Received: 30 August 2024 | Revised: 30 September 2025 | Accepted: 14 January 2026 | Published online: 17 March 2026

RESEARCH ARTICLE

Artificial Intelligence and Applications
2026, Vol. 4(2) 187–196
DOI: 10.47852/bonviewAIA62024212

BON VIEW PUBLISHING

Toward a Causal $PM_{2.5}$ Concentration Forecasting by Inferencing from Local Vehicle Tracking with a Low-Cost End-to-End Sensor System

Chuong Dinh Le[1,*], Hoang Viet Pham[1], Thinh Gia Tran[2], An Dinh Le[3], Anh-Duy Pham[4], Dat Thanh Vo[5], Hien Bich Vo[2] and Huy-Dung Han[6]

[1]*Institute of Computer Science, Universität Bonn, Germany*

[2]*Electrical and Computer Engineering Department, Vietnamese-German University, Vietnam*

[3]*Department of Electrical and Computer Engineering, University of California, United States*

[4]*Joint Lab of Artificial Intelligence & Data Science, Osnabrück University, Germany*

[5]*Department of Mechanical, Automotive and Materials Engineering, University of Windsor, Canada*

[6]*School of Electrical and Electronic Engineering, Hanoi University of Science and Technology, Vietnam*

Abstract: This paper introduces a pioneering, end-to-end system designed for the accurate estimation of $PM_{2.5}$ concentrations, which uniquely integrates custom-designed hardware with advanced computational algorithms. A central innovation of this work is the development of a seamless data pipeline that connects real-time vehicle tracking with sophisticated air quality prediction, offering a novel and comprehensive solution for urban environmental monitoring. The system employs a cost-effective, custom-built sensor package, including PMS7003 and BME280 sensors alongside a 5MP camera, with a bespoke hardware design specifically engineered to enhance operational stability. For the estimation component, a rigorous comparative analysis was conducted, demonstrating that the Cubist regression model significantly outperforms other contemporary machine learning and traditional mathematical models; this success is attributed to its superior ability to model the complex, nonlinear relationship between observed traffic density and ambient $PM_{2.5}$ levels. Furthermore, a streamlined vehicle counting algorithm, leveraging a fine-tuned YOLOv7 model, ensures robust and accurate traffic detection performance across various lighting and environmental conditions. This research successfully establishes an optimal $PM_{2.5}$ estimation pipeline based on real-world vehicle counts, presenting an integrated framework for dynamic urban air quality analysis.

Keywords: $PM_{2.5}$, IoT, object detection, forecasting, computer vision

1. Introduction

Vietnam, like many developing countries, is grappling with severe air pollution issues. A research in 2023 [1] stated that the annual average $PM_{2.5}$ concentration in Ho Chi Minh City, Vietnam, exceeded WHO guidelines. This was due to the large number of vehicles plying on the roads [2], especially motorcycles [3], resulting from rural-urban migration and a high rate of population growth [4, 5]. This directly increased the amounts of pollutants emitted into the atmosphere, particularly $PM_{2.5}$—fine particles with diameters less than 2.5 micrometers. Short- and long-term exposure to this particle increases the risk of severe diseases, such as cardiovascular and respiratory diseases, lung cancer, and stroke [6].

Addressing this critical issue, the present study introduces a novel, integrated framework that seamlessly combines hardware development and algorithmic design to estimate $PM_{2.5}$ levels using traffic density. This holistic approach not only improves the precision of $PM_{2.5}$ estimation but also highlights the potential of traffic density as a reliable indicator of urban air quality. With an estimated 1 billion such cameras globally [7], the potential for widespread deployment is significant. Our contributions are as follows:

1) A working hardware system for measuring $PM_{2.5}$ and counting vehicles
2) A fine-tuning pipeline for the YOLO vehicle detection model so that it is robust to lighting conditions
3) A fast and effective vehicle counting algorithm
4) Various machine models for estimating $PM_{2.5}$ from vehicle density

2. Literature Review

Numerous studies have investigated vehicle counting using cameras [8–10]. However, their research did not examine the relationship between traffic density and air pollution. Furthermore, their system relied on conventional surveillance cameras, while our system integrates the camera with a Raspberry Pi minicomputer to enhance Internet of Things (IoT) capabilities.

*Corresponding author: Chuong Dinh Le, Institute of Computer Science, Universität Bonn, Germany. Email: s56dle@uni-bonn.de

In 2018, a study from Bangkok, Thailand [11], suggested the causal effect of traffic flow on the concentration of $PM_{2.5}$ by linear regression and path analysis. Moreover, it is also shown that, by analyzing the path of traffic flow, the result would be more accurate than using linear regression at low and high PM concentration levels.

A number of studies regarding the impact of vehicle emissions on the $PM_{2.5}$ level [12, 13] have been conducted, which emphasized the idea that traffic emissions contribute greatly to the $PM_{2.5}$ level in cities. The studies also suggested how these emissions are affected by total distance traveled by vehicles, the vehicle technology, and the volume of traffic.

A more extensive project is the WeCount initiative [14] in Europe, which measured traffic density in major cities using computer vision and correlated this data with air pollution levels. While our research shares similarities in using a Raspberry Pi and a camera, our system introduces specific hardware and algorithmic improvements. Our hardware design, for example, features an enclosure optimized with computational fluid dynamics (CFD) simulations for thermal management and airflow to improve sensor reading stability and protect components, enhancing its anti-interference ability. Algorithmically, our approach advances beyond simple correlation by implementing a latency lag compensation method in our mathematical model to address the delay between emission and detection and employing machine learning to directly predict $PM_{2.5}$ levels from traffic. In contrast, the WeCount project focused primarily on widespread data collection through citizen science, inviting volunteers to host devices to gather traffic data.

In recent studies, various approaches have been explored to enhance the accuracy of low-cost air quality monitoring systems. For example, a sheaf-theoretic self-filtering network [15] was proposed that integrates similar sensors to ours with a causal model based on vehicle counts. Their method applies sheaf theory to self-correct sensor data in real time, without external calibration, and scales across multiple sensor nodes. While our approach leverages machine learning to model the relationship between traffic density and $PM_{2.5}$ levels, the sheaf-theoretic framework offers an alternative method for handling inconsistencies in sensor data, especially when working with low-cost sensors. Both approaches share the goal of developing cost-effective solutions for urban air quality monitoring, though the methodologies differ significantly in terms of data integration and correction.

While previous studies have explored various aspects of $PM_{2.5}$ estimation or traffic monitoring, a notable gap exists in the development of integrated systems that combine hardware and algorithmic innovations. This research addresses this gap by creating a comprehensive system that uniquely integrates these two components to estimate $PM_{2.5}$ concentrations based on traffic density, demonstrating a novel approach in environmental monitoring.

3. Method

The methodology used in this study is structured around a comprehensive, integrated system designed for estimating $PM_{2.5}$ concentrations from traffic density. This system uniquely combines custom-developed hardware with sophisticated algorithms, marking a novel approach in environmental monitoring. An overview of the data flow is displayed in Figure 1. The hardware, which does the data acquisition, transfers the images of the road to the vehicle object detection model, which outputs the number of four-wheeled and two-wheeled vehicles at each timestamp, which are then turned into $PM_{2.5}$ concentrations. The $PM_{2.5}$ concentration sampled from the hardware goes through a cleaning process. It is used as the ground truth for both of the $PM_{2.5}$ estimation approaches, namely, $PM_{2.5}$ estimation using the mathematical method and $PM_{2.5}$ estimation using machine learning. The following subsections extensively discuss firstly how the hardware is configured for the data acquisition task, secondly how the vehicle object detection is fine-tuned, and thirdly the inner workings of the two $PM_{2.5}$ estimation approaches.

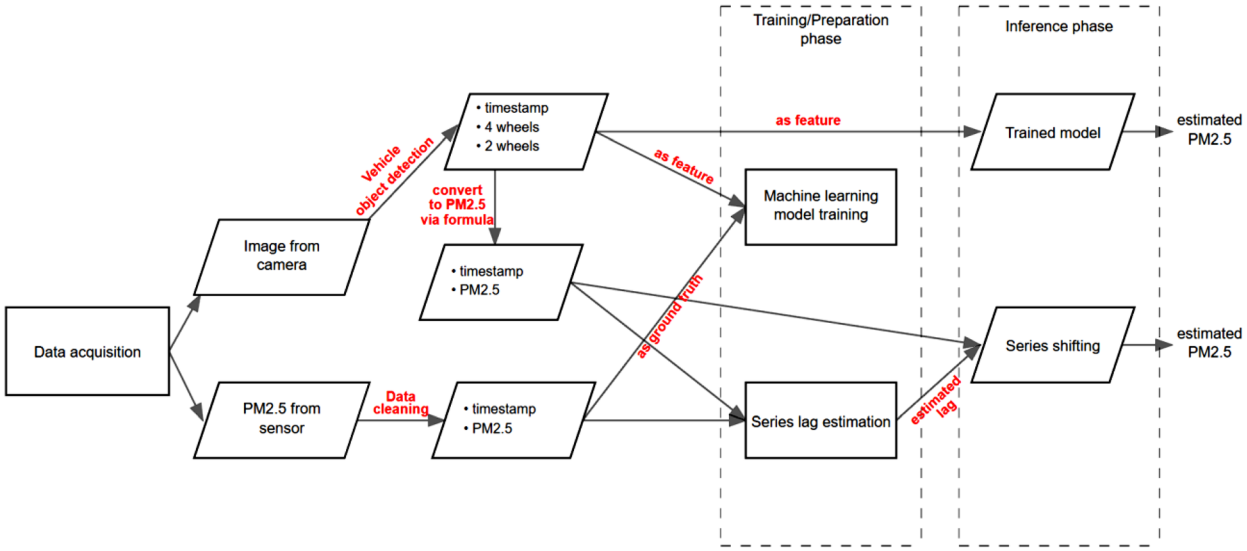
3.1. Hardware setup for data acquisition

This section provides a comprehensive analysis of the device's design, detailing its external and internal architecture. It addresses the rationale behind its configuration, its functional capabilities, and its operational mechanisms.

3.1.1. Design

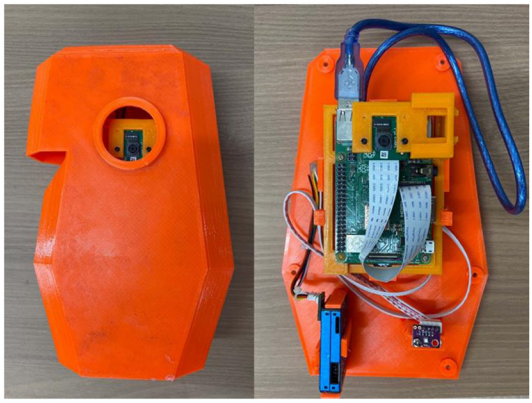
The device is developed in a low-cost fashion to collect video data of the traffic, together with meteorological measures such as

Figure 1
Data flow of the system



temperature, humidity, atmospheric pressure, and particulate matter (PM). This design, which is shown in Figure 2, is a refined version of a previous design proposed in our earlier work [16] and is deployed to roadside houses for data collection.

Figure 2
IoT box



Note: The IoT box, including a camera mounted on the Raspberry Pi, a temperature, humidity, atmospheric pressure sensor at the bottom right, and a particulate matter sensor located at the bottom left are connected to a microcontroller that lies under the Pi. The blue cable is used to connect the Pi to the microcontroller.

The enclosure is designed for durability and optimized airflow, manufactured with ultraviolet-resistant polyethylene terephthalate glycol plastic. To ensure reliable sensor readings, the system's thermal performance was validated using Autodesk CFD airflow simulations. The simulations confirmed that the design effectively dissipates heat from the Raspberry Pi's CPU, maintaining the sensor area at ambient temperature (27 °C) under no-wind conditions, thus preventing heat from the electronics from affecting environmental measurements. Even with a 20 km/h wind, the system maintains thermal stability, ensuring data reliability.

To ensure reliable sensor readings, the system's thermal performance was validated using Autodesk CFD simulations for both no-wind and 20 km/h wind conditions, as can be seen in Figure 3. These simulations modeled the CPU with 5W of heat generation and an active inlet fan with a 471.947 cm^3/s flow rate [17]. The results confirmed that the design effectively dissipates heat, critically maintaining the sensor area at the ambient temperature of 27 °C, which ensures component heat does not affect the environmental measurements.

3.1.2. Camera and sensor setup

The system is designed for deployment on roadside buildings to collect video data on traffic and measure PM concentration. Deployment sites must meet the following criteria: they should be at least 5 m above ground level to avoid vehicle turbulence, offer a clear view of the road and vehicles, have access to power and a strong Wi-Fi connection, and be located outdoors while protected from rain.

3.1.3. Electrical system and data storage

The electrical system features a Raspberry Pi 3B with a camera, which connects via USB to a custom-developed Environmental Monitoring Module (EMM). The Raspberry Pi records video, gathers data from the EMM, and uploads it to Azure Blob Storage [4].

The EMM is controlled by an STM32 microcontroller and integrates a BME280 sensor for temperature, humidity, and atmospheric pressure, alongside a PMS7003 sensor for particulate matter (PM1.0, $PM_{2.5}$, and PM10). This PMS7003 sensor was chosen for its superior accuracy and durability [18]. The calibration of the sensor is done against a reference instrument DustTrak 8534 [4]. For precise timekeeping, the module uses a DS1307 real-time clock. Programmed in C, the EMM sends compiled sensor data to the Pi every 10 seconds for logging. The entire system operates on a standard 5V DC USB power supply, with a measured energy consumption of 147 mAh over 4.5 hours. Figure 4 displays an overview of the system.

3.2. Vehicle counting pipeline

Figure 5 illustrates the complete workflow of the vehicle counting system. The process begins with the input of surveillance video frames, which are fed into the pipeline one at a time. The first step is image acquisition, where each frame is fetched for processing. Next, the system performs vehicle object detection using a fine-tuned YOLOv7 model. This model identifies vehicles in the frame and outputs a tuple containing the position (centroid coordinates) and the type of each detected vehicle (e.g., motorbike or car).

Once detection is complete, the system proceeds to assign unique IDs to each vehicle centroid, enabling consistent tracking across frames. These IDs are managed in an ID pool, which stores active vehicle IDs and discards them when they are no longer visible or relevant.

The tracking module then checks whether the centroid of a tracked vehicle crosses a predefined counting line. This is a critical decision point in the pipeline. If the centroid crosses the line, the system triggers the update function, which increments the count for the corresponding vehicle type. If the centroid does not cross the line, the vehicle continues to be tracked until it either crosses or exits the frame.

The output of the system is a running count of vehicles, categorized by type (motorbike or car), based on those that have successfully crossed the counting line. The surveillance videos used in this study are divided into two groups: one set is used to fine-tune the YOLOv7 detection model, and the other is used to evaluate the counting performance.

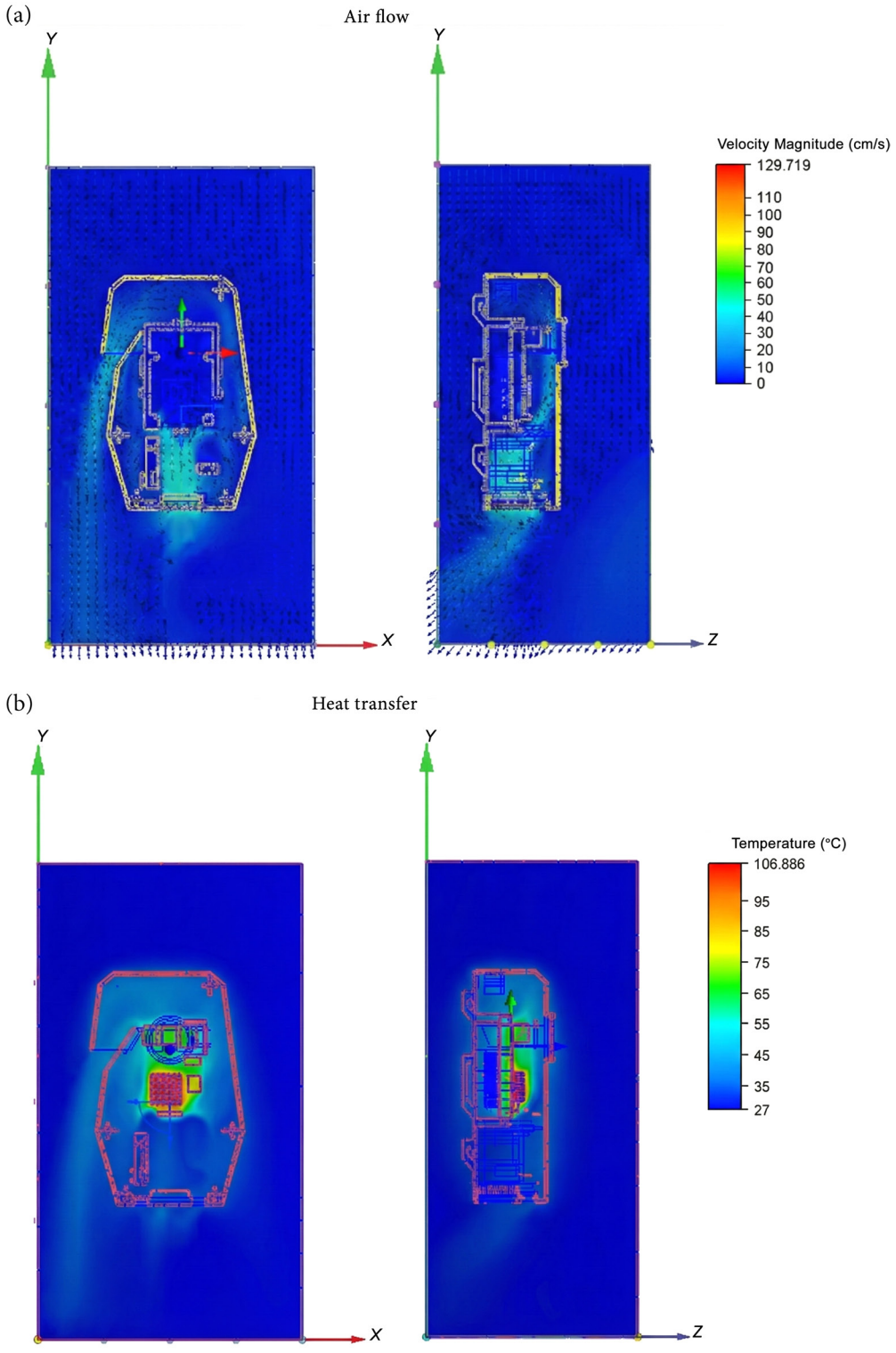
3.2.1. Fine-tuning object detection model

We resampled the frames to keep only frames that are highly useful. The frames were divided into daytime and nighttime groups, and each group went through different processing pipelines. The daytime group processing pipeline is inspired by active learning with a human-in-the-loop approach [19] and is supported by YOLOv7-based detection and classification [20]. The pre-trained model performs vehicle recognition on 10-minute traffic videos and returns a list of all bounding box detections with associated labels, as can be seen in Figure 6(a). For nighttime group, the data preparation process used unsupervised learning techniques to group frames based on visual context. Frame groups satisfying diversity and the minimum number of vehicle conditions were manually filtered. Raw frames were input into EfficientDet [21] model to extract context features, and then K-means clustering [22] was applied to cluster image features from each video into five groups. The output clusters were reviewed and selected for labeling by humans. Figure 6(b) shows this specifically.

3.2.2. Counting algorithm

YOLOv7 is used to detect and classify vehicles. These vehicles are then counted using a line counting method [23], which counts each vehicle whose center passes the counting line. The method is demonstrated in Figure 7. To avoid repeated counting, the Kalman filter [24] and Hungarian algorithm [25] are used to assign and track these centers. After the next position of a detected vehicle is predicted by the Kalman filter, the Hungarian algorithm associates new detections with

Figure 3
CFD simulation result of air flow and heat transfer inside the enclosure



existing tracks using an Intersection over Union (IoU) cost matrix. To handle temporary occlusions where a vehicle might not be detected in a few frames, a track is only de-registered after it has been missing for 30 consecutive frames. Table 1 shows in detail how the algorithm works.

To test the effectiveness of this approach, a simulation is done by randomly spawning multiple vehicles on the road with predefined trajectories, and then a comparison of the counting result with tracking and without tracking against the ground truth is done. The result is discussed in Section 4.

3.3. $PM_{2.5}$ estimation algorithm

After obtaining $PM_{2.5}$ data from the sensor and vehicle counts for each vehicle class, we began by cleaning the $PM_{2.5}$ data and then

Figure 4
IoT pipeline design

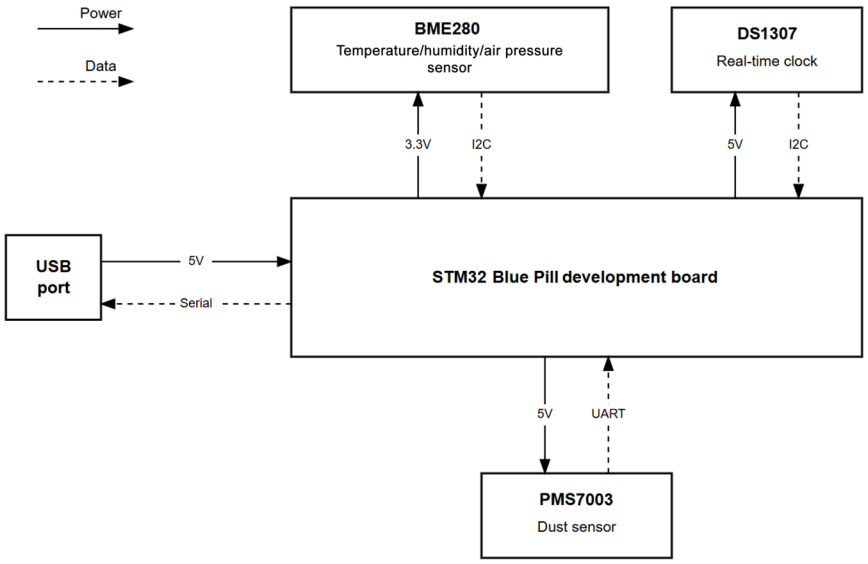
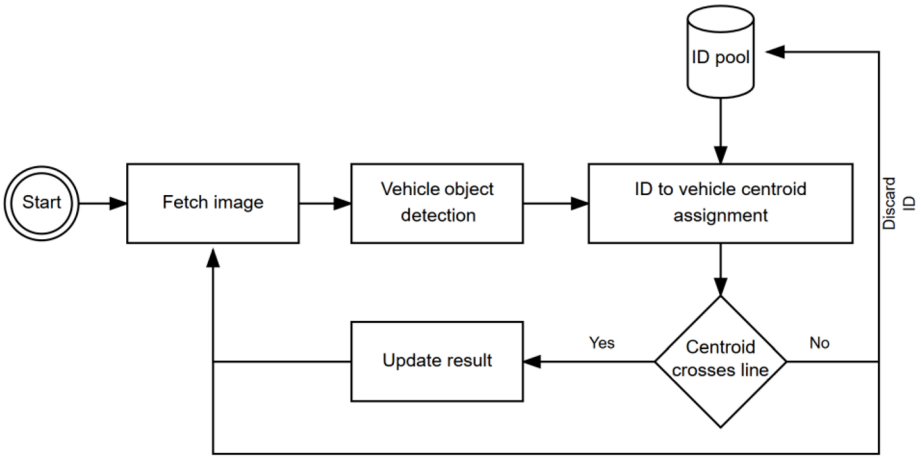


Figure 5
Overview of the vehicle counting pipeline



mapping it to the vehicle count data. We investigated two estimation approaches in order to find the best performance method: mathematical method, which is a linear mapping from vehicle counting data to $PM_{2.5}$ data with latency lag compensation, and machine learning methods.

3.3.1. Cleaning $PM_{2.5}$ data

Outliers can significantly distort visual assessments. Therefore, interquartile range filters are applied to detect and correct outliers. The data is resampled at 1-hour intervals, and the trend is extracted using Seasonal-Trend decomposition using LOESS (STL) [26]. Figure 8 shows the transformation of data after cleaning.

3.3.2. $PM_{2.5}$ estimation using mathematical method

For calculating the amount of $PM_{2.5}$ emitted by a vehicle, we used the following formula [27]:

$$Em = Nm \times EFm \times VKTm,$$

where Em is the mass of emitted $PM_{2.5}$ (g); Nm is the number of vehicle type m; EFm is the emission factor of vehicle type m ($g*km^{-1}$), referenced from the California Air Resources Board Emission Factor Table [28]; and VKTm is the length of the recorded street segment (km).

By dividing Em, given by the result of the above formula, by a volume, the $PM_{2.5}$ concentration can be estimated. To approximate the resulting concentration, a simplified dispersion model was assumed. The total emitted mass (Em) was considered to be distributed within a fixed volume corresponding to the road segment, conceptualized as a cube with a side length equal to VKTm. Although this is a simplification, it provides a baseline for estimation. Therefore,

$$Cm = Em \times 106 / (VKTm \times 1`000)^3,$$

where Cm is $PM_{2.5}$ concentration from the vehicle, measured in $\mu g/m^3$.

3.3.3. Latency lag compensation methods for the mathematical method

This approach is based on the intuition that $PM_{2.5}$ generated by vehicles isn't detected by the sensor instantly, but rather after a latency period. Here, we use cross-correlation to find the lag.

Figure 6
Illustration for the image dataset preparation process

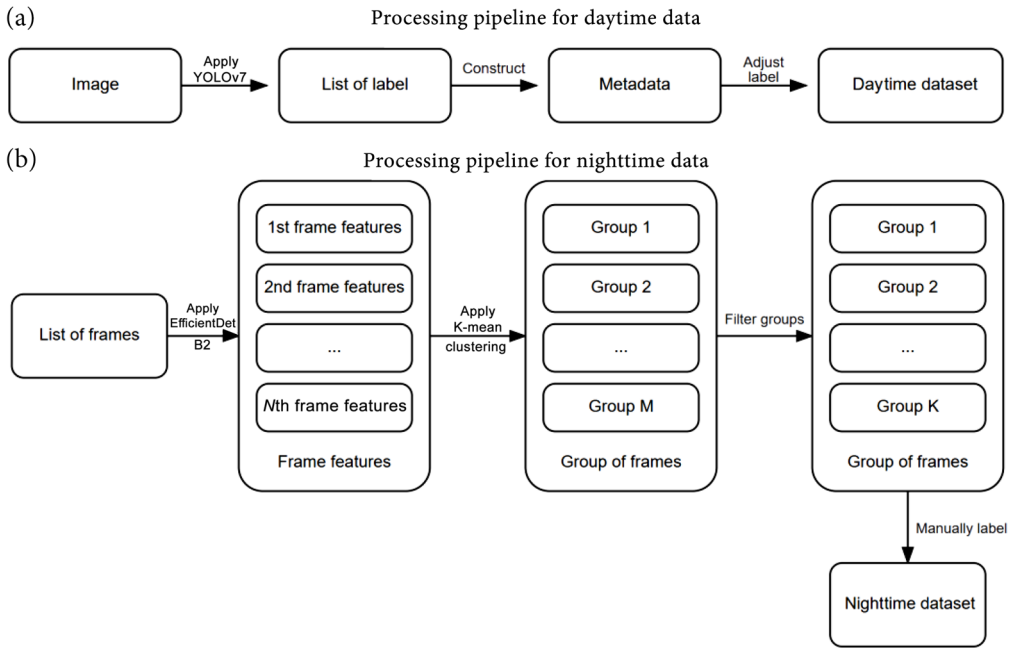
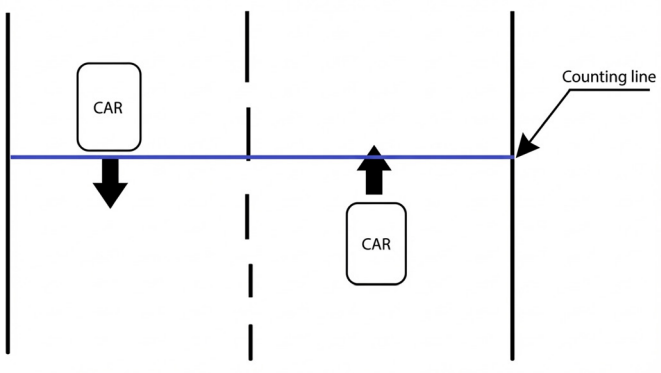


Figure 7
Visualization of the line counting algorithm



Cross-correlation identifies the lag that maximizes the cross-correlation between the two time series. This is achieved by shifting one time series relative to the other and calculating the correlation at each shift. The lag that maximizes this correlation is selected as the final lag, with an upper limit of 50 time units (here, one time unit is 10 seconds).

Once the lag is determined, we shift the $PM_{2.5}$ values inferred from the vehicle count on the previous day by that lag. This shifted data is then subtracted from the actual $PM_{2.5}$ readings obtained from the sensor on that day, giving us an estimate of the $PM_{2.5}$ contribution from other sources. To calculate $PM_{2.5}$ levels on the estimation day, we shift the vehicle count-inferred $PM_{2.5}$ data by the calculated lag and add it to the estimated $PM_{2.5}$ contribution from other sources. Table 2 demonstrates how this method works.

3.3.4. $PM_{2.5}$ estimation using machine learning

The approach discussed above involves several assumptions that may have overlooked various natural factors, such as wind,

Table 1
Pseudocode for vehicle counting algorithm

```
I: current frame
Tracks: a list for keeping track of vehicles
Vehicles = Yolo.detect(I)
Centers = computer_centers(Vehicles)
For vehicle in Vehicles:
    If Tracks is empty:
        For each center in Centers:
            Tracks.append(new_track_with_KalmanFilter(center)
    Else:
        Prediction = KalmanFilter.predict(Tracks)
        Cost_matrix = compute_IoU(Prediction, centers)
        Assignments = HungarianAlgorithm.assign(cost_matrix)
        For assignment in Assignments:
            If assignment.match == True:
                Tracks[assignment.track_index] =
Centers[assignment.detection_index]
                Reset missing frame count
            Else:
                Increase missing frame count
                If missing frame count > threshold
                    Tracks.remove(assignment.track_index)
    For track in Tracks:
        If track crosses counting line and not counted:
                Increase vehicle count, mark track as counted
```

Figure 8
(a) Raw $PM_{2.5}$ data and (b) $PM_{2.5}$ data after preprocessing

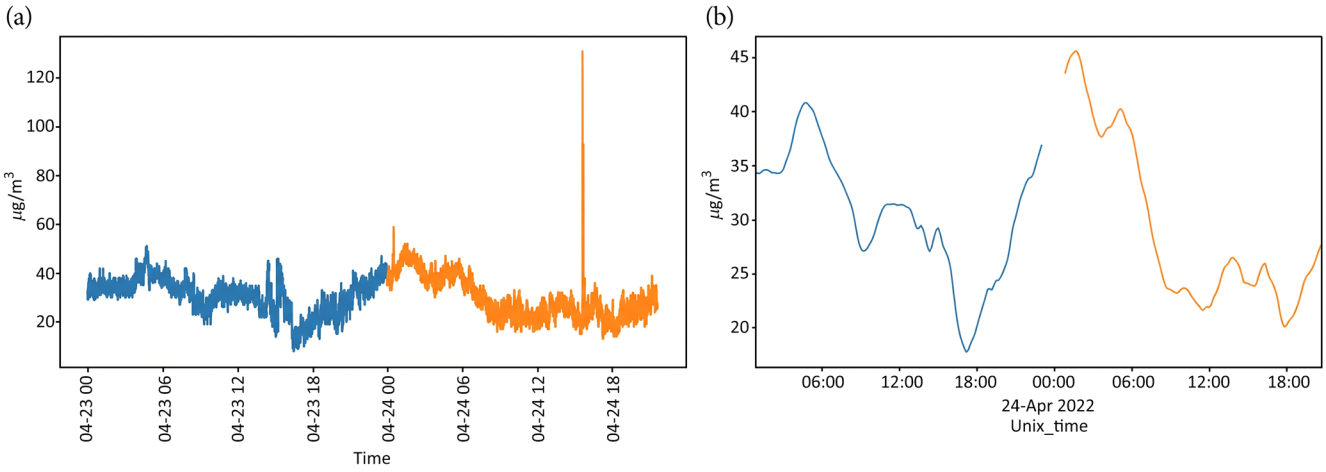


Table 2
$PM_{2.5}$ concentration estimation using a mathematical method

	X: $PM_{2.5}$ concentration of the base date, converted directly from the formula in Section 3.3.2
	X_gt: actual $PM_{2.5}$ concentration of the base date
	Lag = cross_correlation(X,X_gt)
	Daily_factor: $PM_{2.5}$ concentration contributed from other sources
	Daily_factor = shift(X,lag) – X_gt //shift function shifts the time series by the lag
	Y: $PM_{2.5}$ concentration of the estimated date, converted directly from the formula in Section 3.3.2
	Y_result: the final $PM_{2.5}$ concentration of the estimated date, initially is empty
	For i in [0, Y.length]:
	Y_result[i].time = Y[i].time
	Y_result[i].value = Y[i-lag].value + Daily_factor[i]

temperature, and humidity, which influence the dispersion of $PM_{2.5}$ particles. To avoid making assumptions, we implemented five machine learning models, ranging from the simplest to state-of-the-art models, and then compared them with the mathematical model. These models address the regression problem by taking the hour of the day and the number of vehicles at that specific timestamp as input and producing the predicted $PM_{2.5}$ level as output. The data was split chronologically for training (70%) and testing (30%) to prevent data leakage and evaluate true forecasting performance. The data is the cleaned data using the methods mentioned in Section 3.3.1. We evaluated five machine learning approaches: Cubist [29, 30], ExtraTrees, CatBoost [31], random forest, and linear regression. The hyperparameters for each model were determined through grid search. To be more specific,

1) Cubist: number of committees ∈ {1, 3, 5, 7, 10, 50}, number of rules ∈ {1, 3, 5, 7, 10, 25, 50, 75, 100, 200}.
2) ExtraTrees and random forest: number of estimators ∈ {50, 100, 200}.
3) CatBoost: learning rate ∈ {0.03, 0.01}, depth ∈ {2, 4, 6, 8, 10, 12}, L2_leaf_reg ∈ {1, 3, 5, 7}.

Table 3
Machine learning model configurations

CatBoost	Cubist	ExtraTrees and random forest
Learning rate = 0.1	Number of rules = 7	Number of estimators = 50
Depth = 10	Number of committees = 3	
L2_leaf_reg = 1		

The final configurations used are shown in Table 3. This rigorous evaluation on a separate test set helps ensure the models generalize well and are not overfitting to the training data.

4. Results and Discussion

4.1. Effectiveness of tracking algorithm

Figure 9 shows an example of the simulation. After testing on ten simulation videos, each one is 10 seconds long, it is observed that, with tracking, the counting is 10 vehicles less than ground truth on average, while without tracking, the counting is around 20 vehicles more on average. This proves that tracking is necessary for this experiment.

4.2. Prediction result

The estimated results are compared with ground truth from our sensor in two metrics: MAE, and MSE.

Table 4 presents the performance metrics for each machine learning model used in the study. As seen in Table 4, the Cubist model consistently outperforms other models in terms of MAE, and MSE, while the runner-up is CatBoost. This superior performance is attributed to Cubist's ability to effectively handle nonlinear relationships and interactions between variables, making it well suited for modeling the complex relationship between traffic density and $PM_{2.5}$ concentrations. This is further proven by the fact that the mathematical method has the worst performance, meaning a linear transformation of number of motorbikes and cars is not complex enough to model the $PM_{2.5}$ concentration of a day.

The performance result can also be explained via the generalization ability of each model. For linear regression, this is trivial,

Figure 9
One frame of the simulated video for testing the effectiveness of tracking algorithm

Table 4
Comparison between various methods on the 2 tested days

Metrics	Models	April 20 estimates using April 18 and 19 information	April 24 estimates using April 23 information
MAE	Mathematical method	11.88	8.34
	Random forest	8.16	6.40
	Linear regression	8.31	6.02
	Cubist	**4.01**	**4.5**
	ExtraTrees	6.76	5.00
	CatBoost	5.79	4.68
MSE	Mathematical method	163.82	107.83
	Random forest	95.49	68.29
	Linear regression	92.99	52.01
	Cubist	**19.76**	**21,79**
	ExtraTrees	37.13	40.87
	CatBoost	25.65	36.67

Note: The date before is used to estimate the date after.

as the problem is not linear. For random forest and ExtraTrees, these models observe overfitting, where the MAE for training is 0 but larger than 5 for testing. For Cubist, training MAE is 3.88 and testing MAE is 4.01, an indication that the model is well trained. Finally, for CatBoost, the training and validation loss is observed in Figure 10, where both lines converge, also an indication of it being well trained.

As depicted in Figure 11, Cubist delivered the best overall performance by staying close to the true values during the long stable period and avoiding the large swings seen in CatBoost and ExtraTrees. While Cubist tended to smooth out sharp peaks and troughs, slightly underestimating spikes and overestimating dips, this conservative behavior kept its average error low. CatBoost reacted more aggressively to changes, which helped in some transitions but introduced timing errors and higher variance. ExtraTrees tracked the final spike better than the others but remained too high for much of the middle segment. Overall, the results suggest that the data has regime shifts and occasional abrupt changes, and Cubist's rule-based, piecewise-linear approach is well suited for this structure.

Figure 10
Plot of CatBoost training and validation losses

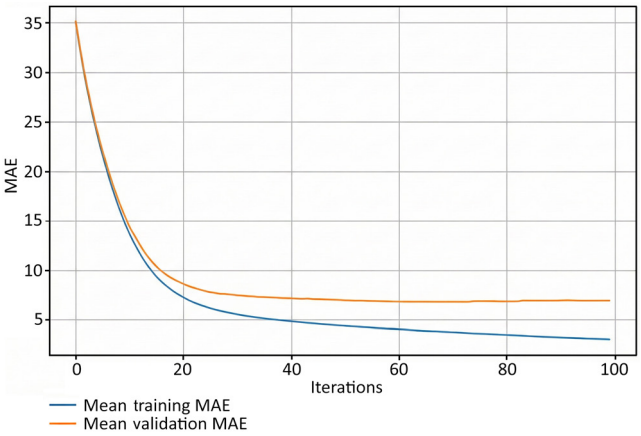
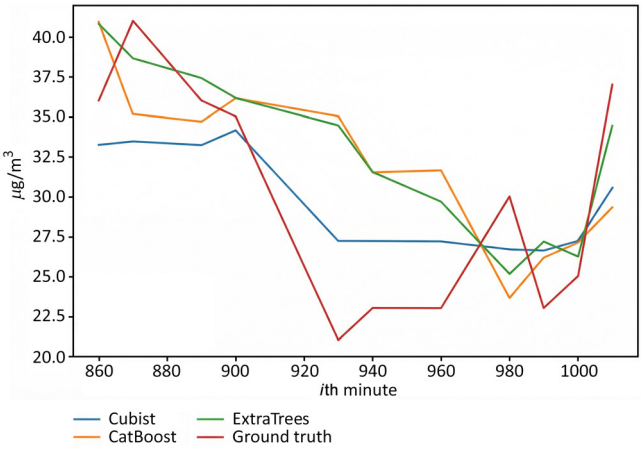


Figure 11
Estimation results of the top three ML methods versus ground truth value from the 860th to 1020th minute of April 20, 2022



5. Limitations and Future Work

Our current model primarily relies on vehicle density for $PM_{2.5}$ estimation. However, other environmental factors such as wind,

temperature, and humidity can significantly influence $PM_{2.5}$ dispersion and concentration. Neglecting these factors could lead to estimation inaccuracies.

A key limitation of this study is that the model's performance was evaluated using data collected over a short period. This is insufficient to establish robust model performance across a wider range of real-world conditions. This limited duration prevents the capture of weekly traffic patterns, diverse weather events, or seasonal effects that influence both traffic density and pollution levels. To address this, future work will focus on extending the data collection timeframe significantly, ideally over several months or a full year, to create a more comprehensive and representative dataset, and testing the model's generalizability by deploying monitoring systems in multiple locations with different urban characteristics (e.g., street canyons, open residential areas) to ensure the model performs well across various environments.

Our mathematical estimation of $PM_{2.5}$ concentration relies on several simplifying assumptions. Specifically, it assumes a fixed dispersion volume and a constant emission factor (EF). This approach does not account for how real-world urban topology (i.e., the layout of buildings) and meteorological conditions (e.g., wind speed and direction) affect pollutant dispersion. Furthermore, the EF can vary significantly with vehicle type, speed, and acceleration, which our constant factor does not capture. Future iterations of this research will aim to improve the model's physical realism by incorporating a basic atmospheric dispersion model, such as a Gaussian plume model [32], that can utilize meteorological data to more accurately simulate how pollutants spread from the source.

The current design of the IoT box enclosure does not effectively separate electrical components from the airflow used for sampling. Air is drawn in by an external fan through a lower inlet, flows over the components, and exits through an upper outlet. This design leads to the accumulation of dust and particles on the components, significantly reducing their lifespan. Additionally, the design poses a risk of insect infiltration, particularly in tropical regions like Vietnam, where insects such as ants are prevalent. Insects entering the internal chamber could damage the electrical components, presenting a serious challenge. In order to address these issues, it is recommended that the enclosure be redesigned to isolate the components in a separate chamber, allowing air to pass through an open-ended, tube-shaped compartment. Sensors can be mounted on the sides of this tube to enable exposure while maintaining the necessary isolation.

6. Conclusion

To sum up, there is a great relationship between Western rhetoric and English writing; rhetoric is not only the use of rhetoric. However, the use of figures of speech can provide some very useful writing skills for English writing. Rhetoric, as a whole, is an art of persuasion. In the process of using various rhetorical devices to write, English writers should pay attention to strengthening their understanding of rhetoric itself and the knowledge covered within it and combine the rhetorical perspective to improve their ability to use rhetorical devices such as contrastive, exaggeration, and so on.

Ethical Statement

This study does not contain any studies with human or animal subjects performed by any of the authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

Data Availability Statement

The data that support the findings of this study are openly available in GitHub at https://github.com/comrang-altf4/$PM_{2.5}$.

Author Contribution Statement

Chuong Dinh Le: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Hoang Viet Pham:** Conceptualization, Methodology, Software, Investigation, Data curation, Writing – original draft, Visualization. **Thinh Gia Tran:** Conceptualization, Methodology, Software, Validation, Investigation, Writing – original draft, Visualization. **An Dinh Le:** Conceptualization, Methodology, Resources, Writing – review & editing, Supervision, Project administration. **Anh-Duy Pham:** Conceptualization, Methodology, Resources, Supervision, Project administration. **Dat Thanh Vo:** Software, Writing – original draft, Visualization. **Hien Bich Vo:** Resources, Supervision. **Huy-Dung Han:** Supervision.

References

[1] Ho, T. H., van Dang, C., Pham, T. T. B., & Wangwongwatana, S. (2023). Assessment of health and economic benefits of reducing fine particulate matter ($PM_{2.5}$) concentration in Ho Chi Minh City, Vietnam. *Hygiene and Environmental Health Advances*, *6*, 100045. https://doi.org/10.1016/j.heha.2023.100045

[2] Tang, V. T., Oanh, N. T. K., Rene, E. R., & Binh, T. N. (2020). Analysis of roadside air pollutant concentrations and potential health risk of exposure in Hanoi, Vietnam. *Journal of Environmental Science and Health, Part A*, *55*(8), 975–988. https://doi.org/10.1080/10934529.2020.1763091

[3] Huu, D. N., & Ngoc, V. N. (2021). Analysis study of current transportation status in Vietnam's urban traffic and the transition to electric two-wheelers mobility. *Sustainability*, *13*(10), 5577. https://doi.org/10.3390/su13105577

[4] Pham, V. A. T., Tran, T. K., Nguyen, K. M., Tran, T. V., Vu, L. H., & Pham, H. T. T. (2024). The relationship between population growth and precipitation change in some regions across Vietnam: Implications for urbanization effect. *Environmental Science and Pollution Research*, *31*(10), 15007–15025. https://doi.org/10.1007/s11356-024-32039-0

[5] Simelton, E., Duong, T. M., & Houzer, E. (2021). When the "strong arms" leave the farms—Migration, gender roles and risk reduction in Vietnam. *Sustainability*, *13*(7), 4081. https://doi.org/10.3390/su13074081

[6] Sukuman, T., Ueda, K., Sujaritpong, S., Praekunatham, H., Punnasiri, K., Wimuktayon, T., & Prapaspongsa, T. (2023). Health impacts from $PM_{2.5}$ exposure using environmental epidemiology and health risk assessment: A review. *Applied Environmental Research*, *45*(3), 010. https://doi.org/10.35762/AER.2023010

[7] Lin, L., & Purnell, N. (2019). A world with a billion cameras watching you is just around the corner. *The Wall Street Journal*. https://www.wsj.com/articles/a-billion-surveillance-cameras-forecast-to-be-watching-within-two-years-11575565402

[8] Ciampi, L., Gennaro, C., Carrara, F., Falchi, F., Vairo, C., & Amato, G. (2022). Multi-camera vehicle counting using edge-AI. *Expert Systems with Applications*, *207*, 117929. https://doi.org/10.1016/j.eswa.2022.117929

[9] Gomaa, A., Minematsu, T., Abdelwahab, M. M., Abo-Zahhad, M., & Taniguchi, R.-I. (2022). Faster CNN-based vehicle detection and counting strategy for fixed camera scenes. *Multimedia Tools*

and Applications, *81*(18), 25443–25471. https://doi.org/10.1007/s11042-022-12370-9

[10] Bouaich, S., Mahraz, M. A., Riffi, J., & Tairi, H. (2021). Vehicle counting based on road lines. *Pattern Recognition and Image Analysis*, *31*(4), 739–748. https://doi.org/10.1134/S1054661821040076

[11] Sahanavin, N., Prueksasit, T., & Tantrakarnapa, K. (2018). Relationship between PM_{10} and $PM_{2.5}$ levels in high-traffic area determined using path analysis and linear regression. *Journal of Environmental Sciences*, *69*, 105–114. https://doi.org/10.1016/j.jes.2017.01.017

[12] Pongpiachan, S., Kositanont, C., Palakun, J., Liu, S., Ho, K. F., & Cao, J. (2015). Effects of day-of-week trends and vehicle types on $PM_{2.5}$-bounded carbonaceous compositions. *Science of the Total Environment*, *532*, 484–494. https://doi.org/10.1016/j.scitotenv.2015.06.046

[13] Shen, X., Yao, Z., Huo, H., He, K., Zhang, Y., Liu, H., & Ye, Y. (2014). $PM_{2.5}$ emissions from light-duty gasoline vehicles in Beijing, China. *Science of the Total Environment*, *487*, 521–527. https://doi.org/10.1016/j.scitotenv.2014.04.059

[14] Momirski, L. A., & Berčič, T. (2022). Southern inner ring road in Ljubljana: 2021 data set from traffic sensors installed as part of the citizen science project WeCount. *Data in Brief*, *41*, 107878. https://doi.org/10.1016/j.dib.2022.107878

[15] Pham, A.-D., Le, A. D., Le, C. D., Pham, H. V., & Vo, H. B. (2023). Harnessing sheaf theory for enhanced air quality monitoring: Overcoming conventional limitations with topology-inspired self-correcting algorithm. In *Proceedings of the Future Technologies Conference 2023, Volume 1*, 102–122. https://doi.org/10.1007/978-3-031-47454-5_8

[16] Tran, T. G., Vo, D. T., Tran, L. C., Pham, H. V., Le, C. D., Le, A. D.,…& Vo, H. B. (2024). Scalable, low-cost, and versatile system design for air pollution and traffic density monitoring and analysis. In *Proceedings of the 8th International Conference on Sustainable Urban Development*, 321–337. https://doi.org/10.1007/978-981-99-8003-1_18

[17] Same Sky. (2024). CFM-4010-03-10: Axial fans, DC fans [Product specifications]. https://www.cuidevices.com/product/resource/cfm-40.pdf

[18] Badura, M., Batog, P., Drzeniecka-Osiadacz, A., & Modzel, P. (2018). Evaluation of low-cost sensors for ambient $PM_{2.5}$ monitoring. *Journal of Sensors*, *2018*(1), 5096540. https://doi.org/10.1155/2018/5096540

[19] Mosqueira-Rey, E., Hernández-Pereira, E., Alonso-Ríos, D., Bobes-Bascarán, J., & Fernández-Leal, Á. (2023). Human-in-the-loop machine learning: A state of the art. *Artificial Intelligence Review*, *56*(4), 3005–3054. https://doi.org/10.1007/s10462-022-10246-w

[20] Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2023). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7464–7475. https://doi.org/10.1109/CVPR52729.2023.00721

[21] Tan, M., Pang, R., & Le, Q. V. (2020). EfficientDet: Scalable and efficient object detection. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10778–10787. https://doi.org/10.1109/CVPR42600.2020.01079

[22] Jin, X., & Han, J. (2011). K-means clustering. In C. Sammut & G. I. Webb (Eds.) *Encyclopedia of machine learning* (pp. 563–564). Springer. https://doi.org/10.1007/978-0-387-30164-8_425

[23] Le, C. D., Pham, H. V., Pham, D. A., Le, A. D., & Vo, H. B. (2022). A $PM_{2.5}$ concentration prediction framework with vehicle tracking system: From cause to effect. In *2022 RIVF International Conference on Computing and Communication Technologies*, 714–719. https://doi.org/10.1109/RIVF55975.2022.10013864

[24] Pei, Y., Biswas, S., Fussell, D. S., & Pingali, K. (2019). An elementary introduction to Kalman filtering. *Communications of the ACM*, *62*(11), 122–133. http://dx.doi.org/10.1145/3363294

[25] Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, *2*(1–2), 83–97. https://doi.org/10.1002/nav.3800020109

[26] Cleveland, R. B., Cleveland, W. S., McRae, J. E., & Terpenning, I. (1990). STL: A seasonal-trend decomposition. *Journal of Official Statistics*, *6*(1), 3–73.

[27] Phung, N. K., Long, N. Q., Tin, N. V., & Le, D. T. T. (2020). Development of a $PM_{2.5}$ forecasting system integrating low-cost sensors for Ho Chi Minh City, Vietnam. *Aerosol and Air Quality Research*, *20*(6), 1454–1468. https://doi.org/10.4209/aaqr.2019.10.0490

[28] California Air Resources Board. (2022). *Methods to find the cost-effectiveness of funding air quality projects*. https://ww2.arb.ca.gov/sites/default/files/2023-01/Cost%20Effectiveness%20Tables%202022_final.pdf

[29] Quinlan, J. R. (1992). Learning with continuous classes. In *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*, 343–348.

[30] Quinlan, J. R. (1993). Combining instance-based and model-based learning. In *Proceedings of the Tenth International Conference on Machine Learning*, 236–243.

[31] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: Unbiased boosting with categorical features. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 6639–6649.

[32] Kawka, M., Struzewska, J., & Kaminski, J. W. (2023). Downscaling of regional air quality model using Gaussian plume model and random forest regression. *Atmosphere*, *14*(7), 1171. https://doi.org/10.3390/atmos14071171

How to Cite: Le, C. D., Pham, H. V., Tran, T. G., Le, A. D., Pham, A.-D., Vo, D. T., Vo H. B., & Han, H.-D. (2026). Toward a Causal $PM_{2.5}$ Concentration Forecasting by Inferencing from Local Vehicle Tracking with a Low-Cost End-to-End Sensor System. *Artificial Intelligence and Applications*, *4*(2), 187–196. https://doi.org/10.47852/bonviewAIA62024212