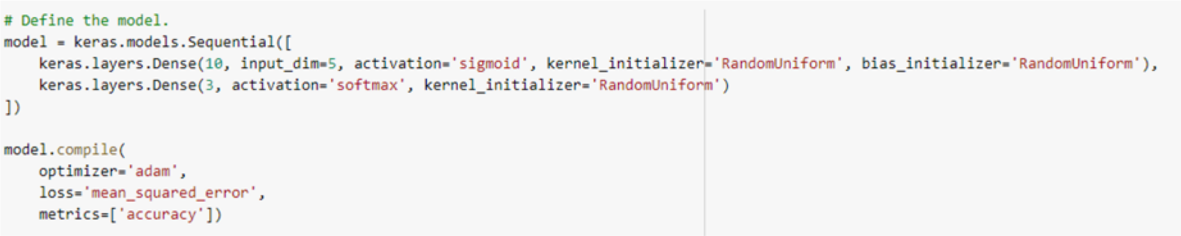
1. Problem Description

In the case study, I considered the chemical vapor deposition of diamonds. A carbon dimer (a bound pair of carbon atoms) is projected toward a diamond surface. We want to determine the probabilities for various reactions based on the characteristics of the projected dimer. The input variables consist of such properties as translational energy and incidence angle, and the response variables consist of the probabilities of the potential reactions, such as chemisorption and scattering.

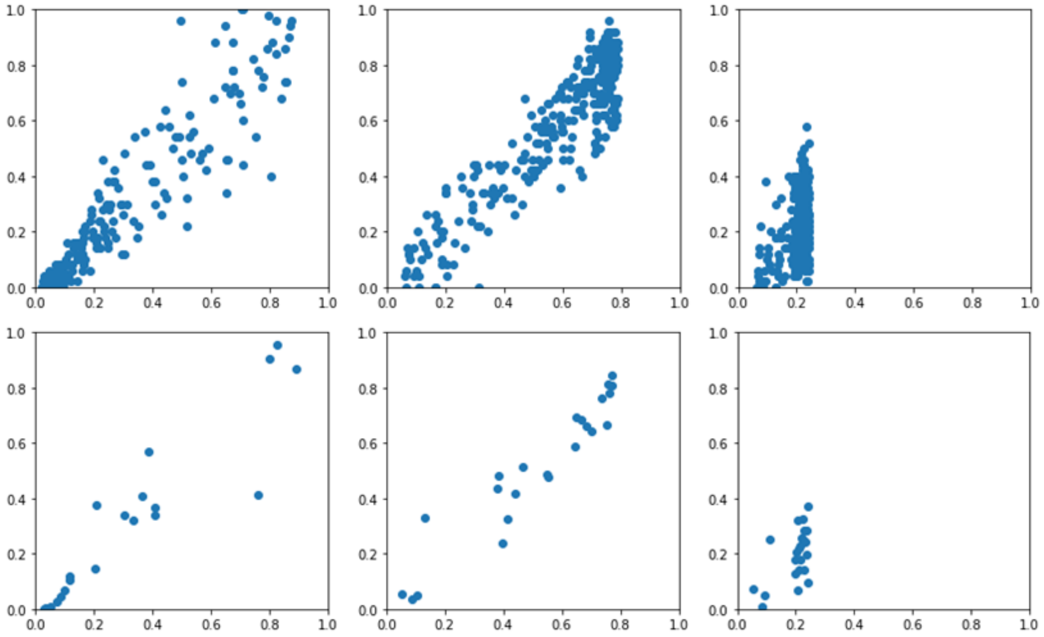
During the CVD process, a carbon dimer is projected toward a diamond substrate. For this study, we will assume that the dimer can react with the substrate in one of three ways: chemisorption scattering, or desorption. A neural network is trained to estimate the probabilities of each of the reactions, based on various characteristics of the carbon dimer.

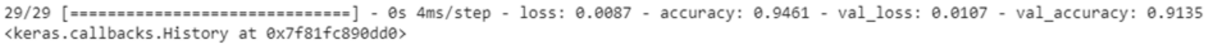
2. Analysis and simulation studies

1. Changes made



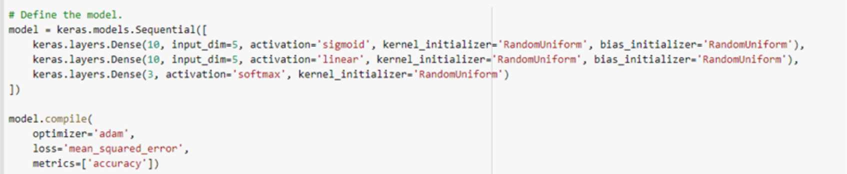
```
# Define the model.
model = keras.models.Sequential([
    keras.layers.Dense(10, input_dim=5, activation='sigmoid', kernel_initializer='RandomUniform', bias_initializer='RandomUniform'),
    keras.layers.Dense(3, activation='softmax', kernel_initializer='RandomUniform')
])

model.compile(
    optimizer='adam',
    loss='mean_squared_error',
    metrics=['accuracy'])
```



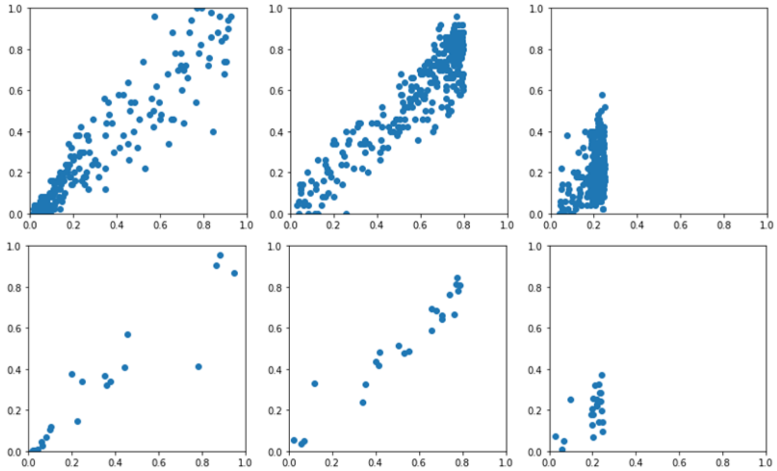


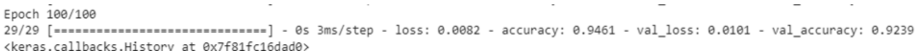
```
29/29 [==============================] - 0s 4ms/step - loss: 0.0087 - accuracy: 0.9461 - val_loss: 0.0107 - val_accuracy: 0.9135
<keras.callbacks.History at 0x7f81fc890dd0>
```

2. Changes made



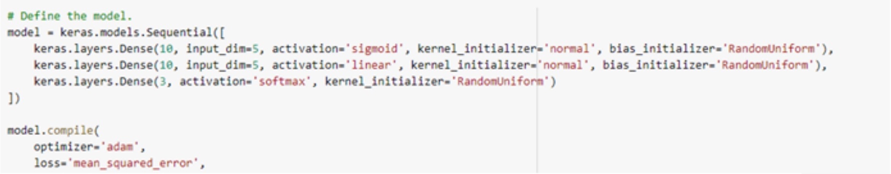
```
# Define the model.
model = keras.models.Sequential([
    keras.layers.Dense(10, input_dim=5, activation='sigmoid', kernel_initializer='RandomUniform', bias_initializer='RandomUniform'),
    keras.layers.Dense(10, input_dim=5, activation='linear', kernel_initializer='RandomUniform', bias_initializer='RandomUniform'),
    keras.layers.Dense(3, activation='softmax', kernel_initializer='RandomUniform')
])

model.compile(
    optimizer='adam',
    loss='mean_squared_error',
    metrics=['accuracy'])
```



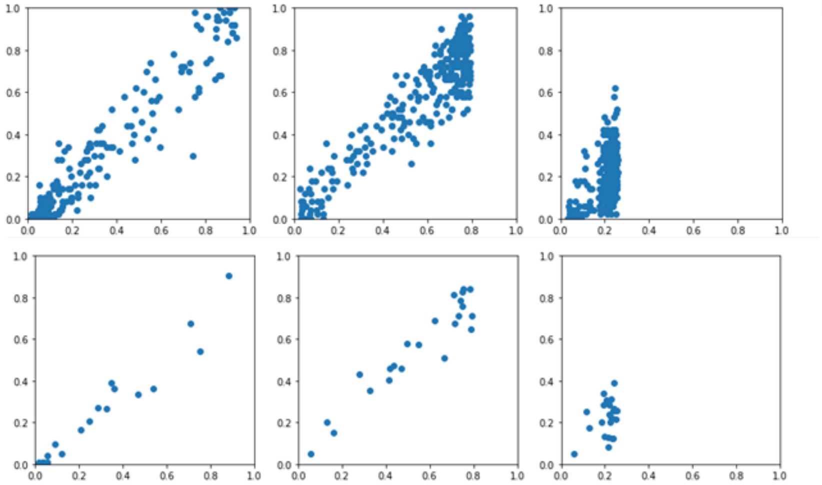


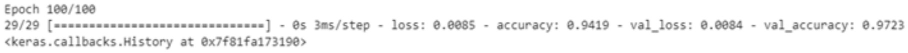
```
Epoch 100/100
29/29 [==============================] - 0s 3ms/step - loss: 0.0082 - accuracy: 0.9461 - val_loss: 0.0101 - val_accuracy: 0.9239
<keras.callbacks.History at 0x7f81fc16dad0>
```

3. Changes made



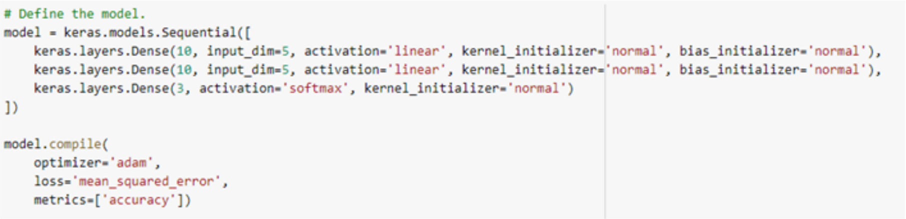
```
# Define the model.
model = keras.models.Sequential([
    keras.layers.Dense(10, input_dim=5, activation='sigmoid', kernel_initializer='normal', bias_initializer='RandomUniform'),
    keras.layers.Dense(10, input_dim=5, activation='linear', kernel_initializer='normal', bias_initializer='RandomUniform'),
    keras.layers.Dense(3, activation='softmax', kernel_initializer='RandomUniform')
])

model.compile(
    optimizer='adam',
    loss='mean_squared_error',
```



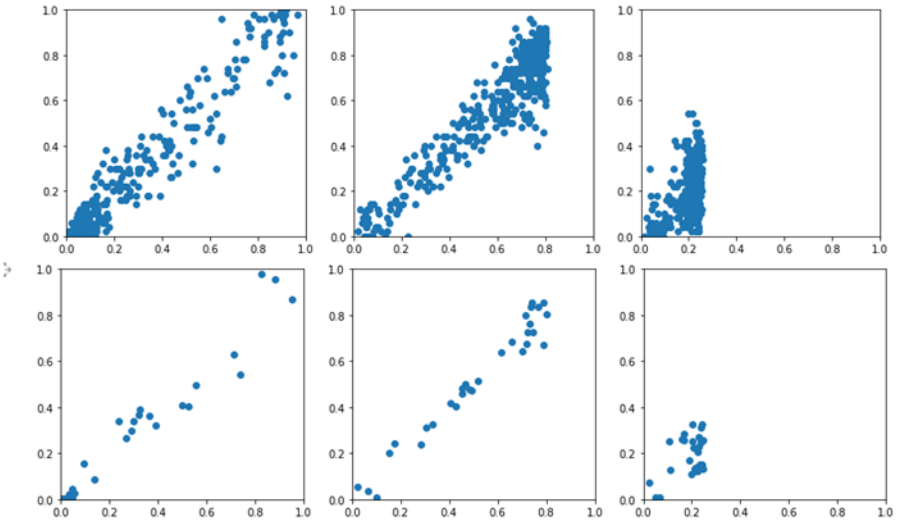


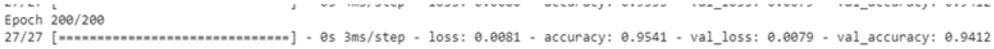
```
Epoch 100/100
29/29 [==============================] - 0s 3ms/step - loss: 0.0085 - accuracy: 0.9419 - val_loss: 0.0084 - val_accuracy: 0.9723
<keras.callbacks.History at 0x7f81fa173190>
```

4. Changes made



```
# Define the model.
model = keras.models.Sequential([
    keras.layers.Dense(10, input_dim=5, activation='linear', kernel_initializer='normal', bias_initializer='normal'),
    keras.layers.Dense(10, input_dim=5, activation='linear', kernel_initializer='normal', bias_initializer='normal'),
    keras.layers.Dense(3, activation='softmax', kernel_initializer='normal')
])

model.compile(
    optimizer='adam',
    loss='mean_squared_error',
    metrics=['accuracy'])
```





```
Epoch 200/200
27/27 [==============================] - 0s 3ms/step - loss: 0.0081 - accuracy: 0.9541 - val_loss: 0.0079 - val_accuracy: 0.9412
```

3. Discussion

After the neural network has been trained, it becomes a simple matter to investigate the effect of the input parameters on the reaction probabilities. we see the effect of the impact parameter b on the reaction probabilities, as determined by the neural network. As the impact parameter is increased, the probability of chemisorption decreases, while the probabilities of scattering and desorption increase.

With standard methods, this case study would take thousands of simulations. The trained neural network has fully captured the relationships between the parameters in and the reaction probabilities. Therefore, we can perform arbitrary studies by simply computing the network responses at a varying set of input points. Note that the network interpolated smoothly through a noisy set of data points to capture the true underlying function. By using the early stopping technique, we prevented the network from overfitting the noise in the data.

4. Conclusion

This case study has illustrated the use of neural networks for probability estimation on a chemical vapor deposition problem. Monte Carlo simulations were used to provide estimates of the reaction probabilities. These estimates were used as targets for the neural network. The network was able to capture the true underlying probability function without overfitting to the errors in the Monte Carlo estimates. This was accomplished by using the early stopping procedure, which stops network training if the error on an independent validation set increases.