Feature Selection for Lightweight Deep Learning Based CO_2 Level Prediction in Edge Device

Khairi Hindriyandhito*, Radityo Fajar Pamungkas*, Ida Bagus Krishna Yoga Utama*, ByungDeok Chung†, and Yeong Min Jang*
*Department of Electronics Engineering, Kookmin University Seoul 02707, Korea
†ENS. Co. Ltd, Ansan 15655, Korea
Email: khairihindri@gmail.com, radityofajar@gmail.com, ibkyutama@ieee.org, bdchung@ens-km.co.kr, yjang@kookmin.ac.kr

***Abstract*—To properly predict and determine which features are critical for CO_2 level prediction, this work proposes a data collecting Intelligent IoT Platform that incorporates human detection algorithms using YOLOv8 into a time-series data collection, as well as a comprehensive analysis of the feature selection for deep learning models prediction performance by utilizing objective measurements such as MSE, RMSE, MAE, MAPE, and R^2. The results suggest that incorporating human detection techniques into the deep learning model improves its performance.**

***Index Terms*—Deep Learning, Internet of Things, Indoor Air Quality, Forecasting**

I. Introduction

To properly maintain CO_2 level, complete and accurate data for monitoring motives would be required, thus monitoring for CO_2 level is a common practice. Nowadays, to take a further step forward, a CO_2 level prediction is being done in some cases to prevent CO_2 level from dropping too low.

Many studies has already covered predicting CO_2 levels, such as [1], found that by incorporating additional datasets, the performance of the ARIMA prediction model can be significantly improved. However, it is important to note that their analysis was based on a limited dataset consisting of temperature, humidity, and CO_2 concentration measurements collected throughout a single day with 10-second intervals. Meanwhile [2] conducted a study on predicting using GRU and LSTM models by utilizing a dataset consisting of CO_2 concentration, fine dust, temperature, light quantity, and volatile organic compounds. The dataset spanned three months, and the results showed that the GRU model outperformed the LSTM model. For [3] in their study utilize several sensor data such as temperature, humidity, PIR, CO_2, and air pressure, their focuses were on doing a prediction for dataset generated from their data collecting sensors. They generated a dataset for one-year which was then made publicly accessible. From their studies it is stated that machine learning is a valid approach when using a multi-dimensional data for modeling building behavior, and adjusting indoor conditions based on forecast would support energy efficiency while simultaneously enhancing modern indoor spaces' occupational health and well-being properties. Furthermore, [4] address occupancy detection for office rooms from light, temperature, humidity, and CO_2 by utilizing classification and regression tree, with their result being a greater performance when using light sensor. For this study, we are looking further into the features that would be used for CO_2 level prediction and determine which of the features deemed important and unimportant. Mentioned by [5] in their study, the utilization of edge server which would act to lessen the cloud server's workload and to prevent a single point of failure to increase the platform reliability, we are implementing this model in an edge device, on which the computational power would be limited. Hypothetically, should the feature had more correlation to the CO_2, it would be better not to exclude that. The main contributions of our work are summarized as follows:

- Comparison of multiple features for predicting CO_2 levels in lightweight deep learning that would be implemented for edge device.
- A human detection scheme in edge device to enhance CO_2 levels prediction.

The rest of this study is documented as follows: Section II explains the methodology being used for this study. Section III emphasize the result received. Section IV concludes the study and future directions for this research.

II. Methodology

A. Data Collection and Human Detection Scheme

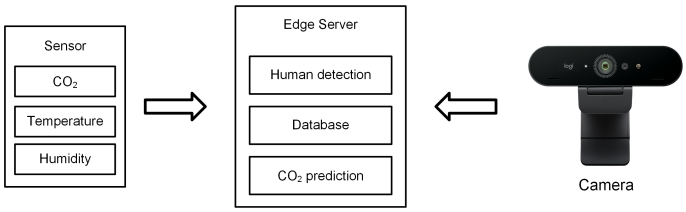

Fig. 1. Architecture Scheme

For data collection, we are utilizing an industrial CO_2 sensors that could measures several data for IAQ. Which are temperature, humidity, and CO_2. This sensor communicate through Modbus RTU communication protocol, an industrial standard for communication protocol. The data collected from this sensor would be sent to the edge server directly to be

stored and for further data processing. In this setup, we are collecting the data for 25 hours and 45 minutes with one second interval on which produces a total of 92700 data.

We are utilizing the YOLOv8 [6] algorithm, which is an open-source effort in the field of computer vision, for the purpose of human detection. Renowned for its exceptional velocity and proficiency in conducting instantaneous object identification [7]. In this scenario, it is crucial to have the capability to do real-time human object identification in order to accurately determine the number of individuals present at a given timestamp. Once the program identifies humans, it will then transmit data to the database in edge server at intervals of 1 second in order to compare it with data from other sensors. Therefore, the data received would match between the human detection and the sensor.

B. CO_2 Level Prediction

For predictions, the size of the deep learning model is important in this study as it would be implemented in an edge environment on which the computational power is limited. With lightweight in consideration, two of the basic types of deep learning models were chosen, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU).

LSTM [8] is a specific iteration of RNN that was developed to address the limitations of traditional RNNs in retaining information over long periods of time. The LSTM possesses a distinct architecture compared to RNN, which facilitates the acquisition of long-term dependencies. In order to accomplish that, LSTM utilizes memory units to preserve the cell state throughout a given period. The equation is formulated as follows:

$$i_t = \sigma(W_i i x_t + b_i i + W_h i h_(t-1) + b_h i) \tag{1}$$

$$f_t = \sigma(W_i f x_t + b_i f + W_h f h_(t-1) + b_h f) \tag{2}$$

$$c_t u = tanh(W_i c x_t + b_i c + W_h c h_(t-1) + b_h c) \tag{3}$$

$$c_t = f_t \cdot c_(t-1) + i_t \cdot c_t u \tag{4}$$

$$o_t = \sigma(W_i o x_t + b_i o + W_h o h_(t-1) + b_h o) \tag{5}$$

$$h_t = o_t \cdot tanh(c_t) \tag{6}$$

On which i_t is an input gate, f_t is the forget gate, o_t is the output gate, $c_t u$ is the cell state update, c_t is the cell state. W and b represents weight matrices and bias vectors, meanwhile subscripts ${}_i i$, ${}_i f$, ${}_i o$, ${}_i c$, ${}_h i$, ${}_h f$, ${}_h o$, and ${}_h c$ denotes different weights and biases for input, forget, output, input cell, hidden input, hidden forget, hidden output, and hidden cell connections. σ represents tanh activation function.

The gated recurrent unit (GRU) network is a variation of the long short-term memory (LSTM) network that has only two gates: the reset gate and the update gate. z_t represents the update gate, r_t represents the reset gate, $h_t u$ for hidden state update, and ht for the hidden state. The update gate decides how much of the prior hidden state should be kept, while the reset gate defines how much of the past information to forget. The proposed update for the hidden state is the hidden state update. The update gate controls the final hidden state, which is a combination of the hidden state update and the previous hidden state.

$$z_t = \sigma(W_i z x_t + b_i z + W_h z h_(t-1) + b_h z \tag{7}$$

$$r_t = \sigma(W_i c x_t + b_i r + W_h r h_(t-1) + b_h r \tag{8}$$

$$h_t u = \sigma(W_i c x_t + b_i c + r_t \cdot (W_h c h_(t-1) + b_h c)) \tag{9}$$

C. Prediction Validation

For validation, an objective evaluation are being used for every prediction made by each deep learning models. To properly evaluate the performance the prediction to the actual data. Five evaluation matrices are MSE, RMSE, MAE, MAPE, and R^2. Each of the matrices are represented as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2 \tag{10}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2} \tag{11}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{N} \mid Y_i - \hat{Y}_i \mid \tag{12}$$

$$MAPE = (\frac{1}{n}) \times \sum_{i=1}^{N} \frac{\mid Y_i - \hat{Y}_i \mid}{Y_i} \times 100 \tag{13}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{14}$$

III. Implementation Results

From the data collection, a total of 370800 data point was received. On which it consist of temperature, humidity, CO_2, and human detection data each consist of 92700 data point. The data match the data collection duration which is 25 hours and 45 minutes with 1 second interval. After receiving the data, a correlation coefficient for feature selection is being done. On which from Fig. 2 we can see that humidity has the highest correlation to CO_2 with 0.94, meanwhile human detection is lower with 0.77. Based on this correlation coefficient, we are making 4 scenarios: to do a prediction with all features to see how well the model would perform with every feature included, then a prediction without human detection feature to see the model performance without this feature, prediction without temperature feature because based from coefficient correlation it has the lowest value, and the last scenario is to predict without temperature and humidity to see how significant human detection feature is for CO_2 level prediction.

This study utilizes TensorFlow2.10 and Python3.10 to conduct the experiment for the prediction. The prediction is set for 300 seconds or five minutes, the prediction validation is being done by measuring MSE, MAE, MAPE, RMSE, and R^2 which can be seen at Table 1.

For Table 1, Fig. 3, and Fig. 4, an additional suffixes of "-P" means without the human detection feature, "-T" without

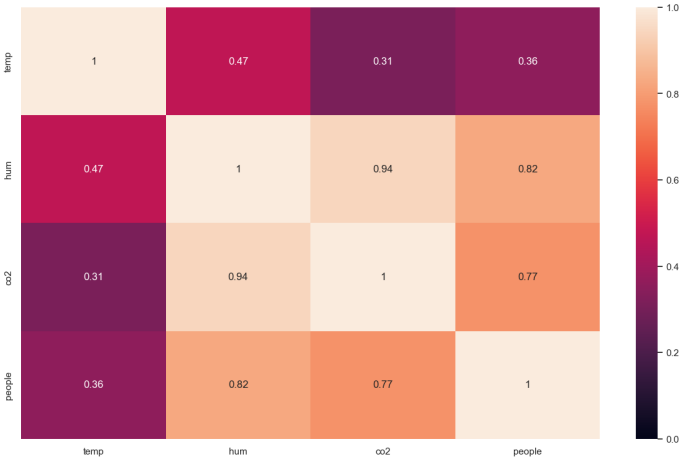

Fig. 2. Correlation Coefficient

TABLE I
PREDICTION VALIDATION VALUE

Model	*MSE*	*MAE*	*MAPE*	*RMSE*	R^2
GRU	7.6825	2.1690	0.0088	2.7717	0.7061
GRU-P	9.6177	2.5021	0.0102	3.1012	0.6076
GRU-T	6.7075	2.0144	0.0082	2.5899	0.7281
GRU-TH	6.4668	1.9660	0.0080	2.5430	0.7207
LSTM	18.9362	3.7634	0.0154	4.3516	0.2625
LSTM-P	6.3213	2.0073	0.0082	2.5142	0.7907
LSTM-T	6.3102	1.9719	0.0080	2.5120	0.7678
LSTM-TH	4.8767	1.7279	0.0070	2.2083	0.8081

the temperature feature, and "T-H" without temperature and humidity feature. As the result suggest, almost all of the scenarios and models has a good performance based on the visualized data. Meanwhile LSTM without temperature and humidity feature has the best performance when compared with the other scenarios, it has the lowest value for MSE, MAE, MAPE, and RMSE, meanwhile the R^2 value is the closest to 1.

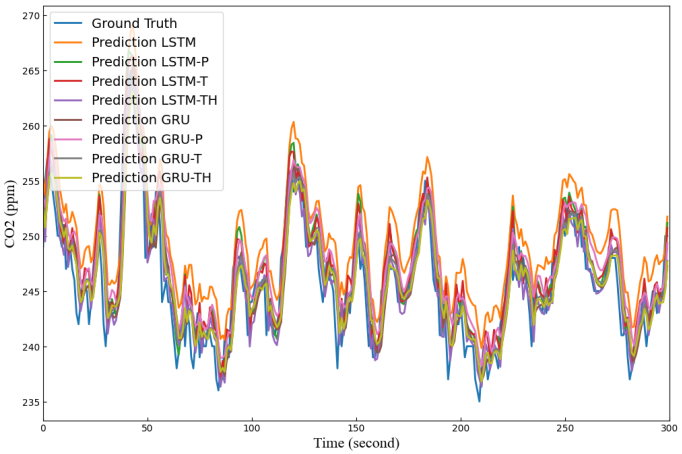

Fig. 3. Data Plot Visualization

TABLE II
MODEL AND PARAMETER SIZE

Model	*Parameter Size*	*Model Size*	*Time Elapsed*
GRU	19861	237 kB	0.0088
GRU-P	19771	235 kB	0.0102
GRU-T	19771	236 kB	0.0082
GRU-TH	19681	235 kB	0.0080
LSTM	26101	311 kB	0.0154
LSTM-P	25981	310 kB	0.0082
LSTM-T	25981	309 kB	0.0080
LSTM-TH	25861	308 kB	0.0070

IV. CONCLUSION

This human detection feature is playing a significant role for improving the CO_2 level prediction, even though the correlation coefficient shows that human detection feature has a lower score than humidity, the prediction results shows the opposite. Based on Table 1, when comparing "GRU" and "GRU-P" it can be seen that GRU has a better performance when including human detection feature, aside from that when removing humidity feature from the prediction, the performance got better in GRU and LSTM. On Fig. 4, it is a

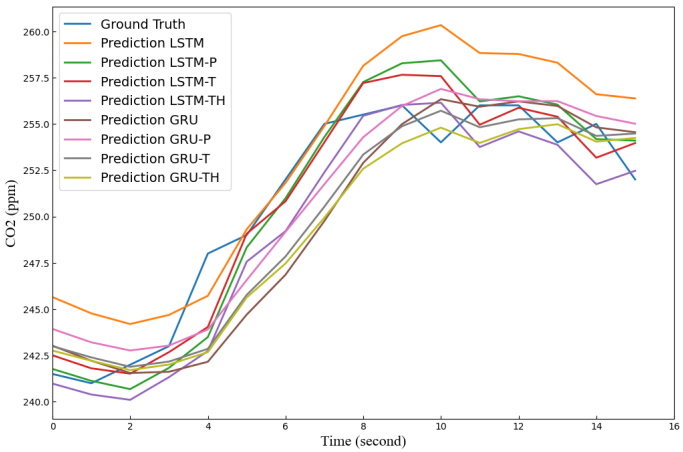

Fig. 4. Localized Data Plot Visualization

localized visualization for data on point 110 to 125 to give a further clear view of how the prediction performs when compared to ground truth. This shows that all models have a decent performance as it could roughly follows the ground truth pattern and managed to stay within below 10 ppm of error margin. All of the models could do a prediction in under one second, and the parameter size of both GRU and LSTM is not that high based on Table 2, GRU have a lower parameter size, model size, and time needed to run the model. For future works of this study, we can explore more to the state-of-the-art lightweight model and do a comparison with its former model. For the human detection algorithm scheme it could be applied for CO_2 level prediction as it shows a significant impact in this study.

ACKNOWLEDGMENT

This work was supported by the Technology development Program (S3098815) funded by the Ministry of SMEs and Startups(MSS, Korea).

REFERENCES

[1] T.-C. Yu and C.-C. Lin, "An Intelligent Wireless Sensing and Control System to Improve Indoor Air Quality: Monitoring, Prediction, and Preaction," International Journal of Distributed Sensor Networks, vol. 11, no. 8, p. 140978, Aug. 2015.

[2] J. Ahn, D. Shin, K. Kim, and J. Yang, "Indoor Air Quality Analysis Using Deep Learning with Sensor Data," Sensors, vol. 17, no. 11, p. 2476, Oct. 2017, doi: 10.3390/s17112476.

[3] J. Kallio, J. Tervonen, P. Räsänen, R. Mäkynen, J. Koivusaari, and J. Peltola, "Forecasting office indoor CO2 concentration using machine learning with a one-year dataset," Building and Environment, vol. 187, p. 107409, Jan. 2021, doi: 10.1016/j.buildenv.2020.107409.

[4] L. M. Candanedo and V. Feldheim, "Accurate occupancy detection of an office room from light, temperature, humidity and CO 2 measurements using statistical learning models," Energy and Buildings, vol. 112, pp. 28–39, Jan. 2016, doi: 10.1016/j.enbuild.2015.11.071.

[5] I. B. K. Y. Utama, R. F. Pamungkas, M. M. Faridh, and Y. M. Jang, "Intelligent IoT Platform for Multiple PV Plant Monitoring," Sensors, vol. 23, no. 15, p. 6674, Jul. 2023, doi: 10.3390/s23156674.

[6] Ultralytics, "GitHub - ultralytics/ultralytics: NEW - YOLOv8 in PyTorch ¿ ONNX ¿ OpenVINO ¿ CoreML ¿ TFLite," GitHub, Dec. 27, 2023. https://github.com/ultralytics/ultralytics

[7] [11] F. M. Talaat and H. ZainEldin, "An improved fire detection approach based on YOLO-v8 for smart cities," Neural Computing and Applications, vol. 35, no. 28, pp. 20939–20954, Jul. 2023, doi: 10.1007/s00521-023-08809-1.

[8] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.