Machine Learning-Aided Process Design Using Limited Experimental Data: A Microwave-Assisted Ammonia Synthesis Case Study

**Supplementary Material**

Md Abdullah Al Masud, Alazar Araia, Yuxin Wang, Jianli Hu, Yuhe Tian∗

Department of Chemical and Biomedical Engineering, West Virginia University, Morgantown, WV 26506, United States

**This file includes:**

Section A: Validation of prior experimental knowledge

Section B: Pseudo algorithm for SMOTE regression

Section C: Python code for SMOTE regression

Section D: Description of SMOTE data generation and plotting of synthetic data

Section E: Description of the comparison of ReLU and Soft Plus activation function

Section F: Description of SMOTE integrated neural networks with soft plus activation function for 4 inputs 1 output model

Section G: Description of SMOTE integrated neural networks with soft plus activation function for 2 inputs 1 output model

Section H: Description of the validation of the proposed model

Section I: Description of the effect of duplicate data on trendline prediction

Section J: Description of the Support Vector Regression

Section K: Description of the neural networks without SMOTE for continuous data prediction

All the referred python (.ipynb) files are available as open-source codes via via Github repository at <https://github.com/masud045/SMOTE-Integrated-NNs>. These code files include complete information on the machine learning set ups and model parameters in Python.

**Section A. Validation of prior experimental knowledge**

Fig. A1 depicts the impact of each individual input factor on NH3 concentration. Namely, one input value is varied (e.g., P) with the remaining input values fixed at the optimal reaction conditions (H2:N2 = 1, T = 597.37K, F = 1.6710-3 m3/(skgcatalyst). Based on prior experimental knowledge, pressure increase always results in higher ammonia production which is well reflected in Fig. A1. The NH3 concentration variation trends as a result of varying T, H2:N2, or F are also consistent with that shown in the original experimental results (Fig. 2). It can also be validated from Fig. A1 that this optimal reaction condition is an extreme point in its neighbourhood.



Figure A1: Validation of prior experimental knowledge at the optimal reaction conditions.

**Section B. Pseudo algorithm for SMOTE regression**

Input:

- Dataset X (input features)

- Y (target variable)

- N (desired number of synthetic samples)

- k (number of nearest neighbors)

Output:

- Augmented dataset with synthetic samples

1. Initialize an empty list synthetic\_samples

2. For each data point (xi, yi) in the dataset:

a. Find the k-nearest neighbors of (xi, yi) in the feature space X

b. For each neighbor (xj, yj) of (xi, yi):

i. Compute the difference (delta\_x, delta\_y) = (xj - xi, yj - yi)

ii. Generate a random number r between 0 and 1

iii. Create a new synthetic sample:

synthetic\_x = xi + r \* delta\_x

synthetic\_y = yi + r \* delta\_y

iv. Add (synthetic\_x, synthetic\_y) to synthetic\_samples

3. Select N synthetic samples from synthetic\_samples randomly

4. Append the selected synthetic samples to the original dataset

5. Return the augmented dataset

**Section C. Python code for SMOTE regression**

import numpy as np

from sklearn.neighbors import NearestNeighbors

def smote\_regression(X, y, N, k):

"""

SMOTE algorithm for regression.

Parameters:

- X: np.array, input features of shape (n\_samples, n\_features)

- y: np.array, target variable of shape (n\_samples,)

- N: int, desired number of synthetic samples

- k: int, number of nearest neighbors

Returns:

- X\_augmented: np.array, augmented input features

- y\_augmented: np.array, augmented target variable

"""

# Initialize list to store synthetic samples

synthetic\_samples\_X = []

synthetic\_samples\_y = []

# Fit nearest neighbors model

nn = NearestNeighbors(n\_neighbors=k)

nn.fit(X)

# Loop over each sample

for i in range(len(X)):

# Find k-nearest neighbors

neighbors = nn.kneighbors([X[i]], return\_distance=False)

for neighbor in neighbors[0]:

if neighbor != i:

# Compute difference

delta\_x = X[neighbor] - X[i]

delta\_y = y[neighbor] - y[i]

# Generate synthetic sample

r = np.random.rand()

synthetic\_x = X[i] + r \* delta\_x

synthetic\_y = y[i] + r \* delta\_y

# Append synthetic sample to the list

synthetic\_samples\_X.append(synthetic\_x)

synthetic\_samples\_y.append(synthetic\_y)

**Section D. Data Source Description of The Synthetic Data Generated via SMOTE**

This section explains the generation of synthetic data following SMOTE regression based on the 46 experimental data points. The file named “3D\_Plotting\_SMOTE\_Data\_Fig4.ipynb” contains the coding for SMOTE algorithm. Detailed model parameters can be found in the file. By running the file, the program will generate 3D plotting and scattered 2D plotting with color gradient of SMOTE data. The results correspond to that reported in Fig. 4.

**Section E. Data Source Description of The Role of Activation Function**

In this section, a comparison between the ReLU activation function and Soft Plus activation are shown for the continuous prediction of ammonia concentration at varying reaction conditions. The file named “Comparison\_between\_ReLU\_and\_softplus\_Fig6.ipynb” contains the coding to visualize the effect of these two activations functions. Detailed model parameters can be found in the file. The results correspond to that reported in Fig. 6.

**Section F. Data Source Description of The SMOTE Integrated Neural Networks with Soft Plus Activation Function (4 inputs - 1 output)**

This file named “Final\_SMOTE\_Integrated\_NN(4input\_1Output)\_Fig7.ipynb” contains the final proposed methodology for the design of microwave-assisted ammonia reactor. Detailed model parameters can be found in the file. By running the file, the program will generate the results shown in Fig. 7 in the manuscript and optimal reaction conditions for microwave reactor. The model is developed for 4 inputs 1 output. The results correspond to that reported in Fig. 7.

**Section G. Data Source Description of The SMOTE Integrated Neural Networks with Soft Plus Activation Function (2 inputs - 1 output)**

The “SMOTE\_Integrated\_NN\_Fig8.ipynb” also contains the SMOTE integrated neural network models following proposed methodology but for 2 inputs 1 output. Detailed model parameters can be found in the file. The plot for continuous data prediction will be generated after running the code file. The results correspond to that reported in Fig. 8.

**Section H. Data Source Description of The Validation of Proposed Model**

This section explains the validation of the model developed following the SMOTE integrated neural networks with soft plus activation function. The file name is “Validate\_Final\_SMOTE\_Integrated\_NN(4Input-1Output)\_Fig10.ipynb”. Detailed model parameters can be found in the file. After running the python file, four plots will be generated that represent the effect of the individual four input variables on the ammonia concentration in microwave reactor. The results correspond to that reported in Fig. 10.

**Section I. Data Source Description of The Duplicate Data for Trendline Prediction**

The file named “Effect\_of\_Duplicate\_Data(4Input\_1Output)\_Fig11.ipynb” shows the effect of duplicate data points on the trendline prediction for four inputs one output model. Detailed model parameters can be found in the file. By running the file, the program will generate the plot of two models. The results correspond to that reported in Fig. 11.

**Section J. Data Source Description of The Support Vector Regression**

In this section, the work utilizing support vector regression is shown to develop the trendline prediction. The file “SVR\_Performance\_Fig13.ipynb” contains the coding and shows the 3D plotting of data prediction. Detailed model parameters can be found in the file. The results correspond to that reported in Fig. 13.

**Section K. Data Source Description of The Neural Networks without SMOTE**

This section is related to the implementation of neural networks without SMOTE. The file name is “NN\_Without\_SMOTE\_Fig14.ipynb”. Detailed model parameters can be found in the file. By running the python file, four plots will be automatically generated that represent the implementation of neural networks for predicting continuous ammonia concentration. The results correspond to that reported in Fig. 14.