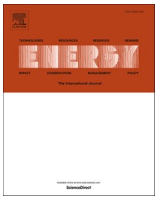
Energy 280 (2023) 128088

Contents lists available at ScienceDirect

Energy

journal homepage: www.elsevier.com/locate/energy



Deep learning models in Python for predicting hydrogen production: A comparative study

Sheila Devasahayam

WASM: Minerals, Energy and Chemical Engineering, Curtin University, Australia

ARTICLE INFO

Handling editor: Krzysztof (K.J.) Ptasinski

Keywords:
Predicting hydrogen production
Co-gasification
Deep learning models
Model evaluation

ABSTRACT

This study relates to predicting hydrogen production using deep learning models. The co-gasification of biomass and plastics dataset used gasification temperature, particle size of biomass rubber seed shell (RSS) and High-Density Polyethylene (HDPE), and the amount of plastic in the mixture as the independent variables, and the amount of hydrogen produced as the dependent variable. It was found that during the co-gasification particle size is a controlling factor for hydrogen production due to the influence on surface reactions, while temperature had no significant effect. The neural network models were developed using Keras and two different architectures were compared with and without L1 and L2 regularizers. The values for L1 and L2 are determined using the gridserach: for the 1 archtecture, the ideal L1 value = 0.010; and the ideal L2 value = 0.000001 and for the 2nd architecture, The ideal L1 value is 0.100; and the ideal L2 value is 0.000010 using the lowest mean squared error values for the test sets. The mean cross-validation scores indicated that the second architecture performed better. The mean cross_val_score using the negative mean square error, for the 1st architecture, with l2 regularizers (0.000001) is determined as −20.05 (13.10) nMSE for Kfold, 10; and for the 2nd architecture l2 regularizers (0.000010) as −8.22 (7.77) nMSE for Kfold, 10, indicate the 2nd architecture performs better. The best model parameters for both architectures were determined using Grid Search CV. The best model hyperparameters using Grid Search is batch_size, 3; epochs,100; optimizer, rmsprop for the first architechure with negative mean square error, −20.95; and for the 2nd architecture, batch_size, 5; epochs,100; optimizer, adam with negative mean square error, −7.38, indicating the 2nd architecture to be a better model. The Keras Wrapper improved the performance of the model for the first architecture, but not for the second architecture. The permutation feature importance for architecture 1 (in descending order) is: size of RSS, size of HDPE, per cent plastics in mixture and temperature. For architecture 2, in descending order: size of HDPE, size of RSS, per cent plastics in mixture and temperature. Overall, the study demonstrates the potential of deep learning models for predicting hydrogen production.

1. Introduction

Plastics and biowastes can reduce the carbon footprint of industrial processes like iron and steel and cement industries, while simultaneously converting the carbon to clean energy like hydrogen and syngas [1,2]. Co-gasification of plastic and biomass mixtures yields H2 via dry and steam reformation of CO2, and factors like feed composition and catalyst type influence the conversion of waste plastic to fuel products [3–7]. Factors contributing to the production of H_2 include the temperature, plastics/biomass ratio, CO_2/CH_4 ratios and catalysts [2,8–11].

The significance of this study relates to the ability to predict the H2 production using AI and machine learning for optimizing H2 production, reducing costs, improving sustainability, determining scalability, and fostering innovation. Generating H_2 from waste or renewable sources can address energy and environmental challenges, such as reducing greenhouse gas emissions, enhancing energy security, and minimizing waste going to landfills. By analyzing large amounts of data and identifying patterns, it will be possible to optimize and predict future H2 production processes using machine learning algorithms.

Waste plastics, such as polyethylene and polypropylene, have relatively high volatile content, viscosity, and heating value, but very low moisture and ash content. Among the different plastics tested, polypropylene is the most beneficial for producing hydrogen. However, plastics require more energy to gasify and produce less hydrogen compared to biomass, which contains high levels of hydrogen-rich compounds, such as cellulose and hemicellulose, and lignin [12,13].

E-mail address: sheila.devasahayam@curtin.edu.au.

https://doi.org/10.1016/j.energy.2023.128088
Received 17 February 2023; Received in revised form 6 June 2023; Accepted 9 June 2023
Available online 19 June 2023

Co-pyrolysis and co-gasification convert waste plastics and biomass into H2 and other valuable products, increasing resource efficiency. Co-gasification is more efficient than biomass gasification, improving product yields, heating value, and carbon conversion to gas [14]. This process can reduce the environmental impact of waste plastics by converting them into useful gases rather than landfilling or incinerating them. Co-pyro-gasification of plastics and biomass has benefits such as improving syngas quality and composition. H2 production through these thermochemical processes offers a promising solution for mixed wastes by reducing the need for waste separation [12,13], [15].

The co-pyrolysis of waste plastics and biomass can improve the quality and uniformity of the products while minimizing coke deposition. The optimal temperature for co-pyrolysis is typically in the range of 450 °C–550 °C, which can increase the rate of reaction and yield of H_2, while also reducing the formation of undesirable by-products. The opposite composition of high oxygen to carbon ratio and low hydrogen to carbon ratio of biomass during co-pyrolysis results in increased product quality and uniformity, further minimizing coke deposition.

Co-pyrolysis requires more energy input than co-gasification, but it can reduce greenhouse gas emissions with biochar residue, while co-gasification can potentially produce a renewable product with a renewable feedstock. Co-pyrolysis of mixed wastes of biomass and plastics requires less activation energy than waste plastic, but more than the solid biomass [16]. The energy usage for co-pyrolysis falls in between that of waste plastic and solid biomass [17]. It was reported Co-pyrolysis of HDPE with pine sawdust enhances H_2 production and reduces catalyst deactivation and breaks down organic matter into smaller molecules, releasing H_2 [18]. The blend ratio of plastic to biomass is a crucial factor in determining the synergistic effects in co-pyrolysis. The ratio of biomass to waste plastics has a direct impact on the H_2 yield, with a higher biomass-to-plastic ratio leading to higher H_2 yields due to the availability of more hydrogen-rich compounds for conversion. Increasing the plastic-to-biomass ratio up to 5 improves H_2 and CO yields.

Raising the operating temperature and air-to-fuel equivalence ratio (ER) during plastic gasification increases gas yield and reduces tar yield. As temperature rises, CO, CH_4, H_2, and C_2H_2 yields increase while CO_2 and C2–C3 hydrocarbons decrease. As ER rises, CO, CH_4, H_2, and C2–C3 hydrocarbon concentrations decrease, and CO_2 concentrations increase. The H_2/CO ratio increases with temperature and decreases with ER. The highest carbon conversion efficiency occurs at 800 °C [19]. Co-gasification can result in a more balanced fuel mix. For example, biomass provides a high amount of hydrogen-rich compounds, while plastics provide a source of carbon-rich compounds. This balanced mix of feedstocks can result in a more complete gasification process and a higher yield of valuable gases, including hydrogen. Plastics can also increase the energy content of the feedstock, leading to a higher overall energy yield from the gasification process.

The co-gasification of biomass and plastics can have complementary benefits, such as reducing the formation of tars that can clog reactors. This is because the properties of the two feedstocks are complementary, leading to a more efficient gasification process. According to Pinto et al. adding plastics to pine wastes decreased CO content but increased H2 release up to 50% (v/v).

In one study, co-gasification of PE waste and biomass increased product gas volume and hydrogen yield. The highest H2 content of 76.18 vol% was achieved with 25 wt% of PE mixed with palm kernel shell at 800 °C [20]. Another study found that at 900 °C, gas output increased with higher HDPE concentration in the feed, with pure HDPE producing more than 2.5 times the gas output of biomass. Increasing the amount of HDPE in the combination from 25% to 50% increased H2 concentration from 40% to 57%, but no further increases in HDPE produced more H2 [12,21,22].

Another study reported increased product gas volume percent and hydrogen yield for co-gasification of PE waste with biomass. The highest H_2 content 76.18 vol% was achieved at 800 °C using 25 wt % of PE mixed with palm kernel shell [20]. At 900 °C, the gas output grew as the feed's HDPE concentration rose, and it was more than 2.5 times higher for pure HDPE than for biomass. The H_2 concentration grew from 40% to 57% when the amount of HDPE in the combination was increased from 25% to 50%, but no further increases in HDPE produced more H_2 [12].

The particle size of the waste plastics and biomass affects the reaction rate and the yield of H_2, with smaller particle sizes leading to higher surface area and higher reaction rates. The ratio of the mixture has only a small effect on gas composition. The rise of temperature favoured the formation of H2 and decreased the formation of hydrocarbons, tars, and char. At 885 °C and in presence of 40% (w/w) of plastic, conversion to char is around 2%, whilst feedstock conversion to gas is around 90% [23].

This study examines the use of Artificial Neural Network (ANN) in deep learning (DL) machine learning (ML) algorithms to predict hydrogen production by co-gasification. DL methods simplify the feature engineering process in supervised learning. The neural networks architecture consists of several layers, and the simplest DL model is the Multi-Layer Perceptron (MLP) used for regression and classification problems [24,25].

1.1. Neural networks architecture

1.1.1. Layers in deep neural networks

Layersin Deep neural networks include:

- Input Layer
- Where the input data is sent for model training
- Hidden Layers

There can be multiple hidden layers from 1 to 100 and many more. More the Hidden layers, complex the DL model and more time it takes to train. Neurons on this layer do the major computation to find hidden data patterns and to learn from them.

- Output Layer, where the results can be seen or collected from model.
- Activation Functions used in hidden layers introduce non-linearity in the model.
- Sigmoid
- ReLU [Rectified Linear Unit]
- SeLU [Scaled Exponential Linear Units]
- Leaky ReLU
- Softmax

The ReLU activation function returns 0 for negative inputs and the input value for positive inputs, giving an output range of 0 to infinity. It outperforms Sigmoid and Tanh activation functions, but SeLU performs better than ReLU [26]. SeLU is a self-normalizing activation function that maintains the mean and variance of the inputs during training, preventing the vanishing or exploding gradients problem. The SeLU function has pre-defined constants alpha (1.67326324) and lambda (1.05070098) chosen to preserve input mean and variance.

1.1.2. Loss function

Evaluating the model on different metrics helps to optimize the performance, fine-tune it, and obtain a better result [27].

1.1.2.1. Mean absolute error (MAE) [27]. MAE calculates the absolute difference between actual and predicted values. The goal is to minimize the MAE, which is the sum of all errors divided by the total number of observations. MAE has advantages such as being in the same unit as the output variable and being robust to outliers. However, its disadvantage is that its graph is not differentiable, making it challenging to apply various optimizers like Gradient Descent, which require differentiability.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_i)$$

MAE = mean absolute error
n = number of data points
Yi = observed values
Ŷi = predicted values

1.1.2.2. Mean squared error (MSE). MSE is the most used and a simple metric slightly different from MAE. Mean squared error is the squared difference between actual and predicted value. MSE avoids the cancellation of negative terms, and it is the benefit of MSE.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2$$

MSE = mean squared error
n = number of data points
Yi = observed values
Ŷi = predicted values

Disadvantages of MSE:

- MSE is a squared unit of output. for example, if the output variable is in meter(m) then the output is m^2.
- It penalizes the outliers most and the calculated MSE is bigger. It is not Robust where outliers are concerned, which was an advantage in MAE.

1.1.2.3. Root mean squared error (RMSE). RMSE is the square root of the mean squared error, and the output is in the same unit as the dependent variable. However, RMSE is not as robust to outliers as MAE.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2}$$

RMSE = Root Mean Squared Error

1.1.2.4. Root Mean Squared Log Error (RMSLE). This metric is very helpful when developing a model without calling the inputs, but, the output will vary on a large scale. To control this situation the log of calculated RMSE error as RMSLE can be used to slows down the scale of error.

It is used by most of the datasets hosted for Machine Learning competitions.

$$RMSLE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(Y_i + 1) - \log(\widehat{Y}_i + 1))^2}$$

RMSFE = Root Mean Squared Log Error

1.1.2.5. R squared (R^2). The R2 score measures a model's performance independently of context and is scale-free, with values always less than one. While MAE, MSE, and RMSE indicate accuracy, a higher R2 value is considered more desirable, as it provides a baseline model for comparison. R2 measures how much the regression line outperforms a mean line, and is also known as the Coefficient of Determination or Goodness of Fit.

$$R^2 = 1 - \frac{\text{sum squared regression (SSR)}}{\textit{total sum of squares (SST)}}$$

$$R^2 = 1 - \frac{\sum(Y_i - \widehat{Y}_i)^2}{\sum(Y_i - \overline{Y}_i)^2}$$

1.1.2.6. Adjusted R squared. R2 score can be misleading when adding irrelevant features to the data, as it assumes that more data always leads to more variance. This can cause the R2 score to increase or stay the same even when new features do not contribute to the model. To address this issue, the Adjusted R Squared metric is used, which considers the number of independent variables in the model. The formula for Adjusted R Squared takes into account the number of observations and independent variables in the data, and its value is always less than or equal to R^2.

$$\text{Adjusted } R^2 = 1 - \frac{(1 - R^2)(N - 1)}{n - p - 1}$$

R^2 = R-squared
n = total sample size
p = number of independent variables

2. Rationale for the study

The rationale for conducting a comparative study of deep learning models for hydrogen production prediction in Python is to identify the most effective model with better model metrics to predict hydrogen production in various contexts. The dataset used for the study is based on the study by Ref. [28]. The sample size of 30 is common across statistics. A sample size of 30 can increase the confidence interval of a population data set enough to warrant assertions against the finding [29]. However, a higher sample size is more likely to be representative of the population set. A sample size of 30 is sufficient for most distributions, according to statisticians. A comparative study of deep learning models was undertaken to identify the effective model given the sample size was 30 and to validate using different techniques. This study would

Table 1
Hydrogen production data [28].

Temperature (C)	RSS particle size (mm)	HDPE particle size (mm)	Percentage of plastics in mixture (wt%)	H2 (vol %)
800	0.25	0.25	10	46.676
700	0.125	0.375	20	50.123
600	0.5	0.25	30	47.751
800	0.5	0.25	10	45.952
500	0.375	0.375	20	44.781
700	0.375	0.625	20	43.031
600	0.5	0.25	10	45.324
900	0.375	0.375	20	49.23
800	0.5	0.5	30	44.355
600	0.5	0.5	30	44.208
700	0.375	0.375	0	44.466
700	0.375	0.375	40	46.603
700	0.625	0.375	20	43.072
800	0.25	0.5	30	47.396
700	0.375	0.375	20	39.98
800	0.25	0.5	10	46.338
700	0.375	0.375	20	38.569
700	0.375	0.125	20	49.868
800	0.25	0.25	30	46.545
700	0.375	0.375	20	38.612
600	0.5	0.5	10	41.032
700	0.375	0.375	20	38.625
600	0.25	0.5	30	47.123
700	0.375	0.375	20	38.621
600	0.25	0.25	10	48.634
800	0.5	0.25	30	48.475
600	0.25	0.5	10	48.132
700	0.375	0.375	20	39.262
600	0.25	0.25	30	46.502
800	0.5	0.5	10	41.93

have practical applications in industries such as energy, where accurately predicting hydrogen production can inform decisions about resource allocation, energy production, and emissions reduction. By comparing different deep learning models, the study could also provide insights into the strengths and weaknesses of different approaches to hydrogen production prediction, which could inform further research in this area. The goal of this study was to develop accurate and efficient models for predicting hydrogen production, which could have significant implications for the energy industry and for global efforts to reduce greenhouse gas emissions.

most plot in the second row shows the scatter plot of RSS particle size versus temperature, which shows no variation as expected. We can observe positive correlation between H_2 production and temperature; and H_2 production and percentage of plastics in mixture. We can also observe negative correlation between H_2 production and RSS particle size; and H_2 production and RSS particle size. The histograms indicate that the hydrogen (vol%) variables are Skewed Left (Negative Skew) [30].

Proximate, elemental analysis, and LHV of characteristic of solid biomass and solid plastics in co-pyrolysis [13].

Material	Proximate analysis (wt.%)				Elemental analysis (wt.%)						Heating Value (Mj/kg)	Hydrogen/carbon ratio (H/Ceff)
	V	FC	A	M	C	H	N	O	S	Cl		
HDPE	99.9	0[d]	0.1	0	85.5[c]	14.5[c]	0[c]	0[c]	0[c]	–	46.4[H]	2.04
PP*	100	0	0	0.08	84.80	14.55	0.14	0.28	0.23	–	45.80[H]	2.05
Rubber seed shell	80.98[a]	6.62[a]	3.81[a]	8.59[a]	44.31[b]	4.38[b]	0.51[b]	50.67	0.13[b]			−0.56

*PP composition given for comparison.
[Proximate analysis] V: volatile; A: ash; FC: fixed carbon; M: moisture.
[Elemental analysis] C: carbon; H: hydrogen; O: oxygen; N: nitrogen; S: sulfur; Cl: chlorine.
[Heating value] H: higher heating value.
L lower heating value.
G gross heating value.
a dry basis.
b dry ash free basis.
c received.
d by difference, e: calculated; -: not reported.
m moisture-free base.

2.1. Objectives

The focus of the work is to develop DL neural network model in Python using Keras to predict the Hydrogen production from co-pyrolysis of plastics and biomass.

3. Artificial deep learning neural network modeling

The DL model, using a Keras Sequential model, consists of the following steps: loading data, defining the model, compiling it, fitting the model to the data, evaluating the model's performance, and making predictions. The Sequential model is a type of model in Keras, which is a high-level deep learning library built on top of TensorFlow, i.e. The Sequential model is a specific type of model within the Keras library.

3.1. Loading the data

A CSV dataset of co-gasification of waste plastic and biomass to produce hydrogen, described by Ref. [28], has been uploaded for the present study. The dataset includes 30 experimental runs with gasification temperature, particle size of biomass rubber seed shell (RSS) and High-Density Polyethylene (HDPE), and the amount of plastic in the mixture as the independent variables, and the amount of hydrogen produced as the dependent variable (Table 1). The input variables are numerical and can be directly used with a neural network in Keras. The input and output variables are split into columns in the CSV data, and descriptive statistics are presented in Table 2.

3.2. Visualising the data

Pair plots (Fig. 1) are used to visualize the distribution of single variables and relationships between two variables to identify the trends. The histogram on the diagonal allows us to see the distribution of a single variable, while the scatter plots on the upper and lower triangles show the relationship (or lack thereof) between two variables. The left-

3.3. Define Keras model

In Keras models, sequence of layers added one at a time to develop the network architecture. The input layer receives network inputs, the hidden layer processes the information, and the output layer provides the network response [31]. The number of neurons in the input layer equals the number of inputs. Similarly, the number of output layer neurons corresponds to the number of outputs of the ANN. The input layer is assigned the number of input features which is four in the present case: gasification temperature, the particle size of biomass rubber seed shell (RSS), and the HDPE, and the amount of plastic in the mixture (Fig. 2).

In the present study, two fully connected network structure are developed to evaluate their performance. 1. With three layers and 2. With four layers and are defined using the Dense class [31]. ReLU and SeLU are used on all the layers except the output. Number of neurons in the layer is the first argument. The correct number of neurons in the hidden layers needs to be determined based on the input layer and the size of the output layer. However, when using a "large" network", its architecture is less important, when well-tuned regularization parameters are used [32]. Adding more layers improves the performance of a neural network and allows the model to extract and recombine

Table 2
Descriptive statistics of the data.

index	Temperature (°C)	RSS particle size (mm)	HDPE particle size (mm)	Percentage of plastics in mixture (wt. %)	H_2 (vol %)
count	30.0	30.0	30.0	30.0	30.0
mean	700.0	0.375	0.375	20.0	44.707
Std	90.97	0.114	0.114	9.097	3.652
Min	500.0	0.125	0.125	0.0	38.569
25%	600.0	0.25	0.25	10.0	42.205
50%	700.0	0.375	0.375	20.0	45.638
75%	800.0	0.5	0.5	30.0	47.327
Max	900.0	0.625	0.625	40.0	50.123

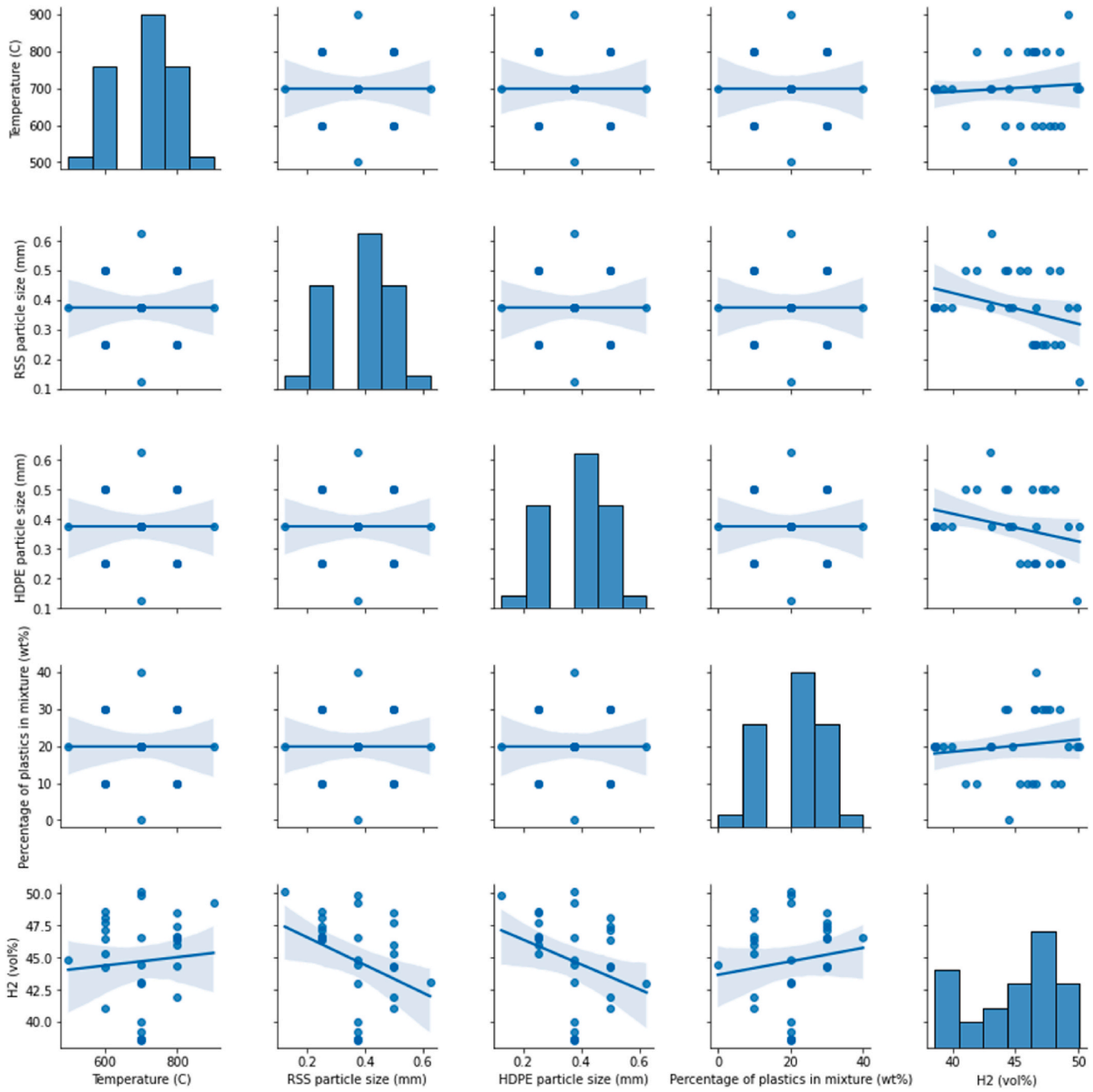


Fig. 1. Visualising the data.

higher-order features embedded in the data.

Two neural network architecture considered for the present study are:

The 1st neural network architecture considered is: [4, 16, 8, 1], a smaller network with a few layers, as depicted below [33,34]:

model = Sequential()

model.add(Dense(16, input_dim = 4, kernel_initializer = 'normal', activation = 'relu'))

model.add(Dense(8, activation = 'selu'))

model.add(Dense(1,kernel_initializer = 'normal'))

Description of the architecture (Fig. 2, Table 3):

- The model has 30 rows of data with 4 independent variables (Xs) in the input layer, L1 (the input_shape is (4)).
- 16 neurons in the first hidden layer, L2
- 8 neurons in the second hidden layer, L3, and
- 1 in the output layer L4 with single nodes, Q_1

No activation function is used for the output layer because it is a regression problem, we are interested in predicting numerical values directly without transformation [35].

3.4. Determination of number of weights

Number of weights for a particular layer is computed by taking the product of (number of nodes/input variables + bias term of each node) of the previous layer and the number of neurons in the next layer [25, 36]:

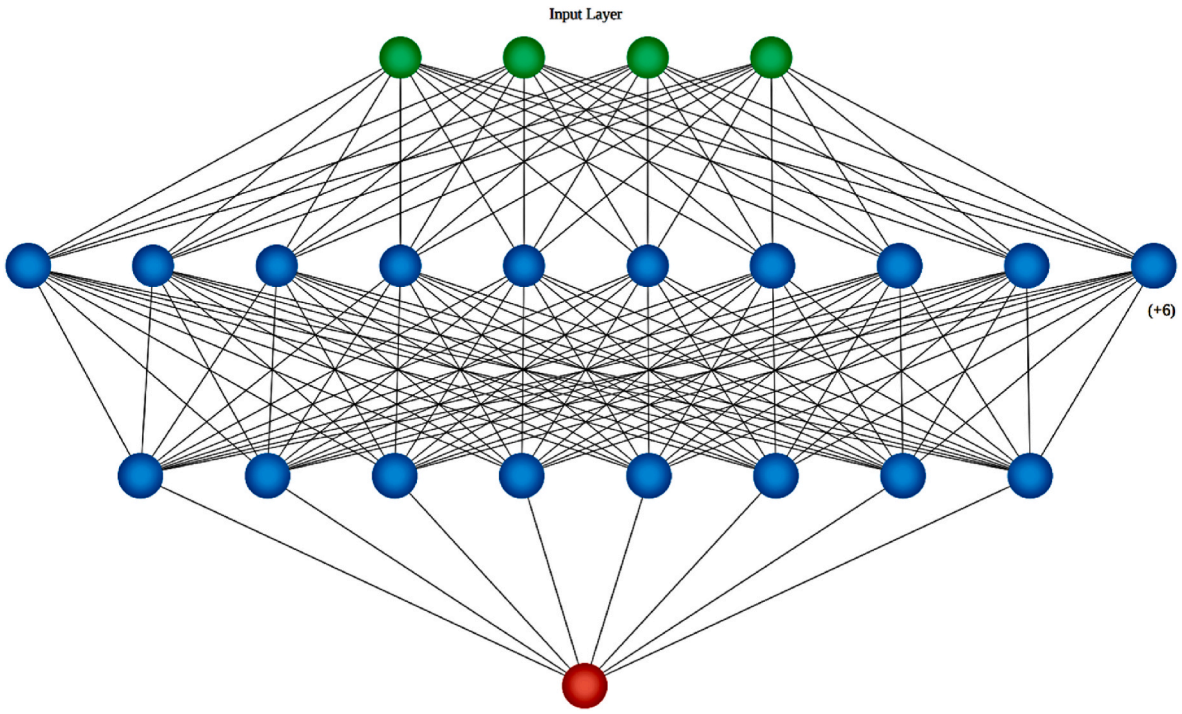


Fig. 2. Schematic diagram of Deep neural network (first Architecture).

Table 3
Summary of Network1.

dense_1120_input	input:	[(None, 4)]
InputLayer	output:	[(None, 4)]
dense_1120	input:	(None, 4)
Dense	output:	(None, 16)
dense_1121	input:	(None, 16)
Dense	output:	(None, 8)
dense_1122	input:	(None, 8)
Dense	output:	(None, 1)

Table 4
Visualization of Network1.

Model: "sequential_1"		
_Layer (type)	Output Shape	Param #
= dense_1 (Dense)	(None[a], 16)	80
dense_2 (Dense)	(None[a], 8)	136
dense_3 (Dense)	(None[a], 1)	9

Total params: 225.
Trainable params: 225.
Non-trainable params: 0.
[a] None are placeholders for batch size. In Keras 'None' means this dimension is variable and any batch size will be accepted.

Number of weights/parameters for the first neural network architecture (4, 16, 8, 1) is:

The number of weights for the first hidden layer, L2 = (4 + 1) * 16 = 80,

Similarly, the number of weights for the hidden layer, L3 = (16 + 1) * 8 = 136 wt,

Similarly, the number of weights for the output layer layer, L4 = (8 + 1) * 1 = 9.

The total number of weights for this neural network is the sum of the weights from each of the individual layers which is = 80 + 136 + 9 = 225 (Table 4)

The Second neural network architecture considered is: [4, 64, 32, 13, 1] slightly larger with more layers. The added layers in the 2nd architecture are depicted as (Fig. 3 and Table 5 and Table 6):

```
model = Sequential()
model.add(Dense(64, input_dim = 4, kernel_initializer = 'normal', activation = 'relu'))
model.add(Dense(32, activation = 'selu'))
model.add(Dense(13, activation = 'selu'))
model.add(Dense(1, kernel_initializer = 'normal'))
```

- The model has 30 rows of data with 4 variables the input_shape is (4), in L1.

The shape of the input to the model is defined as an argument on the first hidden layer.

- The first hidden layer has 64 nodes and uses the relu activation function in L2.
- The second hidden layer has 32 nodes and uses the selu activation function, in L3.
- The third hidden layer has 13 nodes and uses the selu activation function, in L4.
- The output layer has one node in L5.

Table 5 depicts the model containing 4 layers [3 hidden + 1 output]. The Input Layer does not have any weights and hence not represented here. Param represents number of weights/coefficients learned in each connected layer. Overall this 4 layers DNN model had learned/adjusted values for 2843 weights. Number of weights learned by DNN model increases exponentially as the number of layers is increased [24].

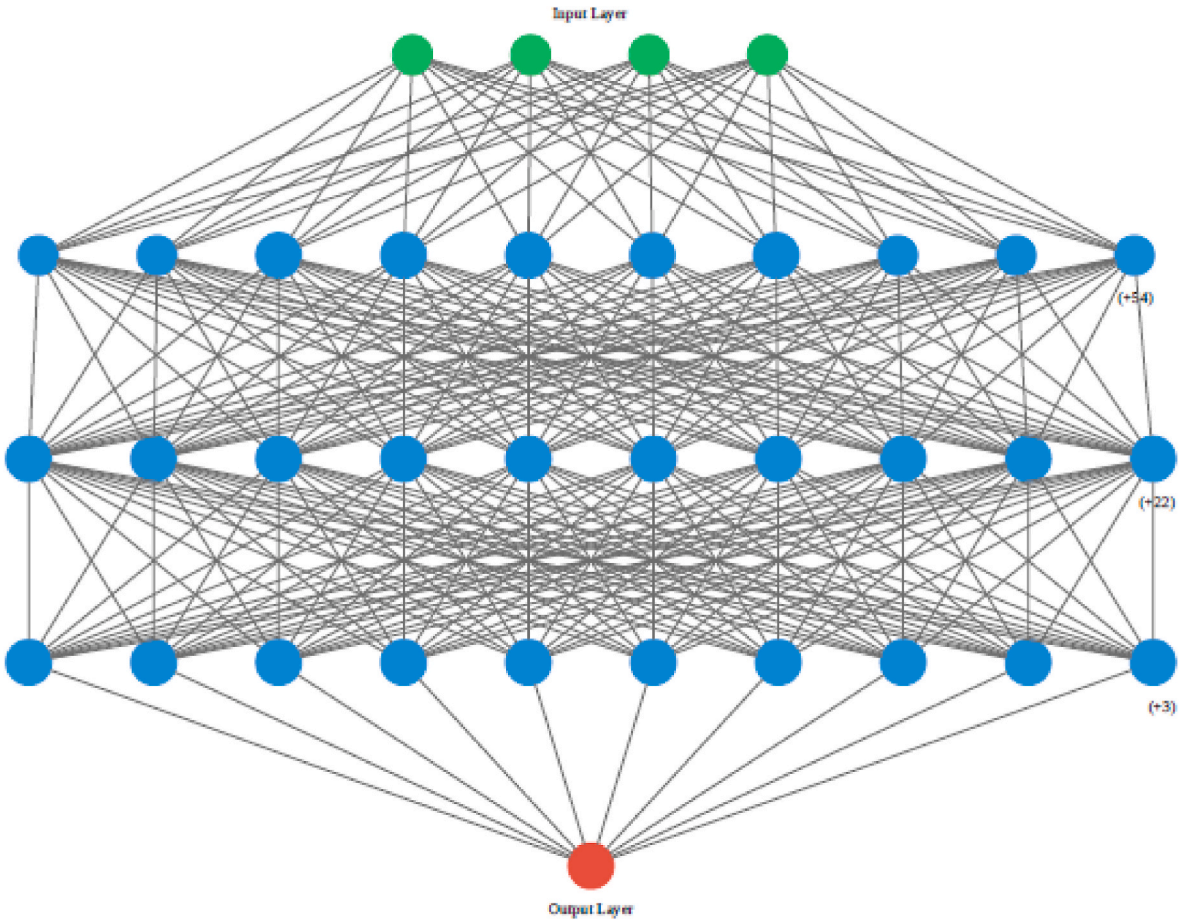


Fig. 3. Schematic diagram of Deep neural network (2nd Architecture).

Table 5
Summary of network 2.

Model: "sequential_263"		
Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None*, 64)	20
dense_2 (Dense)	(None, 32)	2080
dense_3 (Dense)	(None, 13)	429
dense_4 (Dense)	(None, 1)	14

Total params: 2843.
Trainable params: 2843.
Non-trainable params: 0.

Table 6
Visualization of Network 2.

dense_2_input	input:	[(None, 4)]
InputLayer	output:	[(None, 4)]
dense_2	input:	(None, 4)
Dense	output:	(None, 64)
dense_3	input:	(None, 64)
Dense	output:	(None, 32)
dense_4	input:	(None, 32)
Dense	output:	(None, 13)
dense_5	input:	(None, 13)
Dense	output:	(None, 1)

3.5. Regularization parameters

Complex neural network are prone to overfitting. Regularization is important in deep learning to prevent overfitting, where the model fits the training data too well and does not generalize well to new data. Regularization penalizes weight matrices of nodes, and it is necessary to optimize the regularization coefficient to obtain a well-fitted model. The two most common types of regularization are L1 and L2, which can be applied to any layer in Keras using regularizers. L2 regularization forces weights to decay towards zero, while L1 regularization penalizes the absolute value of weights and can be useful for compressing the model. High regularization coefficients can result in underfitting, so it is important to find the right balance.

In the present study regularizers L1 and L2 *were optimized* using the grid-search method. The values for L1 and L2 are determined using the gridserach: for the 1st archtecture, the ideal L1 value = 0.010; and the ideal L2 value = 0.000001 and for the 2nd architecture, The ideal L1 value = 0.100; and the ideal L2 value = 0.000010 (lowest mse values for the test sets). Corresponding L1 or L2 are added to the first layer of each architecture (Figs. 4–7).

3.6. Compile Keras model

The Sequential model was compiled with MSE as the loss function,

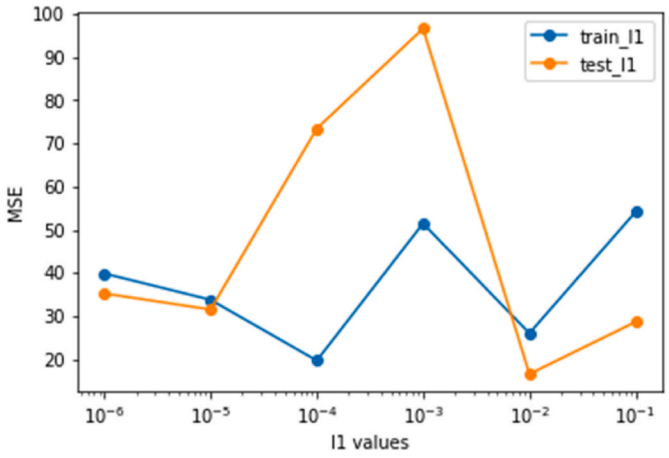


Fig. 4. MSE values of regularization parameter L1 values for 1st architecture.

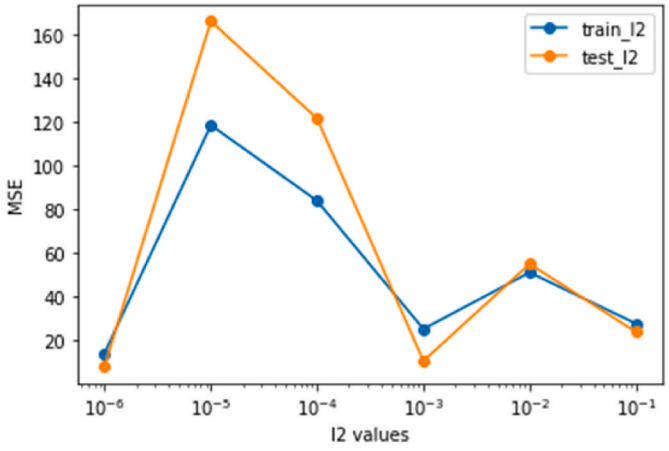


Fig. 5. MSE values of regularization parameter L1 values for 1st architecture.

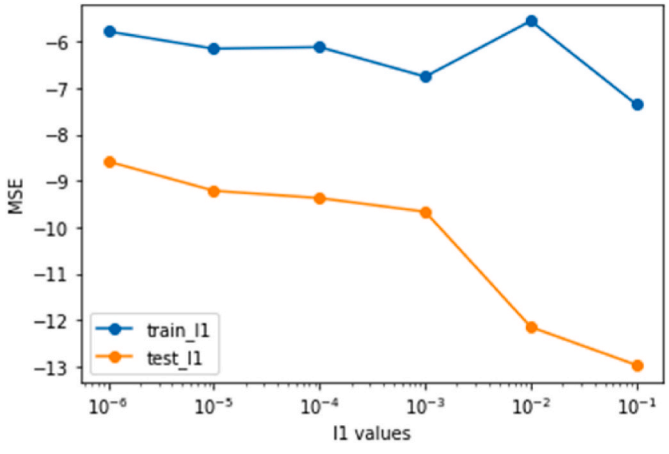


Fig. 6. MSE values of regularization parameter L1 values for 2nd architecture.

coeff_determination as the metric, and the "adam" optimizer. As "adam" optimizes the learning rate on its own, it doesn't need to be specified, making it suitable for a variety of problems.

3.7. Fit Keras model

The complied Sequential model was trained or fitted on the loaded data (Table 1) after standardizing (scaling the data to fit a standard normal distribution). The deep learning model was implemented 1. Using Sequential API of Keras, without wrapping in any particular class or function; 2. Using the Keras wrapper which provides a higher-level API that makes it easy to define and train models. Keras Regressor provides additional functionalities such as cross-validation and evaluation metrics. Training occurs over specified epochs, where each epoch is split into batches.

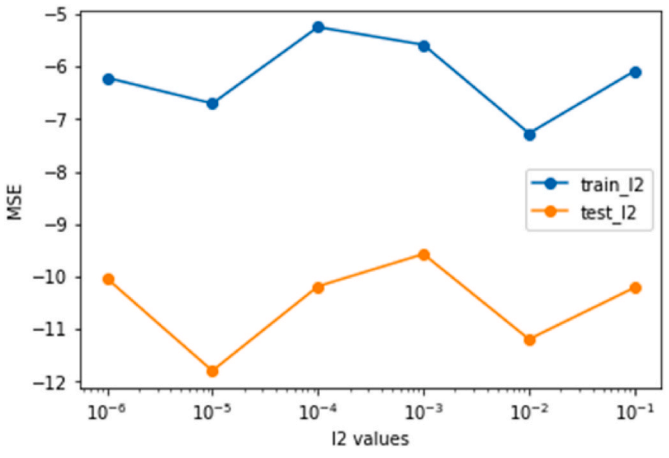


Fig. 7. MSE values of regularization parameter L2 values for 2nd architecture.

- Epoch: One pass through all the rows in the training dataset
- Batch: One or more samples considered by the model within an epoch before weights are updated

One epoch means that each sample in the training dataset has had an opportunity to update the internal model parameters. An epoch that has one batch is called the batch gradient descent learning algorithm. For the first architecture epochs (250) and batch size of 5 were chosen as the epoch 250 offered a better performance. For the present problem epochs (100) and batch size of 5 were chosen for the second architecture. After the model is fit, predictions are made by calling the predict() function on the model.

Figs. 8–13 depict prediction and model performance of the two architectures. Effect of using L1 and L2 regularization parameters as well as Sequential API of Keras with and without the wrapper are shown.

The MSE and the Coefficient of determination/R^2 indicate that the 2nd architecture performs better than the first architecture when used without the Keras wrapper. While the L1 and L2 have similar effects on first architecture, prediction results are not better without the regularization parameter. The effect was noticeable for the 2nd architecture which gave high scores but better scores when not using regularization parameter. The Keras Wrapper improved the performance of the model and predictability better for the first architecture. The second architecture performed better without the wrapper.

The model's architecture affects performance. The first model and the second model having different architectures, order of the layers, number of neurons in each layer, and the activation functions will affect the model's performance. Nonlinear data may benefit from a non-linear activation function like ReLU, while linear data might benefit from SeLU. That the first architecture with less layers and with one Relu and one Selu function as opposed to the second architecture with one ReLU and two SeLU functions can also mean that the data might not be highly non-linear. Kernel_regularizer was applied to prevent overfitting by penalizing large weights which can be beneficial for the model performance depending on the dataset and the regularization values.

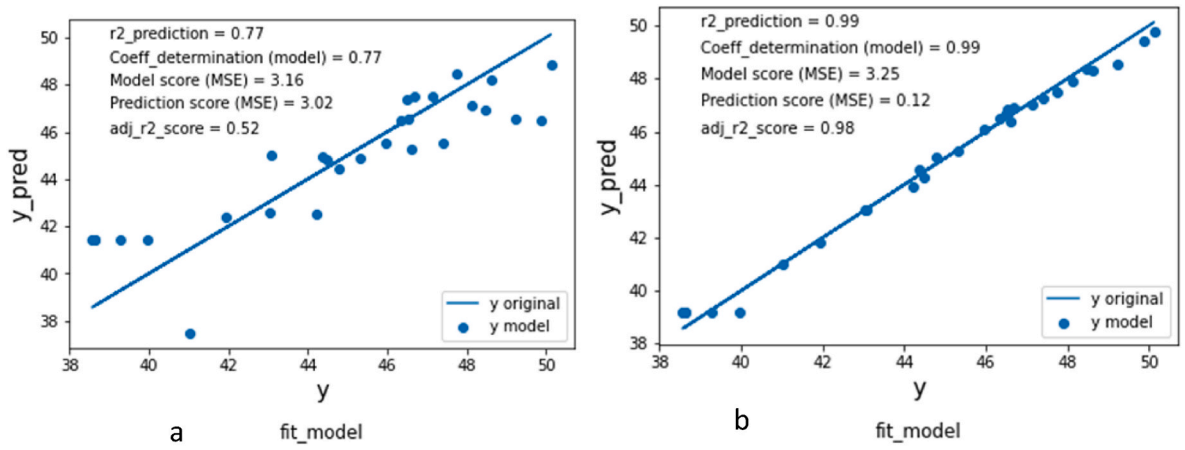


Fig. 8. Fitting (Y pred Vs real y) a. Architecture 1, Regularization, L1 = 0.010, epochs 250; b. Architecture 2, Regularization, L1 = 0.10 without Keras wrapper, epochs 100.

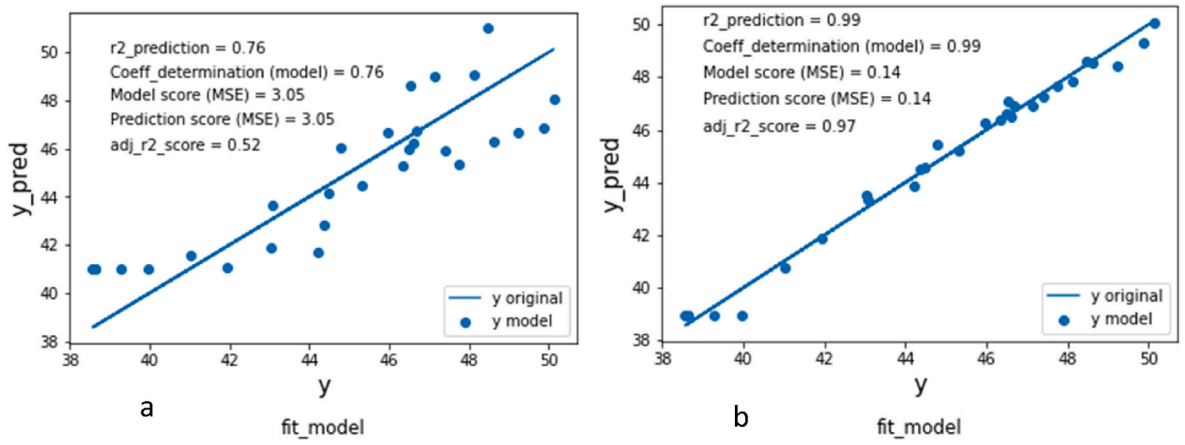


Fig. 9. Fitting (Y pred Vs real y) a. Architecture 1, Regularization, L2 = 0.000001 without Keras wrapper, epochs 250; b. Architecture 2, Regularization, L2 = 0.000010 without Keras wrapper, epochs 100.

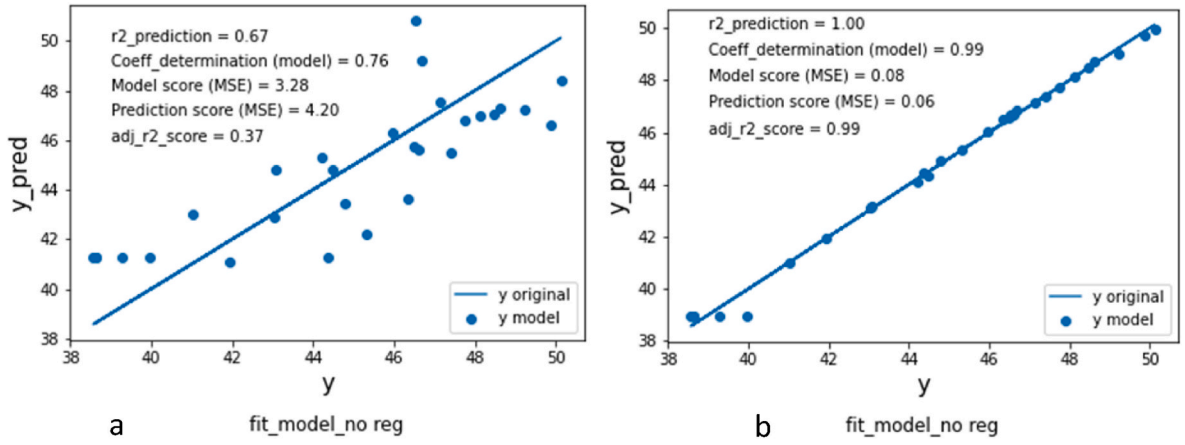


Fig. 10. Fitting (Y pred Vs real y) a. Architecture 1, without Regularization and without Keras wrapper, epochs 250; b. Architecture 2, without Regularization and without the Keras wrapper, epochs 100.

3.8. Evaluating the model

After training the model on the entire dataset, the performance of the network is evaluated on the same dataset. The loss function (mean squared error and the coeff_determination) is indicative of the model performance. The data was split into train and test datasets for training and evaluation of your model. As there will be always have some error in the model, it is ideal to choose a model configuration and training configuration that achieve the lowest loss (MSE) and highest r2/Coeff. Determination ~1 possible for a given dataset. Model performance of 1st

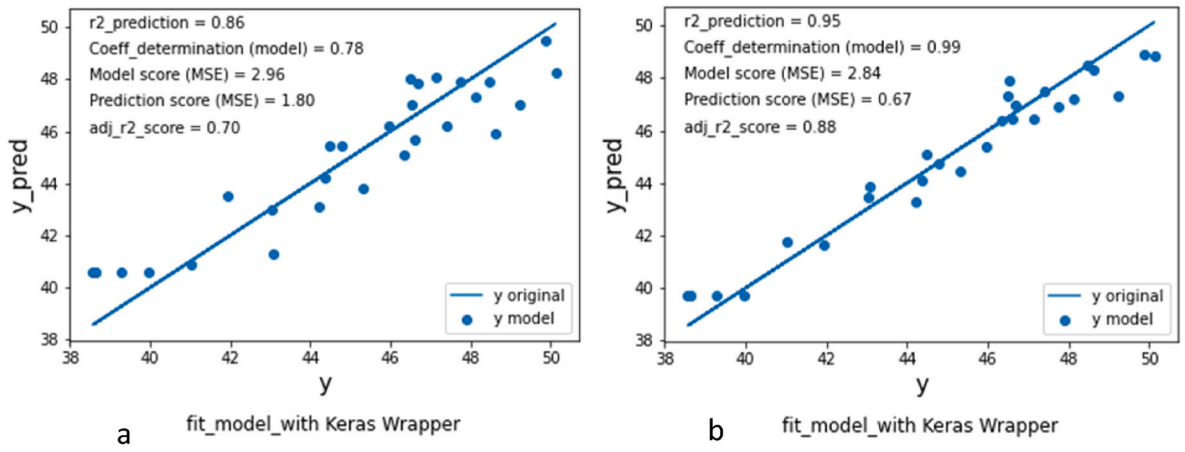


Fig. 11. Fitting (Y pred Vs real y) a. Architecture 1, L1 = 0.01; b. Architecture 2 epochs 250; Regularization, L1 = 0.10, epochs 100, with Keras wrapper.

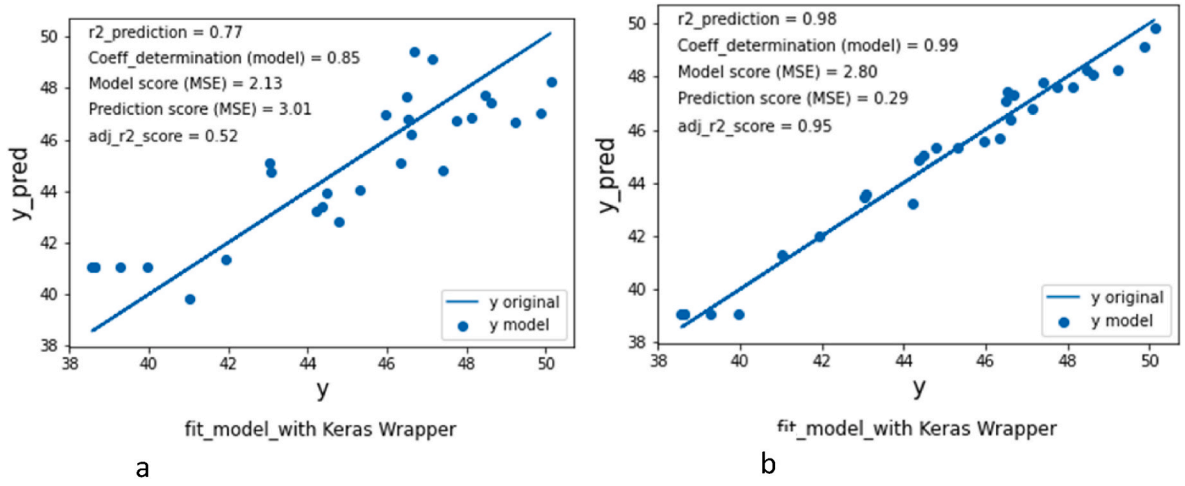


Fig. 12. Fitting (Y pred Vs real y) a. Architecture 1, L2 = 0.000001, epochs, 250; b. Architecture 2, Regularization = L2 = 0.000010, epochs = 100 with Keras wrapper.

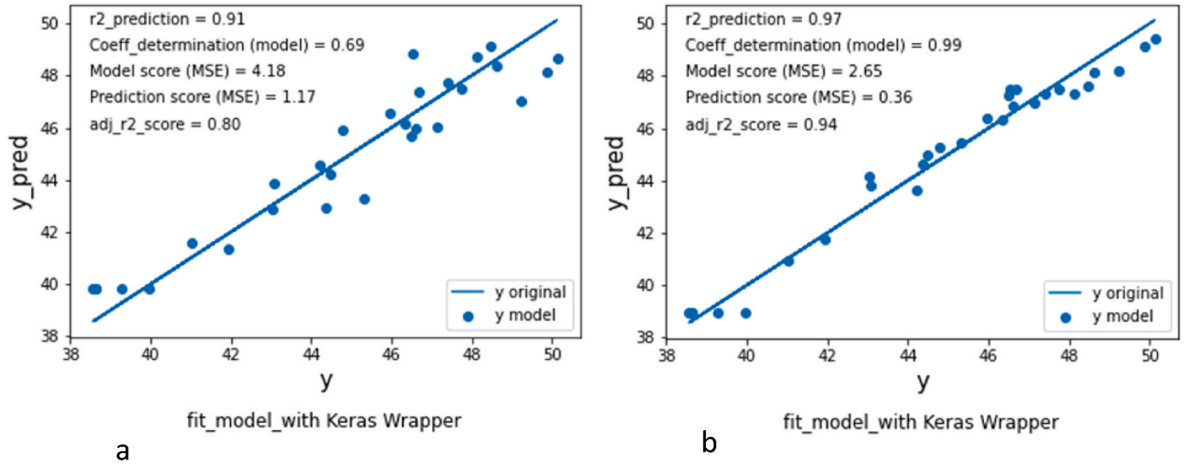


Fig. 13. Fitting (Y pred Vs real y) Architecture 1, epoch = 250; Architecture 2, without Regularization using Keras wrapper.

Architecture is depicted in Fig. 14 a, b, c: While for the full data set the optimum values are reached for epoch~100. However, for validation set the optimum values are reached ~ >150. For the 2nd Architecture the optimum loss and Coeff. Determination is achieved at epoch ~40.

3.9. Learning curves

Keras's history callback records training metrics for each epoch, to create learning curves (Figs. 14 and 15): E.g. plot of coefficient of determination on the training and validation datasets over training

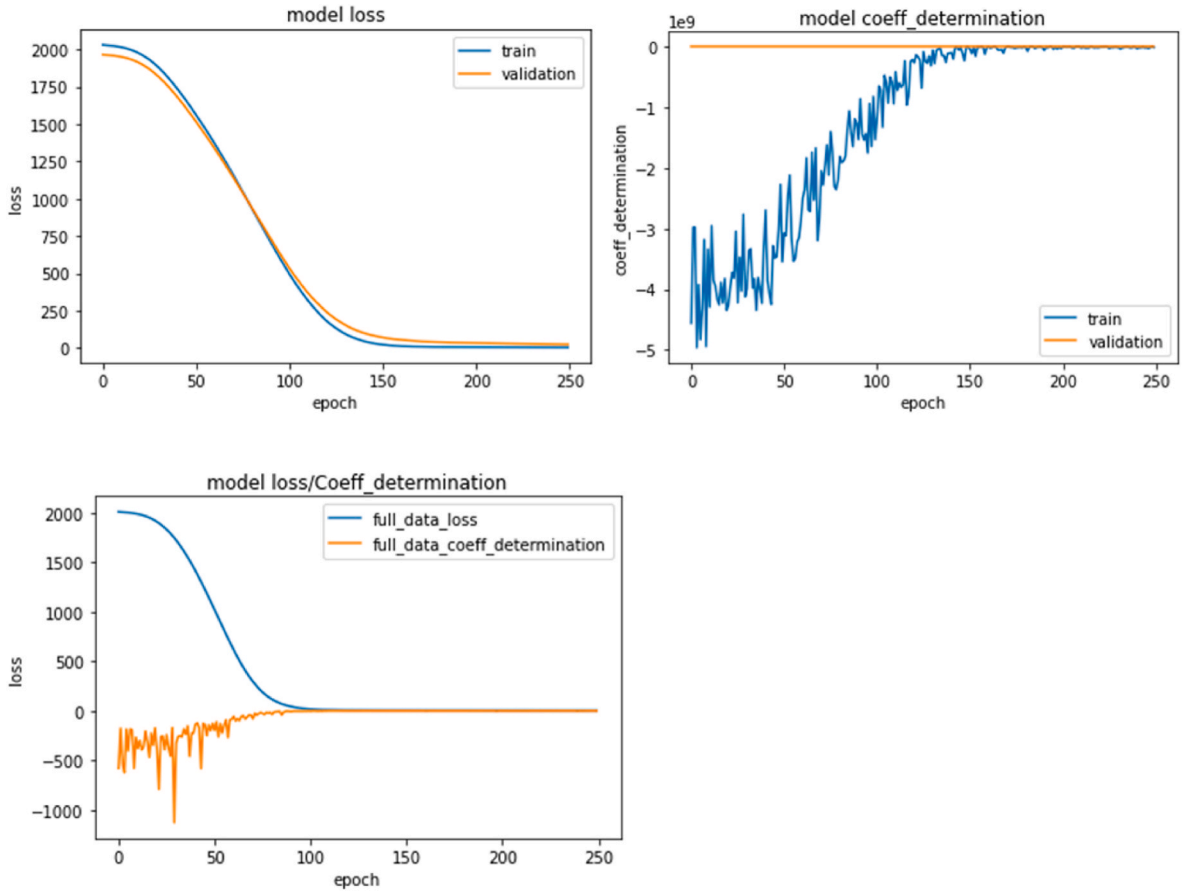


Fig. 14. Model performance vs number of Epochs for 1st Architecture. A. Loss vs epoch on train and test data; b. Coeff. determination vs epoch on train and test data; c. loss and Coeff. determination on full data set, l2(0.000001).

epochs; plot of loss (MSE) on the training and validation datasets over training epochs, which Indicate:

- speed of convergence over epochs (slope)
- Whether the model is already converged (plateau of the line)
- Whether the model is over-learning the training data (inflection for validation line)
- And more

The learning curves can be underfit, overfit or good fit curves.

3.9.1. Underfit learning curves

The learning curves shows underfitting if the training loss remains flat regardless of training.

And/or the training loss continues to decrease until the end of training. Figs. 14 and 15 indicate that the learning curves are not underfitted.

3.9.2. Overfit learning curves

The learning curves show overfitting when the model is too well trained on the data, as seen in a continuously decreasing training loss and/or a decreasing and then increasing validation loss. When the validation loss reaches an inflection point, training should be stopped to avoid overfitting. Figs. 14 and 15 show that the plots plateau after the inflection point, indicating that the learning curves are not overfitted.

3.9.3. Good fit learning curves

A good fit is identified by a training and validation loss that decreases to a point of stability with a minimal gap between the two final loss values. The loss of the model is mostly lower on the training dataset than the validation dataset showing some gap between the train and the validation loss learning curves.

A plot of learning curves shows a good fit if:

- The plot of training loss decreases to a point of stability.
- The plot of validation loss decreases to a point of stability and has a small gap referred to as the "generalization gap" with the training loss.

Continued training of a good fit will likely lead to an overfit. Figs. 14 and 15 indicate that the plots of the learning curve are good fit.

3.9.4. Unrepresentative train dataset

The study's training and validation datasets appear representative as there is no gap between the curves after improvement, indicating their similarity. Unrepresentative datasets can cause problems when the training dataset has too few examples compared to the validation dataset. A large gap between the training and validation curves indicates an unrepresentative training dataset, while a noisy validation curve indicates an unrepresentative validation dataset [37].

3.10. Cross validation (CV)

In K-Fold Cross-Validation (CV), the data is split into K-folds and the model is trained and evaluated K times, each time using a different fold for evaluation and the remaining K-1 folds for training. The performance of the model is then averaged over the K iterations. The purpose of cross validation score (cross_val_score) is to evaluate the performance of a model, by considering its performance on multiple random subsets of the data.

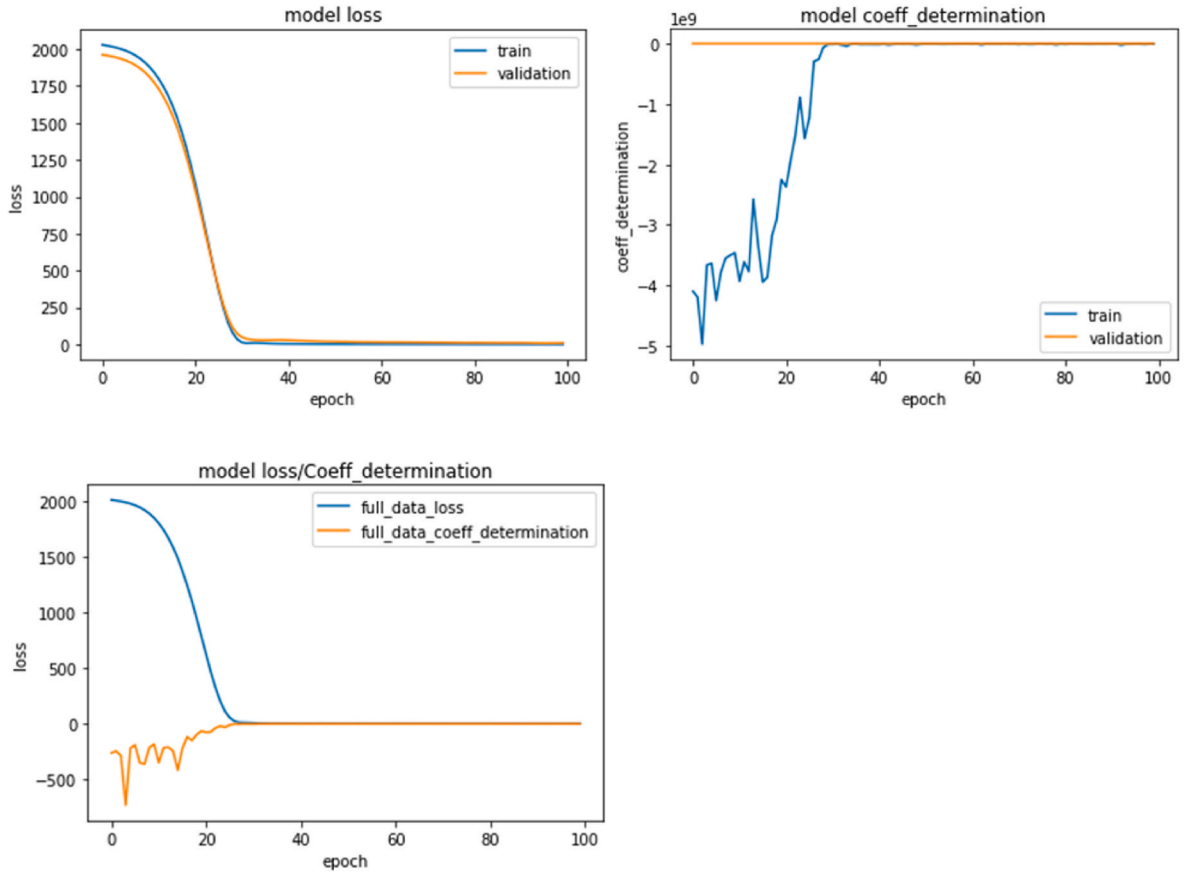


Fig. 15. Learning Curves for 2nd Architecture. A. Loss vs epoch on train and test data; b. Coeff. determination vs epoch on train and test data; c. loss and Coeff. determination on full data set, for l2(0.000010).

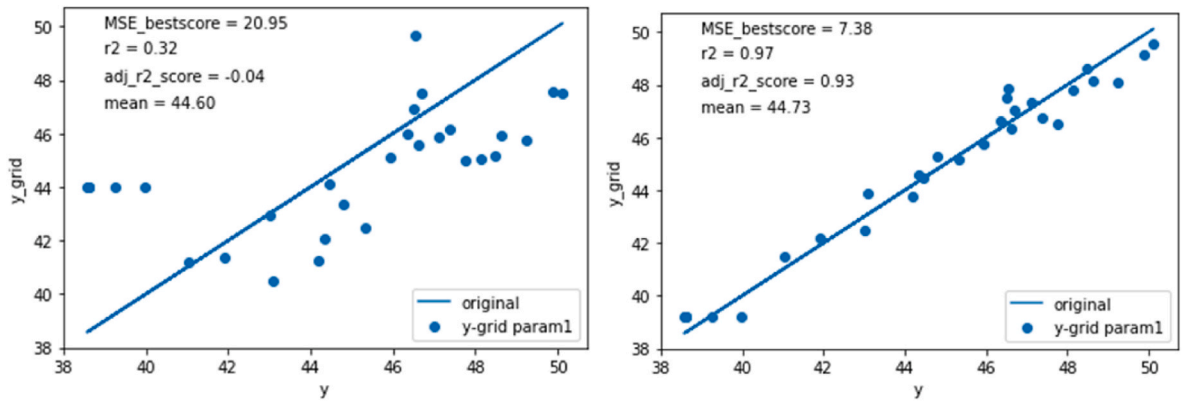


Fig. 16. a. Y Predicted vs Y measured for/Grid Search model for Kfold = 10 and for regularization (12), a. 1st Architecture; b. for 2nd Architecture.

The mean cross_val_score (negative mean square error, nMSE) for the 1st architecture, with l2 regularizers (0.000001) is −20.05 (13.10) nMSE for Kfold = 10. The mean cross_val_score, nMSE for the 2nd architecture l2 regularizers (0.000010) is −8.22 (7.77) nMSE for kfold = 10;

The nMSE values for Kfold = 1 to Kfold = 10 are given in Table 7 for both the architectures.

The cross val score (nMSE) is lower for the 2nd architecture indicating it is a better model.

Cross-validation evaluates the performance of a model and reduces the risk of overfitting by using multiple train/test splits of the data. Regularization also helps reduce overfitting by adding a penalty term to the loss function during training and discourages the model from learning parameters that are too large.

The model performance improved when using the Keras wrapper suggests that the wrapper may be providing some additional regularization or other benefits, such as optimized training algorithms. This highlights the importance of considering multiple models and techniques when working on a machine learning problem and evaluating the results carefully.

3.11. GridSearchCV

GridSearchCV (GS), a model hyperparameter optimization technique was carried out for 1st and 2nd architectures over the hyper parameters indicated in Table 8. GS uses a brute-force approach to model selection

Table 7
nMSE values for each Kfold.

Kfold	nMSE	
	1st Architecture	2nd Architecture
1	−21.57	−8.29
2	−11.66	−4.94
3	−30.68	−12.02
4	−18.89	−11.76
5	−12.01	−1.94
6	−10.88	−6.89
7	−10.76	−7.59
8	−17.97	−3.56
9	−54.86	−17.92
10	−11.21	−1.81

Table 8
GridSearchCV hyper parameters.

Architectures	hyper parameters			
	batch_size	epochs	optimizer	For Regularization, L2
1st	3,5,10,40,100	10,25,50,75, 100	'adam', 'rmsprop'	0.000001
2nd	3,5,10,40,100	10,25,50,75, 100	'adam', 'rmsprop'	0.00001

using CV, to fine-tune the hyperparameters during training using every value and/or combination of values defined by the user. The model with the best performance score is the best model.

In CV, the fold held out is the test set not seen, and thus not part of fitting any pre-processing steps (e.g., scaling or standardization). For this reason, the data cannot be pre-processed when running GS and must be a part of the set of actions taken by GS. CV (Cross-Validation) in GS is used to determine the optimal hyperparameters of a model by performing K-fold cross-validation on the training data, where K is the number specified in the CV parameter. The purpose of CV in GS is to prevent overfitting and to obtain an unbiased estimate of model performance. CV in GS is used to determine the best hyperparameters while CV in cross_val_score is used to evaluate the performance of a model (see Fig. 16).

The optimal combination of hyperparameters and the best optimization algorithm selected based on the performance results for GS are indicated in Table 9. The best model parameters of different models and their scores (nMSEs) are listed in Table 9. The y predicted vs y using the best model parameters are shown in Figures, 16 a, b, for both the architectures. Fig. 17 a. and b illustrate the Grid Search model for the best batch size for Kfold = 10 and for regularization (12) for both the architectures. Fig. 18 a. and b illustrate the Grid Search model for the best optimizer for Kfold = 10 and for regularization (12) for both the architectures.

Performance of Root Mean Squared Propagation. (RMSProp) and Adaptive Moment Estimation (Adam) optimizers by training the models with combinations of hyperparameters defined in Table 8, and the performance of each combination evaluated using Grid search is shown in Fig. 18a and b.

RMSProp and Adam are optimization algorithms used to train deep neural networks. RMSProp avoids oscillations in gradients by dividing the gradient by an exponentially weighted moving average of its recent magnitude to improve convergence speed. Adam is a more advanced optimization algorithm that uses moving averages of both the gradient and squared gradient to adapt the learning rates for each weight and prevent overfitting. The choice between RMSProp and Adam depends on several factors, such as problem type, dataset size, model architecture, regularization, activation functions, and computational resources.

3.12. Validation/sensitivity analysis

Sensitivity analysis indicates how sensitive a deep learning model's output is to changes in its input data by systematically varying the input data to see how the model's output changes in response. Sensitivity analysis helps identify the most important input features for the model's predictions. In deep learning, sensitivity analysis can be particularly useful for identifying areas where a model may be overfitting or making predictions based on noise in the training data rather than the underlying patterns. By analysing which input features have the greatest impact on the model's predictions, researchers can identify areas where

Table 9
GridSearchCV best hyper parameters for different Architecture for CV = 10.

Architectures	Negative mean square error (nMSE) for Kfold = 10	Best hyper-parameters			
		batch_size	epochs	optimizer	For Regularization, L2
1st	−20.95	3	100	rmsprop	0.000001
2nd	−7.38	5	100	adam	0.00001

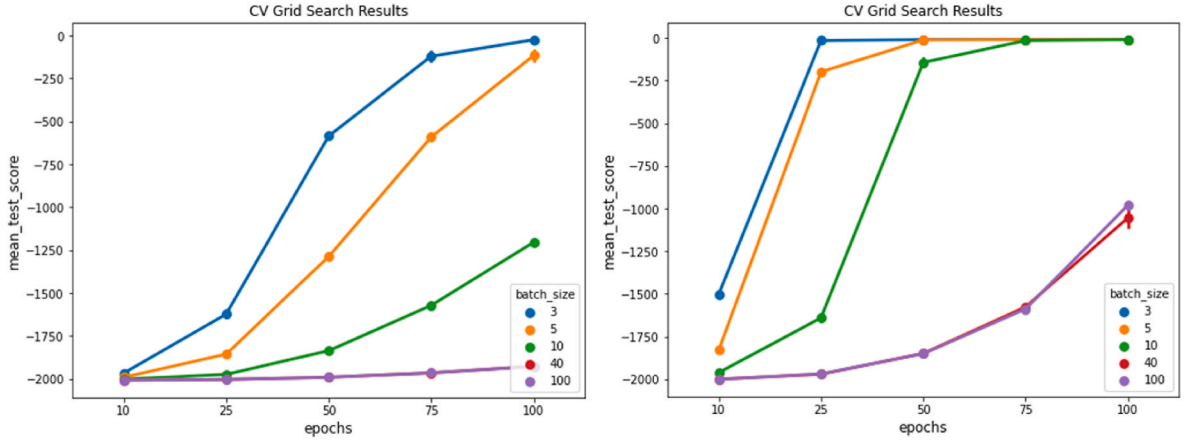


Fig. 17. a. Grid Search model for best batch size for Kfold = 10 and for regularization (12), a. 1st Architecture; b. for 2nd Architecture.

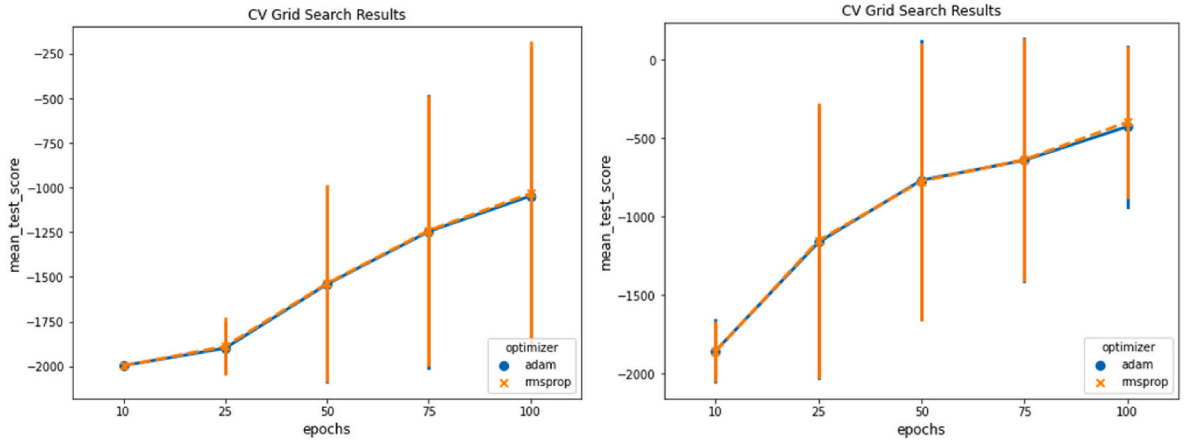
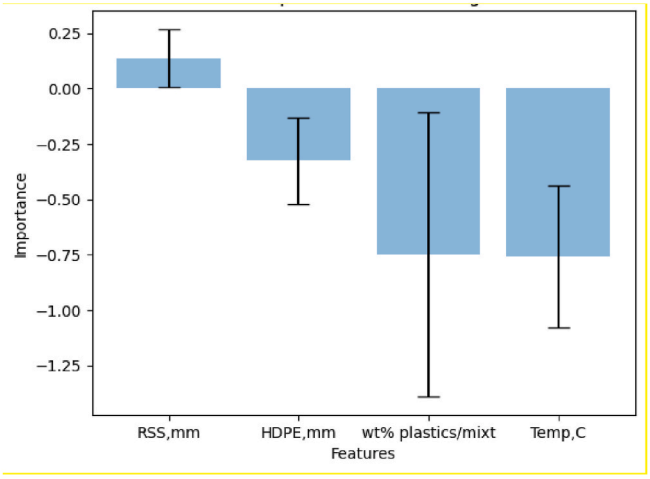
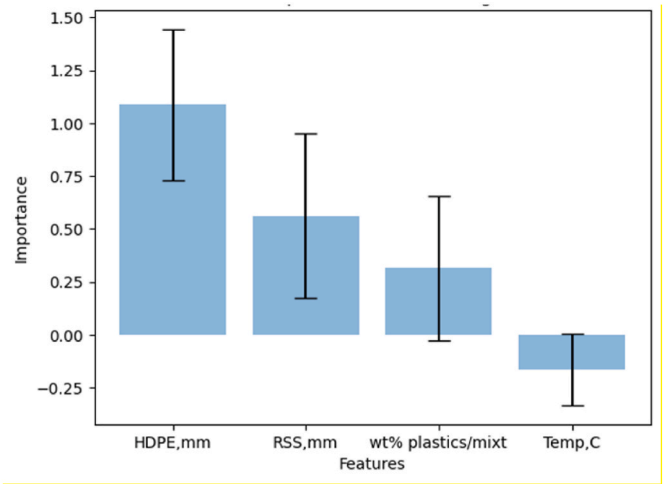


Fig. 18. a. Grid Search model for best optimizer for Kfold = 10 and for regularization (12), a. 1st Architecture; b. for 2nd Architecture.



a



b

Fig. 19. a. The permutation feature importance for Architecture 1 (descending order). b. The permutation feature importance for Architecture 2 (descending order).

the model may be overemphasizing certain aspects of the data and adjust the model accordingly to improve its accuracy and generalization ability.

3.12.1. *Permutation feature importance*

Permutation feature importance, a good validation technique, provides a simple and reliable way to evaluate the importance of features in a machine learning model by randomly shuffling the values of a feature in the test dataset and measuring the resulting decrease in model performance. Permutation feature importance is a method to evaluate the importance of features in a machine learning model. It works by randomly shuffling the values of a feature in the test dataset and measuring the resulting decrease in model performance. This approach is useful for feature selection and can be applied to any machine learning model. By ranking the features according to their importance, permutation feature importance can help identify which features are most informative and should be included in the final model [38].

Fig. 19 a and b illustrate the permutation feature importance for Architecure 1 and 2 respectively. As can be seen from the figures, the particle sizes of RSS and HDPE are the most important features. However, the order has changed with respect to the RSS and HDPE sizes from first architecture to the 2nd. The other features follow a similar trend.

3.12.2. *Hyperparameter sensitivity analysis*

This involves analysing the impact of changes to hyperparameters, such as regularization parameters or learning rates, on the model's performance. Different values of the hyperparameters are tested and the impact on the model's accuracy and generalization ability is measured. This was carried out using Grid Search (section 5 and 3.11) in the present study. The regularization parameters for both the architecture were optimized using GridSearch and applied to prevent overfitting (Figs. 4–13).

The Loss functions (mean squared error and the coeff_determination) were used to determine model performance using the train and validation (test) datasets (Figs. 14 and 15, Section 3.8). This also allows the early stop fitting. The learning curves (Fig. 15) help identify the speed of convergence over epochs, if the model is over-learning the training data or underfitting or good fit. It also indicates whether the training and validation datasets are representative.

Cross-validation (Section 3.10) and grid search (section 3.11) are not typically considered sensitivity analysis in the context of deep learning, but they are related techniques that can help improve the robustness and accuracy of machine learning models.

3.12.3. *Cross-validation*

Cross validation scores (cross_val_score) evaluate the performance of

a model, by considering its performance on multiple random subsets of the data. The cross-validation can help identify areas where a model may be overfitting or underfitting the data, which can be considered a form of sensitivity analysis. If a model is overfitting the data, it may perform very well on the training data but poorly on the validation data. If a model is underfitting the data, it may perform poorly on both the training and validation data. The mean cross_val_score (negative mean square error, nMSE) for the 1st architecture, with l2 regularizers (0.000001) is −20.05 (13.10) nMSE for Kfold = 10. The mean cross_val_score, nMSE for the 2nd architecture l2 regularizer (0.000010) is −8.22 (7.77) nMSE for Kfold = 10.

3.12.4. Grid search

Grid search selects the best hyperparameters for a machine learning model by systematically testing different combinations of hyperparameter values. Hyperparameters are settings that are not learned by the model during training, such as the learning rate or the number of hidden layers in a neural network. By testing different combinations of hyperparameter values, grid search helps identify the optimal settings for a given model architecture and dataset, which can improve the model's performance and generalization ability. The model with the best performance score is the best model (Tables 8 and 9).

3.12.5. Regularization parameters

Regularization parameters L1 and L2 are not typically considered sensitivity analysis in deep learning (Section 3.5), but they are related techniques that help improve the performance and generalization ability of machine learning models. L1 and L2 regularization prevent overfitting in machine learning models. They work by adding a penalty term to the loss function of the model, which encourages the model to learn simpler and more generalizable representations of the data. The penalty term is based on the magnitude of the weights of the model, with L1 regularization encouraging sparsity in the weight values and L2 regularization encouraging small weights overall. By adding a penalty term to the loss function, L1 and L2 regularization can help prevent the model from learning noise in the data and instead focus on the most important features for making accurate predictions.

Overall, while L1 and L2 regularization are not typically considered sensitivity analysis in deep learning, they are important techniques for improving the performance and generalization ability of machine learning models.

3.12.6. Model architecture analysis

Model architecture analysis: This involves analysing the impact of changes to the model architecture, such as the number of layers or the number of neurons in each layer, on the model's performance. In the present study analyses of two architectures with varying layers, neurons, optimizers, activation functions are described and their performance are described using MSE/nMSE, R^2, adj R^2.

The optimisers e.g. adam, optimizer optimizes the learning rate on its own. In the present study the best optimizer for both the architectures. The best optimizers are rmsprop for first architecture and adam for the 2nd architecture Table 9, Figs. 17 and 18.

4. Hydrogen production from plastics and biomass

It is concluded from Fig. 1 that the most important variables having influence on the H2 production is the particle size of the rubber seed (bio waste) and the HDPE (plastic waste). As the size increase H2 production decreases. Smaller particle size results in an increase in the rate of reaction (H2 production) because the surface area of the reactants (rubber seed and HDPE) is increased. It may be concluded the reaction is controlled by the chemical or surface control mechanism. Smaller molecules like H2 (product) can diffuse faster out of the reaction zone, therefore, may not be contributing to the slower rates, i.e., the reaction is not controlled by the diffusion process.

The temperature, however, does not seem to have great importance (Fig. 1). The test temperatures are 500 °C, 600 °C, 700 °C, 800 °C and 900 °C. Reactions 2, 3, 5, 6, 7 and 8 contribute to hydrogen production above 500 °C. Plastics pyrolysis show two phases, solid carbon and the gas, namely CH4 and H2, thermodynamically stable at 1100 °C and experimentally confirmed [39]. CH4 and the solid carbon further undergo catalytic transformation to syn gases. Following reactions characterize the chemical feed stock recycling of waste plastics:

Plastics decomposition (pyrolysis) results in reaction 1:

$$\text{Polymers} \rightarrow \text{CnHm (g)} \tag{1}$$

Pyrolysis product from reaction 2 undergoes Methane cracking (greater than 557 °C) (reaction 3):

$$\text{CnHm (g)} \rightarrow \text{nC (s)} + \text{H2 (g)}; \tag{2}$$

$$CH_4 = C\ (s) + 2H_2;\ \Delta H = 75.6\ \text{kJ/mol} \tag{3}$$

Syngas production is governed by the sequence of following reactions:

The Boudouard reaction (reaction 4, ~701 °C):

$$C + CO_2 \rightarrow 2\ CO;\ \Delta H = 172\ \text{kJ/mol} \tag{4}$$

Water gas shift reaction (reaction 5):

$$CO + H_2O\ (g) \rightarrow CO_2 + H_2;\ \Delta H = -41.2\ \text{kJ/mol} \tag{5}$$

Water gas reaction or char reforming (reaction 6, greater than 700 °C):

$$C + H_2O \rightarrow H_2(g) + CO;\ \Delta H = 131\ \text{kJ/mol} \tag{6}$$

Dry reforming reaction (endothermic), (~700 °C −1100 °C) (reaction 7), and

$$CH_4 + CO_2 \rightarrow 2CO + 2H_2;\ \Delta H = 247\ \text{kJ/mol} \tag{7}$$

Methane reforming reaction (reaction 8) ~450 °C [40] (730 °C–845 °C)

$$CH_4 + H_2O \rightarrow 3H_2 + CO;\ \Delta H = 206\ \text{kJ/mol} \tag{8}$$

Reactions 6, 7 and 8 result in various ratios of syn gas. These reactions are endothermic and high temperature reactions, requiring catalytic support. Energy input for CH_4 dry forming reaction (reaction 7) is 20% higher than steam reforming (or methane reforming) reaction 8, resulting in syngas of varying H_2/CO molar ratios.

The choice of the optimum temperature, particle size, and ratio of biomass to waste plastics in co-pyrolysis is based on trade-offs between the H_2 yield, reaction rate, and the formation of undesirable byproducts. The optimal conditions will depend on the specific system and the desired outcome [41,42].

5. Limitations of the study

The study was based on the work of Chin et al., 2015 [28]. The future study will include the effects of other parameters including the pressure, the reaction time, residence time, plastics and biomass types, and heating rates [43]. The output values can be included not just H2, but also the other possible byproducts including, CH_4, CO and CO_2. Thus the output neurons can be more than one.

1. Limited dataset: The size of the dataset is small; it could affect the ability of the deep learning models to generalize to new data. The model could learn the specific characteristics of the training data and not be able to accurately predict on new data.
2. Overfitting: Overfitting can occur when the deep learning model is too complex or when there is not enough regularization. The model may perform well on the training data but fail to generalize to new

data However, the present study has applied appropriate regularization parameters and presented in Figs. 4–13. The optimum values for different architectures were determines using the Grid Search optimization. (Sections 35, 3.6 and 3.7).

3. Lack of interpretability: Deep learning models can be considered as a black box, which makes it difficult to interpret the results and understand how the model is making its predictions. This issue is addressed using feature importance analysis, Visualization techniques such as scatter plots, help understand the relationships between the input features and the model's predictions. The model's performance on various metrics as used in the present study (MSE, R^2, adj-R^2) can provide insights into how the model is making its predictions.
4. Parameter tuning: Finding the optimal values for hyperparameters such as learning rate, number of layers, and regularization strength can be time-consuming and require significant computational resources. In the present study the learning curves are used to determine the over fitting, under fitting and good fit curves. Section 3.9, Figs. 14 and 15. The optimisers used, e.g., adam optimizer optimizes the learning rate on its own.
5. Biased or incomplete data: If the data used to train the model is biased or incomplete, the model may not be able to accurately predict on new data, leading to poor generalization performance. It was determined in Section 3.9.4 the training and validation datasets appear representative as there is no gap between the curves after improvement, indicating their similarity.
6. Performance metrics: Choosing appropriate performance metrics that capture the objectives of the study can be challenging. For example, accuracy is not the best metric for regression problems. Mean squared error or root mean squared error are more appropriate. In the present study, the loss function (mean squared error and the coeff_determination) was chosen as indicators of the model performance. The data was split into train and test datasets for training and evaluation of the model. The Cross validation in Grid Search prevents overfitting to obtain an unbiased estimate of model performance. CV in GS is used to determine the best hyperparameters while CV in cross_val_score is used to evaluate the performance of a model (Section 3.10, 3.11). The model performance improved when using the Keras wrapper suggests that the wrapper may be providing some additional regularization or other benefits, such as optimized training algorithms (Section 3.10).

6. Conclusions

The study examined hydrogen prediction using deep learning neural network models. The production of hydrogen was found to be controlled by surface reactions due to the dependence on particle size, as H2 diffusion is faster. The reaction temperature, between 500 °C and 900 °C did not have a significant effect on H2 production as multiple reactions, eg. methane cracking, methane steam reforming, water gas, methane dry reforming reactions produced H2 above 500 °C. The permutation feature importance for Architecture 1 and 2 (descending order) indicated the particle size of RSS and HDPE to be the most important features.

The deep learning model was implemented using Sequential API of Keras, with and without wrapping in any particular class or function. Two fully connected network structure with different sizes are developed to evaluate their performance. Regularizers, L1 and L2 were optimized using thegrid-search method. The values for L1 and L2 determined using the gridsearch for the 1st archtecture, L1 value = 0.010; and the L2 value = 0.000001. For the 2nd architecture, the L1 value determined using the gridsearch = 0.100; and the L2 value = 0.000010. These values are determined based on the lowest mse values for the test sets. The corresponding L1 or L2 are added to the first layer of each architecture. The plots of the learning curve on training and validation set indicated a good fit for both the architectures and the data to be representative.

While the L1 and L2 have similar effects on first architecture, prediction results are not better without the regularization parameter. The effect was noticeable for the 2nd architecture which gave high scores but better scores when not using regularization parameter. The Keras Wrapper improved the performance of the model and predictability for the first architecture. The MSE and the Coefficient of determination/R2 indicate that the 2nd architecture performs better than the first architecture when used without the Keras wrapper.

The mean cross_val_score (negative mean square error, nMSE) for the 1st architecture, with l2 regularizers (0.000001) is −20.05 (13.10) nMSE for Kfold = 10. The mean cross_val_score, nMSE for the 2nd architecture l2 regularizers (0.000010) is −8.22 (7.77) nMSE for kfold = 10. The cross-validation scores indicated that the 2nd architecture performance was better than the first.

The best model parameters determined using Grid Search CV based on model scores (nMSEs) for the 1st architecture is −20.95 (nMSE); 3 (batch size); 100 (Epochs); rmsprop (optimizer) for the regularization parameter, L2 (0.000001) for Kfold = 10. For the 2nd architecture: −7.38 (nMSE); 5 (batch size) 100 (epochs), adam 0.00001 (optimizer) for the regularization parameter, L2 (0.00001) for Kfold = 10. The lower nMSE indicates the 2nd architecture to be a better performer than the first.

CRediT authorship contribution statement

Sheila Devasahayam: Conceptualization, Methodology, Software, Data curation, Writing – original draft, Visualization, Investigation, Validation, Writing – review & editing.

Declaration of competing interest

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

Data availability

Data will be made available on request.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.energy.2023.128088.

References

[1] Devasahayam S, Guntamadugu BR, Hussain CM, Devasahayam S, Guntamadugu BR, Hussain CM. Utilization and recycling of end of life plastics for sustainable and clean industrial processes including the iron and steel industry. Mater Sci Energy Technol 2019;2:634–46.
[2] Devasahayam S. Decarbonising the Portland and other cements—via simultaneous feedstock recycling and carbon conversions sans external catalysts. Polymers 2021; 13(15):2462. https://doi.org/10.3390/polym13152462. Retrieved from.
[3] Devasahayam S. Review: opportunities for simultaneous energy/materials conversion of carbon dioxide and plastics in metallurgical processes. Sustain Mater Technol 2019;22:e00119. https://doi.org/10.1016/j.susmat.2019.e0011. Retrieved from.
[4] Block C, Ephraim A, Weiss-Hortala E, Minh DP, Nzihou A, al e. Co-Pyrogasification of plastics and biomass: a review. Waste Biomass Valor 2019;10(3):483–509. https://doi.org/10.1007/s12649-018-0219-8ff.ffhal-01700742f.
[5] Sepe AM, Li J, Paul MC. Assessing biomass steam gasification technologies using a multi-purpose model. Energy Convers Manag 2016;129:216–26. Retrieved from, https://www.sciencedirect.com/science/article/pii/S0196890416309177.
[6] Sterner M. Bioenergy and renewable power methane in integrated 100% renewable energy systems. Kassel: Kassel university, Faculty of Electrical Engineering and Computer Science. Kassel University Press GmbH; 2009.
[7] Siming Y, Yong Sik O, Tsang DC, Kwon EE, Wang C-H. Towards practical application of gasification: a critical review from syngas and biochar perspectives. Crit Rev Environ Sci Technol 2018;48(22–24):1165–213. https://doi.org/10.1080/10643389.2018.1518860.

[8] Devasahayam S. Decarbonising the Portland and other cements—via simultaneous feedstock recycling and carbon conversions sans external catalysts-correction. Polymers 2022;14(2):281.
[9] Devasahayam S. Catalytic actions of MgCO3/MgO system for efficient carbon reforming processes. Sustain Mater Technol 2019;22:e00122.
[10] Devasahayam S, Vladimir S. Thermal decomposition of magnesium carbonate with biomass and plastic wastes for simultaneous production of hydrogen and carbon avoidance. J Clean Prod 2018;174(10):1089–95.
[11] Saad JM, Williams PT. Manipulating the H2/CO ratio from dry reforming of simulated mixed waste plastics by the addition of steam. Fuel Process Technol 2017;156:331–8. https://doi.org/10.1016/j.fuproc.2016.09.016. 0378-3820.
[12] C. Block AE-H. Co-Pyrogasification of plastics and biomass, a review. Waste Biomass Valor 2019;10:483–509.
[13] Zhiwei Wang KG. Co-pyrolysis of waste plastic and solid biomass for synergistic production of biofuels and chemicals-A review. Prog Energy Combust Sci 2021;84:100899.
[14] Kaydouh M-N, El Hassan N. Thermodynamic simulation of the co-gasification of biomass and plastic waste for hydrogen-rich syngas. Results Eng 2022;16:100771.
[15] A.-C. Johansson LS. Co-pyrolysis of woody biomass and plastic waste in both analytical and pilot scale. J Anal Appl Pyrolysis 2018;134:102–13.
[16] Ö. Çepelioğullar AP. Thermal and kinetic behaviors of biomass and plastic wastes in co-pyrolysis. Energy Convers Manag 2013;75:263–70.
[17] A.O. Oyedun TG. Mixed-waste pyrolysis of biomass and plastics waste – a modelling approach to reduce energy usage. Energy 2014;75:127–35.
[18] Aitor Arregi MA. Hydrogen-rich gas production by continuous pyrolysis and in-line catalytic reforming of pine wood waste and HDPE mixtures. Energy Convers Manag 2017;136:192–201. https://doi.org/10.1016/j.enconman.2017.01.008.
[19] Han SW, Lee JJ, Tokmurzin D, Lee SH, Nam JY, Park SJ, Seo MW. Gasification characteristics of waste plastics (SRF) in a bubbling fluidized bed: effects of temperature and equivalence ratio. Energy 2022;238:121944.
[20] Moghadam RA. Hydrogen production from mixture of biomass and polyethylene waste in fluidized bed catalytic steam Co-gasification process. Chem Eng Trans 2013;35:565–70.
[21] A.V. Kondra DN. Cogasification of mixed plastics and biomass waste: process optimization and economic evaluation. Energy 2012;41:402–9.
[22] Zhang J, Chen B, Wang L. Synergistic effects of cogasification of coal and plastic waste. Fuel Process Technol 2005;86:1089–98.
[23] Pinto F, Franco C, André R, Miranda M, Gulyurtlu I, Cabrita I. Co-gasification study of biomass mixed with plastic wastes. Fuel 2002;81(3):291–7.
[24] Aggarwal R. Regression analysis using deep learning. 2022, June 13. Retrieved from Time to Deep Dive the Data and Perform Advance Analytics: https://guardiancoder.com/2022/07/13/regression-deep-learning/.
[25] Vaibhav Kumar aM. Deep learning as a frontier of machine learning: a review. Int J Comput Appl 2018;182(1):22–30. https://doi.org/10.5120/ijca2018917433.
[26] Böhm T. A first introduction to SELUs and why you should start using them as your activation functions. 2018, Aug 28. Retrieved january 3, 2023, from Towards Data Science, https://towardsdatascience.com/gentle-introduction-to-selus-b19943068cd9.
[27] Agrawal R. https://www.analyticsvidhya.com/blog/2021/05/know-the-best-evaluation-metrics-for-your-regression-model/; 2021, May 19.
[28] Chin B, Yusup S, Al Shoaibi A, Kannan P, Srinivasakannan C, Sulaiman S. Optimization study of catalytic Co-gasification of rubber seed shell and high density polyethylene waste for hydrogen production using response surface methodology. In: Ravindra P, editor. Advances in bioprocess technology; 2015. p. 209–21. https://doi.org/10.1007/978-3-319-17915-5_11.
[29] Chang HJ. Determination of sample size in using central limit theorem for weibull distribution. Int J Info Manag Sci 2006;17(3):153–74.
[30] Koehrsen W. Visualizing data with Pairs plots in Python. 2018, April 7. Retrieved 9 1, 2023, from, https://towardsdatascience.com/visualizing-data-with-pair-plots-in-python-f228cf529166.
[31] Brownlee J. Your first deep learning project in Python with Keras step-by-step. 2022, June 18. Retrieved December 19, 2022, from Machine Learning Mastery: https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/.
[32] Jain S. An overview of regularization techniques in deep learning (with Python code). 2021, July 23. Retrieved December 30, 2022, from Analytics Vidhya: https://www.analyticsvidhya.com/blog/2018/04/fundamentals-deep-learning-regularization-techniques/.
[33] Seth N. Estimation of neurons and forward propagation in neural net. 2021. Retrieved January 9, 2023, from Analytics Vidhya:a, April 26, https://www.analyticsvidhya.com/blog/2021/04/estimation-of-neurons-and-forward-propagation-in-neural-net/#h2_6.
[34] Hyndman RJ, Athanasopoulos G. Forecasting: principles and practice. In: Melbourne, Australia: OTexts, OTexts.com/fpp2. second ed. 2018. from https://otexts.com/fpp2/nnetar.html. [Accessed 9 January 2023].
[35] Brownlee J. Regression Tutorial with the Keras deep learning Library in Python. 2016, june 9. Retrieved December 28, 2022, from Machine Learning Mastery: https://machinelearningmastery.com/regression-tutorial-keras-deep-learning-library-python/.
[36] Seth N. Estimation of neurons and forward propagation in neural net. 2021. Retrieved January 12, 2023, from Analytics Vidhya:b, April 26, https://www.analyticsvidhya.com/blog/2021/04/estimation-of-neurons-and-forward-propagation-in-neural-net/#h2_6.
[37] Brownlee J. How to use learning curves to diagnose machine learning model performance. 2019, August 6. Retrieved December 24, 2022, from Machine Learning Mastery: https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/.
[38] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Duchesnay E. sklearn.ensemble.RandomForestClassifier, scikit-learn: machine learning in Python. J Mach Learn Res 2011;12:2825–30. Retrieved from, https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.
[39] VOEST. High temperature pyrolysis of plastic waste. Austria: Styrian Provincial Government - Specialised Division 1c . Styrian Provincial Government; 1997. Plant Construction of VOEST-Alpine., Styrian Material Flow Management. Retrieved from, http://www.abfallwirtschaft.steiermark.at/cms/dokumente/10177122_46583/baee943b/FA19D_CPC_Studie_13_e.pdf.
[40] Ferreira-Aparicio P, Guerrero-Rui A, I R-R. Comparative study at low and medium reaction temperatures of syngas production by methane reforming with carbon dioxide over silica and alumina supported catalysts. Appl Catal Gen 1998;170(1):177–87. https://doi.org/10.1016/S0926-860X(98)00048-9. Retrieved from.
[41] Tsang WL. Gasification of biomass and waste plastics: a review of the technology and research progress. Renew Sustain Energy Rev 2015;47:730–41.
[42] Sontakke CH. Gasification of mixed plastics and biomass waste: a thermodynamic analysis. Waste Manag 2010;30:1620–7.
[43] Zhang B, Guo S, Jin H. Production forecast analysis of BP neural network based on Yimin lignite supercritical water gasification experiment results. Energy 2022;246:123306. https://doi.org/10.1016/j.energy.2022.123306. ISSN 0360-5442.