Carbon Nanotube Gas Sensor Using Neural Networks

Introduction: The need to identify the presence and quantify the concentrations of gases and vapors is ubiquitous in NASA missions and societal applications. Sensors for air quality monitoring in crew cabins and ISS have been actively under development (Ref. 1). In particular, measuring the concentration of CO_2 and NH_3 is important because high concentrations of these gases pose a risk to ISS crew health. Detection of fuel and oxidant leaks in crew vehicles is critical for ensuring mission safety. Accurate gas and vapor concentrations can be measured, but this typically requires bulky and expensive instrumentation. Recently, inexpensive sensors with low power demands have been fabricated for use on the International Space Station (ISS). Carbon Nanotube (CNT) based chemical sensors are one type of these sensors. CNT sensors meet the requirements for low cost and ease of fabrication for deployment on the ISS. However, converting the measured signal from the sensors to human readable indicators of atmospheric air quality and safety is challenging. This is because it is difficult to develop an analytical model that maps the CNT sensor output signal to gas concentration. Training a neural network on CNT sensor data to predict gas concentration is more effective than developing an analytic approach to calculate the concentration from the same data set. With this in mind a neural network was created to tackle this challenge of converting the measured signal into CO_2 and NH_3 concentration values.

Language and API: The neural network was built in Python using Keras with TensorFlow as a backend, and the data to train the network was acquired from a CNT sensor developed by researchers at NASA Ames Research Center. Data preprocessing steps were conducted using Pandas and Sci-Kit Learn.

CNT Sensor Data: The CNT sensors studied detect gas concentration by measuring the change in resistance of the CNT material when different gases are present. Resistance changes are caused by gas adsorbing and desorbing from the surface of the CNTs. In a controlled laboratory environment, the CNT sensor was alternately exposed to CO_2 and NH_3 gas. Approximately 14 hours of gas exposure data from a 16-channel CNT sensor, along with the measured ambient pressure and relative humidity, were used to train the neural network. The logarithm of the exponential weighted moving average (ewm) of the data was concatenated with the raw 16-channel data, creating a data frame consisting of 34 columns. Each row in the data frame represented a five second data collection interval. The data was then transformed to form 5-minute running windows, and the average CO_2 and NH_3 gas concentration for each window was calculated from the measured values at each time step within a window (Figure 1). Raw and processed data were scaled separately using Scikit-Learn's RobustScaler. Through scaling the data from all source are centered and set to the same standard thus improving the predictive performance and convergence of the neural network model. The data was split into training (70%) and test (30%) sets.

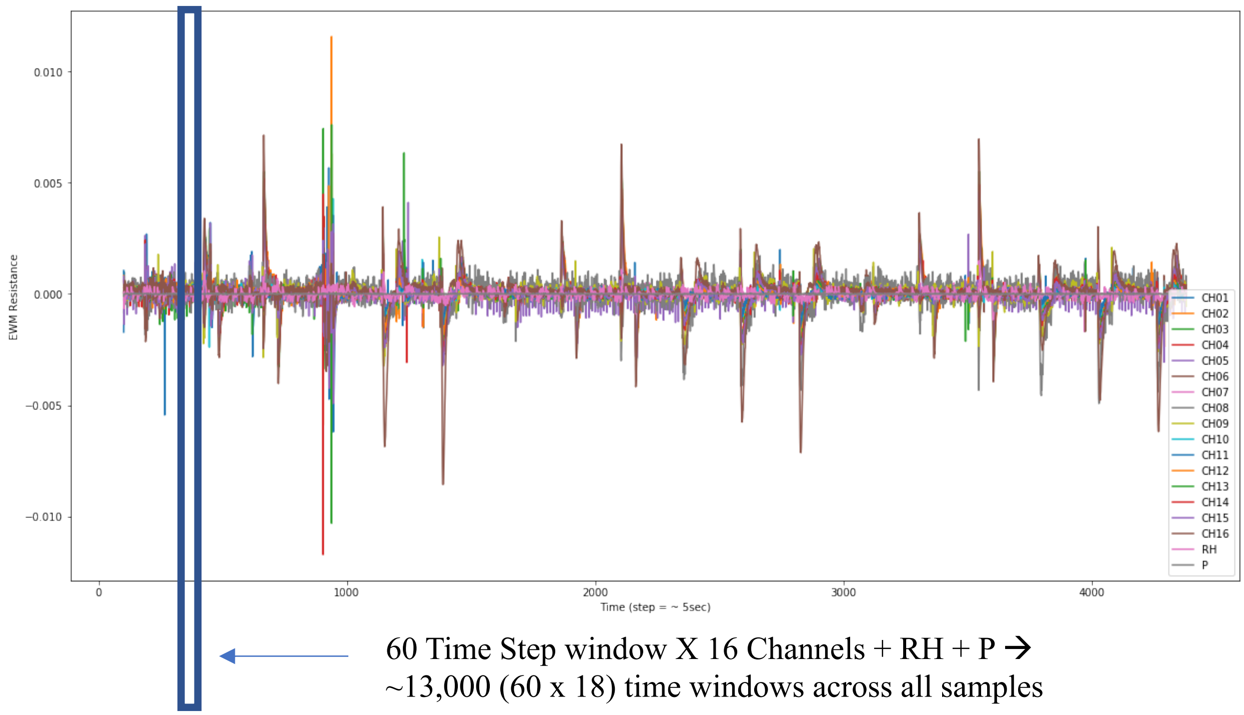

Figure 1: A plot of data after the processing steps and before the creation of the sliding window. The blue box rectangular box is to represent the running window. The dimensions of the box are not to scale.

Neural Network Architecture: The model chosen for the neural network architecture is based on two GRU (gated recurrent unit) neural network cells (Figure 2). One cell received scaled raw signal data, along with the pressure and humidity data, as input. The other received the processed CNT sensor data as input. The GRU cells are bidirectional, i.e., the data sequence is read into the cell in both forward and reverse order. The output of the GRU cells were combined and then passed into a fully connected neural network. The model was trained using the Adam optimizer with the mean squared logarithmic error (MSLE) as the loss function. The model was trained utilizing 33% cross-validation, and early stopping was used to reduce the training time. Dropout regularization was employed to mitigate overfitting.

```python
def lstm_model_02():
    #Inputs: a = differenced data, b = real data
    input_a = Input(shape=(60, 16))
    input_b = Input(shape=(60, 18))

    #LSTM layers
    L1 = Bidirectional(GRU(30))(input_a)
    L2 = Bidirectional(GRU(30))(input_b)

    #Concatentate L1 & L2
    c = keras.layers.concatenate([L1, L2])

    #FC layers
    fc1 = Dense(120, kernel_initializer = glorot_uniform(seed=None), activation = 'relu')(c)
    dr1 = Dropout(0.25)(fc1)
    fc2 = Dense(75, kernel_initializer = glorot_uniform(seed=None), activation = 'relu')(dr1)
    dr2 = Dropout(0.25)(fc2)
    fc3 = Dense(25, kernel_initializer = glorot_uniform(seed=None), activation = 'relu')(dr2)
    dr3 = Dropout(0.25)(fc3)

    #Target
    prediction = Dense(2, kernel_initializer = glorot_uniform(seed=None), activation = 'relu')(dr3)
    model = Model(inputs=[input_a, input_b], outputs=prediction)
    return model
```

```python
lstm_02 = lstm_model_02()
lstm_02.summary()
```

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 60, 16)	0	
input_2 (InputLayer)	(None, 60, 18)	0	
bidirectional_1 (Bidirectional)	(None, 60)	8460	input_1[0][0]
bidirectional_2 (Bidirectional)	(None, 60)	8820	input_2[0][0]
concatenate_1 (Concatenate)	(None, 120)	0	bidirectional_1[0][0] bidirectional_2[0][0]
dense_1 (Dense)	(None, 120)	14520	concatenate_1[0][0]
dropout_1 (Dropout)	(None, 120)	0	dense_1[0][0]
dense_2 (Dense)	(None, 75)	9075	dropout_1[0][0]
dropout_2 (Dropout)	(None, 75)	0	dense_2[0][0]
dense_3 (Dense)	(None, 25)	1900	dropout_2[0][0]
dropout_3 (Dropout)	(None, 25)	0	dense_3[0][0]
dense_4 (Dense)	(None, 2)	52	dropout_3[0][0]

Total params: 42,827
Trainable params: 42,827
Non-trainable params: 0

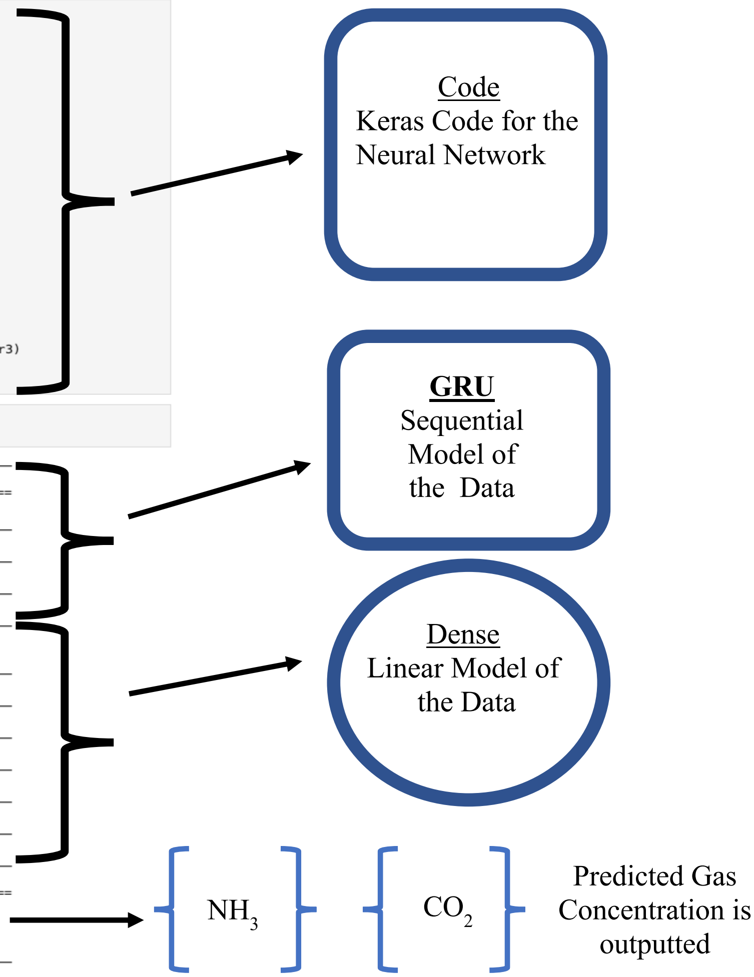

Figure 2: Keras code for the bidirectional neural network model. The model takes in two tensors: a 60 x 16 tensor composed of raw signal data and a 60 x 18 tensor composed of processed data. The inputs are passed first into separate GRU cells and then fed into the fully connected layers. The output is the prediction of CO_2 and NH_3 concentration sensed.

Results: The trained model produced a MSLE loss of 0.026 when evaluated against the test set. The calculated loss is a reflection of how close the predicted values are to the actual values. So small loss values are better than large ones. Plots of ground truth values, which are the actual gas concentration measured independently by a metal oxide gas sensor, versus the values predicted by the CNT gas sensor for CO_2 and NH_3 concentrations (Figure 3) confirmed that the model performed well at determining the gas concentrations. Therefore, a GRU cell is capable of learning complex sequential patterns such as those produced from a chemical sensor. The addition of bidirectionality to the model may also allow it to capture any hysteresis or lag in the adsorption and desorption of gas from the surface of the CNT that might be reflected in the resistance measurement. Two GRU cells, one fed with scaled raw data and the other with processed data, were observed to perform better at predicting the gas concentration than one. The two-GRU-cell approach was chosen because models using only one GRU fed with only processed data tended to predict negative gas concentrations (which has no physical meaning), while those fed with only scaled raw data could not predict gas concentrations at all. Finally, the details of the methodology and results can be found in Ref. 2.

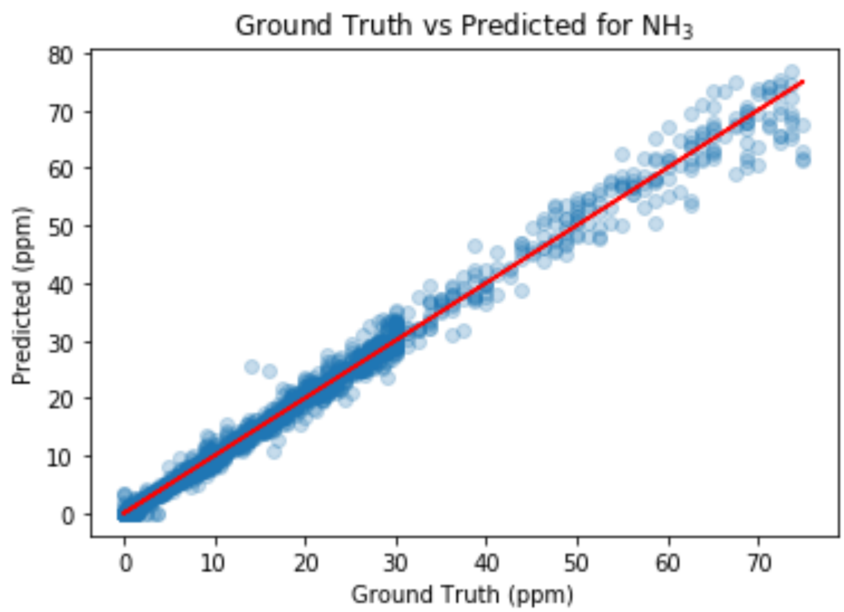

Figure 3a

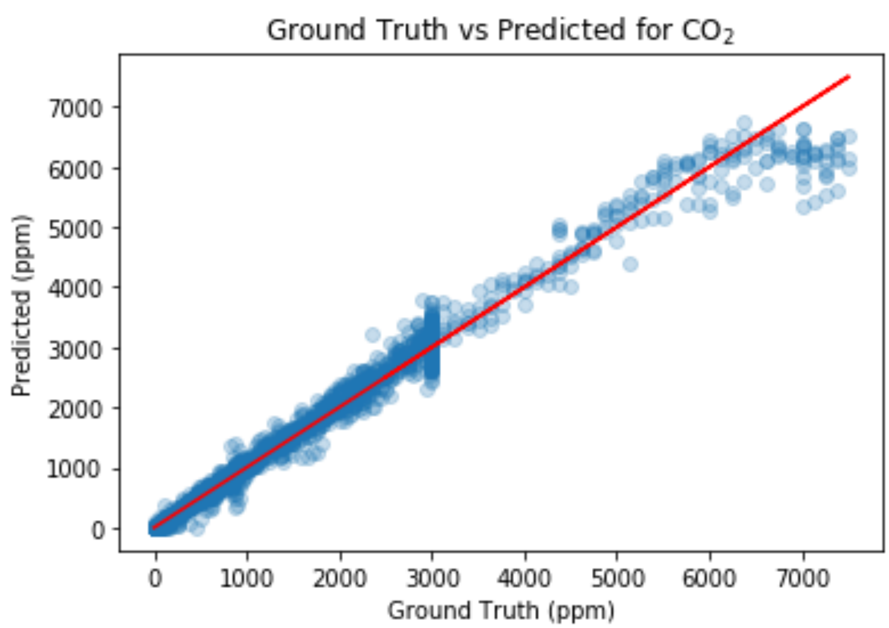

Figure 3b

Figure (3a): Plot of ground truth versus predicted NH_3 gas concentration evaluated using the test data set. **Figure (3b):** Plot of ground truth versus predicted CO_2 gas concentration evaluated using the test data set. Blue dots: concentration predicted by the CNT sensor at the concentration stated on the x-axis. Red line: concentrations if perfectly predicted.

Conclusion: A trained neural network model may be an effective method for converting CNT sensor signals to actual human interpretable gas concentrations. The next steps for this project are to collect vastly more gas sensor data (~1000 hours), retrain the model with this new data, and add other gases of interest, such as chlorine, in crew cabin air quality monitoring. Once this final step is completed, real-world testing in the ISS environment can begin.

References:

1. Pilgrim, J. S.; Wood, W. R.; Casias, M. E.; Vakhtin, A. B.; Johnson, M. D.; Mudgett, P. D. Optical MultiGas Monitor Technology Demonstration on the International Space Station. 44th International Conference on Environmental Systems; Tucson, AZ, Paper No. 58 July 13, 2014.

2. B. Kim, T.J. Norman, R.S. Jones, D.I. Moon, J.W. Han, and M. Meyyappan, Carboxylated single-walled carbon nanotube sensors with varying pH for the detection of ammonia and carbon dioxide using an artificial neural network, *ACS Applied Nanomaterials*, https://doi.org/10.1021/acsanm.9b01401