Demonstration of networked gas sensor data-driven calibration

Elin Berglund, Laura Briffa, Yu Qin and Yuqi Zheng
For courses EQ2443 / EQ2444 / EQ2445

***Abstract*—Non-dispersive infrared gas sensing is a popular technology for CO_2 level monitoring. Using machine learning algorithms to replace humans for auto-calibrating the gas sensor has been an important topic recently. Previously, an algorithm based on Hidden Markov Model and Dempster–Shafer theory has been proposed. In this paper, the algorithm is implemented and verified on our own dataset. The result is that even though the methods worked as expected, the success of the proposed algorithm is sensitive to the non-fixed and human-involved aspects of the process.**

Supervisors: Cheng Yang, Tobias Oechtering

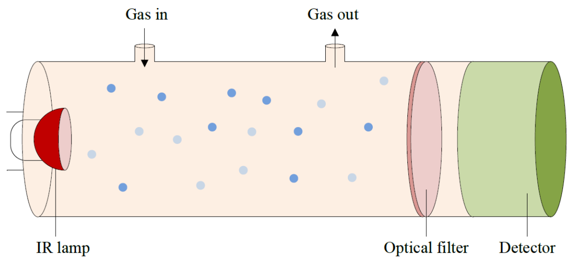


Fig. 1: NDIR sensor [1].

I. Background

The CO_2 concentration is an important measurement in many areas, for example, in health and safety applications. Due to large areas having to be measured and sensors having to be placed close together, there is a need for low-cost CO_2 sensors. One of the most common ways to measure CO_2 concentration is to use non-dispersive infrared (NDIR) sensors. The working principle is illustrated in Fig. 1. When infrared light irradiates the tube, some of the light will be absorbed by the gas molecules. The amount of light that has been absorbed by the gas is then used to calculate the gas concentration. The problem is that these sensors often are sensitive to aging and environmental factors which leads to poor accuracy. One way to mitigate this effect is by calibration [1].

One established calibration method is the Automatic Baseline Correction (ABC) which uses the baseline value, where the CO_2 concentration is assumed to be the same as in fresh air. This method works well when the sensors get regular exposure to fresh air but not in places, for example, in large cities where the gas concentration is always high [1]. In recent research, the use of machine learning algorithms for self-calibration of sensors has been explored. In [1], a Hidden Markov Model (HMM) based calibration method is investigated. The paper develops both an unsupervised and supervised learning model for temperature compensation of the baseline value. The result showed an improvement in accuracy using the data-driven calibration for both the supervised and unsupervised approaches.

Another important aspect of sensor calibration is sensor fusion. This is the process of combining measurements from multiple sensors to achieve a more accurate result than could be attained using only one sensor. One theory that is widely applied is the Dempster-Shafer (DS) theory which is based on belief and plausibility. The theory, also known as belief function theory, combines multiple uncertain results to come to a conclusion [2]. In [2], a self-calibration scheme based on belief function theory is proposed for NDIR gas sensors. The paper implements two methods for sensor fusion, a general belief function fusion, and a weighted average fusion that can handle situations where the sensor belief function has conflicting values. The result shows that the general approach gives a good result when the belief functions are consistent, but that the weighted average approach performs better when the values are conflicting.

In this project, a data-driven calibration scheme for gas sensors is implemented combining the results from [1] and [2]. Both the supervised and unsupervised HMMs are applied to individual sensors for self-calibration. The results from these sensors are then combined using both the general belief function sensor fusion and the weighted average sensor fusion. One important aspect of performing sensor measurement is the spatial setup of the sensors. For this purpose, measurements are performed in different spatial setups to investigate how different distances between sensors affect the result.

II. Theory

A. NDIR sensor technology

The NDIR sensor relies on the ability of gas molecules to absorb light of specific wavelengths. According to the Beer-Lambert law, this ability can be utilized to calculate concentration and absorbance. As seen in Fig. 1, the NDIR sensor works by letting the gas from the surrounding environment into a gas chamber. This chamber is then illuminated by infrared light and a filter on the opposite side of the light source filters out the wavelength corresponding to the target gas. A detector then measures the light intensity of the specific wavelength to find out the attenuation caused by the target gas [3]. By using the IR signal (IR), the temperature (T), and a baseline value called the zero coefficient ($zero$), the amount of absorbed light (Abs) can be calculated by a mapping according to,

$$Abs = f(zero, IR, T). \tag{1}$$

The gas concentration, in this case, the CO_2 concentration, can then be calculated by mapping the difference between (Abs) and the reference value for zero gas concentration (R) according to,

$$CO_2 = L(R - Abs). \tag{2}$$

The zero coefficient can be used to compensate for drifting in measured concentration, as it sets the baseline for when the concentration of CO_2 is zero. When the sensors get older and are exposed to temperature change or other environmental factors, the IR signal will change in value. In order to adjust the zero coefficient after this change, a stochastic model of the coefficient has to be built. Similarly to [1], this project will focus on compensation in temperature.

B. Hidden Markov Model

In this project, we use the Hidden Markov Model (HMM) to calibrate the true zero coefficients due to the varying concentration level of CO_2 over time. The HMM is effective at characterizing a coherent process, making it well-suited for this task. An HMM can be used to represent information that is not directly observable. It can be seen as two stochastic models: one that is hidden and one that is observable, which is produced by the hidden one.

An HMM can be defined by a parameter set $\lambda = \{\pi, A, B\}$, where the initial probability distribution $\pi = \{\pi_j : \forall j \in [1 : N]\}$ and the transition probability matrix $A = \{a_{ij} : \forall i, j \in [1 : N]\}$ define a Markov Chain (MC), i.e., the character of the state sequence, and the emission probability matrix $B = \{b_j : \forall j \in [1 : N]\}$ describes the distribution of the outputs of the model. Let N represent the number of the hidden states, S_t represent the state at time instant t, and X_t represent the observation at t. The elements of the parameter can then be defined as follows:

$$\pi_j = P[S_1 = j]. \tag{3}$$

$$a_{ij} = P[S_{t+1} = j | S_t = i]. \tag{4}$$

$$b_j(X_t = x) = f_{X_t|S_t}(X_t = x | S_t = j). \tag{5}$$

With the true zero coefficients being seen as hidden states, and the CO_2 values as observable states that are produced by the true zero coefficient, an HMM can be created. Furthermore, the temperature difference can be seen as the hidden state together with true zero coefficients [4]. To learn the HMM, we utilize both supervised and unsupervised learning methods. Once the HMM, i.e., the parameter $\lambda = \{\pi, A, B\}$ of the model is learned, given the observation sequence, we could obtain the predicted, i.e., the calibrated true zero coefficients through the Viterbi Algorithm, which will be explained in detail in the following section.

Assuming the collected data is fluctuating within certain range, with an appropriate initialization for training process, we expected the HMM can model the sequential data correctly and show great fitting ability.

C. Dempster-Shafer theory

The Dempster-Shafer(DS) theory is a theoretical framework for modeling uncertainty and combining the probabilities from different observations. It is a modified framework of the formal probability theory, and not self-consistent, which is a weakness that sometimes lead to contradictory results. For further reading, we refer to [5].

D. Belief Function Fusion via Wasserstein Distance based Weighted Average

The Wasserstein Distance is defined as

$$W_2(P_I, P_J) = \sqrt{\min_{P_{IJ}: \sum_j P_{IJ} = P_I, \sum_i P_{IJ} = P_J} \sum_{i \in \mathcal{I}, j \in \mathcal{J}} |i - j|^2 P_{IJ}(i, j)}. \tag{6}$$

This metric measures the distance between two different probability distributions even with different sample spaces. With the definition in 6, the belief function from $sensor_i$ and $sensor_j$ can be calculated with

$$\hat{W}_2(P_I, P_J) = \frac{2 \times W_2(P_I, P_J)}{\sum_i \sum_j W_2(P_I, P_J)}. \tag{7}$$

Having this, we can further define the similarity between P_I and P_J as

$$S(P_I, P_J) = 1 - \hat{W}_2(P_I, P_J). \tag{8}$$

Moreover, we define the support degree of a single probability distribution as $Supp(P_I) = \sum_{j \neq i} S(P_I, P_J)$. Finally, the weighted average probability is $\hat{P} = \sum_{I=1} \alpha_I P_I$, where $\alpha_I = \frac{Supp(P_I)}{\sum_{I=1} Supp(P_I)}$. Lastly, we can obtain the fused belief function by apply method from [6], adding it 4 times within the framework under DS theory, which gives

$$P(x) = (\hat{P} \oplus \hat{P} \oplus \hat{P} \oplus \hat{P})(x). \tag{9}$$

III. Hardware and measurements

The initial phase of the project involved getting to know the hardware. Once we were able to take measurements with the hardware, we began collecting data.

A. Hardware

Five sensor units are used during the project for data collection and testing. Each sensor unit consists of a low-cost NDIR sensor, a more expensive and reliable NDIR sensor, an Adafruit HUZZAH32 - ESP32 Feather Board, and a battery. The expensive sensor is used as a reference as its measurements will be interpreted as the correct CO_2 values. The microcontroller connects, through WiFi, to an MQTT broker where the sensor measurements are uploaded. Measurements are made periodically, and the units go into deep sleep to save battery between the measurements; meaning that every time the unit wakes up to make a measurement, it has to connect to WiFi and the broker. Since wireless connections are not always reliable, it is expected that a few measurements will be lost since the units will not always be able to make the connection. The measurements are made at minute intervals

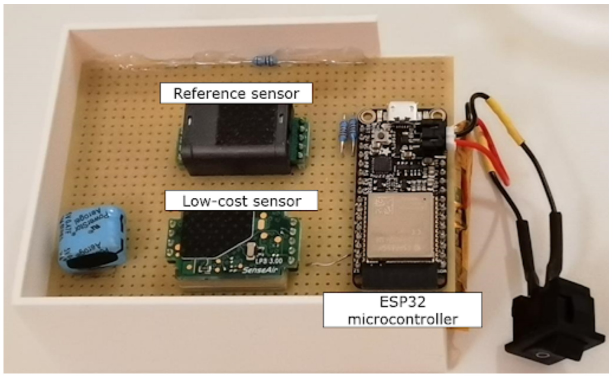


Fig. 2: Units with low-cost sensor and a reference sensor.

and the resulting data is saved to a CSV file on a computer acting as a server.

The measurements made are: date and time, temperature from reference sensor, CO_2 concentration from reference sensor, temperature from low-cost sensor, CO_2 concentration from low-cost sensors, and measured IR signal.

B. Measurements

For data collection, the sensors were placed in an apartment according to Fig. 3. With this allocation scheme, the CO_2 concentration measurements were expected to differ slightly between sensors, while still maintaining similar patterns. It was also expected that the variation of the CO_2 concentration would be good enough to train the HMM on. The variations in concentration would come from the movement of people through the apartment, the opening of doors and windows, ventilation, and other events. The doors between the rooms with sensors in them were maintained open throughout the entire measurement. This was to make sure that the variations in CO_2 concentration stayed similar in the whole apartment. The measurement ran for approximately five days to get enough data for both training and testing of the HMM.

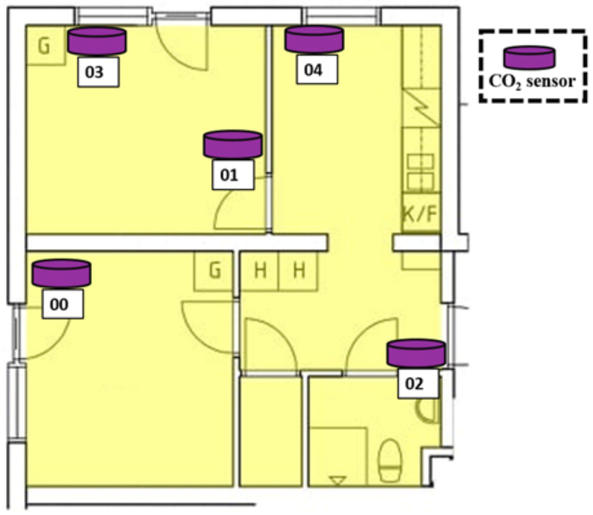
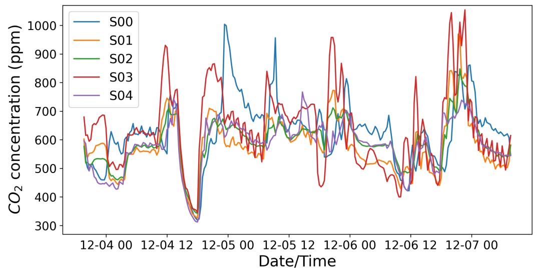


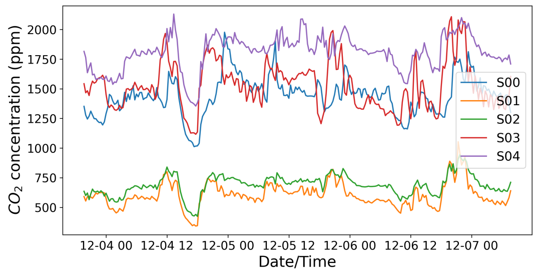
Fig. 3: Placement of sensors for data collection.

The measured concentration by the reference and low-cost sensors from the data collection can be seen in Fig. 4. From the two graphs, it is seen that measurements from the low-cost sensors are noisier and, although they follow a similar pattern, measure different levels of concentration than the reference sensors. Before starting the measurements, an assumption was made that the sensors closest to each other in the set-up would have the most similar measurement result. With that assumption, sensors 01 and 03 would be the most similar (counting the distance between sensors in the number of rooms). In Fig. 5, a comparison is made between the Euclidean distance between measurements from sensor 01 and the rest of the sensors and the Euclidean distance between measurements from sensor 03 and the rest of the sensors. The comparison is made using the reference sensor measurements in Fig. 4. It is seen from the result that the assumption made beforehand was not correct since the measurement from sensor 01 is more similar to the ones made by sensors 04 and 02 than by sensor 03. This shows that the measured concentration is not only affected by the distance between sensors, but also by other environmental factors, for example, the presence of people and ventilation.

To further investigate the impact of different distances between sensors, another measurement was made with the sensors placed along a corridor, see Fig. 6. The corridor is located in an office space where people are present during the day, and the measurement was taken over a period of two days. The Euclidean distance between the CO_2 measurements made by the reference sensors placed at the ends of the corridor and the rest of the reference sensors can be seen in Fig. 7. The result shows, similarly to the above, that there are other factors than the distance that impacts the measurements as the Euclidean distance between the time series does not always



(a) Reference sensor measurements.



(b) Low-cost sensor measurements

Fig. 4: Measured CO_2 concentration in apartment.

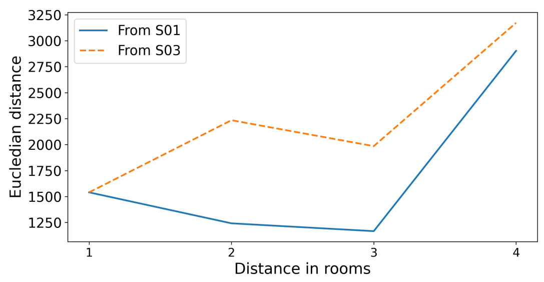


Fig. 5: Euclidean distance between time series in figure 4 (a) in relation to the physical distance between sensors, see figure 3. The blue line is the distance from sensor 1. The orange line is the distance from sensor 3.

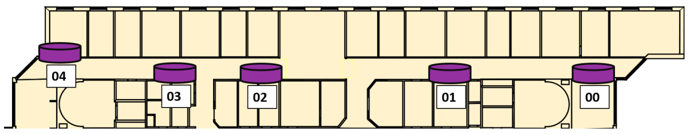


Fig. 6: Placement of sensors along the corridor.

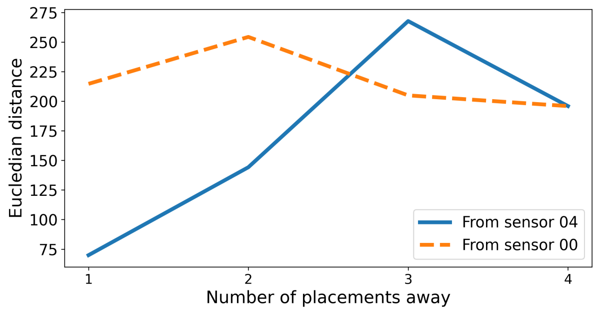


Fig. 7: Euclidean distance between time series from two-day measurement in the corridor in relation to the physical distance between sensors.

increase with the physical distance.

IV. Algorithms

A. General Process

The whole process is illustrated in Fig. 8. Besides, the final results of auto-calibration are outputs generated from ”Viterbi Algorithm” for the adjusted zero parameters and ”Calculate CO_2 with zero” for the data after calibration respectively.

B. Supervised HMM

In supervised learning, we regard the empirical frequency of occurrence as a probability. First, we preprocess the collected data by downsampling and taking the average at appropriate intervals, and removing irrational data using the 3σ rule. Then, we calculate the reference true zero coefficients $zeros$ using CO_2 measured by the reference sensor. Next, we quantize CO_2 measured by the low-cost sensor and the calculated $zeros$ with and without the temperature difference ΔT measured by the reference sensor. Thus, we obtain the hidden states $\{zero_t\}$ or $\{(zero_t, \Delta T_t)\}$, and the observation sequence $\{CO_{2t}\}$. Then, we calculate the frequency of each state as the initial state probability, the frequency of transition from one state to another as the transition probability, and the frequency of each observation given each state as the emission probability, i.e., π, A, and B, respectively. In this way, the HMM is learned in a supervised way.

C. Unsupervised HMM

For unsupervised learning, we utilize the Baum-Welch Algorithm to train the model, i.e., to optimize the HMM iteratively in the Maximum Likelihood sense. The pseudo code is as follows:

Initialize: Select appropriate $\lambda^0 = \{\pi, A, B\}$.
Iterate: For $n = 1, 2, ...,$ update λ^n based on λ^{n-1}, such that

$$\log P[x|\lambda^n] > \log P[x|\lambda^{n-1}], \tag{10}$$

where π, A, and B are updated separately.
Terminate: Repeat iterating until either a fixed number of iterations is reached, or the improvement is smaller that the threshold, i.e.,

$$\log P[x|\lambda^n] - \log P[x|\lambda^{n-1}] < \Delta_{min}. \tag{11}$$

In both supervised and unsupervised learning, the quantization scheme needs to be devised and tuned carefully in order to obtain a better result. For supervised learning, we use a higher-resolution quantization for more accurate predictions, while in unsupervised learning, we use a lower-resolution quantization for a faster convergence. The initialization of the model is especially important for unsupervised learning. We use both probabilities obtained from the supervised learning and the uniform distributions as the initial probabilities. There is not much difference between the results of the two initialization schemes.

D. Viterbi Algorithm

Given HMM $\lambda = \{\pi, A, B\}$ and the observation sequence, we could find the most probable state sequence using Viterbi Algorithm. The pseudo code is as follows:

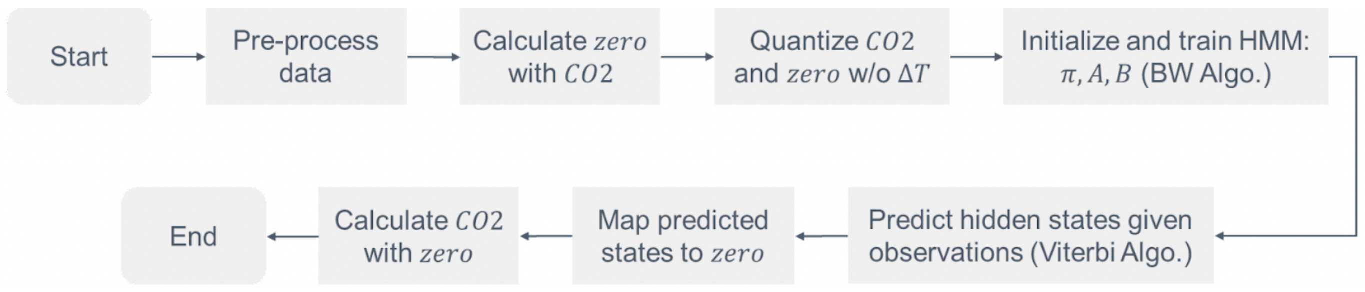

Start
Pre-process data
Calculate $zero$ with $CO2$
Quantize $CO2$ and $zero$ w/o ΔT
Initialize and train HMM: π, A, B (BW Algo.)
Predict hidden states given observations (Viterbi Algo.)
Map predicted states to $zero$
Calculate $CO2$ with $zero$
End

Fig. 8: General Process

Initialize: At $t = 1$,

$$\chi_{j,1} = P[S_1 = j, X_1] = q_j b_j(X_1). \tag{12}$$

Forward: For $t = 2, ..., T-1$,

$$\chi_{j,t} = b_j(X_t) \max_i \chi_{i,t-1} a_{ij}, \tag{13}$$

$$\zeta_{j,t} = \arg\max_i \chi_{i,t-1} a_{ij}. \tag{14}$$

Terminate: At the final step, $t = T$,

$$\chi_{j,T} = b_j(X_T) \max_i \chi_{i,T-1} a_{ij}, \tag{15}$$

$$\zeta_{j,T} = \arg\max_i \chi_{i,T-1} a_{ij}. \tag{16}$$

Backtrack: At $t = T$,

$$\hat{i}_T = \arg\max_i \chi_{i,T}. \tag{17}$$

At $t = T-1, T-2, ..., 1$,

$$\hat{i}_t = \zeta_{\hat{i}_{t+1}, t+1}. \tag{18}$$

Once we learn the HMM, given the CO_2 measured by the low-cost sensor, we can quantize the CO_2 and predict the most probable $zeros$. Then, we can calculate the calibrated CO_2 with the predicted $zeros$.

E. Sensor fusion

Prerequisites for the sensor fusion implementation are the models and a subset of the processed data from the low-cost sensor measurements. The measured CO_2 values of the different sensors are quantized with the same quantization scheme used for training, see Sections IV-B and IV-C. These quantized CO_2 values denote the HMM observations $\mathbf{x}^i = (x_1^i, ..., x_t^i)$, for the ith HMM. The temperature and IR measurements are also needed at a later stage. To construct the belief functions $P_i(x)$, given the CO_2 observations $\mathbf{x}^i$ and the HMMs, two different approaches were implemented.

In one of the approaches, the Forward algorithm is utilized to determine the belief functions. Here, the probability $P[S_t = j | x_1, \ldots, x_t, \lambda]$ is determined in the algorithm procedure, where S_t denotes the state at a time t. As previously established, the states correspond to the true zero coefficients. However, these states can be mapped into CO_2 values using Equations 1 and 2 if also given the IR and temperature measurements.

In the second approach for determining the belief functions, the Viterbi algorithm is used. Using the most probable state sequence from the output, the belief functions can be based on the transition probabilities $P[S_{t+1} = k | S_t = j]$ between each state, which can be acquired with the state transition matrices. As for the case of the Forward algorithm approach, these belief functions give the probabilities for the zero coefficients but can be mapped into CO_2 values.

Since the HMMs can have a different amount of states and correspond to different zero coefficients, the belief functions of the different sensors are mismatched. In other words, state 0 in one HMM is not the same zero coefficient as state 0 in another HMM, and therefore, the jth element in one belief function is not mapped to the same CO_2 value as the jth element in another belief function. Before any sensor fusion algorithm can be applied to these belief functions, they need to be expanded in order for the elements to correspond to the same CO_2. At first, it was accomplished by using linear interpolation, but as this caused the sum of all probabilities to be > 1 (and scaling when normalized), it was decided to fill the missing elements with zeros instead.

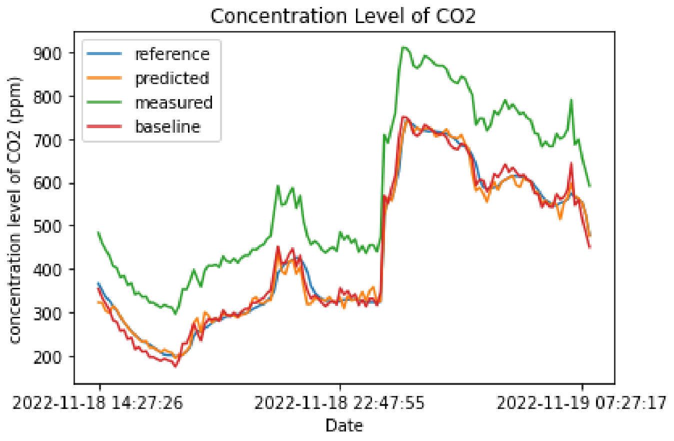
The Wasserstein distance-based weighted average belief fusion, as described in Equations 6 - 9, is implemented in a straightforward fashion with NumPy. A scipy function was used to calculate the Wasserstein distance. After belief fusion, $argmax(P(x))$ is taken to obtain the sensor fusion result at each time instance.

V. Results

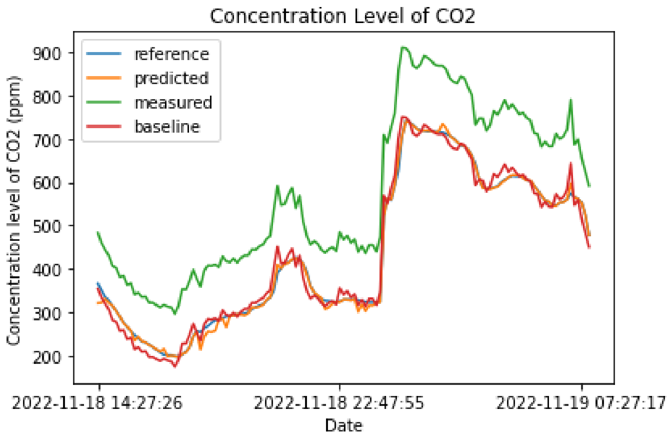
A. Decoded zero coefficient using Viterbi

In this project, we have four kinds of HMMs: $zero$ or $(zero, \Delta T)$ state with supervised or unsupervised learning. After the training process, the HMM can decode the corresponding true zero coefficient for each sensor. We take the average of the reference true zero coefficients as the baseline.

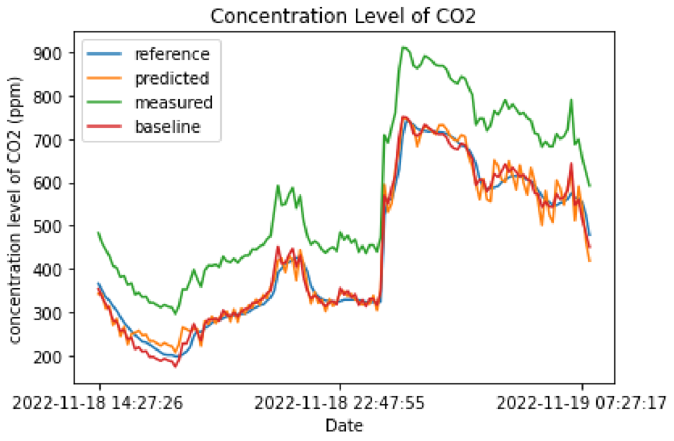
Take the supervised learning model with state $(zero, \Delta T)$ as an example. We take the data collected from 2022-12-01 18:35:11 to 2022-12-04 03:56:40 as the training data. The CO_2 and $zeros$ calculated from the reference CO_2 are shown in Fig. 10.



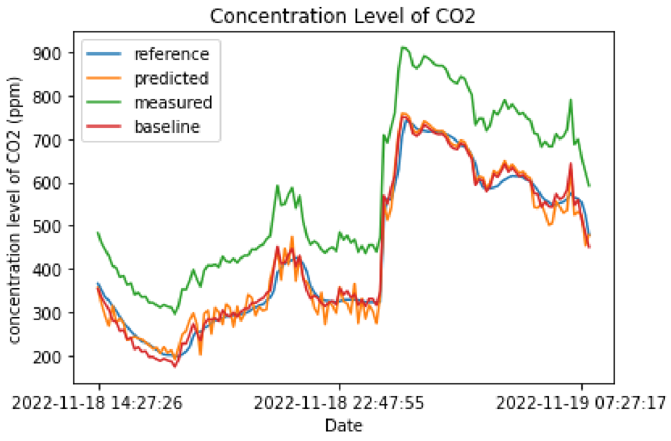
(a) $zero$, supervised



(b) $(zero, \Delta T)$, supervised



(c) $zero$, unsupervised



(d) $(zero, \Delta T)$, unsupervised

Fig. 9: Example of fused belief results at a time instance

After pre-processing the data, we quantize the $zero$, ΔT, and CO_2 according to their distributions, respectively, which is shown in Table I.

We use the CO_2 measured by the low-cost sensor from 2022-12-04 07:24:52 to 2022-12-04 21:19:26 to test the learned HMM. The reference and predicted hidden states, the corresponding $zeros$ and CO_2 are shown in Fig. 11, respectively.

We can see that the learned HMM can successfully predict the hidden states and calibrate the measured CO_2 close to the reference value, even though there is a great error between the reference and the measured data. The root-mean-square error (RMSE) can be diminished to 21.211 from 789.620, which is also smaller than the baseline RMSE of 27.136.

The results of the four models for one particular sensor is shown in Fig. 9. The RMSEs between the reference CO_2 and the baseline CO_2, the measured CO_2, and the calibrated CO_2 of the four models are shown in table II, respectively.

TABLE I: Data information after pre-processing and quantization scheme.

	min	max	q_{min}	q_{max}	stepsize
zero	13333	13541	13360	13480	8
ΔT	-0.126	0.920	-0.0672	0.068	0.0035
CO_2	795	1800	992	1770	22

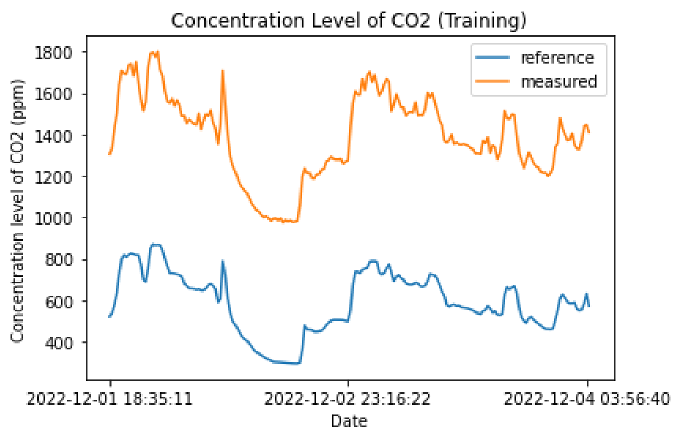
We can see that there is an obvious difference between the raw CO_2 measured by the low-cost sensor, with an RMSE up to 137.818. After the calibration of the HMM, the error is significantly reduced. Among the four models, the supervised learning model with state $(zero, \Delta T)$ has the best performance, with an RMSE of 9.238, which is only 6.70% of the RMSE between the reference and measured CO_2.

The preliminary calibrated results for five low-cost sensors are shown in Fig. 12.

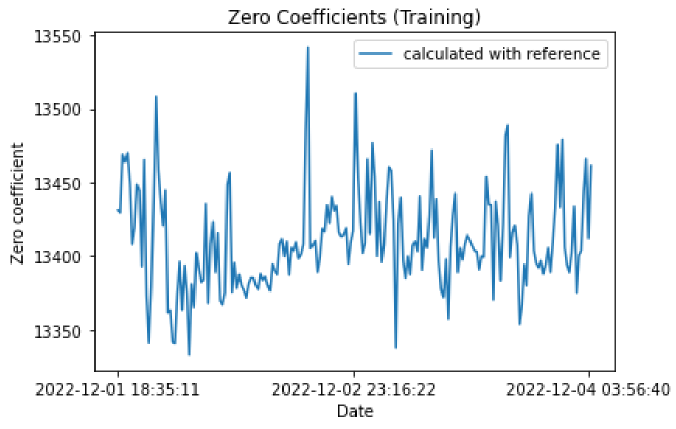
We can see that although the calibrated CO_2 of each sensor is close to the reference CO_2, there are still some minor differences between the results of each sensor. To get a better result, we introduce sensor fusion.

TABLE II: RMSEs between the reference and other concentration levels of CO_2.

$zero$, supervised	17.574
$(zero, \Delta T)$, supervised	9.238
$zero$, unsupervised	26.974
$(zero, \Delta T)$, unsupervised	27.093
baseline	21.556
measured	137.818



(a) Concentration level of CO_2 (Training).



(b) Zero coefficients (Training).

Fig. 10: Training data.

B. Calibrated Data after sensor fusion

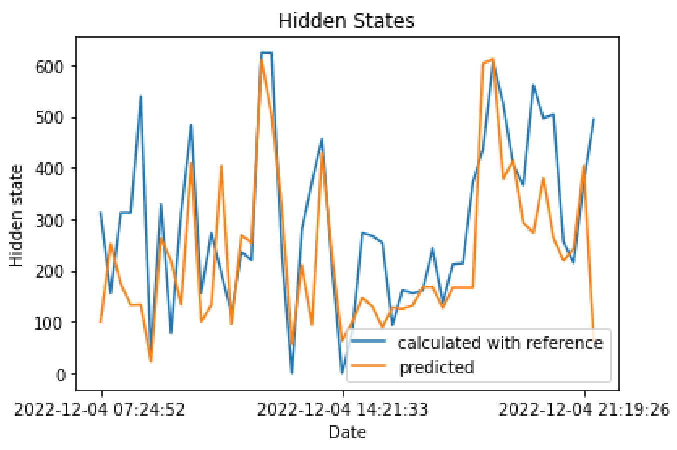
The final result of the calibration is shown in Fig. 14 and Fig. 15. The difference is not noticeable. Thus, we can say the operation of getting the probability from the HMM is not key to our result. The quality of the HMM model itself is crucial.

Meanwhile, we can notice the sensor fusion algorithm here is not better than the mean of the separated predictions. Thus, we can claim that the sensor fusion algorithm is not working as we expected. The reason is the HMM model needs much more data for training to get a non-sparse transition matrix.

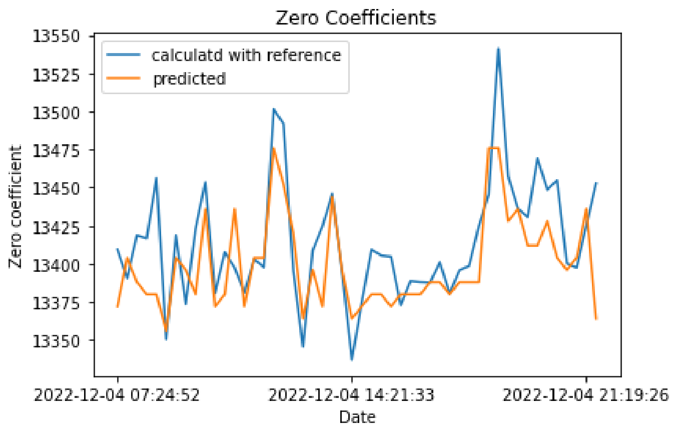
Further, an example of the fused belief results at a time instance is shown in Fig. 13. In this particular example, the conflicting belief function is chosen as truth. In these cases, the higher probability outweighs the distance. This may also have an impact on the results seen in Fig. 14 and Fig. 15. Whilst the HMM is a relevant factor here as well, it might also be relevant to study the effect of other distance metrics.

VI. Conclusion

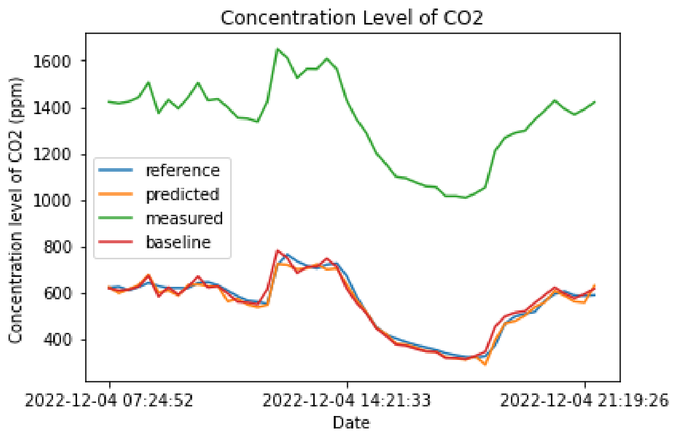
In this project, a combination of methods for data-driven sensor calibration has been studied for NDIR gas sensors. First, data was collected in a test environment using both reference sensors and the sensor to be calibrated. Second,



(a) Hidden states.



(b) Zero coefficients.



(c) Concentration level of CO_2.

Fig. 11: Prediction results of supervised learning model with state $(zero, \Delta T)$.

HMMs were trained on the data and used to calibrate the measurements. Finally, a sensor fusion algorithm was developed to combine the multiple sensor measurements.

The separate results of the HMM and the sensor fusion showed that we successfully demonstrated the networked gas sensor data-driven calibration methods proposed in [1] and [2] by You and Oechtering. The combined results showed

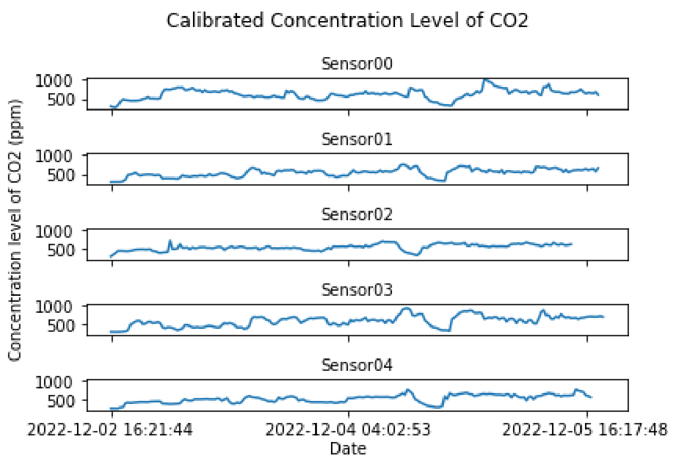


Fig. 12: Calibrated concentration levels of CO_2.

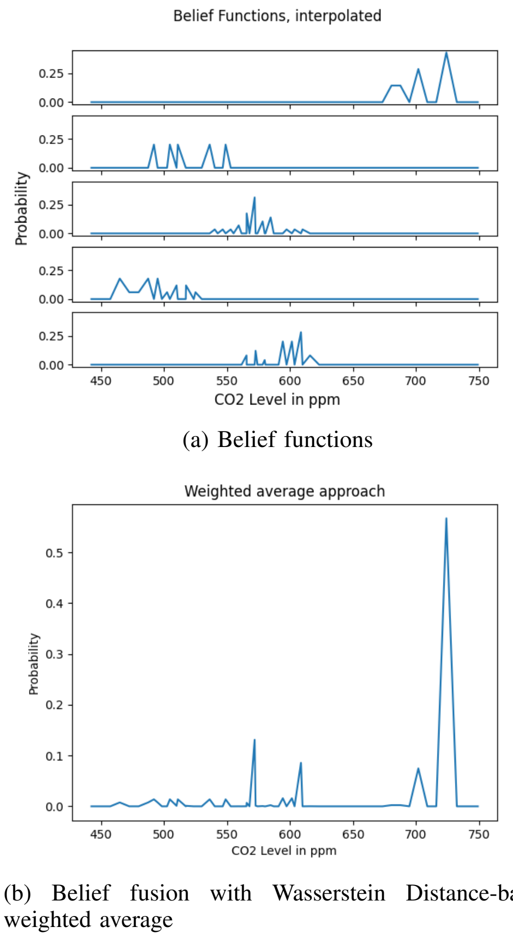


Fig. 13: Example of fused belief results at a time instance

that each part of the process is sensitive to the results of the previous part, which makes the whole process sensitive to errors. The training of the HMM depends on the collected data, and the sensor fusion is highly dependent on the transition probabilities created by the HMM.

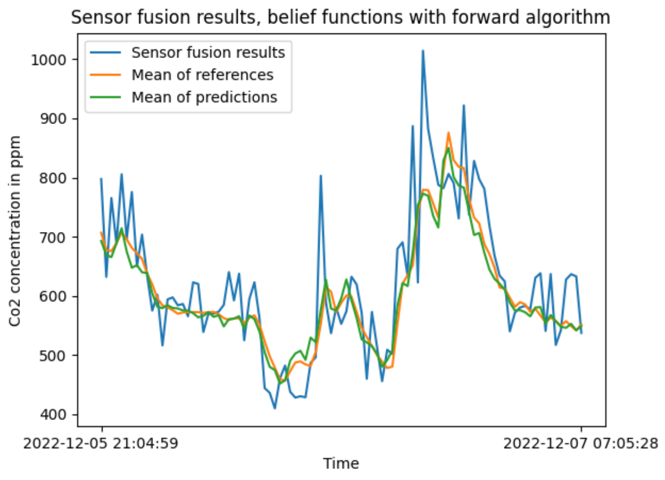


Fig. 14: Sensor fusion forward results

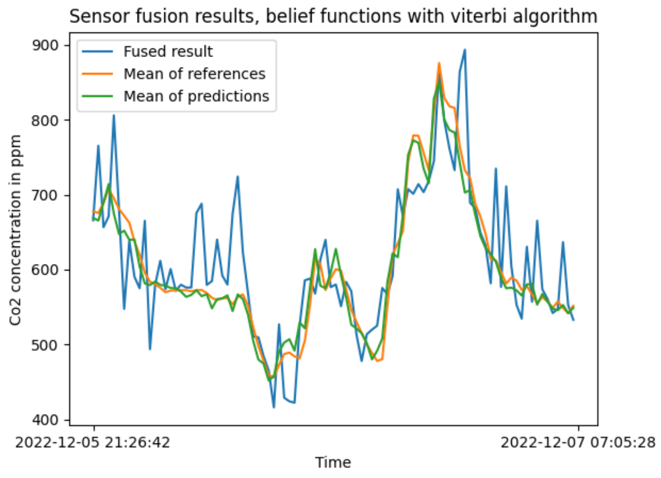


Fig. 15: Sensor fusion viterbi results

VII. Reflection

A. Reflection toward the Model

The HMM created in this project was found to be too strict for use in all situations, indicating that it would need to be trained on more data to be applied to a real application. However, it was shown that the variations in CO_2 concentration indoors depend on many different environmental factors. This implies that it would be time-consuming to collect a sufficient amount of data to train the HMM for every possible scenario.

Another thing that was observed is that the indoor temperature does not vary significantly. Since the HMM is trained to handle temperature fluctuations, it may be more suitable for use in an outdoor environment where the temperature fluctuates more. In an outdoor setting, it could also be the case that the CO_2 concentration is less affected by environmental factors as the air flows more freely than indoors. Hence testing the HMM and sensor fusion on outdoor data could give a better result.

Using a different initialization for the HMM could also give a better result as the current initialization needs to be manually adjusted by amounts of times. Most of the performance of the model relies on the data collected: only when the patterns of the training data and testing data are similar, the model would yield an accurate prediction. However, this cannot be guaranteed in practice in a short period of time, like several days. Unpredictable variations of the environment also lead to the quantization problem, especially when there is a sudden rise or drop of the concentration level of CO_2, since the quantization results directly affect the probabilities used to learn the HMM and irrational probabilities can cause a lot of trouble, not only for prediction, but also for sensor fusion. Besides, synchronization is also an important and tricky issue for sensor fusion.

B. Reflection toward the Project Course

At the last, we have several suggestions for the upcoming students to have a better experience throughout the project.

1) Have consistent meetings (with and without project owner).
2) Pay attention to the time plan throughout the entire project.
3) Do a thorough risk analysis and plan for changes.
4) Difference between theory and real-world application.

Acknowledgment

The authors would like to thank You Yang for his kind supervision and Gustav from the Senseair Company for the help on the sensor chips.

References

[1] Y. You and T. J. Oechtering, "Hidden markov model based data-driven calibration of non-dispersive infrared gas sensor," in *2020 28th European Signal Processing Conference (EUSIPCO)*, 2021, pp. 1717–1721.

[2] Y. You, A. Xu, and T. J. Oechtering, "Belief function fusion based self-calibration for non-dispersive infrared gas sensor," in *2020 IEEE SENSORS*, 2020, pp. 1–4.

[3] J. Wong, "NDIR Gas Sensors," Aug 1995, US Patent 5,444,249. [Online]. Available: https://patents.google.com/patent/EP1695066A1/en

[4] L. E. Sucar, *Hidden Markov Models*. London: Springer London, 2015, pp. 63–82. [Online]. Available: https://doi.org/10.1007/978-1-4471-6699-3_5

[5] A. P. Dempster, "Upper and Lower Probabilities Induced by a Multivalued Mapping," *The Annals of Mathematical Statistics*, vol. 38, no. 2, pp. 325 – 339, 1967. [Online]. Available: https://doi.org/10.1214/aoms/1177698950

[6] C. K. Murphy, "Combining belief functions when evidence conflicts," *Decision Support Systems*, vol. 29, no. 1, pp. 1–9, 2000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167923699000846