JOM, Vol. 74, No. 9, 2022
https://doi.org/10.1007/s11837-022-05371-4

Applications of Machine Learning to Predict the Chord Length Distribution of Droplets in Oil–Water Dispersions

YUNCHAO LI,[1,3] DAQIAN LIU,[1] and LU LIU[2]

1.—State Key Laboratory of Natural Gas Hydrate, National Engineering Laboratory for Pipeline Safety, MOE Key Laboratory of Petroleum Engineering, Beijing Key Laboratory of Urban Oil and Gas Distribution Technology, China University of Petroleum–Beijing, Beijing, China. 2.—Hong Kong Branch of National Precious Metals Material Engineering Research Center, Department of Materials Science and Engineering, City University of Hong Kong, Hong Kong, China. 3.—e-mail: 2020210791@student.cup.edu.cn

The chord length distribution (CLD) of oil–water dispersions were experimentally measured with a focused beam reflectance method (FBRM) probe. With the experimental data, models based on multiple-linear regression (MLG), deep neural network (DNN), random forest (RF), and support vector regression (SVR) have been applied to predict the CLD in oil–water systems and to investigate the connection between shear rate (CUT), rotational speed (RPM), chord length (CL), and the counts of different CLs. Beyond the highest absolute value of correlation coefficients (0.504 for Pearson, 0.813 for Spearman, 0.606 for Kendall), CL influences the counts the most, and they are negatively correlated. The variance and scores of different models were compared, which provided a reference for the application of machine-learning algorithms to the prediction of drop size distribution (DSD) in practical production. From the results, the model based on RF has the highest accuracy (over 0.99) on both the testing and training datasets. To examine our tuned hyperparameters, RF also showed the most stable performance on unseen data, which has 0.9926 R^2 and minimum mean absolute error (MAE) of 0.0159. Our results showed that machine-learning algorithms are stable and precise enough to predict CLD in oil–water dispersions.

INTRODUCTION

Oil–water dispersions are an immiscible system of oil and water.[1] Many investigations have been carried out experimentally and theoretically into oil–water dispersions, both in stirred vessels[2] and in pipes,[3] to figure out how the drop length distribution (DSD) is influenced by parameters such as impeller speed, fluid velocity, phase fraction, and viscosity ratio.[4] Whether in vessels or pipes, the two immiscible liquids often occur as a dispersed flow, where one liquid is present in the other liquid in the form of drops. The morphology and characteristics of these drops affect the safety and efficiency of the production process. Furthermore, chord length distribution (CLD) and DSD can be used to describe the state of oil–water dispersions,[5] especially in the mixing tank. To analysis DSD, the CLD should be observed and analyzed first using a focused beam reflectance method (FBRM) probe[6] The sensor of the FBRM measures a unidimensional characteristic of particles, giving measurements of the CLD of suspended particles as a response.[7] The chord length (CL) is defined as the length of the segments of a line embedded in specific phases of a material.[8]

The fluid dynamic interaction between oil and water phases plays an essential role in predicting the features of dispersion, but is far from being understood.[9] Many aspects, such as the correlation of DSD with energy dissipation,[10] the influence of impellers on the dispersion features,[11] the description of the interaction between the liquid phases in terms of different turbulence conditions,[12] and the use of maximum entropy, gamma, for the prediction

(Received January 31, 2022; accepted May 26, 2022; published online June 28, 2022)

of drop size[13] have been found to be difficult to apply effectively to the prediction of DSD because the connections between these factors are complicated. To simplify the mechanism models, factors like the changing temperature, the influences of impellers, and the average properties are always ignored over the whole system in the calculation processes, and this ignorance in the model building often leads to errors.[2] Thus, it is difficult to use mechanism models to predict CLD and DSD in oil–water dispersions. Our aim is to describe oil dispersions based on several easily controlled features (oil features, rotation speed, and CL) and to search for better solutions for problems in the oil industry.

In recent years, great progress has been made in the prediction of complex variables by the use of machine learning. Machine learning is widely used in many subjects, such as biology,[14] physics,[15] chemistry,[16] and oil engineering.[17] Figure 1 shows eight major steps[18] to build a machine-learning project. Crestani et al.[7] applied machine learning to convert sucrose CLD into DSD. Giulietti et al.[19] used a laser reflection sensor to measure CLD and a neural network to monitor the crystallization processes. Langenbucher et al.[20] used deep learning algorithms and multilinear regression to predict the axial lens position after cataract surgery. Lopez-Exposito et al.[21] combined a neural network and the detection of CLD to estimate *Chlamydomonas reinhardtii* biomass concentration. To sum up, to investigate better solutions to predict CLD more precisely without considering influencing factors, such as temperature, pressure, and the oil properties of oil–water dispersions, we applied machine learning in CLD prediction model building.

Based on our raw experimental data, we built our models using three algorithms, deep neural network (DNN), random forest (RF), and support vector regression (SVR) to look for the connection between CUT (oil cut), CL (chord length), RPM (impeller rotation speed), and COUNTS (counts of different particle sizes). First, our data were split into training sets, testing sets and unseen data sets.[22] Then, a straight shuffle–split method[23] was applied for the cross-validation of our models. Also, dropout[24] layers and the L1 L2 regularization method were employed to avoid overfitting of the training model. Beyond that, we calculated the correlation coefficients to examine which factor influences CLD the most. With our analysis, CUT exerts an important role in the prediction of CLD, which is consistent with the results of our physical analysis.[5] Our present models were tuned over different training sets to determine the most reliable and accurate model. After that, these best models were saved and validated on the testing sets and the unseen data sets. Our evaluation results and analysis showed that machine-learning algorithms are stable and can be used to predict CLD precisely, which support the employment of this model for future designs and applications. Our approach offers a methodology that can be followed for other technological problems in oil fields to facilitate designs.

CLD of Oil–Water Dispersion Droplets in a Mixing Tank

In order to simulate the changes of stirring and particle size distribution in the separator in practical production and application, corresponding experiments were designed to measure the particle CL to indirectly reflect the particle size.

Experimental Equipment, Materials and Procedures

All the parameters in this paper came from our own experiment. D80 solvent oil, Span80 emulsifier, deionized water, and other experimental raw materials were used in this experiment. Firstly, we made emulsions with different oil contents under the condition of constant temperature, then we carried out shear experiments on the oil–water dispersions at different speeds under the conditions of a water bath. Moreover, FBRM was used to detect the CLD

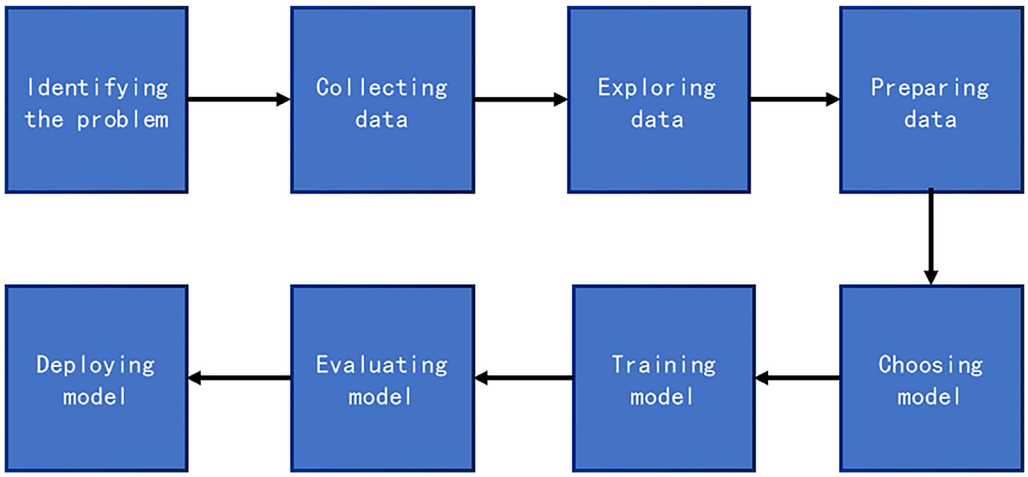

Fig. 1. Eight steps to build a deep learning project.

in the system. In order to avoid the influence of shear history on the oil products, we changed the experimental samples for each testing group. We mainly recorded the relatively stable CLD data for subsequent analysis.

Experimental Results and Analysis of Oil–Water Dispersion System Under Variable Oil Content and Rotating Speed

A FBRM probe (Model D600L; REDMOND, USA) was used to measure the particle size distribution. The principle of particle size measurement is illustrated in Fig. 2. In order to facilitate the fixing of the FBRM probe during measurement, it had a cylindrical shape.[25] There is a laser diode in the probe that can emit a laser beam which enters a beam splitter. The separated laser beam then passes through a high-speed rotating mirror light sheet, so that the beam path deviates from the central axis and focuses to form a light "focus", which is near the outside of the probe. With the high-speed rotation of the mirror light sheet, the laser beam path and the focus also rotate at a constant rate in the direction of the central axis. The focus is emitted into the sample to be measured. The laser is scattered on the particle surface, and a certain proportion of the scattered light will return to the probe, finally reaching the probe detector, which automatically analyzes the data signal.[26]

The scanning speed of the laser beam on the particle surface is generally 2–16 m/s, and the speed can be set as a constant. The product of the scanning speed and the time difference used to scan the particles is the CL of the particles. Because the particle size meter is measured directly on-line, it can dynamically quantify and control the influence of process parameters, such as stirring speed and concentration, on the particle system, as well as the influence of the energy particle system on product performance. The properties of oil we used are shown in Table I. The oil used in our experiment is sold by Exxon Mobil.[27]

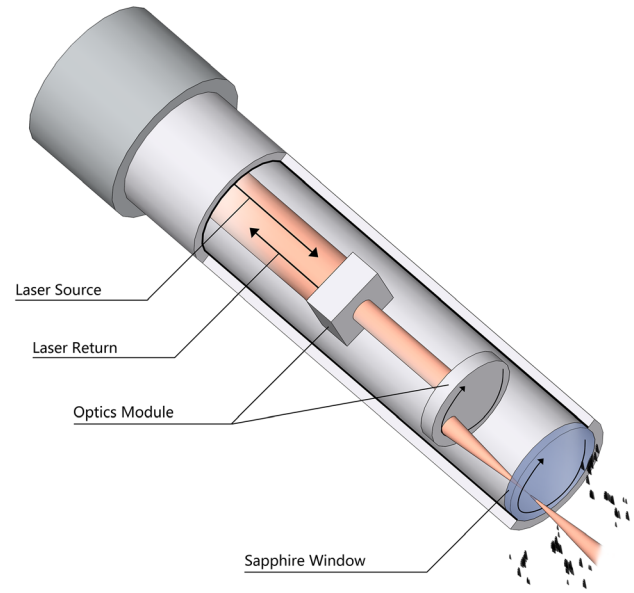

Fig. 2. Schematic of FBRM.

Besides the properties above, Span 80 was used in the experiment as an emulsifier. The mixing equipment used in this experiment was an IKA RW 20 agitator, equipped with a four-blade agitator. The diameter of the impeller was 50 mm, the thickness was 5 mm, the blade inclination angle was 45°, and the mixing speed range was 0∼ 2000 rpm. The measuring equipment for droplet distribution was a d600l focused reflection laser particle sizer produced by Mettler Toledo, Switzerland, and the measurement range of particle diameters was 0.5∼ 1000 μm. The probe of the particle sizer was placed 30 mm below the liquid level and 20 mm from the axial direction of the mixing shaft. The vessel selected for the stirring experiment was a 500-ml beaker with a diameter of 80 mm.[28] For the mixed liquid with each oil content, the total volume of the oil–water mixture remained unchanged at 300 ml, and the liquid level height is 70 mm. The volume of the oil-water is maintained at 300 ml, and the ambient temperature was maintained at 25 ± 1°C. The mixing time of each test sample was kept at around 40 min. The experimental results of a mixing time of 90 min showed that, for an experiment, the CLD of the particles remain unchanged after mixing for 40 min. In order to avoid the influence of the sample shear history on the experimental results, the experimental emulsion was reconfigured at the end of each experiment, and the experiment with shear rate from high to low was not set.

The experimental steps were as follows. Firstly, the D80 solvent oil and deionized water were configured according to the volume of oil content, and the amount of each phase was determined by the weighing method when taking the liquid from the material. Then, the gas switch of the particle sizer was opened, and the cleanliness of the probe of the particle sizer was tested by wiping it until the experimental conditions were met, when it was placed into the beaker and the probe into the experimental frame, the speed was adjusted and the measurement started. In order to eliminate errors caused by different measuring positions, the stirring paddle and measuring equipment were set at the 150-ml scale of the beaker to measure the distribution of the particle CL at different stirrer speeds when the moisture content was 10%, 20%, 30%, and 40%, respectively.

Table I. Properties of experimental fluids

Properties	D80 Oil (Exxol)	Deionized water
Density (kg/m^3)	792.5	992.6
Viscosity (mPa s)	1.77	0.94

CLD Under Different Oil Content

In order to investigate the influence of oil content on CLD, two groups of working conditions with oil contents of 10%, 20%, 30%, and 40% at 500 rpm and 10%, 20%, 30% and 40% at 650 rpm were selected for experiment. In this experiment, water was a continuous phase, and the oil content did not exceed the inverse point. After the experiment was stable, the CLDs given by the particle sizer at 500 rpm and 650 rpm are shown in Fig. 3. The abscissa in the figure is the CL, and the ordinate is the number of CLs swept per second.

It can be seen from Fig. 3 that the droplet distribution curves under various working conditions had a log-normal distribution, and that they had only one peak. Moreover, the changing trend of particle size distribution with oil content can also be seen: when the oil content increases, the peak of the log-normal distribution of the droplet distribution curve shifts to the right and the peak is larger. The wave crest moves to the right, indicating that the particle size distribution increases with the increase of oil content; The peak value of the wave is getting larger, mainly because, when the oil content increases, the number of droplets in the system increases. At the same speed, the number of droplets scanned by the particle sizer is larger.

CLD at Variable Speeds

In this experiment, in order to investigate the influence of rotation speed on CLD, two groups of working conditions with stirring speeds of 650 rpm, 800 rpm and 950 rpm at 20% oil content and 650 rpm, 800 rpm and 950 rpm at 30% oil content were selected for the experiment. In the experiment, water was a continuous phase, and the oil content did not exceed the inverse point. After the experiment was stable, the CLDs given by the particle sizer at three speeds of 20% and 30% are shown in Fig. 4. The abscissa in the figure is the CL, and the ordinate is the number of CLs swept per second.

As can be seen from the figure, the droplet distribution curves under each working condition had a nearly log-normal distribution, and that they had only one peak. At the same time, the changing trend of particle size distribution with rotating speed can be seen: when the rotating speed increases, the peak of lognormal distribution of droplet distribution curve shifts to the left, and the peak becomes larger. The wave peak shifts to the left, indicating that the particle size distribution tends to decrease with the increasing rotating speed. Therefore, the abscissa is the CL, and the CL becomes smaller when it goes to the left.

To sum up, machine-learning algorithms can be used to link different data more concisely. We fixed some parameters, such as ambient temperature, the diameter of impeller, and the viscosity of the solute and solvent to simplify our models. According to Srilatha et al.[10] for some parameters in mechanism models, like energy dissipation, turbulent kinetic energy, and the breakage rate of the drops, it is difficult to obtain accurate values due to the average properties, like the calculating timestep, the viscosity and density of the oil phase, and interfacial tension. These factors are always considered unchanged over the whole system. It is hard to describe oil–water dispersions precisely by simply physical formulas. To investigate a more effective way to predict CLD, we simplified some of the parameters, including the oil the properties, the dynamic effect of the impeller on the mixture, and the particle size to CUT, RPM, and CL separately. Furthermore, machine-learning algorithms including multiple linear regression (MLG), DNN, RF, and SVR were applied to build our prediction models.

MODELS BUILDING

In this work, we used machine-learning methods, including MLG, DNN, RF, and SVR, to predict the CLD of the water droplets and the prediction effects of the different algorithms were compared. Our

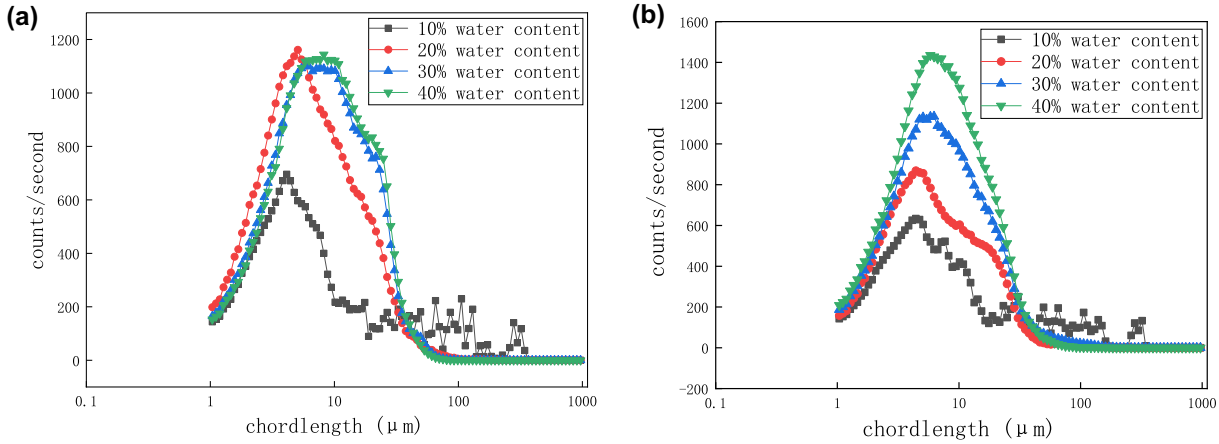

Fig. 3. (a) CLD measured by FBRM with oil contents of 10%, 20%, 30%, and 40% at 500 rpm, and (b) CLD measured by FBRM with oil contents of 10%, 20%, 30%, and 40% at 700 rpm.

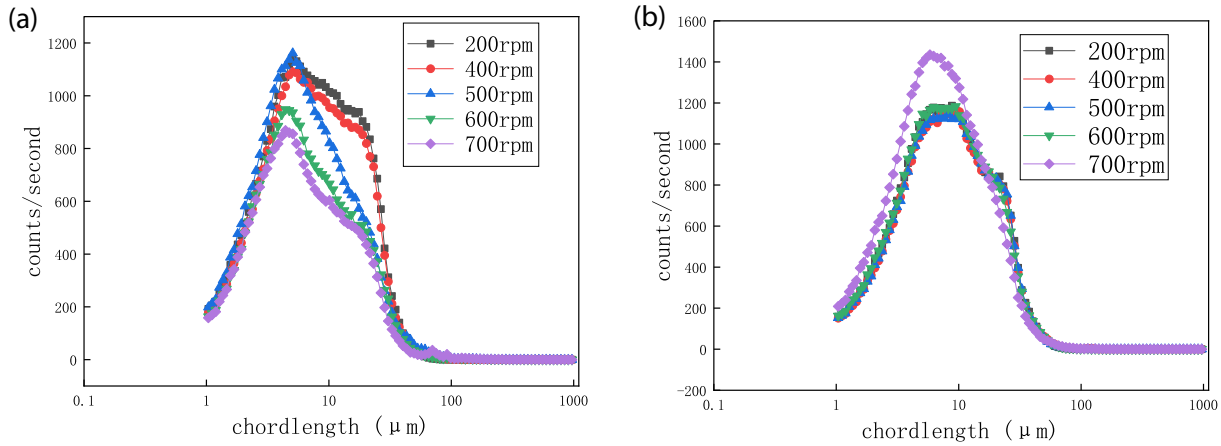

Fig. 4. (a) CLD measured by FBRM with 20% oil fraction at 200 rpm, 400 rpm, 500 rpm, 600 rpm and 700 rpm, and (b) chord length measured by FBRM with 40% oil fraction at 200 rpm, 400 rpm, 500 rpm, 600 rpm, and 700 rpm.

codes were accomplished using Python. In these models, we took CUT, RPM, and CL as independent variables, x, and COUNTS as the dependent value, y.

To detect and limit overfitting issues of our models, our experimental data were split into training, testing, and unseen data sets[22] before data scaling to avoid data leakage.[29] Of our data, 65% formed the training set, 20% the testing set, and 15% the unseen data. The training and testing sets were shuffled[30] using the *selection.-train_test_split* function. Also, the *optuna* framework[31] was used to tune the hyperparameters in the DNN model, especially the number of neurons and activation functions. Based on a genetic algorithm, the *geatpy* framework[32] was used to tune the hyperparameters in the SVR and RF models. We took the average value of 20 cross-validation scores on the training set as the optimization goal using the *cross_val_score* function. L1 L2 Regularization was used to reduce overfitting. Combined with the cross-validation results on training set, the best model was saved and the performances on the testing and unseen data sets were evaluated.

Overall, several functions were used to avoid overfitting and reduce the dimensions of the features. The statistical estimation was validated by satisfying the hypothesis testing procedure with respect to the form of a probability distribution using Kolmogorov–Smirnov goodness-of-fit tests.[33] Also, we used the *kstest* function in Python to judge whether the feature had a normal distribution. Box–Cox transformation[34] was used to alleviate the asymmetry of residuals. The PauTa criterion[35] was used to exclude anomalous equations in the phasor estimation process. We normalized and scaled our data using the *minmaxscalar* function in Python to reduce the interaction between the dimensions and the calculation times.

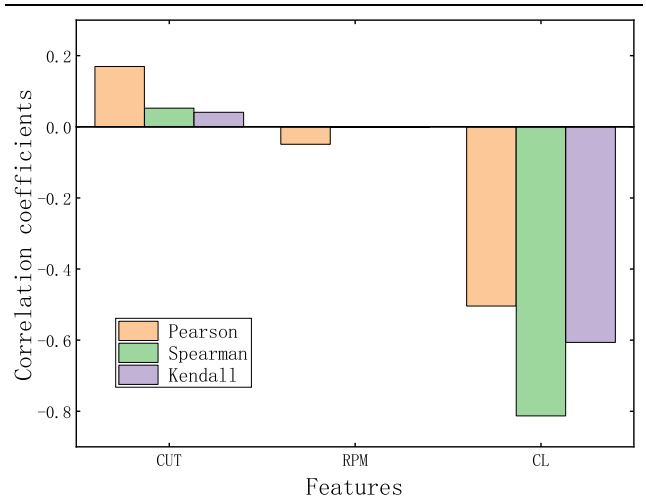

Fig. 5. Calculated correlation coefficients using Pearson, Spearman, and Kendall correlations.

To figure out which input factor has the greatest impact on the prediction results of CLD, Pearson,[36] Spearman ,and Kendall correlation coefficients were calculated.

In Fig. 5, we can see that the CL plays the most important role in the prediction of CLD, which is in line with physical reality because the CLD is observed based on the CL (as shown in Figs. 3 and 4). From the logarithmic axis in Fig. 4, it is obvious that the changes of CL and CLD are mostly negatively correlated. The feature RPM has the least influence on the value of CLD. Combined with the analysis of Oezkaya[37] and Wang,[4] it can be concluded that the influences of RPM on CLD are based on the value of CUT. When CUT is higher, the effect of shear on the particles will be more significant. Last but not the least, the correlation coefficients of CUT show that the properties of oil–water dispersion are also important. Also, more

experiments should be applied to explore the connection between the oil parameters, such as the viscosity, density, flash point, and CLD.

To measure the fluctuation of the prediction results, the mean absolute error (MAE) and mean square error (MSE) have been calculated to measure the deviation between the actual and predicted target values of y for n data points, as shown in Eqs. 1[22] and 2. We used an early-stopping[38] mechanism to optimize the training process: when the MAE of the model in the verification set began to decline, the training was stopped.

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n} y_i - \hat{y}_i \tag{1}$$

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{2}$$

where $\hat{y}_i$ is the predicted value for the ith entry in the input dataset. To evaluate the accuracy of our models, R^2 (goodness-of-art) was calculated. R^2 quantifies how close the predicted values are to the actual values, and is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \tag{3}$$

Model Based on MLG

To start with, we analyzed the linear nature between the inputs (CUT, RPM, CL) and the output (COUNTS). In theory, MLG can be used to fit the relationship between multiple independent variables and dependent variables, and the weight of each independent variable can be given according to the fitting results.[39] For the independent variables x, x_P, ... x_M, when studying their MLG to the dependent variable y, $x \sim x_M$ has a linear correlation to y. If the independent variables x_1, x_2, ... x_M were general variables that can be accurately measured and controlled, and the corresponding dependent variable is y, the MLG model is expressed according to Eq. 4, where $\beta_0, \beta_1, \beta_2 \ldots \beta_m$ represent unknown parameters and ε represents the random error.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_m x_m + \varepsilon \tag{4}$$

$$(y_i, x_{i1}, x_{i2}, \ldots, x_{im}), i = 1, 2, \ldots, n (n > m) \tag{5}$$

Equation 5 can be obtained through analysis of n groups of data. These are the basic principles of MLG.

In matrix form, this is shown as Eq. 6:

$$\vec{y} = \vec{X}\vec{\beta} + \vec{\varepsilon} \tag{6}$$

The number of N different equations can be derived from Eqs. 5 and 6, so we only need to solve the equations to obtain the solutions. In order to facilitate the estimation of the model, the following assumptions are made for Eq. 4:

I. The independent variables (x, x_P, ..., x_m) are deterministic variables, not random variables, the number of rows and columns of the matrix consisting of measurement data transformed from Eq. 6 is required to be m + 1 < n (m is the number of rows and n is the number of columns);
II. Random error term ε Satisfy as $E(\varepsilon) = 0$ and $Var(\varepsilon) = \sigma^2$;
III. It is assumed that the random error term obeys the normal distribution.

Therefore, the evaluation of MLG algorithm can reveal the linear nature between the inputs and the output. The prediction results on the test data are shown in Fig. 6.

From Fig. 6, it can be seen that the MLG model shows a certain accuracy in predicting the trend, but there are still large deviations between the predicted values and the real values. According to our calculation, the value of R^2 for the testingset is less than 0.2, even after being optimized by the genetic algorithm, and high scores of MAE and MSE (around 0.25) are still high. These results suggest that MLG may not fit our datasets very well.

From Fig. 5, considering that CLD can be seen as a function of CL, we can see that CUT influences CLD very significantly. As shown in Fig. 7, although the value of the feature CL influences CL the most, it is obvious that the variation of CL is not linear.

To sum up, it can be concluded that the impacts of the three input features (CUT, RPM, and CL) on the dependent variable (COUNTS) are not linear.

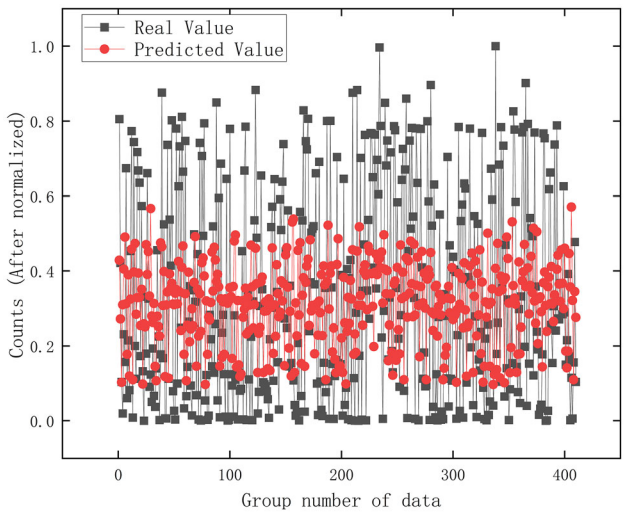

Fig. 6. Prediction results of MLG on the testing set.

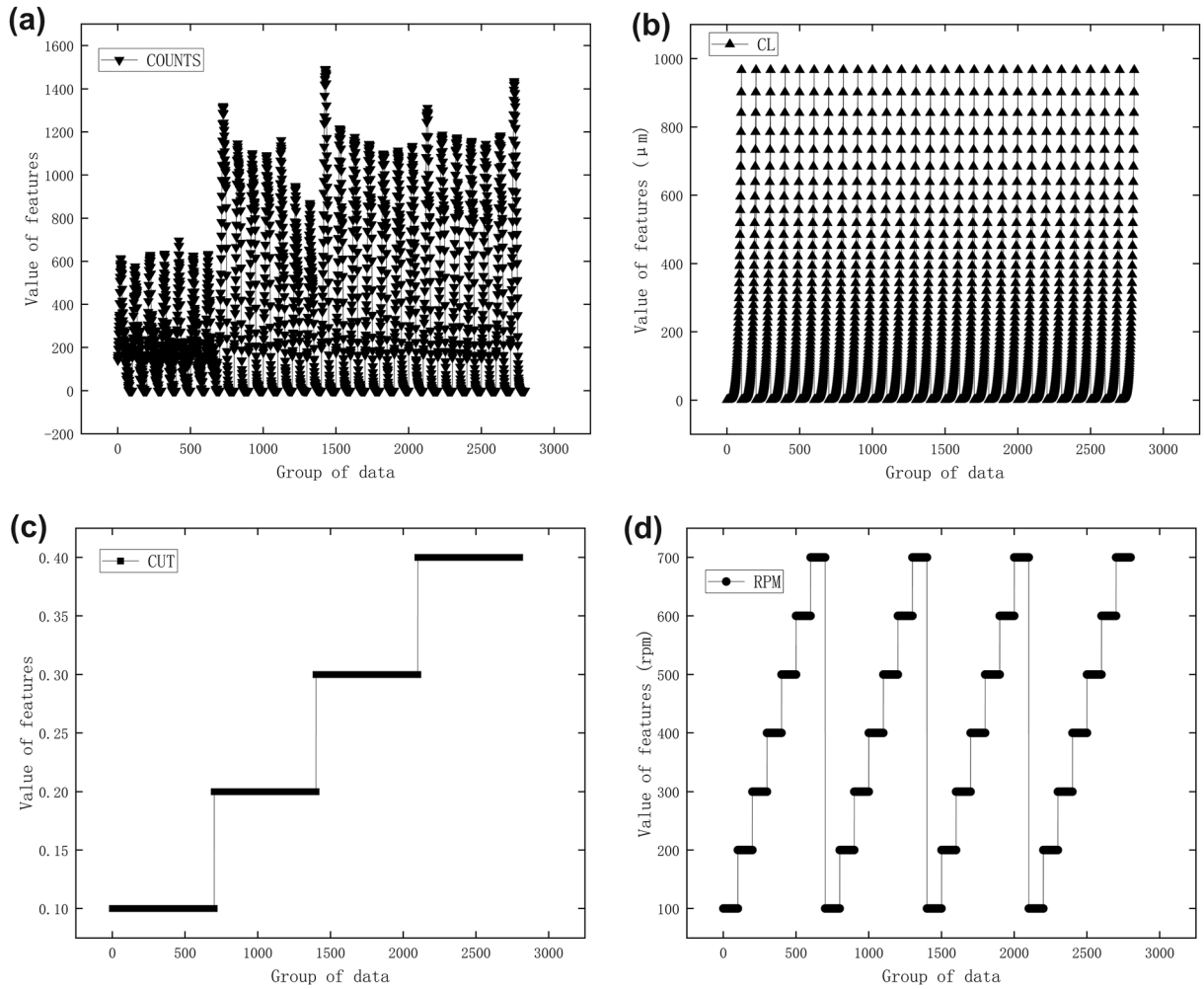

Fig. 7. (a) The values of the dependent variable, COUNTS; (b) of CL; (c) of CUT; and (d) of RPM.

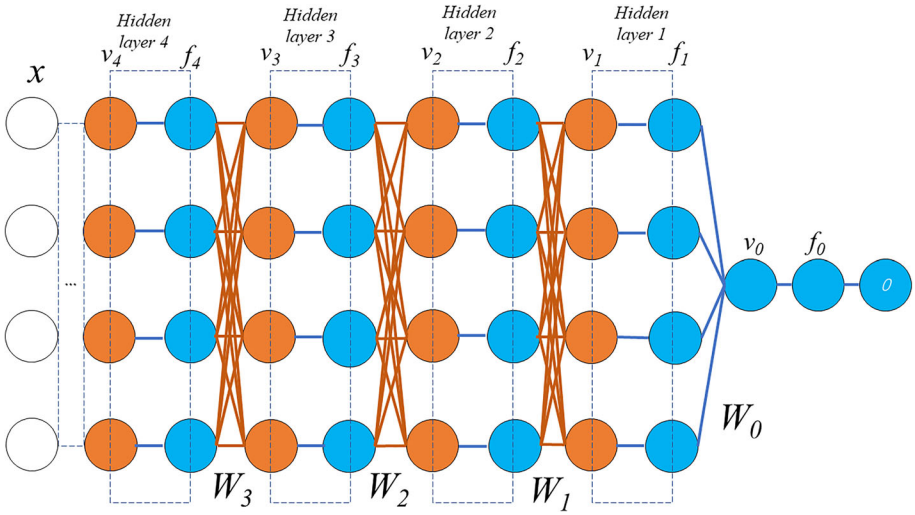

Fig. 8. Structure diagram of DNN.

DNN Model

Generally, neural networks can provide processing methods for a large amount of data. As shown in Fig. 8, this is a common structure of DNN, which is composed of an input layer, five hidden layers, and an output layer.

As shown in Fig. 8, x represents the input layer, $v_1 \sim v_4$, $f_1 \sim f_4$ represent the hidden layers, and v_0 and f_0 represent the output layer. For an intermediate layer I, its input or output mathematical expression can be written as shown in Eqs. 7 and 8.

$$v_{out}^{(i)} = X_{in}^{(i)} w^{(i)} + b^{(i)} \tag{7}$$

$$f_{out}^{(i)} = ACT^{(i)} v_{out}^{(i)} \tag{8}$$

In Eq. 7, $X_{in}^{(i)}$ is the input matrix of each layer, $w^{(i)}$ is the weight matrix of each layer (i indicating different nerve layers), $b^{(i)}$ is the offset of each layer, and $v_{out}^{(i)}$ is the output of each layer. In Eq. 8, f is the final output of each layer, and ACT(i) is the activation function selected for each layer.

The derivative of each layer parameter is solved, as shown in Eq. 9.

$$\frac{\partial E}{\partial W^{(i)}} = X_{out}^{(i)T} \times Error \tag{9}$$

As shown in Eq. 9, $\frac{\partial E}{\partial W^{(i)}}$ represents the derivative of the parameters of each layer, $X_{out}^{(i)}$ represents the transpose of the input of each layer, and Error represents the error, where this is equal to the derivative of the loss function to the network output.

According to the above analysis, the recurrence Eq. 10 of s neural network weight matrix component can be obtained, and, as shown in the equation, Δw is the variation of the component of the neural network matrix:

$$\Delta w = -\alpha(0 - y) \prod_{i=0}^{k-1} w_i f_i'(v_i) f_i'(v_k) g(k) \tag{10}$$

In Eq. 10, α represents the learning rate, and $f_i'(v_i)$ and $f_i'(k_i)$ represent the derivatives of the corresponding activation function, respectively.

Therefore, a four-layer neural network is established by using characteristic variables for CUT, RPM, and CL as input variables and the particle number as output variables.

The optuna[31] optimizer was used to tune the hyperparameters in DNN. In detail, the framework can optimize the parameters of the neural network based on the set of the number range of each layer, a random inactivation rate, and the activation function.[40] The best model was saved after training. The prediction of COUNTS on the testing set is shown in Fig. 9.

In the tuning process, we mainly focus on the activation functions, learning rate, and the number of neurons in each layer. The number of neurons in the best saved model are $3 \times 2 \times 4$, and the learning rate is 0.0178.

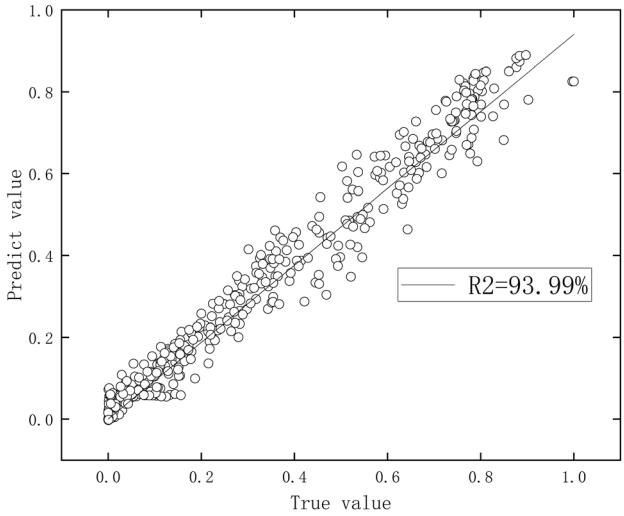

Fig. 9. The prediction of COUNTS using the DNN model on the testing set.

$$LINEAR(x) = x \tag{11}$$

$$RELU(x) = \max(0, x) \tag{12}$$

$$SELU(x) = scale \times (\max(0, x) + \min(0, \alpha e^x - 1)) \tag{13}$$

$$TANH(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{14}$$

Our optimizer searched the best activation functions in LINEAR, RELU, SELU, and TANH. After tuning, the activation function TANH, SELU, and TANH were applied on the layers 1, 2, and 3, respectively. According to Eq. 12, RELU can maintain the derivative activation function at 1, so the problems of gradient disappearance and gradient explosion can be effectively alleviated. Also, overfitting can be solved in some extent. However, RELU will kill neurons when the input $x < 0$. In Eq. 13, Klambauer[41] deduced the value of scale and α, and the code can be seen on github.[42] Compared with RELU, SELU is more flexible, but the effect of the convolution network has not been proved, and it may cause overfitting. According to Eq. 14, the mean value of TANH is 0. Therefore, TANH can distinguish features better.

By our evaluation, the MSE of our DNN model on the test data is 0.47%, and the MAE is 5.22%. As shown in Fig. 9, the R^2 is over 93%. To sum up, the DNN model based on the optuna optimizer can be used to predict the CLD in a steady state of shear.

Model Based on RF

RF is an algorithm that integrates multiple decision trees through the idea of integrated learning.[43] As mentioned in Sect. "DNN Model", our data are pretreated by the use of the PauTa criterion to reduce the data noise which may lead to overfitting. We used the sklearn.ensemble in Python to apply the RF algorithm.

The basic calculation process of RF is shown in Fig. 10.

In the RF algorithm, the values of n_estimators should be carefully tuned, because it can cause significant overfitting. The values of min_samples_leaf present the minimum number of samples in newly created leaves. We set min_samples_leaf to the default value of 1. After tuning, the value of n_estimators is 61, and the value of min_sample_split is 2. By the use of the genetic algorithm, the parameters (n_estimator $\times$ min_sample_split) are controlled. Other factors such as random_state and max_features, can be seen in our code.[44] The R^2 of this RF model is 99.18% on the test data.

Model Based on SVR

SVR is a machine-learning algorithm based on statistical learning theory, which was first proposed by Vapnik.[45] The principle of SVR is that the error rate of machine learning on test data is bounded by the sum of the training error rate and a term dependent on the Vapnik–Chervonenkis dimension.[46] In the case of the separable mode, the value of SVR for the former term is zero and the second term is minimized. Therefore, although SVR does not use the domain knowledge of the problem, it can still provide a good generalization performance in pattern classification. This attribute is unique to SVR.

We tuned the hyperparameters in sklearn.svm.SVR. The parameter epsilon specifies the epsilon-tube within which no penalty is associated in the training loss function, with points predicted within a distance epsilon from the actual value. We set epsilon as the default (0.1). The rbf kernel was used in our model. In the saved best SVR model, hyperparameter C is 1.0, and gamma is 4.053. If the value of gamma is too high, overfitting will be caused, while if the value is too low, the accuracy on the testing set will not be guaranteed. On the testing set, R^2 is 95.03%, MSE is 0.39%, and MAE is 5.30%.

From our analysis above, it is obvious that the connection between the input features (CUT, RPM and CL) on the dependent variable (COUNTS) can be evaluated by the use of the SVR (RBF) algorithm. From our model building described above, we can conclude that the connection between input and output is nonlinear.

RESULTS AND DISCUSSION

Under our evaluation mentioned above, it can be concluded that the connection between input and output is nonlinear. Many algorithms and functions have been applied to avoid overfitting. The hyperparameters of our models are shown in Table II.

It is obvious that the values of MSE, MAE, and R^2 have a high reliability in the evaluation of the accuracy of algorithms. With the use of the optuna optimizer and the genetic algorithm, the tuned hyperparameters can better fit the algorithms,

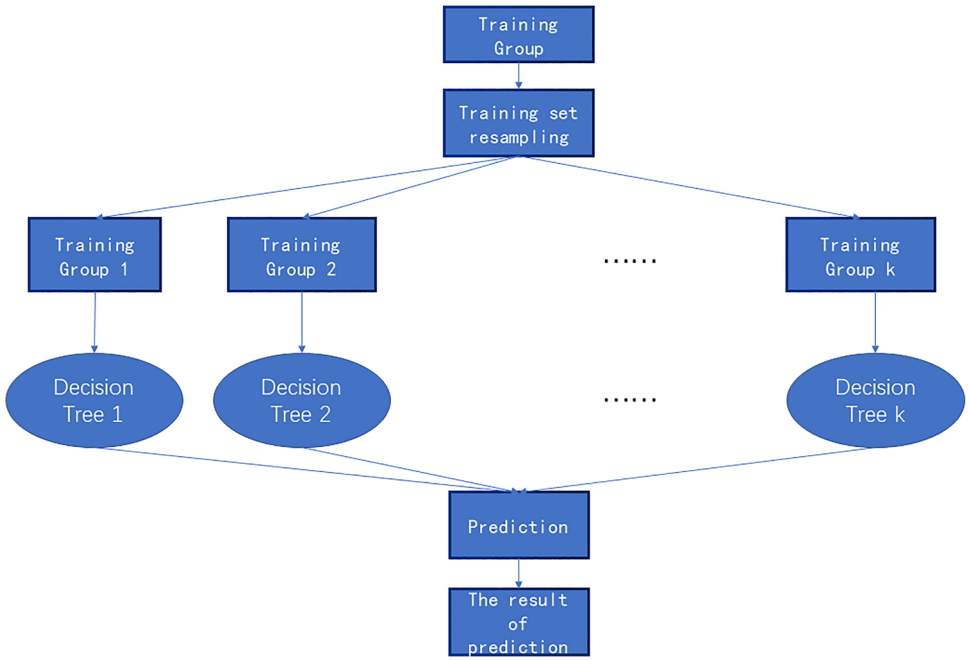

Fig. 10. The basic calculation process of RF.

Table II. Hyperparameters of best saved models

Method	Hyperparameters	Function/value
DNN	Learning rate	0.017
	Activation functions of each layer	TANH; SELU; TANH
	Number of neurons (layer 1 × layer 2 × layer 3)	3 × 2 × 4
RF	n_estimators	61
	min_samples_split	2
	min_samples_leaf	1
SVR (RBF)	Kernel	rbf
	Gamma	4.05
	C	1.0

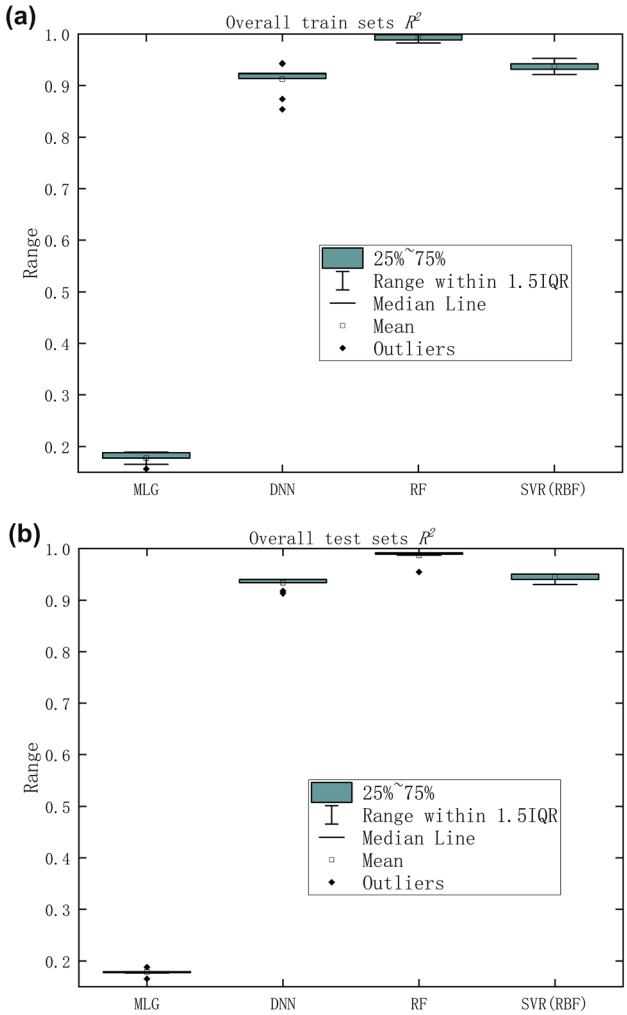

Fig. 11. (a) Overall training sets performance evaluation of DNN, RF, and SVR (RBF); (b) overall testing sets performance evaluation of DNN, RF, and SVR (RBF).

which shows that RF has the best accuracy to predict CLD. In addition, the R^2 values of 20 different splits of data to the testing and training sets are shown in Fig. 11 to identify the impact of their variation on each model performance. It is worth mentioning that each point in Fig. 11 is the score of a model on one distinct split data to the training/testing sets; i.e., the highest average and least deviations of the training set R^2 value belong to RF with mean testing set R^2 values of nearly 0.99 and deviation of 0.01. On the other hand, the testing set R^2 values, as expected, are close to the evaluation of the overall testing sets. Also, from Table III, it can be seen that the values of MSE and MAE in each model between the training and testing sets are very close. This result shows that the overfitting is controlled to some extent. Thus, DNN, RF, and SVR (RBF) can be used to predict CLD, with RF showing a relatively stable behavior.

According to the analysis of Gukeh et al.,[22] to further investigate the universality of each model, a new set of unseen data were split before data scaling. A total of 300 groups of experimental data were split as unseen data. Figure 12 shows the predicted versus measured counts for all the methods. To make the trend more clearly, we have normalized and denoised our data. Figure 12 shows that our nonlinear models can be used to describe the trend of CLD. Also, Table IV shows the corresponding metrics of Fig. 12, which is the same as the condition on the training and testing data. MLG still performs poorly on the unseen data, suggesting overfitting, while other nonlinear models with overall $R^2 > 0.93$ predict CLD with fairly high accuracy. RF has been consistently obtaining the maximum R^2 value (> 0.99) and minimum MAE of 0.0159 on the unseen data, which shows that it can be used to capture the pattern and trend of CLD on the unseen data.

CONCLUSION

Based on our experimental data, CLD in oil–water dispersions can be described and predicted by the use of machine-learning algorithms. We used several methods to normalize our data and to reduce the overfitting. In total, four algorithms have been used to predict CLD, by our evaluation. The overall performance of each model was evaluated by 20 different training and testing sets, with 65% of the data employed for each training model, with 20%

Table III. Evaluation of models

	Training set				Testing set			
Method	R^2	**MAE**	**Expected variance**	**MSE**	R^2	**MAE**	**Expected variance**	**MSE**
MLG	0.1802	0.2016	0.1802	0.0569	0.1808	0.2146	0.1847	0.0637
DNN	0.9237	0.0168	0.9357	0.0044	0.9399	0.0522	0.9517	0.0047
RF	0.9986	0.0067	0.9986	0.0001	0.9918	0.0175	0.9926	0.0006
SVR	0.9428	0.053	0.9490	0.0038	0.9503	0.0530	0.9546	0.0039

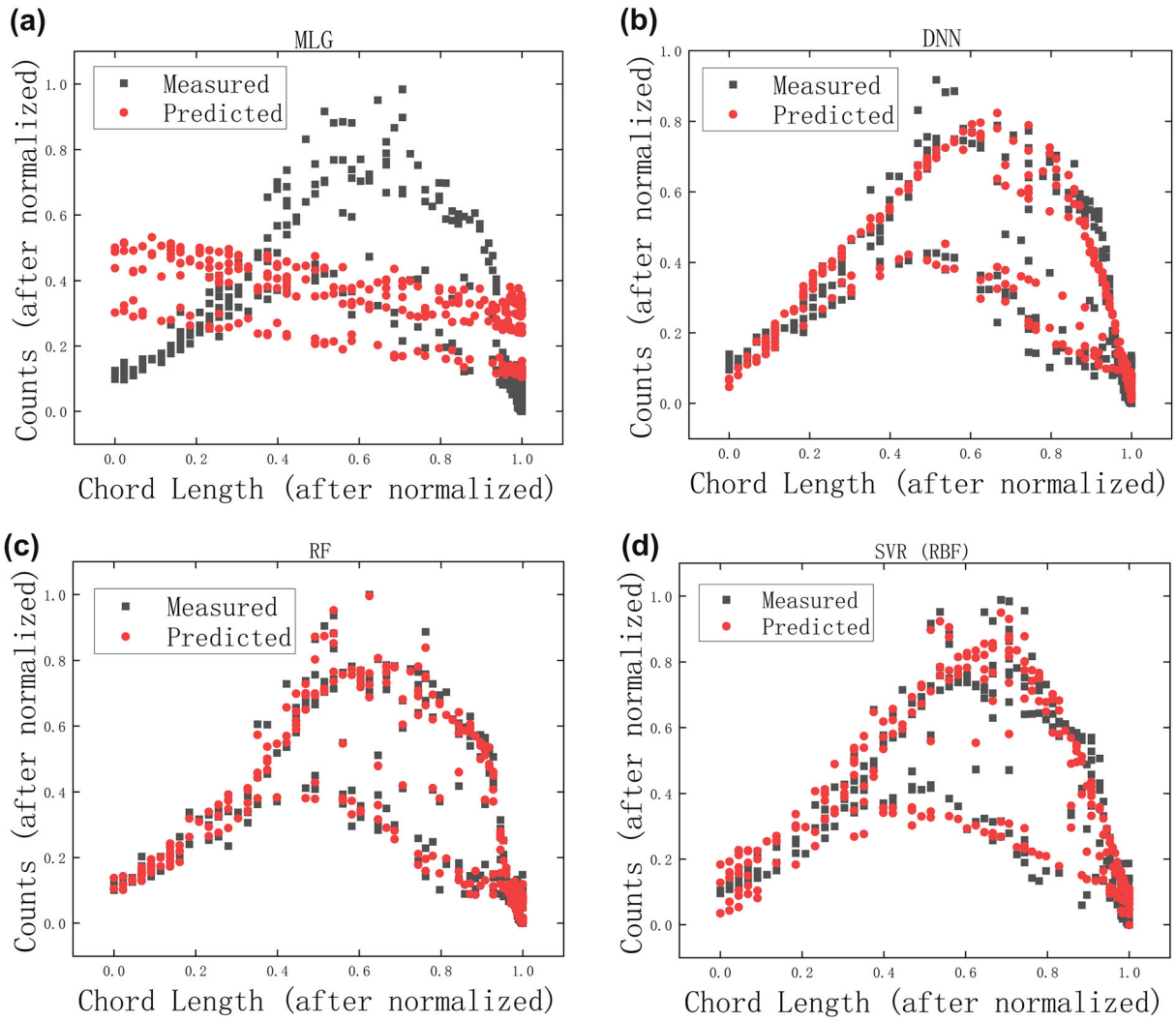

Fig. 12. (a) Prediction of CLD on unseen data by the MLG model; (b) prediction of CLD on unseen data by the DNN model; (c) prediction of CLD on unseen data by the RF model; (d) prediction of CLD on unseen data by the SVR (RBF) model.

Table IV. Evaluation of models on unseen data

Method	**Expected variance**	R^2	**MSE**	**MAE**
MLG	0.1559	0.1493	0.0559	0.1996
DNN	0.9464	0.9442	0.0034	0.0458
RF	0.9927	0.9926	0.0005	0.0159
SVR (RBF)	0.9523	0.9484	0.0039	0.0527

used for testing, and 15% for unseen data. We scaled and optimized our model after the data split. The model based on MLG revealed the poor linear nature of the data. Other algorithms (DNN, RF, SVR) can be well tuned and fitted for our datasets. Pearson, Spearman, and Kendall correlations were also used and it was found that oil CUT mostly influences CLD. By comparing our calculation results, we found out that all three algorithms obtained a high value of R^2 (above 0.85) while RF can be used to predict best CLD. To further investigate the universality of each model, we evaluated our models on the unseen data set. RF also showed the most stable performance with the lowest variation in all accuracy metrics and highest R^2 of 0.9926 on the unseen data set, which supports the employment of this method for future designs and applications. The present approach shows that machine learning can be used to predict CLD well. Our evaluation steps can be followed to solve other technological problems in oil fields.

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

REFERENCES

1. S. Al-Zuhair, K.B. Ramachandran and M. Hasan, *J. Chem. Technol. Biotech.* 79, 706 https://doi.org/10.1002/jctb.1039 (2004).
2. F. Laurenzi, M. Coroneo, G. Montante, A. Paglianti and F. Magelli, *Chem. Eng. Res. Des.* 87, 507 https://doi.org/10.1016/j.cherd.2008.12.007 (2009).
3. K. Piela, R. Delfos, G. Ooms, J. Westerweel, R.V.A. Oliemans and R.F. Mudde, *Int. J. Multiph. Flow* 32, 1087 https://doi.org/10.1016/j.ijmultiphaseflow.2006.05.001 (2006).
4. W. Wang, W. Cheng, J. Duan, J. Gong, B. Hu and P. Angeli, *Chem. Eng. Sci.* 105, 22 https://doi.org/10.1016/j.ces.2013.10.012 (2014).
5. S. Maaß, S. Wollny, A. Voigt and M. Kraume, *Exp. Fluids* 50, 259 https://doi.org/10.1007/s00348-010-0918-9 (2010).
6. M.W. Hermanto, P.S. Chow and R.B.H. Tan, *Ind. Eng. Chem. Res.* 51, 13773 https://doi.org/10.1021/ie301626c (2012).
7. C.E. Crestani, A. Bernardo, C.B.B. Costa and M. Giulietti, *Powder Technol.* 384, 186 https://doi.org/10.1016/j.powtec.2021.01.075 (2021).
8. C.J. Gommes, Y. Jiao, A.P. Roberts and D. Jeulin, *J Appl Crystallogr.* https://doi.org/10.1107/S1600576719016133 (2020).
9. E. Charlafti, J. Steinhoff, L. Hohl, Z. Huang, L. Reinecke, H.-J. Bart and M. Kraume, *Chem. Eng. Res. Des.* 168, 465 https://doi.org/10.1016/j.cherd.2021.02.004 (2021).
10. C. Srilatha, V.V. Morab, T.P. Mundada and A.W. Patwardhan, *Chem. Eng. Sci.* 65, 3409 https://doi.org/10.1016/j.ces.2010.02.035 (2010).
11. D. Pinelli, A. Bakker, K.J. Myers, M.F. Reeder, J. Fasano and F. Magelli, *Chem. Eng. Res. Des.* 81, 448 https://doi.org/10.1205/026387603765173709 (2003).
12. J.J. Derksen and H.E.A. Van Den Akker, *Chem. Eng. Res. Des.* 85, 697 https://doi.org/10.1205/cherd06161 (2007).
13. M. Asadollahzadeh, R. Torkaman, M. Torab-Mostaedi and J. Safdari, *Chem. Eng. Res. Des.* 117, 637https://doi.org/10.1016/j.cherd.2016.08.025 (2017).
14. Y. Masoudi-Sobhanzadeh, H. Motieghader, Y. Omidi and A. Masoudi-Nejad, *Sci Rep* 11, 3349. https://doi.org/10.1038/s41598-021-82796-y (2021).
15. J. Carrasquilla and R.G. Melko, *Nat. Phys.* 13, 431 https://doi.org/10.1038/nphys4035 (2017).
16. G.B. Goh, N.O. Hodas and A. Vishnu, *J Comput Chem* 38, 1291 https://doi.org/10.1002/jcc.24764 (2017).
17. M.A. Ahmadi and Z. Chen, *Petroleum* 5, 271 https://doi.org/10.1016/j.petlm.2018.06.002 (2019).
18. F. Nelli, *Python Data Analytics: With Pandas, Numpy, and Matplotlib*, (2018).
19. M. Giulietti, R. Guardani, C.A.O. Nascimento and B. Arntz, *Chem. Eng. Tech.* 26, 267 https://doi.org/10.1002/ceat.200390039 (2003).
20. A. Langenbucher, N. Szentmary, A. Cayless, J. Wendelstein and P. Hoffmann, *Acta Ophthalmol* 1, 1 https://doi.org/10.1111/aos.15108 (2022).
21. P. Lopez-Exposito, A.B. Suarez and C. Negro, *J Appl Phycol* 28, 2315 https://doi.org/10.1007/s10811-015-0749-4 (2016).
22. M. Jafari Gukeh, S. Moitra, A.N. Ibrahim, S. Derrible and C.M. Megaridis, *ACS Appl. Mater. Interf.* https://doi.org/10.1021/acsami.1c13262 (2021).
23. C. Risk and P.M.A. James, *Earth Space Sci.* 9, 1 https://doi.org/10.1029/2021ea002019 (2022).
24. D. Zhu, C. Qian, C. Qu, M. He, S. Zhang, Q. Tu and W. Wei, *Int. J. Adv. Manuf. Tech.* https://doi.org/10.1007/s00170-022-08836-7 (2022).
25. P. Hu, L. Liang, B. Li and W. Xia, *Fuel* 286, 1119445 https://doi.org/10.1016/j.fuel.2020.119445 (2021).
26. A. Khalil, F. Puel, Y. Chevalier, J.-M. Galvan, A. Rivoire and J.-P. Klein, *Chem. Eng. J.* 165, 946 https://doi.org/10.1016/j.cej.2010.10.031 (2010).
27. M. Cozier, *Biofuel. Bioprod. Biorefin.* 13, 432 (2019).
28. C. Wei, In *College of Mechanical and Storage and Transportation Engineering*, (China University of Petroleum, Beijing: China University of Petroleum, Beijing, 2013).
29. S. Kaufman, S. Rosset, C. Perlich and O. Stitelman, *ACM Trans. Knowl. Discov. Data* 6, 1 https://doi.org/10.1145/2382577.2382579 (2012).
30. H. Gao and T. Gao, *Clust. Comput.* 25, 707 https://doi.org/10.1007/s10586-021-03446-6 (2021).
31. T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama and M. Assoc Comp, In *25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, (Assoc Computing Machinery: Anchorage, AK, 2019), pp 2623-2631.
32. P. Zibin, "Geatpy – The Genetic and Evolutionary Algorithm Toolbox for Python with High Performance." (2020), http://geatpy.com/index.php/home/. Accessed 01/07 2020.
33. H. Ku and J.H. Maeng, *Nat. Hazard.* 107, 617 https://doi.org/10.1007/s11069-021-04598-9 (2021).
34. L. Fang, Z. Zhou and Y. Hong, *Symmetry* 14, 22 https://doi.org/10.3390/sym14010022 (2021).
35. Y. Liang, Z. Zhang, H. Li, J. Ding, G. Wang, L. Chen and I.E.T. Generation, *Transm. Distrib.* https://doi.org/10.1049/gtd2.12408 (2022).
36. M. Cappelloni, M. Gallo and A. Cesarani, *Ital. J. Anim. Sci.* 21, 488 https://doi.org/10.1080/1828051x.2022.2050471 (2022).
37. E. Oezkaya, S. Michel and D. Biermann, *Adv. Manuf.* https://doi.org/10.1007/s40436-021-00383-w (2022).
38. Y.L. Zhang, L.F. Li and Acm, In *28th ACM International Conference on Information and Knowledge Management (CIKM)*, (Assoc Computing Machinery: Beijing, PEOPLES R CHINA, 2019), pp 2053-2056.
39. C.J. Fraza, R. Dinga, C.F. Beckmann and A.F. Marquand, *Neuroimage* 245, 118715 https://doi.org/10.1016/j.neuroimage.2021.118715 (2021).
40. O. Eguasa, E. Edionwe and J.I. Mbegbu, *J Appl Stat*. https://doi.org/10.1080/02664763.2022.2026895 (2022).
41. G. Klambauer, T. Unterthiner, A. Mayr and S. Hochreiter, "Self-Normalizing Neural Networks" (2017), https://arxiv.org/pdf/1706.02515.pdf.
42. G. Klambauer, "Tutorials and implementations for "Self-normalizing networks"(SNNs) " (2017), https://github.com/bioinf-jku/SNNs.

43. V. Plakman, J. Rosier and J. van Vliet, *GI Sci. Remote Sens.* 59, 461 https://doi.org/10.1080/15481603.2022.2036056 (2022).
44. Y. LI, "Code-to-predict-CLD" (2022), https://github.com/LiYunchao2CityU/Code-to-predict-CLD. Accessed 04 April 2022.
45. V. Vapnik and R. Izmailov, *Ann. Math. Artif. Intell.* 81, 3 https://doi.org/10.1007/s10472-017-9538-x (2017).
46. J. Zhang, T. Liu and D. Tao, *IEEE Trans Neur. Netw. Learn. Sys.* https://doi.org/10.1109/tnnls.2021.3109942 (2021).