart carbon capture with machine learning," *Cell Reports Physical Science*, p. 100396, 2021. doi:10.1016/j.xcrp.2021.100396.

[11] A. Shalaby, A. Elkamel, P. L. Douglas, Q. Zhu, and Q. P. Zheng, "A machine learning approach for modeling and optimization of a CO_2 post-combustion capture unit," *Energy*, vol. 215, p. 119113, 2021. doi:10.1016/j.energy.2020.119113.

[12] F. Li, J. Zhang, E. Oko, and M. Wang, "Modelling of a post-combustion CO_2 capture process using extreme learning machine," *International Journal of Coal Science & Technology*, vol. 4, no. 1, pp. 33–40, 2017. doi:10.1007/s40789-017-0158-1.

[13] G. Mirlekar, B. Gebreslassieb, U. Diwekar, and F. V. Lima, "Biomimetic model-based advanced control strategy integrated with multi-agent optimization for nonlinear chemical processes," *Chemical Engineering Research and Design*, vol. 140, pp. 229–240, 2018. doi:10.1016/j.cherd.2018.10.005.

[14] G. Mirlekar, G. Al-Sinbol, M. Perhinschi, and F. V. Lima, "A biologically-inspired approach for adaptive control of advanced energy systems," *Computers Chemical Engineering*, vol. 117, pp. 378–390, 2018. doi:10.1016/j.compchemeng.2018.07.002.

[15] H. Saboori and R. Hemmati, "Considering carbon capture and storage in electricity generation expansion planning," *IEEE Transactions on Sustainable Energy*, vol. 7, no. 4, pp. 1371–1378, 2016. doi:10.1109/TSTE.2016.2547911.

[16] N. A. Menad, A. Hemmati-Sarapardeh, A. Varamesh, and S. Shamshirband, "Predicting solubility of co2 in brine by advanced machine learning systems: Application to carbon capture and sequestration," *Journal of CO2 Utilization*, vol. 33, pp. 83–95, 2019. doi:10.1016/j.jcou.2019.05.009.

[17] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016. doi:10.1145/2939672.2939785.

[18] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004. doi:10.1023/B:STCO.0000035301.49549.88.