Classroom Occupancy Monitoring System using IoT Device and the k-Nearest Neighbors Algorithm

Yarnish Dwi Sagita Fidarliyan*, Agung Budi Prasetijo, and Dania Eridani

Department of Computer Engineering, Universitas Diponegoro
Jl. Prof. Soedarto, SH., Semarang, 50275, Indonesia

**Corresponding author. Email: yarnishdwisagitaf@gmail.com*

Abstract— The occupancy monitoring system is one of the substantial aspects of building management. Through monitoring the occupancy in the area in a building, the obtained information can be used for building management purposes such as controlling indoor area air quality and improving building security. Some technologies such as video surveillance cameras, Radio Frequency Identification (RFID), and motion sensors have been used in the occupancy monitoring system. However, those technologies pose several disadvantages including privacy concerns and limited information generated. A classroom occupancy monitoring system using an Internet of Things (IoT) device and the k-Nearest Neighbors (k-NN) algorithm was built to monitor classroom occupancy by classifying the number of occupants based on classroom environmental data into occupancy levels by using the k-NN classifier model. By utilizing IoT devices, CO_2, temperature, and humidity data in a naturally ventilated classroom were recorded using the MQ-135 and BME280 sensors, as well as WiFi-based NodeMCU, was used to distribute data to the cloud. The collected data were trained and tested by the k-NN algorithm to produce a k-NN classifier model. From the tests conducted, the performance of the k-NN classifier model in classifying the number of occupants into occupancy levels resulted in an accuracy of 88%. In addition, the proposed system also produces a web-based classroom occupancy monitoring application that has been integrated with the k-NN classifier model so the classification can be done for real-time data and monitored directly.

Keywords— IoT; k-NN; occupancy levels; occupancy monitoring system

I. Introduction

Occupancy monitoring is a substantial component of building management. The occupancy information obtained through monitoring an area in the building is beneficial for estimating hourly sales rates in commercial buildings, improving building security services, and providing opportunities to implement automated Heating, Ventilation, and Air Conditioning (HVAC) systems in the building [1].

Several technologies have been used in a building occupancy monitoring system such as video surveillance cameras, Radio Frequency Identification (RFID), and motion sensors. However, it has several disadvantages: privacy concerns and limited information generated to name a few. The use of IoT to monitor the number of occupants in a building is widely used to overcome the limitations of previous technologies by providing low-cost computing and data distribution flexibility [2], [3].

Regarding occupancy monitoring, the indirect method is considered a good approach compared to the use of image and video processing which have privacy concerns in obtaining occupancy information [4]. The indirect method derives occupancy information by measuring the effect of the presence of occupants in their environmental conditions, where the change in its environment correlates with the number of occupants in the building [5]. The changes in environmental conditions due to the presence of occupants can be in the form of changes in CO_2 concentration, temperature, humidity, and so on.

The IoT device utilizes sensors to capture data on environmental changes in buildings due to the presence of occupants and distribute them for processing. Data processing using statistical and analytical methods such as machine learning is carried out to produce more accurate occupancy information. As in the previous study [1], a machine learning regression model was used in processing CO_2, humidity, and temperature data to obtain classroom occupancy information. The study resulted in the quantile regression model having better performance than the linear regression model in estimating occupancy with the Mean Absolute Percentage Error (MAPE) metric of 2.51%.

The study [6] found that the use of non-linear machine learning models such as k-Nearest Neighbors (k-NN), Support Vector Machine (SVM), and Decision Tree (DT) resulted in better accuracy in providing occupancy information than linear models. It is said that the linear model has strong assumptions on the distribution of the data so that it is vulnerable to outliers and produces lower performance accuracy than non-linear models. As in study [7], eight machine learning models were used to detect occupancy in an office, including linear and non-linear models. Using the office space temperature and humidity data, the study suggested the non-linear model, k-NN performed the best on the dataset with an auROC metric value of 0.77. A similar study [3] used three non-linear models to estimate occupancy levels in closed spaces. By using temperature, humidity, and pressure data, the results show that the k-NN model worked the best with an accuracy of 97%.

Previous studies [3], [7] have motivated this study to explore the use of a non-linear machine learning model, k-NN,

Received 18 June 2022, Accepted 16 December 2022, Published 30 December 2022.

DOI: https://doi.org/10.15294/jte.v14i2.37141

in occupancy monitoring by using different environmental variables. In this study, the CO_2, temperature, and humidity data of the classroom will be used to train a model/classifier using the k-NN algorithm. The use of CO_2, temperature, and humidity data have a correlation that can provide the information needed in the occupancy monitoring system [8].

In a recent study, the use of CO_2 to obtain occupancy information has been carried out, as in [9], Bayesian filtering with Extreme Learning Machine (ELM) was used to estimate indoor occupancy and resulted in an estimation accuracy of 77.29%. A similar study [10] with a single sensor to capture CO_2 was carried out to estimate occupancy levels in lab offices using SVM, Infinite Hidden Model Markov (IHMM), and Classification and Regression Trees (CART) which obtained an estimation accuracy of 82.5%. The combining use of CO_2, temperature, and humidity along with humidity ratio and light information to obtain the occupancy levels was carried out in [11], using the ELM model and CART parameters, resulting in an accuracy of 94.52%.

The generated occupancy information in this study is the result of classifying the number of occupants into 3 occupancy levels, namely low, medium, and high. Most of the previous similar studies found that information about occupancy was limited to the presence of occupants and occupied/unoccupied state. Classification into occupancy levels can be considered a good approach to obtaining occupancy information that applies to the use of IoT devices, especially when monitoring occupancy in densely populated environments [3]. As with the monitoring system, a web-based monitoring application was built in this study to display occupancy information resulting from classifying the number of occupants by the k-NN classifier model based on IoT device data which is CO_2, temperature, and humidity in real-time.

This study is presented in four sections: introduction, method, results and discussion, and conclusion. The introduction section describes the background of the problem, relevant literature, and the proposed approach of this study. The second section describes the research methodology of this study. The third section explains the analysis of the results and discussion of this research, and the last part is the conclusion of this study.

II. Method

This section describes how the classroom occupancy monitoring system was built, starting by collecting dataset using IoT device, setting up the cloud to store the data, processing dataset using the k-NN algorithm, and integrating the k-NN model classifier with a web-based user interface for classroom occupancy monitoring.

Figure 1 shows the system flow in a block diagram consisting of Input, Process, and Output. The Input part is the IoT device consisting of the NodeMCU ESP8266, the MQ-135, and the BME280 sensors, which collect the dataset. The IoT device schematic and its explanation are shown in detail in Figure 2 and Table I. The Process part shows the system flow from storing the collected dataset by the IoT device to ThingSpeak cloud, processing the exported dataset from the cloud by the k-NN algorithm in Jupyter Notebook, and integrating the k-NN model with a web-based monitoring application in Visual Studio Code. The Output part shows the k-NN classifier model that has been integrated with a web-based classroom occupancy monitoring application. In addition, the flow represented by the line between ThingSpeak and the Web User Interface in the block diagram shows data in the cloud will be displayed in real-time in the web-based occupancy monitoring application during the testing phase.

A. IoT device for Dataset Collection

The proposed system used an IoT device to collect the dataset, where the IoT device was designed using a WiFi-based NodeMCU ESP8226 as the main microcontroller to receive and transmit sensor data, as well as the MQ-135 and BME280 sensors which were used to capture changes in CO_2, temperature and humidity in the classroom. Figure 2 shows a circuit schematic of the IoT device that connects each component according to their needs based on a data sheet. Table I provides an explanation of the interface of each component in this IoT device.

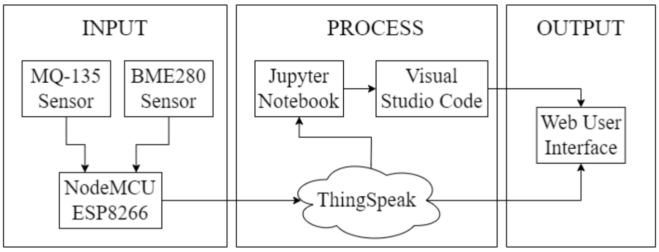


Figure 1. Block diagram of system

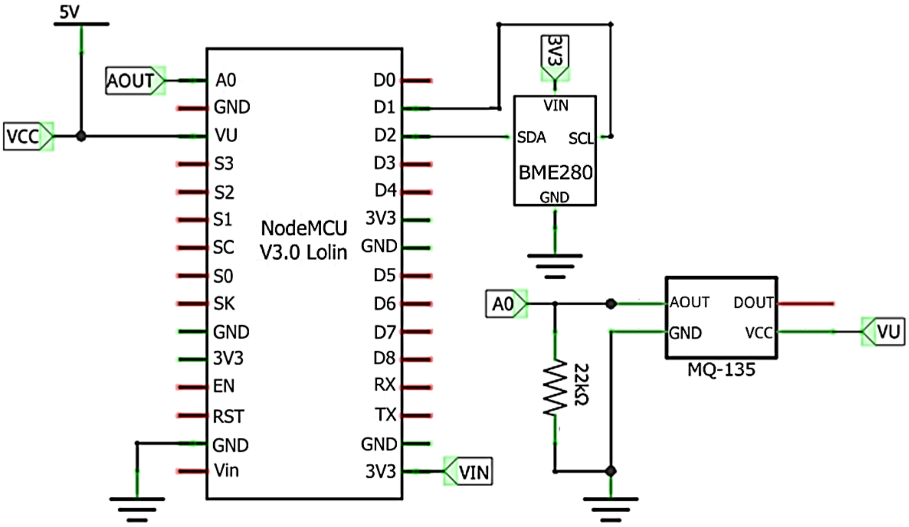


Figure 2. Schematic of an IoT device in classroom occupancy monitoring system

TABLE I. NODEMCU ESP8266 PIN INTERFACE WITH SENSOR COMPONENTS

No.	Pin		Function
	NodeMCU	Peripheral	
1.	A0	AOUT	Receiving and sending CO_2 data between NodeMCU and MQ-135 sensor
2.	D1	SCL	Synchronization clock between NodeMCU and BME280 sensor
3.	D2	SDA	Receiving and sending temperature and humidity data between NodeMCU and BME280 sensor
4.	VU	VCC	5V power source from NodeMCU USB socket, used by MQ-135 sensor
5.	3V3	VIN	3.3V power source, used by BME280 sensor
6.	GND	GND	Ground power source

The calibration of the MQ-135 sensor was conducted before being implemented on the IoT device to properly capture the CO_2 target gas. Prior to that, pre-heating of the sensor was carried out for 24 hours in an empty classroom to ensure that the sensor gets calibrated properly to produce a stable Ro resistance value [12]. Calibration begins by taking the Ro value at a certain temperature and humidity conditions in the classroom. After obtaining the value of Ro, the resistance Rs can be calculated using the voltage divider rule in (1). Based on the sensitivity characteristic diagram of MQ-135 [13], the ratio of the two resistances (Rs/Ro) is used to obtain the CO_2 concentration or the relationship between the two is expressed in Rs/Ro = f(ppm).

$$R_S = \frac{V_C \; x \; R_L}{V_{RL}} - R_L \qquad (1)$$

where V_C is the test voltage, V_{RL} is the drop of voltage on load resistance and R_L is the load resistance of MQ-135 based on datasheet recommendation (22KΩ).

The proposed system also addresses the possible influence of external conditions on sensor readings due to the presence of natural ventilation in the classroom. The magnitude of this influence is represented as a sensor error calculation due to the presence of natural ventilation to see whether the dataset read by the sensor can represent the actual conditions in the classroom. To find out the magnitude of the sensor error, a comparison was made between the readings of the MQ-135 and the calculation of CO_2 levels involving the opening area in the classroom. This comparison was made only on the MQ-135 sensor considering that CO_2 gas is more susceptible to being disturbed by external influences such as wind and so on, compared to temperature and humidity whose changes take time and are relatively constant.

Calculations involving the opening area are calculated using the equations below. Equation (2) is used to calculate the CO_2 concentration in the classroom after the t interval, involving the average concentration of CO_2 produced per occupant (G) at medium to low-intensity activities and conditions exchanged with the outside through natural ventilation [14], [15], where G for the number of occupants of 14 people is 2,856 l/hour. Then, the values of k and ACH in (2) are expressed in (3) and (4), respectively.

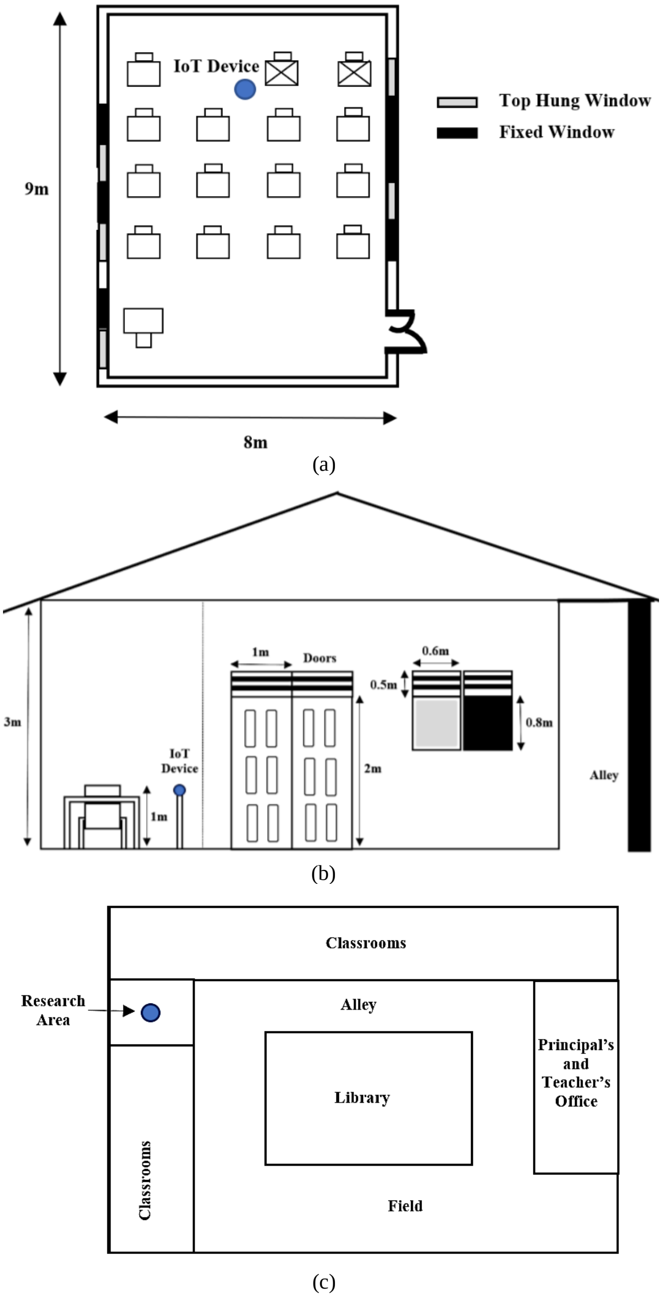
$$C(t) = C_{ext} + k - (C_{ext} - C_o + k).e^{-ACH.t/3600} \qquad (2)$$

$$k = \frac{G.10^6}{0.65 \; x \; openings \; (m^2)} \qquad (3)$$

$$ACH = \frac{0.65 \; x \; windspeed \; x \; openings \; (m^2) x \; 3600}{room \; volume \; (m^3)} \qquad (4)$$

with C_{ext} is the external CO_2 concentration (ppm), C_o is the initial CO_2 concentration in t interval (ppm), t is the measurement time interval (s), G is the average concentration of CO_2 per occuppants (m^3/s), and ACH is the air change per hour for ventilated room (h^{-1}).

After the IoT device setup was complete, dataset collection was carried out for seven days at 4-hour occupants' active sessions per day in a classroom from the building. The IoT device is placed at the back of the class to avoid direct interference from occupants and in the breathing height zone for occupants in a sitting position [16], as shown in Figure 3.



(a)

(b)

(c)

Figure 3. Placement of IoT device at (a) behind the classroom, (b) refers to the height and area opening of the classroom, and (c) the position of the classroom from the whole building

The collected dataset by the IoT device will be sent to the ThingSpeak cloud in 15-second intervals. In addition, calculations involving the opening area were carried out on the same day during dataset collection to directly compare the values under the same temperature and humidity conditions. The classroom opening area can be calculated referring to the class opening area described in Figure 3(b).

B. ThingSpeak Cloud Configuration for Dataset Storage

ThingSpeak cloud configuration was conducted to create channels for storing the dataset and generating an API Key. The IoT device microcontroller, NodeMCU ESP8266, used the Write API Key to send the dataset to the created cloud channels. In the main channel of the cloud, 3 fields will store CO_2 (field 1), temperature (field 2), and humidity (field 3) data in real-time with an update time of 15 seconds, while the second channel was made to calculate the value of the standard deviation and kurtosis of each field in the main channel which will only be used in testing phase on the occupancy monitoring application. In addition, the values on the two channels will be displayed during the testing phase on the web-based occupancy monitoring application every 5 minutes as new input data to classify the number of occupants into classroom occupancy levels.

C. Dataset Processing by k-NN Algorithm

The dataset collected in the ThingSpeak cloud was exported into a CSV file for processing by the k-NN algorithm in Jupyter Notebook. Figure 4 shows a dataset processing flow diagram using the k-NN algorithm starting from dataset labeling, dataset pre-processing, and dataset training and testing with the k-NN algorithm to produce the k-NN classifier model.

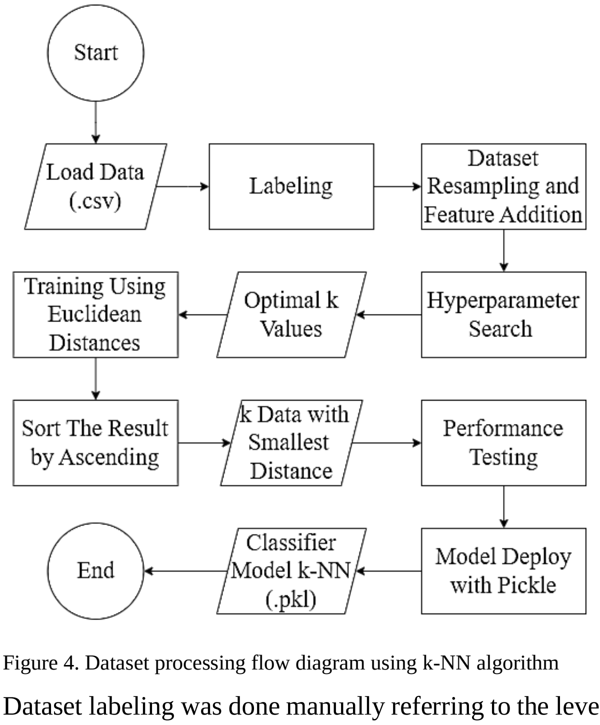

Start
Load Data (.csv)
Labeling
Dataset Resampling and Feature Addition
Hyperparameter Search
Optimal k Values
Training Using Euclidean Distances
Sort The Result by Ascending
k Data with Smallest Distance
Performance Testing
Model Deploy with Pickle
Classifier Model k-NN (.pkl)
End

Figure 4. Dataset processing flow diagram using k-NN algorithm

Dataset labeling was done manually referring to the level of occupancy recorded during dataset collection. The level of occupancy was established by dividing the number of occupants proportionally based on the maximum number of class occupants, namely 14 occupants, so the labels for each class are Low (0-4), Medium (5-9), and High (10-14).

After labeling, pre-processing was done by resampling the dataset for different time resolutions of 10 min, 5 min, and 1 min, with the aim of overcoming the sensor waiting time due to the non-temporary effect of occupants on classroom environmental conditions [17]. In addition, standard deviation and kurtosis features were added to each of the CO_2, temperature, and humidity, so there were a total of 9 features in the dataset. The addition of these features aims to make the k-NN classifier model able to distinguish the occupancy levels with precise accuracy since the standard deviation can provide information about the distance between the attribute with its mean, while the kurtosis represents the frequency of extreme attributes [3], [7].

In k-NN, k is a hyperparameter that represents the number of nearest neighbors in the dataset. The starting point for searching the k value is typically by using the square root of the total number of data [18], where 918 collected data points in this study can use the range of k up to $\sqrt[2]{918} \approx 30$ to find the optimal k value. The search for the optimal k was carried out by using the k-fold cross-validation method which is preferable over splitting the data method that is impractical and does not represent the generalization of the dataset. By using the k-fold cross-validation, each k value in the range 1-30 will be tested using 10 folds. The selection of the number of folds was based on the size of the data since there is no universal rule, but 10 folds were often used to obtain a lower error prediction [19], [20]. In this study, the 918 data points were distributed evenly into 10 folds which resulted in a fairly balanced data variation compared to the use of other folds, where the composition of the data for high, medium, and low occupancy levels in each fold was 2:1:1, respectively.

The pre-processed dataset will be trained and tested by dividing it into a ratio of 80%:20%, taking into account that the composition of each occupancy level is 2:1:1 in each training and testing set. The comparison ratio was chosen by considering the size of the dataset where 20% of the data is considered sufficient to validate the k-NN classifier model with a single hyperparameter. Furthermore, training of 80% of the data with the Euclidean Distance of the k-NN algorithm was carried out to produce a k-NN classifier model that can find the k-nearest neighbors. K-nearest neighbors will later determine the occupancy level of the new input data based on the majority of the classifications on the k data. Using (5), the distance between the new data point (x) and data point (y) in the dataset is calculated as follows.

$$d(x, y) = |x - y| = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{5}$$

where $\mathbf{x_i}$ is the Euclidean vector of new data point, $\mathbf{y_i}$ is Euclidean vector starting from the initial point in the dataset, and n is number of observations.

The performance of the trained k-NN model will be tested using test data in the Confusion Matrix. In the Confusion Matrix, the characteristics of True Positive, False Positive, True Negative, and False Negative are used to determine the performance metrics of the k-NN classifier model. The performance metrics from the confusion matrix were then used to measure precision, recall, f-1 score, and accuracy, which were calculated using (6), (7), (8), and (9), respectively.

$$\text{Precision}_{\text{level}} = \frac{TP_{\text{level}}}{TP_{\text{level}} + \sum FP_{\text{level}}} \tag{6}$$

$$\text{Recall}_{\text{level}} = \frac{TP_{\text{level}}}{TP_{\text{level}} + \sum FN_{\text{level}}} \tag{7}$$

$$\text{f1} - \text{score}_{\text{level}} = \frac{2 \text{ x } \text{Precision}_{\text{level}} \text{ x } \text{Recall}_{\text{level}}}{\text{Precision}_{\text{level}} + \text{Recall}_{\text{level}}} \tag{8}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \quad (9)$$

where TP is the True Positive, FP is the False Positive, FN is the False Negative, and TN is the True Negative.

The k-NN classifier model that has been tested was deployed with the help of the Python Pickle library to generate a pickle file (.pkl) which will be integrated with a web-based classroom occupancy monitoring application.

D. Integrating k-NN Classifier Model with a Web-based Classroom Occupancy Monitoring Application

The deployed k-NN classifier model was integrated with a web-based classroom occupancy monitoring application. The web-based application was built using HTML, Bootstrap framework, and Python Flask in Visual Studio Code, where a knn.html page was created to display classroom environment data received via the cloud API in real-time at 5-minute intervals and used for classification by the classifier model k-NN, also an index.html page was created as route app @app.route('/') or the initial page when the user accesses the occupancy monitoring application domain. The k-NN classifier model was integrated into the knn.html page with the help of render_template from the Flask library so that the model can classify the number of occupants based on new input data, classroom environment data, from the cloud and produce classroom occupancy level output on that page.

III. Results and Discussion

This section discusses the results obtained from testing the performance of the k-NN model in classifying the number of occupants into occupancy levels. In addition, it will explain the tuning results of the k-NN hyperparameter values using k-fold cross-validation. Testing on web-based occupancy monitoring applications, testing on IoT sensors, and analysis of the environmental effect on the system will also be discussed in this section.

A. MQ-135 Sensor Testing

In the MQ-135 sensor test, the R_O value was taken in a classroom with a temperature of 26°C and humidity of 69%, with a sensor R_L value of 22KΩ. With the R_O value obtained through measurements in this study, which is 10700Ω, Table II shows the data from the test results on the MQ-135 sensor. The test results were used to see the relationship between the measured CO_2 concentration and the Rs/Ro resistance value shown in the graph in Figure 5.

TABLE II. MQ-135 Sensor Test Results

No.	R_S (Ω)	R_S/R_O	CO_2 (ppm)	CO_2 (Log)
1.	6780.0	0.63000	420.53	2.6237
2.	6816.9	0.64000	413.99	2.6169
3.	6853.9	0.64000	407.58	2.6102
4.	6743.3	0.63000	427.19	2.6306
5.	6670.1	0.62000	440.89	2.6443
6.	6965.2	0.65000	389.02	2.5899
7.	6706.6	0.63000	433.98	2.6374
8.	7040.6	0.66000	377.20	2.5765
9.	7002.6	0.65000	383.05	2.5832
10.	6345.1	0.59000	509.37	2.7070

Note: The R_O value was taken during calibration where this value is a constant sensor resistance at certain classroom temperature and humidity conditions. Extreme changes in both conditions will change the value of R_O. The R_S value is the sensor resistance which is always changing due to interaction with the CO_2 target gas. The CO_2(ppm) value is the R_S/R_O ratio according to the MQ-135 sensitivity chart on the datasheet which is stated in f(ppm) = R_S/R_O.

Figure 5 shows an inverse relationship between CO_2 concentration with the R_S/R_O resistance value. The trend line in the graph in Figure 5 has the characteristic sensitivity of MQ-135 for CO_2 gas which is in accordance with the datasheet, when the sensitive material of the MQ-135 sensor increases in conductivity which indicates an increase in CO_2 levels, the R_S/R_O resistance value will decrease, and vice versa. This concluded that the MQ-135 was able to capture changes in CO_2 gas in the classroom where the research was conducted.

The graph of the relationship between CO_2 concentration and R_S/R_O resistance also produces a regression equation that indicates linearity through y = -0.5134x + 1.9801 and the determination value (R^2) is 0.9808. The determination value represents 98.08% of the variation that can be predicted using the y regression equation and the remaining 1.92% are other factors that affect the concentration of CO_2 gas in addition to sensor resistance such as room temperature and humidity, and gas pressure.

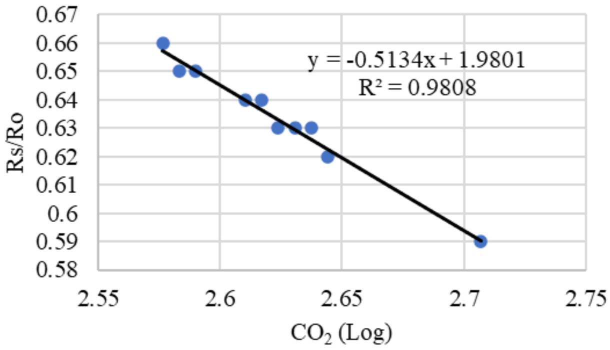


Figure 5. Graph of relationship between CO_2 concentration and resistance ratio R_S/R_O

Furthermore, a comparison was made between the obtained CO_2 levels obtained through MQ-135 and the CO_2 levels calculated by involving the opening area in the classroom. With a total opening area in the classroom of 10,45 m^2, Table III shows comparison results for seven days. In calculating the CO_2 concentration involving the opening area, the CO_2 concentration produces a constant value of 414.78 ppm. These results were obtained as a result of using the outdoor CO_2 concentration value (Cext = 414.67 ppm) and the average CO_2 concentration produced by the occupants (G = 2.856 l/h) was constant.

TABLE III. CO_2 Concetration Comparison Results

Day	CO_2 Concentration by MQ-135 (ppm)	CO_2 Concentration by Calculations Involving Opening Area (ppm)
1.	451.43	414.78
2.	369.86	414.78
3.	417.63	414.78
4.	377.04	414.78
5.	369.31	414.78
6.	456.52	414.78
7.	408.67	414.78
Mean	407.21	414.78

The difference in CO_2 concentration obtained by reading MQ-135 with calculations involving the opening area is 7.57 ppm. The difference between the two concentrations is also calculated using a relative error and produces a percentage of error by the sensor of 1.825%. Based on the relative error value obtained by the sensor, it can be concluded that there is an external effect due to the presence of natural ventilation but does not give a significant effect on sensor readings. Factors that influence these results include the location of the classroom which is not directly exposed to the wind inflow and the use of natural ventilation shows a slow effect on changes in classroom conditions. Even though the MQ-135 sensor has been able to capture data that represents the state of CO_2 in the classroom, there is a drawback in using the MQ-135. Compared to [21] using the Building Integrated Control Test-bed (BICT), the use of MQ-135 to collect ground truth occupancy profiles and CO_2 dataset in a naturally ventilated classroom requires repeated calibration of the constant Ro value when extreme temperature and humidity changes occur.

B. BME280 Sensor Testing

Testing the accuracy of BME280 sensor readings in detecting classroom temperature and humidity was carried out by calculating the average difference between the sensor readings and the digital thermometer as a standard meter. Table IV and Table V are test results for the BME280 sensor, respectively, as a measure of classroom temperature and humidity. Based on the Tables, the tests performed 10 times on the BME280 sensor for temperature and humidity measurements resulted in an average difference of 0.276°C and 0.657%, respectively. The difference in values indicates an accuracy offset where the accuracy tolerance on the datasheet is ±1°C and ±3%, which means that the sensor can capture classroom temperature and humidity with an acceptable error.

TABLE IV. BME280 SENSOR TEST RESULTS FOR TEMPERATURE

No.	BME280 (°C)	Thermometer HTC-1 (°C)	Δ (°C)
1.	28.66	28.00	0.66
2.	28.69	28.00	0.69
3.	28.73	28.30	0.43
4.	28.78	28.70	0.08
5.	28.83	28.70	0.13
6.	28.85	28.70	0.15
7.	28.85	28.70	0.15
8.	28.83	28.70	0.13
9.	28.87	28.70	0.17
10.	28.87	28.70	0.17

TABLE V. BME280 SENSOR TEST RESULTS FOR HUMIDITY

No.	BME280 (%)	Thermometer HTC-1 (%)	Δ (%)
1.	61.05	61.00	0.05
2.	60.51	61.00	0.49
3.	59.58	61.00	1.42
4.	59.40	60.00	0.60
5.	59.33	59.00	0.33
6.	59.07	58.00	1.07
7.	58.97	59.00	0.03
8.	59.69	58.00	1.69
9.	58.18	58.00	0.18
10.	58.29	59.00	0.71

C. k-NN Hyperparameter Search by k-Fold Cross Validation

Testing the value of k for the range 1-30 was carried out with 10-fold iterations resulting in the best k value of k=1 with 86% accuracy and k=5 with 85.9% accuracy, as shown in Figure 6. However, the use of k=1 was not applied to the model because the classification will be made based only on a single nearest neighbor who has similar data, this typically led to overfitting. In addition, there was a decrease in the accuracy along with the increase of k as shown in Figure 6. The accuracy decreased because it was calculated based on an average of 10-folds, where an increase in the k means that there are more nearest neighbors but the distance value is further away so the average accuracy has a larger bias error. Thus, only k in a certain range of values will produce optimum accuracy, but after passing the optimum limit there will be a decrease in accuracy.

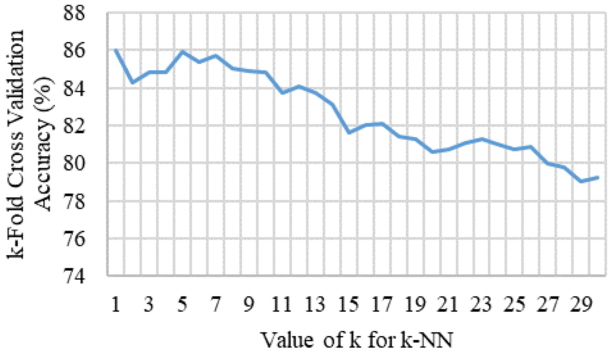


Figure 6. Optimal k values of k-NN obtained through k-fold cross validation

D. k-NN Classifier Model Performance Test

The performance test on the trained k-NN classifier model was carried out by using a Confusion Matrix. The confusion Matrix shown in Figure 7 was generated by using a test set in the testing phase. The metrics used in evaluating the performance of the k-NN classifier model using the Confusion Matrix are precision, recall, and f-1 score for each occupancy level as presented in Table VI.

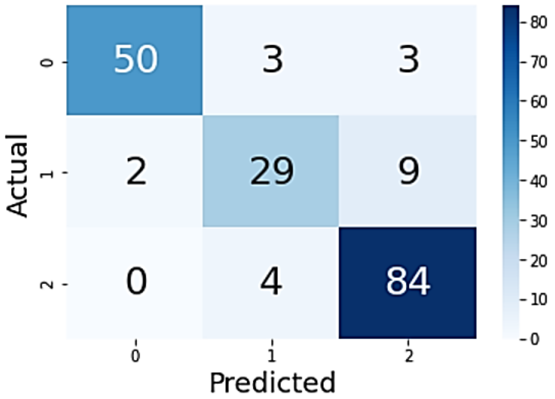
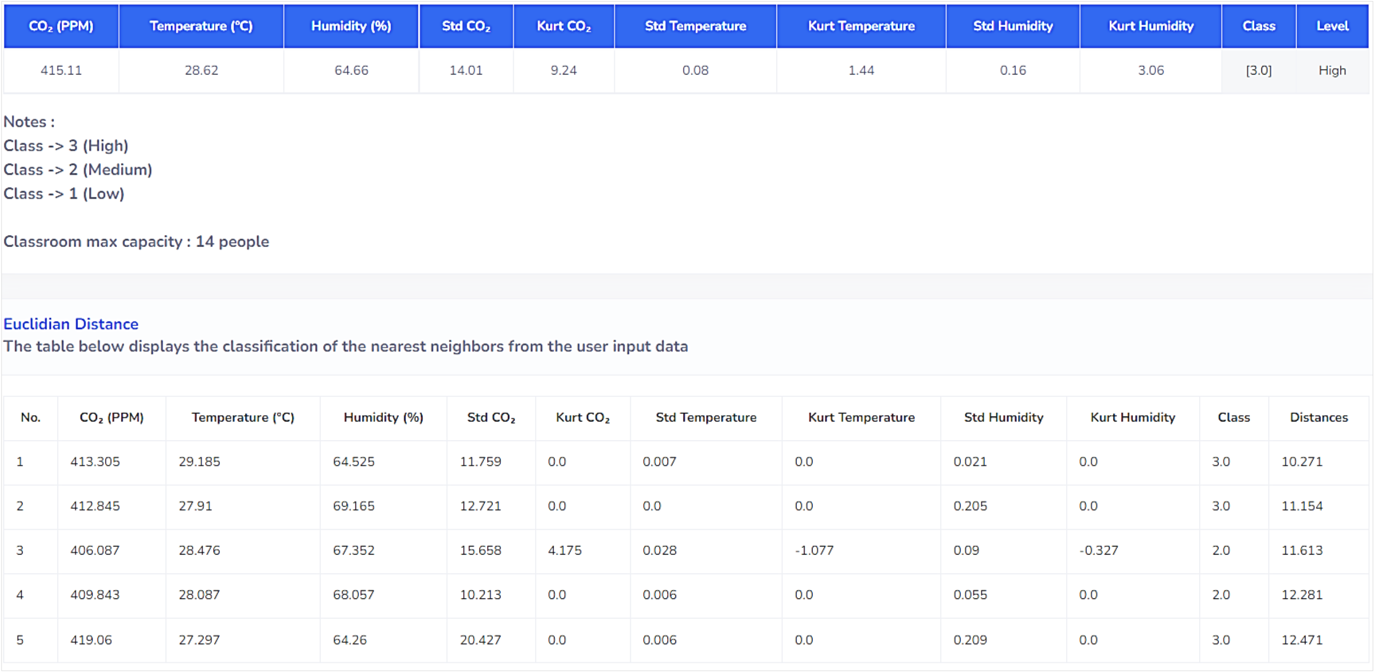


Figure 7. Confusion matrix

TABLE VI. RESULTS OF K-NN MODEL PERFORMANCE

Occupancy Level	Precision	Recall	F-1 Score
Low	96%	89%	93%
Medium	81%	72%	76%
High	88%	95%	91%
Macro Average	88%	86%	88%
Accuracy			88%

Note: Precision, Recall, F-1 score and Accuracy are the results of the Confusion Matrix calculations in Figure 7, where the calculation involves 4 characteristics of the Confusion Matrix which are calculated using equations (6-9).



CO_2 (PPM)	Temperature (°C)	Humidity (%)	Std CO_2	Kurt CO_2	Std Temperature	Kurt Temperature	Std Humidity	Kurt Humidity	Class	Level
415.11	28.62	64.66	14.01	9.24	0.08	1.44	0.16	3.06	[3.0]	High

Notes :
Class -> 3 (High)
Class -> 2 (Medium)
Class -> 1 (Low)

Classroom max capacity : 14 people

Euclidian Distance
The table below displays the classification of the nearest neighbors from the user input data

No.	CO_2 (PPM)	Temperature (°C)	Humidity (%)	Std CO_2	Kurt CO_2	Std Temperature	Kurt Temperature	Std Humidity	Kurt Humidity	Class	Distances
1	413.305	29.185	64.525	11.759	0.0	0.007	0.0	0.021	0.0	3.0	10.271
2	412.845	27.91	69.165	12.721	0.0	0.0	0.0	0.205	0.0	3.0	11.154
3	406.087	28.476	67.352	15.658	4.175	0.028	-1.077	0.09	-0.327	2.0	11.613
4	409.843	28.087	68.057	10.213	0.0	0.006	0.0	0.055	0.0	2.0	12.281
5	419.06	27.297	64.26	20.427	0.0	0.006	0.0	0.209	0.0	3.0	12.471

Figure 8. Classification result by k-NN classifier model tested on classroom occupancy monitoring application

The performance evaluation results of the k-NN classifier model yield a precision of 88%, recall of 86%, and an f-1 score of 88%. Then, the accuracy of the k-NN model to classify correctly based on the actual value of the total data produces an accuracy of 88%. As referred in the k-fold cross-validation test, there is an accuracy difference for the use of k=5. This occurs due to different dataset handling where the k-fold cross-validation used the entire dataset in rotation fold as test data while the confusion matrix used a split dataset where the model was tested by using test data that comes from 20% of the dataset.

This study obtained 88% accuracy in generating occupancy information by utilizing a combination of CO_2, humidity, and temperature data, resulting in better accuracy than studies that used single CO_2 data [5], [6]. In addition, the use of the k-NN algorithm on combined data also results in better accuracy than in the previous study [7]. This result occurs due to several factors, such as standard deviation and kurtosis features addition that mentioned in [3] being able to provide insight to produce better occupancy information.

E. Implementation of k-NN Classifier Model to a Web-based Monitoring Application

In the web-based classroom occupancy monitoring application, the implemented k-NN classifier model will classify the number of occupants based on changes in class environmental conditions to occupancy levels. Testing was carried out on a monitoring web application, where the form on the knn.html page was used to manually fill in class environmental data received via the cloud for an update time every 5 minutes. The form has a submit button that functions to display the results of the classification by the k-NN classifier model and the calculation distance by Euclidean Distances, which were displayed in the table by providing the nearest distance data as much as k data (k = 5). Figure 8 shows the test results on the knn.html where the classification results from the input data produce a high occupancy level. With reference to the Euclidean Distances table on that page, the high-level occupancy classification is correct according to the majority of classifications in the table where there are closest neighbors with 3 data classified as Class 3 (High Level) and 2 data classified as Class 2 (Medium Level). Overall tests on the web-based classroom occupancy monitoring application show that the implemented k-NN classifier model can perform classification and produce classification results in the form of classroom occupancy levels that can be monitored directly.

IV. Conclusion

A system to monitor classroom occupancy has been successfully built. This study used the k-NN machine learning approach on classroom environmental data such as CO_2, temperature, and humidity data captured by the IoT device. The dataset has been successfully trained and tested using the k-NN algorithm, resulting in a k-NN classifier model that can classify the number of occupants into 3 occupancy levels, namely low, medium, and high. The performance testing of the k-NN classifier model produces measurement metrics for each occupancy level of 88% precision, 86% recall, 88% f-1 score, and 88% accuracy. It is important to highlight that performance metrics were achieved under the following conditions: use of temperature, humidity, and CO2 data in a naturally ventilated classroom; use of datasets obtained from occupants' active hours due to limited permits caused by the pandemic. In addition, a web-based classroom occupancy monitoring application has also been built and integrated with the k-NN classifier model, so the classification can be done for real-time data and makes it easier for users to monitor classroom occupancy levels directly. For future research, we suggest exploring the use of data other than CO_2, temperature and humidity, and other machine learning models. Further improvement on a web-based occupancy monitoring application can also be done to automatically classify occupancy levels.

References

[1] K. Rastogi and D. Lohani, “IoT-based Indoor Occupancy Estimation Using Edge Computing,” *Procedia Comput. Sci.*, vol. 171, pp. 1943–1952, Jan. 2020, doi: 10.1016/j.procs.2020.04.208.

[2] V. Chidurala and X. Li, “Occupancy Estimation Using Thermal Imaging Sensors and Machine Learning Algorithms,” *IEEE Sens. J.*, vol. 21, no. 6, pp. 8627–8638, 2021, doi: 10.1109/JSEN.2021.3049311.

[3] A. Vela, J. Alvarado-Uribe, M. Davila, N. Hernandez-Gress, and H. G. Ceballos, “Estimating Occupancy Levels in Enclosed Spaces Using Environmental Variables: A Fitness Gym and Living Room as Evaluation Scenarios,” *Sensors (Switzerland)*, vol. 20, no. 22, pp. 1–21,

2020, doi: 10.3390/s20226579.

[4] J. Wang, N. C. F. Tse, T. Y. Poon, and J. Y. C. Chan, “A Practical Multi-sensor Cooling Demand Estimation Approach Based on Visual, Indoor and Outdoor Information Sensing,” *Sensors (Switzerland)*, vol. 18, no. 11, p. 3591, 2018, doi: 10.3390/s18113591.

[5] K. Mjörnell, D. Johansson, and H. Bagge, “The Effect of High Occupancy Density on IAQ, Moisture Conditions and Energy Use in Apartments,” *Energies*, vol. 12, no. 23, pp. 1–11, 2019, doi: 10.3390/en12234454.

[6] P. Anand, C. Deb, K. Yan, J. Yang, D. Cheong, and C. Sekhar, “Occupancy-based Energy Consumption Modelling Using Machine Learning Algorithms for Institutional Buildings,” *Energy Build.*, vol. 252, no. 4, p. 111478, 2021, doi: https://doi.org/10.1016/j.enbuild.2021.111478.

[7] S. Zemouri, Y. Gkoufas, and J. Murphy, “A Machine Learning Approach to Indoor Occupancy Detection Using Non-intrusive Environmental Sensor Data,” *ACM Int. Conf. Proceeding Ser.*, pp. 70–74, 2019, doi: 10.1145/3361758.3361775.

[8] S. Taheri and A. Razban, “Learning-based CO2 Concentration Prediction: Application to Indoor Air Quality Control Using Demand-Controlled Ventilation,” *Build. Environ.*, vol. 205, no. 2, p. 108164, 2021, doi: 10.1016/j.buildenv.2021.108164.

[9] C. Jiang, Z. Chen, R. Su, M. K. Masood, and Y. C. Soh, “Bayesian Filtering For Building Occupancy Estimation From Carbon Dioxide Concentration,” *Energy Build.*, vol. 206, p. 109566, 2020, doi: 10.1016/j.enbuild.2019.109566.

[10] Y. Zhou *et al.*, “A Novel Model Based on Multi-Grained Cascade Forests with Wavelet Denoising for Indoor Occupancy Estimation,” *Build. Environ.*, vol. 167, no. 10, p. 106461, 2020, doi: 10.1016/j.buildenv.2019.106461.

[11] S. Kumar, J. Singh, and O. Singh, “Ensemble-based Extreme Learning Machine Model For Occupancy Detection with Ambient Attributes,” *Int. J. Syst. Assur. Eng. Manag.*, vol. 11, no. 2, pp. 173–183, 2020, doi: 10.1007/s13198-019-00935-1.

[12] Y. R. Carrillo-Amado, M. A. Califa-Urquiza, and J. A. Ramón-Valencia, “Calibration and Standardization of Air Quality Measurements Using MQ Sensors,” *Respuestas*, vol. 25, no. 1, pp. 70–77, 2020, doi: 10.22463/0122820x.2408.

[13] M. Sebayang, “Stasiun Pemantau Kualitas Udara Berbasis Web,” *J. INFORMATICS Telecommun. Eng.*, vol. 1, no. 1, pp. 24–33, Sep. 2017, doi: 10.31289/jite.v1i1.571.

[14] L. Schibuola and C. Tambani, “Indoor Environmental Quality Classification of School Environments by Monitoring PM and CO2 Concentration Levels,” *Atmos. Pollut. Res.*, vol. 11, no. 2, pp. 332–342, 2020, doi: 10.1016/j.apr.2019.11.006.

[15] S. Jayakumar and M. G. Apte, “Estimation and Analysis of Ventilation Rates in Schools in Indian Context: IAQ and Indoor Environmental Quality,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 609, no. 3, p. 032046, 2019, doi: 10.1088/1757-899X/609/3/032046.

[16] B. Talarosha, “Konsentrasi CO2 pada Ruang Kelas dengan Sistem Ventilasi Alami sebuah Penelitian Awal,” *J. Lingkung. Binaan Indones.*, vol. 6, no. 1, pp. 22–27, Apr. 2017, doi: 10.32315/jlbi.6.1.22.

[17] D. Stjelja, J. Jokisalo, and R. Kosonen, “Scalable Room Occupancy Prediction with Deep Transfer Learning Using Indoor Climate Sensor,” *Energies*, vol. 15, no. 6, pp. 1–21, 2022, doi: 10.3390/en15062078.

[18] A. Pandey and A. Jain, “Comparative Analysis of KNN Algorithm using Various Normalization Techniques,” *Int. J. Comput. Netw. Inf. Secur.*, vol. 9, no. 11, pp. 36–42, 2017, doi: 10.5815/ijcnis.2017.11.04.

[19] Y. Jung, “Multiple predicting K-fold cross-validation for model selection,” *J. Nonparametr. Stat.*, vol. 30, no. 1, pp. 197–215, 2018, doi: 10.1080/10485252.2017.1404598.

[20] B. G. Marcot and A. M. Hanea, “What is an optimal value of k in k-fold cross-validation in discrete Bayesian network analysis?,” *Comput. Stat.*, vol. 36, no. 3, pp. 2009–2031, 2021, doi: 10.1007/s00180-020-00999-9.

[21] S. H. Kim and H. J. Moon, “Case Study of An Advanced Integrated Comfort Control Algorithm with Cooling, Ventilation, and Humidification Systems Based on Occupancy Status,” *Build. Environ.*, vol. 133, pp. 246–264, 2018, doi: 10.1016/j.buildenv.2017.12.010.