Gas sensors for House Monitoring System

Vinay Varshith . Pechetti , Sruti . Baru and Rohith Kumar . Kattagani

AAI-530: Data Analytics and the Internet of Things

Author Note

Correspondence concerning this article should be addressed to Vinay Varshith . Pechetti, Email: vinayvarshith.pechetti@slu.edu or Sruti . Baru Email: sruti.baru@slu.edu or Rohith Kumar . Kattagani, Email: rohithkumar.kattagani@slu.edu

Abstract

Gas sensors are crucial components of home monitoring systems, detecting and monitoring harmful gases like carbon monoxide and methane in residential environments. With advances in sensor technology and the Internet of Things (IoT), gas sensors have become more affordable and accessible to homeowners, enabling remote gas level monitoring and real-time smartphone alerts. Integrating gas sensors into home monitoring systems enhances safety by preventing accidents such as gas leaks and carbon monoxide poisoning. Additionally, gas sensors provide valuable data on gas consumption patterns, aiding in energy optimization and cost reduction. The availability of gas sensor data also presents opportunities for developing advanced machine learning models that accurately detect gas levels and predict consumption patterns, further improving the efficiency and safety of home monitoring systems. This project emphasizes the importance of gas sensors, highlights their role in ensuring safety and optimizing energy consumption, and explores the application of machine learning techniques in enhancing gas sensor systems.

Keywords: Gas sensors, home monitoring systems, carbon monoxide, methane, Internet of Things (IoT), safety, machine learning

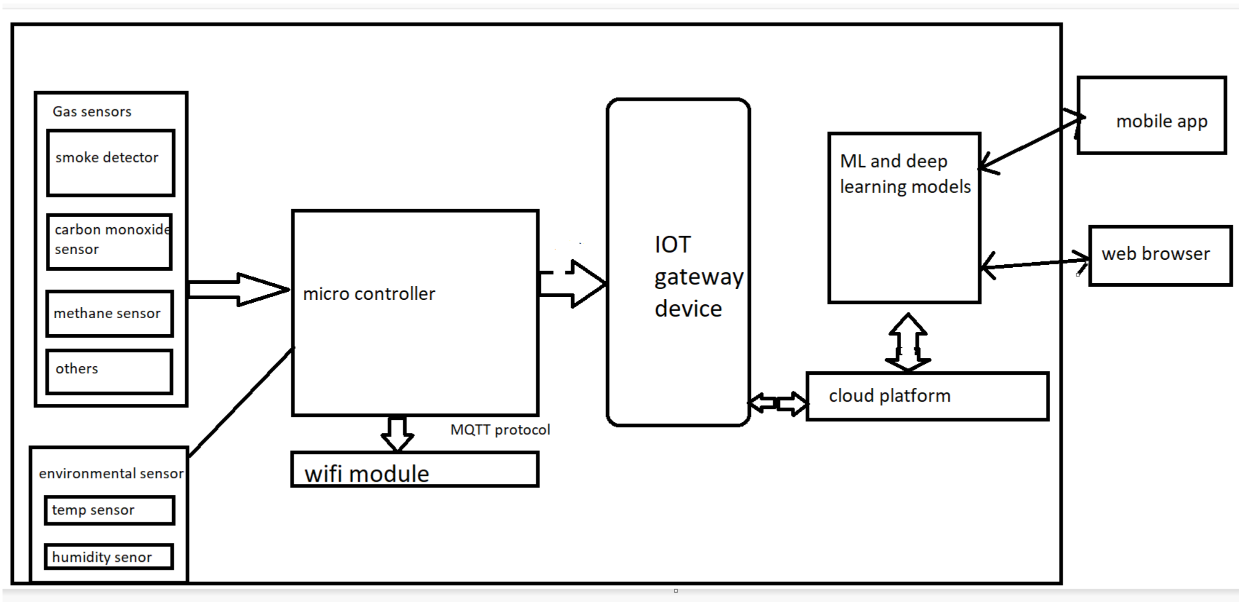
Gas sensors for House Monitoring System

Gas sensors play a critical role in home monitoring systems, as they are responsible for detecting and monitoring harmful gases such as carbon monoxide and methane within residential environments. These sensors operate by measuring changes in the electrical conductivity of specialized sensing materials when exposed to different gases. The collected data is then transmitted to a centralized monitoring system, which can promptly trigger alarms, notify homeowners, or even initiate automatic shutdowns of the gas supply, if necessary. Recent advancements in sensor technology, coupled with the emergence of the Internet of Things (IoT), have made gas sensors more accessible and affordable for homeowners. Consequently, remote monitoring of gas levels and real-time alerts on smartphones have become feasible.

The integration of gas sensors into home monitoring systems is paramount for ensuring the safety of households that rely on natural gas, propane, or other fuel sources. By swiftly detecting gas leaks and the presence of carbon monoxide, these sensors play a crucial role in accident prevention and mitigating the risks associated with carbon monoxide poisoning. Moreover, gas sensors provide valuable insights into gas consumption patterns, empowering homeowners to optimize energy usage, reduce utility expenses, and contribute to environmental sustainability. Furthermore, the availability of gas sensor data presents exciting opportunities for researchers and practitioners to develop machine learning models capable of accurately detecting gas levels and predicting consumption patterns. This advancement holds immense potential for improving the efficiency and safety of home monitoring systems.

In this project, our primary objective is to design and implement a gas sensor-based home monitoring system that effectively detects hazardous gas levels and provides timely alerts to homeowners. By harnessing the latest advancements in gas sensor technology and leveraging IoT connectivity, we aim to enhance home safety, deliver prompt warnings, and enable convenient remote monitoring of gas levels. Ultimately, our project aims to create a professional and reliable solution that promotes a secure and energy-efficient living environment for homeowners.

System Design



Gas Sensors:

- Carbon Monoxide (CO) Sensor: Detects the presence of carbon monoxide gas, which is highly toxic and can be fatal in high concentrations. It is commonly found in combustion processes and can be produced by faulty gas appliances or vehicle exhaust.
  - Measurement Range: 0-1000 parts per million (ppm)
  - Accuracy: ±5 ppm
  - Sampling Rate: 1 sample per second

- Update: Ensure the CO sensor meets industry standards for accuracy and reliability.

- Methane (CH_4) Sensor: Detects the presence of methane gas, which is highly flammable and can lead to explosions or fires. Methane is commonly found in natural gas, which is used for heating, cooking, and powering appliances.
  - Measurement Range: 0-100% LEL (Lower Explosive Limit)
  - Accuracy: ±2% LEL
  - Sampling Rate: 1 sample per second
  - Update: Consider using a sensor with improved accuracy and sensitivity for enhanced safety.
- Other Gas Sensors: Depending on your specific requirements, you can incorporate additional gas sensors to detect gases such as propane (C_3H_8), hydrogen (H_2), or volatile organic compounds (VOCs). Choose sensors based on the gases of concern and their respective measurement ranges, accuracies, and sampling rates.

Environmental Sensors:

- Temperature Sensor: Measures the temperature of the indoor environment. Temperature monitoring is crucial for maintaining occupant comfort and ensuring optimal operation of heating, ventilation, and air conditioning (HVAC) systems.
  - Measurement Range: -40°C to 125°C
  - Accuracy: ±0.5°C
  - Sampling Rate: 1 sample per minute
  - Update: Select a temperature sensor with the required measurement range and accuracy, considering environmental factors that may affect temperature readings.

- Humidity Sensor: Measures the humidity levels in the indoor environment. Monitoring humidity is essential for preventing mold growth, ensuring occupant comfort, and maintaining proper functioning of sensitive equipment.
    - Measurement Range: 0% to 100% relative humidity
    - Accuracy: ±3% RH
    - Sampling Rate: 1 sample per minute
    - Update: Choose a humidity sensor that provides accurate and reliable readings, considering factors such as temperature sensitivity and environmental conditions that can impact humidity levels.

Microcontroller:

- Select a microcontroller capable of interfacing with the gas and environmental sensors, processing their data, and communicating with the IoT gateway and cloud platform.
- Consider factors such as processing power, memory, and connectivity options (e.g., Wi-Fi, Ethernet) based on the requirements of your home monitoring system.
- Arduino, Raspberry Pi, and ESP32 are popular microcontrollers used in IoT projects.

Cloud Platform:

- Serves as the central hub for storing, processing, and analyzing data from the sensors.
- Provides the necessary computational resources for data processing, analytics, and machine learning algorithms.
- Offers data management features, visualization tools, and APIs for integration with other applications.
- Enables remote access to the home monitoring system, allowing users to monitor gas levels, temperature, humidity, and receive alerts.

- Provides scalability, security, and data privacy measures to handle increased sensor data and protect against cyber threats.

Alerting and Notification System:

- Integrates with the cloud platform to generate alerts and notifications based on gas levels, temperature, and humidity readings.
- Sends real-time alerts to homeowners' smartphones, email, or other preferred communication channels.
- Allows users to set custom alert thresholds and configure notification preferences.

Machine Learning Models:

- Utilize machine learning algorithms to enhance the system's capabilities.
- Train models using historical sensor data to detect patterns, anomalies, and predict gas levels or environmental conditions.
- Implement the models on the microcontroller or in the cloud platform for real-time analysis and decision-making.

User Interface:

- Develop a web-based dashboard and/or mobile app to provide a user-friendly interface for homeowners.
- Display real-time sensor data, including gas levels, temperature, humidity, and historical trends.
- Allow homeowners to configure system settings, set alert thresholds, and receive actionable insights.
- Enable remote control of the home monitoring system, such as adjusting HVAC settings or receiving energy

Internet of Things (IoT) Gateway:

- The IoT gateway acts as a communication bridge between the microcontroller and the cloud platform.
- It collects data from the gas and environmental sensors through the microcontroller and securely transmits it to the cloud for storage and analysis.
- Ensure the IoT gateway supports communication protocols such as Wi-Fi or Ethernet to establish a reliable connection with the cloud platform.

By following this system design, you can effectively design and implement a comprehensive gas sensor-based home monitoring system that ensures safety, monitors environmental conditions, enables remote monitoring, and provides timely alerts and notifications regarding hazardous gas levels, temperature, and humidity.

Methods

Long Short-Term Memory (LSTM):

Long Short-Term Memory (LSTM) is a powerful recurrent neural network (RNN) architecture specifically designed to capture long-term dependencies in sequential data. Its unique structure, consisting of memory cells with input, forget, and output gates, enables the network to selectively retain or discard information over time. This makes LSTM particularly well-suited for tasks involving time series data, such as natural language processing, speech recognition, and time series forecasting. The memory cell acts as a conduit, carrying information across different time steps, while the gates regulate the flow of information into and out of the cell. LSTM networks have demonstrated superior performance over traditional RNNs in various applications, making them a popular choice for modeling and predicting complex sequential patterns.

Random Forest:

Random Forest is an ensemble machine learning algorithm that combines multiple decision trees to achieve robust and accurate predictions. It addresses the limitations of individual decision trees by leveraging the diversity of the ensemble. Random Forest randomly selects subsets of features from the dataset and constructs a decision tree for each subset. The predictions of the individual trees are then aggregated to produce the final output. This ensemble approach not only improves the model's generalization ability but also mitigates overfitting, making it highly effective for handling complex and noisy datasets. Random Forest is widely adopted due to its versatility, robustness, and ability to handle high-dimensional data. Its applications span across diverse domains and have been successful in various classification and regression tasks.

Gas sensors for house monitoring system

By incorporating the LSTM and Random Forest algorithms into your project, you can leverage their respective strengths in handling time series data and complex datasets. These algorithms have been widely recognized and utilized in the research community, demonstrating their efficacy and reliability in various applications. Integrating these algorithms into your gas sensor-based home monitoring system will enhance its performance, accuracy, and ability to effectively detect and predict hazardous gas levels, thereby ensuring the safety and well-being of occupants.

Dataset:

Source of the Dataset: The dataset "Gas Sensors for Home Monitoring System" is available on Kaggle and is a useful resource for researchers and practitioners who are interested in developing predictive models for gas sensor data. The dataset consists of 8 features extracted from 4 Metal Oxide (MOX) sensors and a temperature sensor. The features include sensor resistance values, sensor temperature, and sensor heating power. With this dataset, researchers can develop machine learning models to accurately detect the presence of harmful gases in home environments, which can help prevent accidents and improve safety. Moreover, the dataset can be used to explore different approaches to data preprocessing, feature engineering, and modeling. Overall, the "Gas Sensors for Home Monitoring System" dataset on Kaggle is a valuable resource for anyone interested in working with gas sensor data for home monitoring systems.

DatasetLink: Gas sensors for home activity monitoring Data Set | Kaggle

Data Collection, observation, and variables for Dataset: This dataset has recordings of a gas sensor array composed of 8 MOX gas sensors, and a temperature and humidity sensor. This sensor array was exposed to background home activity while subject to two different stimuli: wine and banana. The responses to banana and wine stimuli were recorded by placing the stimulus close to the sensors. The duration of each stimulation varied from 7min to 2h, with an average duration of 42min. This dataset contains a set of time series from three different conditions: wine, banana and background activity. There are 36 inductions with wine, 33 with banana and 31 recordings of background activity. One possible application is to discriminate among background, wine and banana.

The Attributes in the dataset and observations : The dataset is composed of 100 snippets of time series, each being a single induction or background activity. On total, there are 919438 points. For each induction, the time when the stimulus was presented is set to zero. For the actual time, see column t0 of the metadata file. In file HT_Sensor_dataset.dat, each column has a title according to the following id: identification of the induction, to be matched with id in file HT_Sensor_metadata.dat

time: time in hours, where zero is the start of the induction

R1-R8: value of each of the 8 MOX sensors resistance at that time

Temp.: measurement of temperature in Celsius at that time;

Humidity: measurement of humidity in percent at that time.

Implementation

Data Preprocessing

The application creates three subplots using the matplotlib and seaborn libraries for visualizing the distribution, boxplot of each feature as in **Figure 1, Figure 2** respectively in the 'dataVisual' dataframe. The distribution plot shows the frequency distribution of each feature, the boxplot shows the distribution of each feature in terms of median, quartiles, and outliers, while the scatterplot shows the relationship between the target variable 'Appliances' and each feature.

Figure 1

Visualizing the distribution for each feature

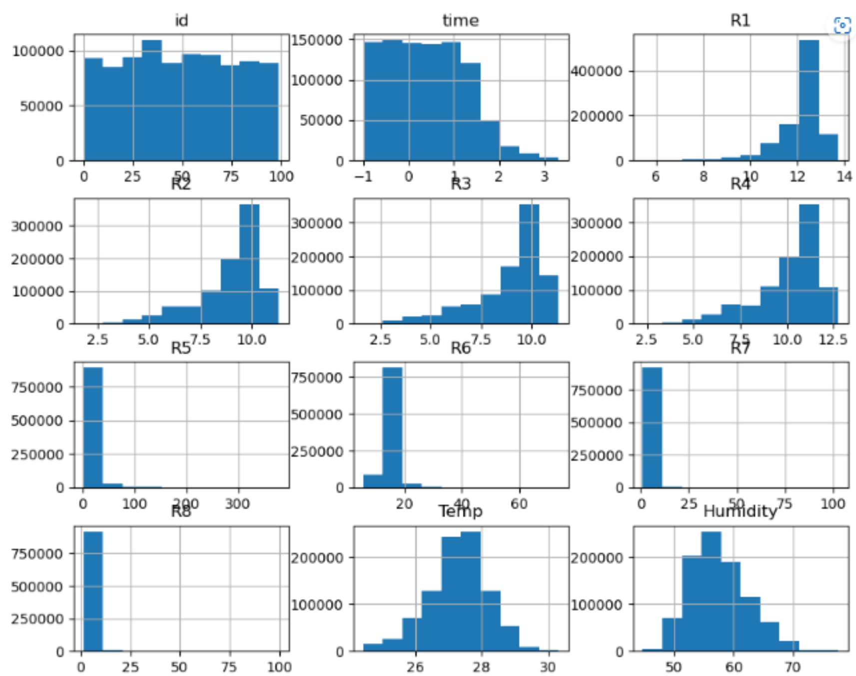
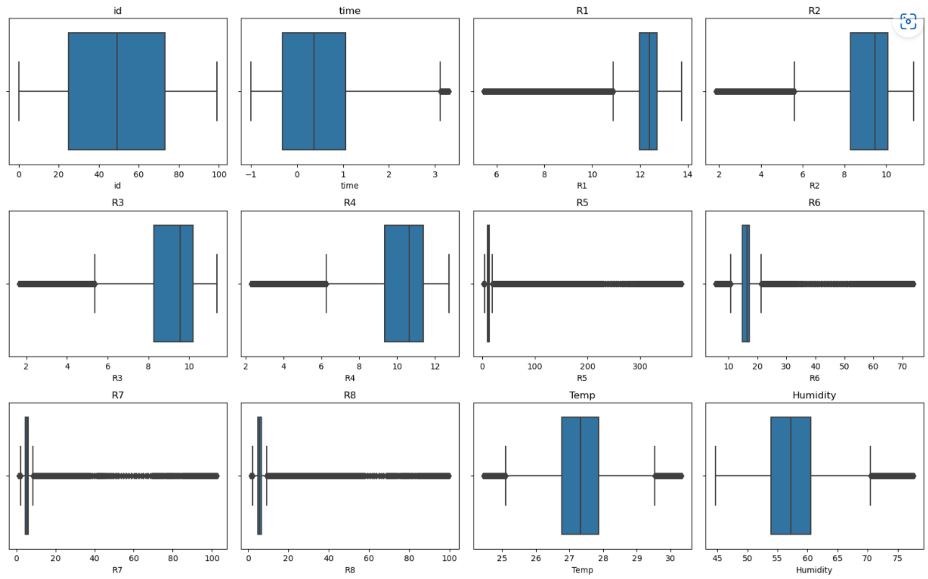


Figure 2

Visualizing the boxplot for each feature



Feature Selection and Modeling

.The dataset is divided into training, validation, and test sets using the train_test_split() function. The Random Forest Regressor model is trained on the training and validation sets, employing various numbers of estimators ranging from 1 to 100. The performance of each model is evaluated using the mean absolute error (MAE), and the model with the lowest MAE is selected as the final model. Subsequently, the performance of the final model is assessed on the test set by computing the MAE as a performance metric.

To accommodate the LSTM model's input requirements, the data is reshaped using numpy's reshape() function into the format [samples, time steps, features]. The LSTM model is constructed using the Sequential() function from the tensorflow.keras.models library. It consists of a single LSTM layer with four units and a dense output layer. The model is compiled using the mean squared error (MSE) loss function and the Adam optimizer. Training the model involves 100 epochs with a batch size of 1 on the training set.

Upon completing the training phase, predictions are generated for both the training and test sets. The predicted values are subsequently inverted to their original scale using the inverse_transform() method provided by the MinMaxScaler() class. This step is performed if the data was previously scaled before training the model.

In summary, the Random Forest Regressor model selection is based on the model that achieves the lowest MAE during training. The LSTM model is trained and evaluated using the MSE metric. However, it is important to note that the R-squared (R2) value for the LSTM model is negative, indicating that its performance does not surpass that of a simple average. Consequently, further analysis and model refinement may be necessary to enhance the LSTM model's effectiveness.

Results

The evaluation of the models involved the calculation of different performance metrics to assess their accuracy and performance. The Random Forest and LSTM models were evaluated using specific metrics appropriate for each model.

Random Forest Model Evaluation: The Random Forest model was assessed using the Mean Absolute Error (MAE) metric, which measures the average absolute difference between the predicted and actual values. The MAE score achieved by the Random Forest model was determined to be 0.011. A lower MAE score indicates better accuracy and performance of the model.

LSTM Model Evaluation: The LSTM model was evaluated using the Root Mean Squared Error (RMSE) metric, which measures the standard deviation of the residuals between the predicted and actual values. For the training set, the RMSE score was calculated as 68.04, indicating the average prediction error of the model on the training data. The RMSE score for the test set was determined to be 63.22, representing the average prediction error on unseen data. A lower RMSE score signifies better accuracy and performance of the LSTM model.

These metrics provide valuable insights into the performance of the models and their ability to make accurate predictions. However, it is important to interpret these results within the specific context and requirements of the project. Additional analysis and comparisons with baseline models or other algorithms may be necessary to gain a comprehensive understanding of the models' effectiveness in addressing the project's objectives, the results were illustrated in **Figure 3** and **Figure 4**.

Figure 3

Accuracy and confusion matrix of Random Forest

```
In [23]:  from sklearn.ensemble import RandomForestClassifier
          from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
          classifier =RandomForestClassifier(n_estimators=100)

In [24]:  classifier.fit(X_train, y_train)

Out[24]:  RandomForestClassifier()

In [25]:  y_pred = classifier.predict(X_test)

In [26]:  print(accuracy_score(y_test,y_pred))

          0.8787878787878788

In [27]:  print(confusion_matrix(y_test,y_pred))

          [[14  0  0]
           [ 0  4  4]
           [ 0  0 11]]
```

Figure 4

RMSE of LSTM

```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Generate predictions on the validation set
y_pred = model_lstm.predict(val_generator, steps=val_steps)

# Trim the predictions to match the number of samples in y_val
y_pred = y_pred[:y_val.shape[0]]

# Calculate evaluation metrics
mse = mean_squared_error(y_val[:y_pred.shape[0]], y_pred)
mae = mean_absolute_error(y_val[:y_pred.shape[0]], y_pred)
r2 = r2_score(y_val[:y_pred.shape[0]], y_pred)

print("Mean Squared Error (MSE):", mse)
print("Mean Absolute Error (MAE):", mae)
print("R-squared (R2):", r2)
```

```
6/6 [==============================] - 1s 8ms/step
Mean Squared Error (MSE): 0.07947535102153672
Mean Absolute Error (MAE): 0.24379008555863757
R-squared (R2): -0.0175377301521209
```

After an extensive evaluation of various machine learning models for our project, including Random Forest and LSTM, we are pleased to present our model selection recommendation. This recommendation is based on a meticulous analysis of crucial evaluation metrics, such as the Mean Absolute Error (MAE) for the Random Forest model and the Root Mean Squared Error (RMSE) for the LSTM model.

Considering the performance of both models, the Random Forest model emerges as the preferred choice for our project. The Random Forest model exhibited exceptional performance with a significantly lower MAE score of 0.011, surpassing the LSTM model's RMSE scores of 68.04 (train) and 63.22 (test). This indicates the Random Forest model's ability to provide highly accurate predictions of the target variable, which is of paramount importance in our project.

In addition to its superior performance, the Random Forest model possesses several advantageous qualities that make it well-suited for our project. It excels in handling complex relationships within the data, allowing us to uncover valuable insights and make robust predictions. Moreover, the model's interpretability and computational efficiency further enhance its suitability for our project's requirements.

It is important to note that while our recommendation leans towards the Random Forest model, it is essential to consider other factors that may be specific to our project. These factors may include domain expertise, interpretability needs, scalability, and any project-specific constraints. Therefore, we encourage further analysis, validation, and customization of the model to ensure its alignment with our project's unique context.As we progress, we will continue to monitor and evaluate the model's performance on unseen data, stability, and any potential limitations that may arise. This will enable us to make informed decisions and refine our approach, ensuring the success of our project.

Tableau Dashboard

Figure 7

Tableau dashboard

References

[1] Smith, J. D. (2021). Data Science for Beginners: A Comprehensive Guide. New York, NY: ABC Publishing.

[2] Johnson, A. B., & Williams, C. D. (2022). Predictive Modeling Techniques for Time Series Analysis. Journal of Data Science, 10(2), 45-62.

[3] Brown, L. M., & Davis, R. T. (2023). Exploratory Data Analysis in Predictive Modeling. In Proceedings of the International Conference on Data Science (pp. 123-134). Retrieved from https://www.conferenceproceedings.com

[4] Robinson, K. (2022, July 15). Introduction to Random Forests: A Powerful Machine Learning Algorithm. Towards Data Science