IoT based monitoring of air quality and traffic using regression analysis

José Ángel Martín-Baos[a], Luis Rodriguez-Benitez[b,*], Ricardo García-Ródenas[a], Jun Liu[c]

[a] *Departament of Mathematics, Escuela Superior de Informática, University of Castilla-La Mancha, Ciudad Real, Spain*
[b] *Department of Information and System Technologies, Escuela Superior de Informática, University of Castilla-La Mancha, Ciudad Real, Spain*
[c] *School of Computing, University of Ulster, Northern Ireland, UK*

Abstract

Dynamic traffic management (DTM) systems are used to reduce the negative externalities of traffic congestion, such as air pollution in urban areas. They require traffic and environmental monitoring infrastructures. In this paper we present a prototype of a low-cost Internet of Things (IoT) system for monitoring traffic flow and the Air Quality Index (AQI). The computation of the traffic flows is based on processing video in the compressed domain. Only using motion vectors as input, traffic flow is computed in real-time over an embedded architecture. An estimation of the AQI is supported by machine learning regression techniques, using different feature data obtained from the IoT device. These automatic learning techniques overcome the need for complex calibration and other limitations of embedded devices in making the needed measurements of the pollutant gases for the computation of the actual AQI. The experimentation with the data obtained from different cities representing different scenarios with a variety of climate and traffic conditions, allows validating the proposed architecture. As regressors, Linear Regression (LR), Gaussian Process Regression (GPR) and Random Forest (RF) are compared using the performance metrics R^2, MSE, MAE and MRE resulting in a relevant improvement of the AQI estimations of our proposal.

Keywords: Air Quality Index, Regression Modelling, Video Motion Vectors, Embedded Systems

1. Introduction

Road transport has emerged as as one of the most significant sources of air contamination in both cities and urban areas and has a major effect on local air conditions and human health. For this reason, it is becoming increasingly necessary to accurately estimate the contribution of road transport to urban air pollution so that measures to reduce contamination can be properly designed and implemented. These pollution reduction actions are becoming even more necessary because of the continual increase in vehicle use and the

*Corresponding author
Email address: Luis.Rodriguez@uclm.es (Luis Rodriguez-Benitez)

deteriorating driving conditions (traffic congestion). Consequently, a first step for the local governments to address their environmental objectives/challenges (e.g. air quality standards or national emission ceilings) is to provide reliable emission systems to accurately assess air pollution and to detect pollution peaks. The most common approach for air quality monitoring is to rely on environmental monitoring stations. Unfortunately, they are very expensive to acquire (have prices ranging between €5000 and €30000 per device [58]), which causes a sparsely deployed, resulting in limited spatial resolution for pollutant measurements. In a second stage, the authorities should have a set of effective traffic management strategies (TMS) to mitigate emissions and reduce the health effects of traffic-related air pollution in urban areas. [23] have identified that empirical evidence on the effectiveness of these TMS is limited. These authors point out that an evidence-based approach to transportation systems planning necessitates additional resource allocation to ex post evaluations and performance monitoring for air quality impacts of TMS. A fundamental fact in designing systems to assess TMS is that certain pollutants generated by traffic emissions ($PM_{2.5}$ and CO) are incorporated to urban background concentrations while other pollutants, like NO y NO_2, have a local contribution [49]. Moreover, the meteorological conditions [24, 50], affect to atmospheric dispersion, resulting in significant spatial and temporal variability. The monitoring systems designed should enable observations on traffic, air quality and weather at high spatial resolution in near-real-time.

As an alternative to environmental monitoring stations, and focusing on delimiting at the neighbourhood scale where air pollution has a more negative health impact on inhabitants, low-cost distributed IoT systems are recommended for monitoring both the levels of contamination and traffic pattern by neighbourhood or street in real time.

With respect to the categorisation of the air pollution, we use the AQI [1] as an indicator for daily reporting of air quality. It shows how clean or polluted the air is, and what associated human health effects must be taken into consideration. The AQI is computed for ground-level ozone, Particulate Matter (PM), carbon monoxide, sulfur dioxide, and nitrogen dioxide. Ground-level ozone and particulate matter in the air are the two pollutants that pose the biggest risk to human health in most countries. It is complex to not only have sensors in the embedded device for all these measures but also to calibrate these sensors. Hence, we propose an estimation of the AQI by machine learning regression techniques. More concretely, we compare LR, RF, and GPR. These regressors have been used as estimators in a great number of previous works. For instance, with respect to RF we can find references in fields as different as breast cancer prediction [2], turbine noise prediction [3], failure prediction of hard drives [4] and our proposal of a Particulate Matter 2.5 ($PM_{2.5}$) prediction in [5] using additionally XGBoost and deep learning. Related to GPR, one sees predictions in solar power [6], daily demand in tourism planning [7] and wind power [8], among others.

To sum up, the main objective of this paper is the design and construction of a low-cost integrated platform prototype for road traffic and air pollution surveillance. It focuses on the design and implementation of the software and hardware architecture of the embedded devices within soft-computing techniques for the estimation of pollution levels represented by the AQI and the determination of traffic flows. With this information, the authorities can asses the effectiveness of implemented TMS and monitor the actions taken. Figure 1 presents the whole architecture of the DTM system.

1.1. Major contributions

In the review of the literature in Section 2 we consider that the following aspects of air pollutant control in cities have received limited attention being challenges to be addressed.

- Build reliable emission monitoring systems to accurately assess the contribution of road transport to urban air pollution so that measures to reduce contamination can be properly designed and implemented.
- Effective integration of pollution control systems with traffic monitoring infrastructure to provide a complete view of the urban area, which can be then used as a warning system to help authorities make decisions.

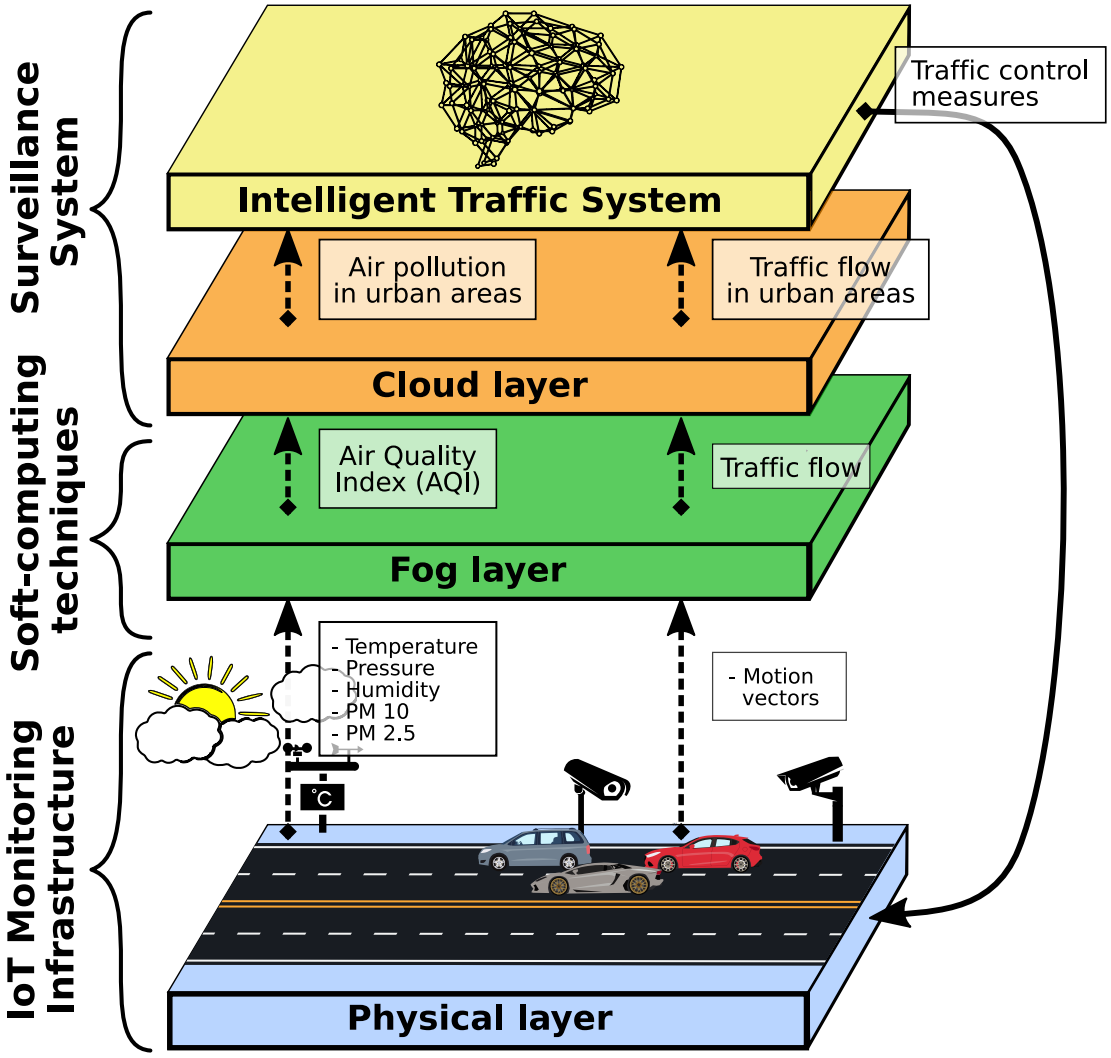


Figure 1: Prototype of the pollution surveillance and traffic control system

- Effective and efficient control of traffic flows in the city as they increase the pollution levels, which have to be supported by IoT infrastructures connected to cloud platforms as well as large data applications based on sensors.

This paper is a step to face the above challenges, being its main contribution a low-cost distributed architecture based on embedded systems whose main component is a Raspberry Pi to simultaneously monitor road traffic and air pollution in urban areas. The purpose of this hardware and software system is to support the acquisition of data to determine the pollution level, in conjunction with information on traffic flows at multiple points in a city in real time. The key elements of the architecture are:

- Low-cost multi-sensor system design and on-board processing by the development of a module for obtaining environmental parameters, particle matter estimates, and video in the Raspberry Pi system.
- Vehicle counting by means of an algorithm based on the compressed video domain that allows obtaining results taking as exclusive input the values of the H264 motion vectors.

- LR, GPR and RF approaches are applied to build models for estimating the AQI based on a reduced number of pollutants and environmental parameters in comparison with the fixed-site air quality monitoring stations.
- Implementation of the regression models and the vehicle counting algorithm in the embedded platform for air pollution monitoring to estimate the AQI and the traffic flows in real time. The level of accuracy obtained in these estimates enables them to be integrated into DTM systems.
- Smart city integration by developing a cloud service to communicate and synchronise data from different Raspberry Pi devices and a front end to monitor the data registered by the different Raspberry Pi devices integrating an alert or notification system for local authorities.

1.2. Structure of this paper

The rest of this paper is organised as follows: First, Section 2 details the main techniques, methods, and projects relating to intelligent transport systems and their relation to traffic pollution. This section also describes several monitoring systems in the context of the IoT. Next, Section 3 introduces our proposal of a three layered architecture with a physical layer, a fog layer, and a cloud layer. Section 4 then describes the experiments completed in order to validate the operation of the complete architecture. Lastly, Section 5 presents some conclusions and lines for future research.

2. Related work

In this section, the main techniques, methods, and projects related to transportation systems and their relations with traffic pollution are detailed. Furthermore, due to the fact that IoT has become an integral part of smart cities, we present a review of the literature related to the relevance of the IoT in smart cities.

2.1. Intelligent transportation systems and traffic pollution

Intelligent Transport Systems (ITS) are tools that combine advanced communication and information technologies to solve transport problems such as traffic congestion, safety, and environmental conservation. Today, different modes of transport are elements that directly influence the activities of everyday life. However, some of these modes of transport (such as road traffic, aviation, maritime transport, etc.) are increasing the emission of greenhouse gases, with negative consequences for the environment. For this reason, it is very important to develop a low-carbon economy, so that the environment is threatened as little as possible. For instance, EcoMobility [9] pursues this goal by focusing on energy-efficient ITS.

Road traffic [10, 11, 12] is often the main source of air pollution in urban areas, so it is increasingly necessary to accurately estimate its contribution to urban air pollution. Therefore, many institutions have developed emission models and systems to predict the contribution of road transport to air pollution. For example, CITEAIR has been created by the European Union [13] in order to develop effective means to collect, present, and compare data on air quality in multiple European cities. Other systems, such as DTM systems [14, 15, 16], can focus on emission reduction, using tools such as variable speed limits, speed measurement, adaptive signal synchronisation [17], routing or prioritisation of vehicle class [18], and so on. These measures may generate some indirect effects, such as longer travel delays, a decrease in traffic performance, or higher levels of greenhouse gas emissions. They should therefore only be activated when air quality conditions warrant it [19] and reliable emission models are needed.

The COPERT emission model [20] can be applied to determine pollutant emissions caused by traffic. This model is the most widely used method in Europe for the preparation of official national inventories of road traffic emissions [21]. The COPERT model estimates the emissions of six different categories of vehicles. Using this model and the traffic flow rate, the levels of the different pollutants can be predicted and therefore active traffic control measures can be taken on the basis of these estimates to mitigate their effects.

The above papers are examples of mathematical models used to quantify vehicular emissions [22]. Currently, the assessment of TMS measures is based on estimates of the capacity of such measures to reduce

emissions. However, [23] point out that there exists limited evidence of effects on air quality for two of the TMS strategies: area road pricing and low emission zones. Insufficient evidence exists for all other TMS and effects. One of the main causes is the lack of infrastructure capable of measuring these effects in the city. Pollution values depend on both the process of emission generation (which can be estimated with mathematical models) and their dissipation depending on meteorological conditions, which makes it necessary to develop approaches based on massive data collected in real time throughout the cities.

2.2. Air pollution and traffic flow monitoring systems in the the smart city framework

Traditionally, air pollution is measured using dedicated instruments at fixed monitoring stations, which are placed sparsely in urban areas. This information together with GIS and traffic datasets allow building air quality models based on techniques like land-use regression (LUR), machine learning, or hybrid methods for assessing intra-urban air pollution contrasts [25, 26, 27]. Nowadays, with the development of the so-called 'smart city', new sensor-based alternatives are appearing. The 'smart city' is a name given to a city that incorporates information and communication technologies (ICTs) in order to improve the efficiency and reliability of city services like electricity [28, 29], mobility [30], and other services [31, 32, 33]. One of the major problems related to the IoT in the smart city is the integration and interconnection of lots of IoT objects, such as sensors, actuators, etc. with the intelligent systems composing the smart city. Basically, an explosion of data collected in real time must be processed, transmitted, and sometimes stored. For this reason, the Cloud appears as the adopted technology to be merged with the IoT in order to enable applications in a large number of different scenarios. This is known as the CloudIoT paradigm. In [34] a literature survey on the integration of both technologies is presented. Additionally, [35] presents a survey focused on integration components. That is, Cloud platforms, Cloud infrastructures, and IoT middleware.

The emergence of low-cost air pollution platforms enables observations at high spatial resolution in near-real-time, especially for traffic-related pollution monitoring. Now, we focus our attention on these environment monitoring systems in the IoT context. For instance, Shat et al. in [36] present a low power system for $PM_{2.5}$ prediction where the correlation between particulate matter and other pollutants is obtained by means of a prediction model combining analytical equations and an artificial neural network. In [37] not only is $PM_{2.5}$ monitored but also Particulate Matter 10 (PM_{10}) and PM_1. Once this information has been transmitted to the IoT platform Thingspeak, they establish an alarm rate for situations where the AQI is degrading and PM is increasing. The concept of fog computing-based appears in [38] where once the sensors information has been collected, this data is sent over the fog nodes. An Arduino microcontroller is used to monitor Ozone (O_3), Sulphur Dioxide (SO_2), and Carbon Monoxide (CO) and particulates in [39]. Again, the ThingSpeak cloud system allows showing the monitoring results through a web page. Hawari et al. [40] used the embedded device to monitor the level of pollution related to a different index than AQI. This index is the Malaysia Air Pollution Index, known as API. Another communication protocol with the cloud like HTTP is introduced in [41]. They use an ESP8266 smart controller which captures the sensor information and creates JSON packets that are sent to the Cloud. The possibility of monitoring the data in real time using a smartphone is presented in [42]. To this end, an application named AirProp is used. In [58] data quality obtained from the use of low-cost sensor technologies for air quality monitoring is analysed, showing that their performance varies spatially and temporally, as it depends on the atmospheric composition and the meteorological conditions while in [59] authors showed how machine-learning-based calibration can improve the accuracy of low-cost sensors.

Now, we present a review of several works related to video processing in embedded systems. For instance, Rodriguez-Benitez et al. in [43] use motion vector information to perform real time analysis of traffic scenes. More concretely, the system applies to the identification of overtakes, particularly those exceeding the maximum allowed speed. In [44] a tracking specific technique is introduced, known as MVint. MVint combines a motion vector based interpolator and a Deep convolutional neural network based detector to achieve high accuracy and energy efficiency by using motion vectors. Another approach is that of [45], who introduce detection devices to screen for traffic flow congestion through the provision of multiple proposed services such as vehicle counting, live video, and re-routing services. Users can access the services using the proposed mobile application connected to the Internet, as these services are integrated with the public map service. Related to this is the proposal of Razavi et al. [46] where a new method for the management of

traffic lights using a combination of IoT and image and video processing techniques is presented. In the proposed models, the scheduling of traffic lights is determined by the intensity and the number of vehicles going by. In [47], authors extract the motion vector features and they are used to classify the traffic patterns into three categories: light, medium and heavy by means of neural network. This method does not allow to estimate the traffic flow but only a congestion level with low granularity. In [48], authors use Raspberry Pi to detect the type of vehicle. They compare a frame with a reference frame by evaluating all pixels considering only abrupt changes. It has an accuracy of over 95.7%.

Finally, it is interesting to mention several studies [49, 51, 52] that highlight the local impact of traffic on air quality that generate differences in average pollutant concentrations between near-road and urban background station pairs. These studies show that excess air pollution associated with proximity to roads is significant. Moreover, the local meteorology is a critical factor determining the extent of near-road impact [24, 50] in the dissipation of emissions. The design of a infrastructure to determinate the contribution of traffic emission on air quality and/or the exposure of the population to pollutants requires the establishment of a densely localised combined traffic, pollution and weather monitoring network. The above review shows that traffic and pollution monitoring systems have evolved separately, although the use of traffic datasets for air quality modelling is well established.

Now that some of the literature related to the proposed solution has been reviewed, a three-layered architecture to deploy the proposed system will be described in next section.

3. A soft-computing solution based on a three-layered hierarchical distributed architecture

In this section the architecture of the proposed system is introduced. We propose a three layered architecture with a physical layer, a fog layer and a cloud layer. The first layer is described in Section 3.1, the second is described in Section 3.2, and the Cloud layer is described in Section 3.3.

3.1. Physical layer

Also known as the user-device layer, this layer is considered the closest to the 'things' or the nodes involved. This is where the data produced by the nodes are collected. Section 3.1.1 describes the extraction of motion vectors from a camera device attached to the Raspberry Pi system. Then, the operations needed to configure and recover data from the environmental and pollutant sensors are described in Section 3.1.2. Before beginning, Table 1 shows the hardware configuration of the IoT device.

Table 1: Hardware resources

Module	**Description**
Raspberry Pi 3	Low cost, credit-card sized computer.
Raspberry Pi Camera Module V2	Capture pictures and record videos using the CSI port.
Sense HAT	It is an add-on board with a 8x8 RGB LED matrix, five-button joystick and sensors of temperature, humidity, barometric pressure, magnetometer, accelerometer, and gyroscope.
SDS011 (PM Sensor)	Low cost $PM_{2.5}$ and PM_{10} sensor. Using principle of laser scattering it can get the particle concentration between 0.3 to 10μm in the air.

3.1.1. Extraction of motion vectors from a camera device

A macroblock is the basic unit in video compression formats based on linear block transformations. Each macroblock contains the information about the luminance, the chrominance, and the motion vectors associated to each 16×16-pixel region of a frame.

A motion vector represents a temporal redundancy pattern detected between two consecutive frames encoded using the H.264/AVC codec, defining a distance and a direction in the form of a two-dimensional vector. In other words, it represents the movement of each macro block in the current frame in relation to the reference frame. For instance, Figure 2 shows an example of the motion vectors generated by the Raspberry

Pi device. Using this information, most video encoding algorithms, instead of storing all the pixels of the current frame, store the motion vectors corresponding to the macroblocks that have been identified in the reference image. This has the advantage, among others, that it is possible to store the video information using much less disk space.

Figure 2: Example of motion vectors obtained using the developed device

The Raspberry Pi Camera Module V2 is connected to the CSI port of the Raspberry Pi device and supports recording video at $1080p30$. This device is able to extract in real time the motion vector estimates that the H.264 encoder calculates while compressing video using low CPU resources.

The advantage of dealing with motion vectors to determine the traffic flow instead of the full information from the video images is the low amount of information, and therefore processing capacity, needed to execute an algorithm to process this data. For example, in a video where each frame has a width of x pixels and a height of y pixels, the total number of pixels can be calculated as $Npx = x \cdot y$. If we assume that one motion vector is associated to each macro block in each frame, and taking into account that each macro block in the H.264 video format has a size of 16×16 pixels, then the number of motion vectors (Nmv) in a frame can be calculated as shown in Equation 1.

$$Nmv = \frac{x}{16} \cdot \frac{y}{16} = \frac{xy}{256} \tag{1}$$

Therefore, Equation 2 computes the percentage of processed video information that is employed by the device if motion vectors are used instead of processing the individual frames. Consequently, it can be stated that using motion vectors, the information to be processed is much less than working at the pixel level.

$$\frac{Nmv}{Npx} \simeq 0.4\% \tag{2}$$

After presenting the procedure for the extraction of the video information, the subsequent section will introduce the configuration needed to capture the environmental data.

3.1.2. Architecture of a system for monitoring environmental parameters

This section presents a system for obtaining environmental data. This goal involves the inclusion of some environmental sensors to measure different parameters such as temperature, humidity, and pressure, as well as a low-cost PM sensor. The term 'PM' refers to a mixture of solid particles and liquid droplets found in the air that come in many sizes and shapes and can be made up of hundreds of different chemicals. Some of them are emitted directly from sources such as construction sites, unpaved roads, fields, smokestacks, or fires. These particles consitute most of the particles formed in the atmosphere as a result of complex reactions of chemicals such as sulphur dioxide and nitrogen oxides, which are pollutants emitted from power plants, industries, and automobiles. Particle pollution includes:

- PM_{10}: particulate matter 10 micrometers or less in diameter.
- $PM_{2.5}$: particulate matter 2.5 micrometers or less in diameter, which is about 3% of the diameter of a human hair.

The Raspberry Pi Sense Hat board [53] is installed directly by placing it correctly on top of the board using the GPIO pins. The PM sensor is connected using the UART protocol through the GPIO pins of the Raspberry Pi.

The SDS011 is an economical PM sensor that uses the principle of laser scattering to obtain the concentration in the air of particles between 0.3 to 10μ m. This sensor includes a built-in fan to assure stable and reliable operation. The sensor data can be accessed digitally using the UART protocol or using the analogue output signal in form of Pulse-width modulation (PWM) output.

This electronic component is connected to the Raspberry Pi system using the GPIO pins (Figure 3). However, as the Sense hat module is already installed, the GPIO pins 3, 5, 16, 18, 22, and 24 are already in use. The GPIO pins 8 and 10 implement the UART protocol and are used to connect the PM sensor. The GPIO 4 and 6 pins are used for 5 volt power and ground, respectively.

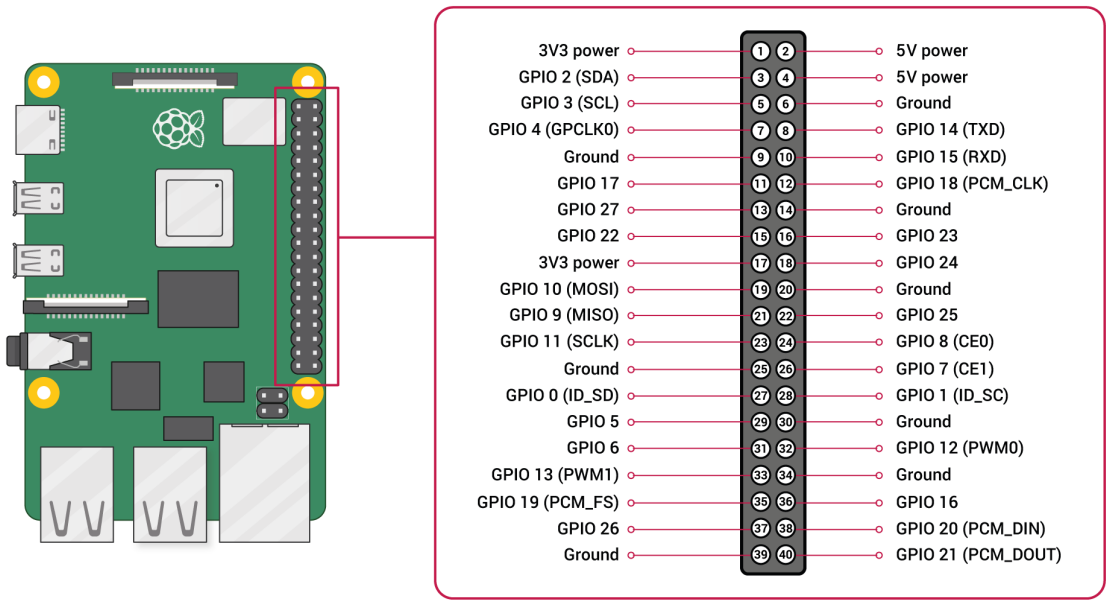


Figure 3: Raspberry Pi GPIO pinout. Source: Raspberry Pi Foundation

To synchronize and collect data from all these sensors, a Python module has been developed. This module is in charge of: Firstly, configure all the GPIO pins needed for the different sensors, as well as the initialisation of the PM sensor. Secondly, collecting all the data obtained from the PM sensor and the environmental sensors installed in the Sense Hat.

To avoid noisy data or sudden changes in data caused by erroneous measurements, a smoothed average is employed, as shown in Equation 3, where w_{t+1} represents the actual data read by the sensor, $\overline{w_t}$ is the value calculated by the previous measurement equation, and α is a weight that indicates how the new values affect the final value represented as $\overline{w_{t+1}}$.

$$\overline{w_{t+1}} = \alpha \cdot w_{t+1} + (1-\alpha) \cdot \overline{w_t} \tag{3}$$

The coefficient α must be calculated so that the influence of the previous n^{th} measurements are insignificant. Therefore, this formula can be expanded as shown in Equation 4, to demonstrate Equation 5, where the maximum influence of the previous n^{th} measurements (ε) is calculated.

$$\begin{aligned}\overline{w_{t+1}} &= \alpha \cdot w_{t+1} + (1-\alpha)\left[\alpha \cdot w_t + (1-\alpha) \cdot \overline{w_{t-1}}\right] = \\ &= \alpha \cdot w_{t+1} + (1-\alpha) \cdot \alpha \cdot w_t + (1-\alpha)^2\left[\alpha \cdot w_{t-1} + (1-\alpha) \cdot \overline{w_{t-2}}\right] = \\ &= ... = \\ &= \alpha \cdot w_{t+1} + (1-\alpha) \cdot \alpha \cdot w_t + ... + (1-\alpha)^n \cdot \overline{w_{t+1-n}}\end{aligned} \tag{4}$$

$$(1-\alpha)^n < \varepsilon \tag{5}$$

It is assumed that ε is insignificant with a value of 10^{-3}. We want to obtain values from the sensors every 2.5 minutes and that only the data from the last half hour influence the current measures. Then, n must be 12. We now operate on Equation 5 to obtain Equation 6.

$$(1-\alpha)^n < \varepsilon \Rightarrow 1-\alpha < \sqrt[n]{\varepsilon} \Rightarrow 1 - \sqrt[n]{\varepsilon} < \alpha \tag{6}$$

Then, considering $n = 12$ and $\varepsilon = 10^{-3}$, by Equation 6, $\alpha = 0.57$. This parameter is used to adjust Equation 3, which is implemented in the data collection module.

3.2. Fog Layer

This is the layer in which the data is saved, interpreted, and evaluated. The most latency-sensitive applications are immediately handled and the data collected for long-term study and prediction is sent to the Cloud. It is also known as the edge-node Layer. Here is where soft computing techniques are introduced. More concretely in Section 3.2.1 an algorithm tolerant to the uncertainty derived from taking as input the motion vectors, a sparse and imprecise approximation to optical flow, is introduced to compute the traffic flow. Furthermore, in Section 3.2.2 two supervised machine learning methods, Random Forest (RF) and Gaussian Process Regression (GPR), and a least squares Linear Regression model are used to estimate the AQI.

3.2.1. Computing the traffic flow based on video information

This section describes an algorithm for determining the traffic flow on a road based on the number of vehicles detected by the Raspberry Pi using the motion data provided by the connected camera. The hardware of the Raspberry Pi computer is powerful enough to execute this algorithm in real time as its execution does not require significant computational power or memory. With respect to vehicle counting there exist several kind of algorithms like [57] based on tracking and deep learning techniques among others. The execution of these techniques is limited in embedded devices as they require a high level of computing resources. However, the one proposed here uses motion vectors as input which allows its execution in on-board devices. It is obvious that the detection efficiency is lower than other techniques that use the whole image information as input, although as it will be shown in the section of experimentation it is enough for this purpose.

The developed algorithm is able to determine the number of vehicles that are travelling on a street and determine their direction (left or right), which allows computing the traffic flow in each direction of the street. The variables and parameters which are needed for the algorithm to operate are shown in Table 2 and its pseudo-code is shown in Algorithm 1.

This algorithm receives as an input a vector with the number of motion vectors captured in a given time interval for a given direction of traffic flow. Therefore, before this algorithm is employed, the number of motion vectors generated by the H.264 codec in each of the possible directions of the traffic flow must be determined for each input frame. Furthermore, it is recommended to smooth the resulting vector to remove possible noise (e.g., due to coding errors, reflected light, shadows, etc.). Once the vector is smoothed, Algorithm 1 can be employed, obtaining the traffic flow in that time interval.

However, before obtaining proper measurements, the algorithm parameters must be adjusted for each device. These parameters are the uppercase variables shown in Table 2.

Table 2: Parameters and variables in Algorithm 1

Parameters	
Name	**Description**
$HEIGHT\_THRESHOLD$	Defines the minimum number of motion vectors needed in a frame to analyse the movement of a car. This variable should be lower when the camera is farther away from the street.
$WIDTH\_THRESHOLD$	Defines the minimum number of $n\_positive\_frames$ needed to analyse a car.
$GROWTH\_LIMIT$	Defines the limit of the growth variable.
Variables	
Name	**Description**
$growth$	Store the trends in the number of motion vectors between consecutive frames.
$n\_positive\_frames$	Store the number of frames in which the number of motion vectors exceed the $HEIGHT\_THRESHOLD$ parameter and the growth trend is positive.
$car\_detected$	Store if a car has been detected in the current frame.

To adjust the values of these parameters for a given device, a series of input videos have to be stored once the device has been set up in the desired location. Then, the collected videos need to be labelled, indicating the total number of vehicles that have crossed the road in each direction throughout the whole video. Finally, the parameters are adjusted with an optimisation process applied over Algorithm 1, using the previously generated videos as input data. These parameters have been adjusted using a method based on trial/error.

Algorithm 1 allows determining the traffic flow for any previously recorded video. However, this algorithm can be easily modified to be applied in real time. To this end, if each iteration of the loop located in the third line of Algorithm 1 is executed every time a new frame is captured by the device's camera, then this algorithm can be executed in real time on the device. This is the procedure that has been adopted in the IoT device in order to determine the traffic flow at each instant.

An example of the execution of the algorithm is shown in Figure 4. This graph shows the number of motion vectors detected by the device during a one minute period. The device was placed perpendicular to the street and slightly elevated. The blue line represents the number of motion vectors per frame of objects that moved leftward across the street, and the red line represents the same but for those moving from left to right. The black line represents the calibrated $HEIGHT\_THRESHOLD$ parameter for the device. After applying Algorithm 1 we obtain that during this period a total of three vehicles traversed the road leftward, whereas a total of eight vehicles did the same in the opposite direction. To continue with the description of

Algorithm 1: Determine the traffic flow using the motion vectors

Input: For a given direction of the road, the number of motion vectors per frame, mv.
Output: The total number of cars detected in that direction of the road, $n\_vehicles$.

```
/* Initialise variables                                                    */
n_vehicles, growth, n_positive_frames ← 0; car_detected ← False
/* Parameters                                                              */
GROWTH_LIMIT ← 5; WIDTH_THRESHOLD ← 10; HEIGHT_THRESHOLD ← 150
foreach frame in the video do
    /* Calculate the video movement tendency                               */
    if mv[frame - 1] < mv[frame] and growth < GROWTH_LIMIT then
        growth ← growth + 1
    else
        growth ← growth − 1
    /* Count the number of vehicles                                        */
    if mv[frame] >= HEIGHT_THRESHOLD and growth > 0 then
        n_positive_frames ← n_positive_frames + 1
        if n_positive_frames ≥ WIDTH_THRESHOLD and car_detected = False then
            car_detected ← True
            n_vehicles ← n_vehicles + 1
    else if growth = −GROWTH_LIMIT then
        car_detected ← False
        n_positive_frames ← 0
```

this layer, the next section introduces the proposal for AQI estimation.

3.2.2. Computing the AQI based on regression models

The AQI is used by government agencies to assess the amount of pollution in the air. This index collects data about certain essential pollutants and synthesises them into a scale. AQI has an associated set of categories that measure qualitatively the impact on human health. Its computation is based on the pollutants described in Table 3. To calculate the AQI, first a piecewise linear function $L_p(c)$ is defined for each pollutant p. Table 3 shows the thresholds used for each pollutant p, which allows converting a given concentration c_p of a pollutant p to a determinate value of the AQI. The AQI considers the following pollutants: O_3, O_3, $PM_{2.5}$, PM_{10}, CO, SO_2, and Nitrogen Dioxide (NO_2). Table 3 shows under each pollutant the number of hours for which the average concentration of this pollutant must be calculated. The reported AQI is the highest value obtained for the pollutants:

$$\text{AQI} = \underset{p \in \{\text{ pollutants }\}}{\text{Maximize}} \; L_p(c_p) \tag{7}$$

Equation 7 can be rewritten as AQI = L(O_3, $PM_{2.5}$, PM_{10},CO, SO_2, NO_2), showing the functional relation between the output and the input. While the environmental stations can monitor the concentrations of these air pollutants, the Raspberry Pi devices can only monitor a limited set of them. For this reason, two types of regression models are proposed to estimate AQI based on the information that the Raspberry Pi device can collect.

In order to model multivariate time series using the supervised machine learning, the sliding window technique have been used. By moving the window w time steps, the lags that acts like new predictors for

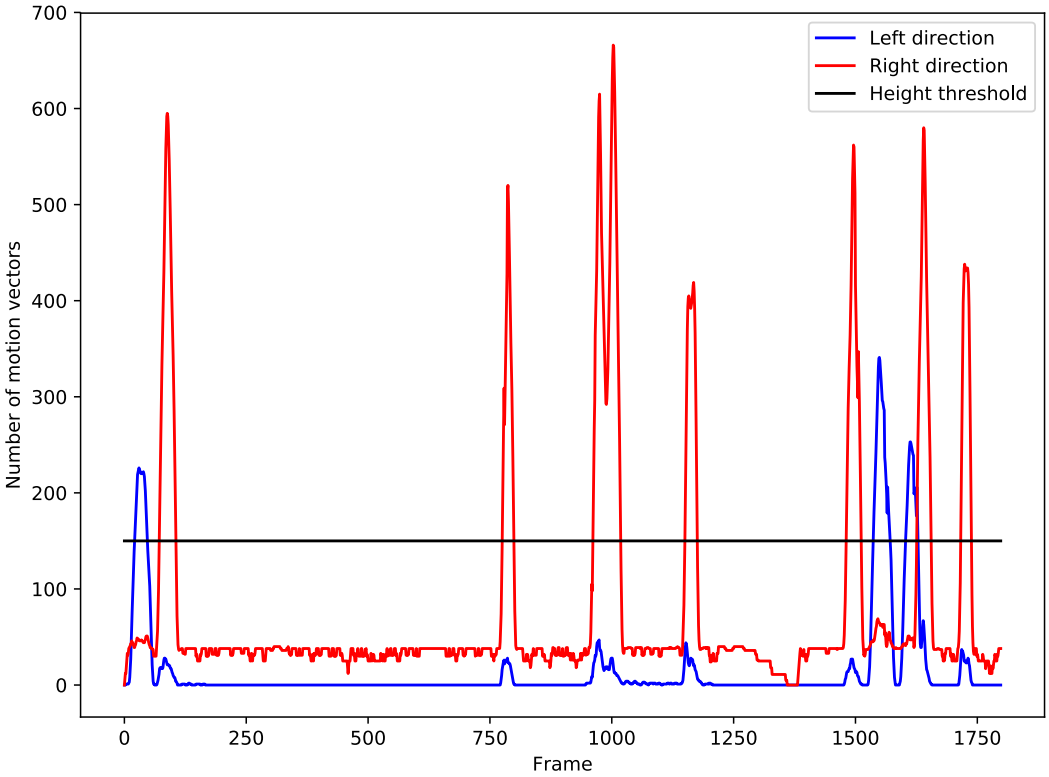


Figure 4: Example of the execution of Algorithm 1

the model appear. We consider the regression model

$$y = f(\mathbf{x}) + \varepsilon, \tag{8}$$

where $\mathbf{x}$ is the vector of lag features $\mathbf{x} = (\mathrm{PM}_{2.5}, \mathrm{PM}_{10}$, temperature, humidity, pressure) and y is the AQI. Note that the dimension of the vector $\mathbf{x}$ is $5w$ where w is the window width. For training these models, we assume that there is available a data-set $\mathcal{Z} = \{(\mathbf{x}_1, y_1) \ldots, (\mathbf{x}_n, y_n)\}$ obtained at n instants.

Three regression approaches are proposed. The first two of them are generic supervised learning methods known as Random Forest (RF) and Gaussian Process Regression (GPR), and the third one is a Linear Regression (LR). RF and GPR approaches outperform conventional multiple regression methods in multiple applications [54]. For instance, with respect to AQI prediction, a more critical concern is memory and storage complexity. Machine learning techniques, such as RF and GPR have a model size that can be consider as reasonably small, but emerging techniques, such as deep learning, often have large a model size, incorporating tens or even hundreds of thousands of parameters. Storing and loading such models on the sensing units will become a bottleneck and that is the reason why RF and GPR have been selected.

Now that the intermediate layer has been described, the next section will describe the highest layer of the proposed architecture.

Table 3: AQI categorisation

		Pollutants						
		O_3 (ppb)	O_3 (ppb)	$PM_{2.5}$ ($\mu g/m^3$)	PM_{10} ($\mu g/m^3$)	CO (ppb)	SO_2 (ppb)	NO_2 (ppb)
	AQI	8h	1 h	24 h	24 h	8 h	1 h	1 h
Good	0-50	0-54	-	0.0-12.0	0-54	0.0-4.4	0-35	0-53
Moderate	51-100	55-70	-	12.1-35.4	55-154	4.5-9.4	36-75	54-100
Unhealthy for Sensitive Groups	101-150	71-85	125-164	35.5-55.4	155-254	9.5-12.4	76-185	101-360
Unhealthy	151-200	86-105	165-204	55.5-150.4	255-354	12.5-15.4	186-304	361-649
Very Unhealthy	201–300	106–200	205–404	150.5–250.4	355–424	15.5–30.4	305–604	650–1249
Hazardous	301-400	-	405-504	250.5-350.4	425-504	30.5-40.4	605-804	1250-1649
	401-500	-	505-604	350.5-500.4	505-604	40.5-50.4	805-1004	1650-2049

3.3. Cloud layer

This section presents the architecture of the components in the Cloud to receive, process (if necessary), and display the data collected from the network of embedded devices. The different stages required are the establishment and setup of an IoT service in the Cloud, the connection of each single device to this service, the setting up of a web page to monitor the data in real time, and, finally, the design of a reporting system to generate alerts.

In this research, it has been decided to integrate the devices with the IBM Watson IoT platform, which is a fully managed, cloud-hosted service that makes it simple to derive value from IoT devices. Message Queuing Telemetry Transport (MQTT) has been used to send the data gathered by each of the Raspberry Pi devices. MQTT is a publication and subscription message transport protocol that has been developed for the exchange of data in real time between sensors and other devices. MQTT, which operates over the TCP/IP protocol, is the principal protocol used by the IBM Watson IoT platform. Once a Raspberry Pi device has been attached to the IoT platform, the platform gathers data from the connected devices and carries out real-time analysis. Figure 5 shows a diagram with the main structure of the MQTT messaging. In this sample, two Raspberry Pi devices collect the temperature and send it to the MQTT service (Publish), while the server states its willingness to receive the temperatures (Subscribe) and therefore receives the temperature data. The next step is to define the configuration file for each of the devices. The parameters to be configured are shown in Table 4.

Table 4: Parameters to configure for each device

Name	**Description**
org	It indicates the organization ID
type	It identifies the type of device which is a grouping for devices that perform a specific task.
id	It is an unique ID that identifies each device which is added to the application.
auth-method	It defines the authentication method. The only supported method by IBM is “token”.
auth-token	An authentication token that allows securely connecting the device to Watson IoT Platform.
clean-session	When it is set to true, the messages are queued while the device is not connected.

Once the platform parameters are set, the different data gathered by each Raspberry Pi device are sent to the MQTT queue. This data is published as an event and stored in an SQL database. Two database tables are required: The data table records all the data received from the Raspberry Pi devices; while the device table stores information from all the different Raspberry Pi devices, such as the time interval in which a sensor is being updated, its position, and the number of lanes in the street where the sensor is placed.

Next, a website has been developed to monitor the real-time data and control all the Raspberry Pi devices. This web page allows the data to be viewed in real time or to display the data collected at a given time interval. Therefore, if the devices are on-line, meaning that they are active at that time, the data displayed corresponds to the current flow of vehicles and the weather conditions at the specific position of

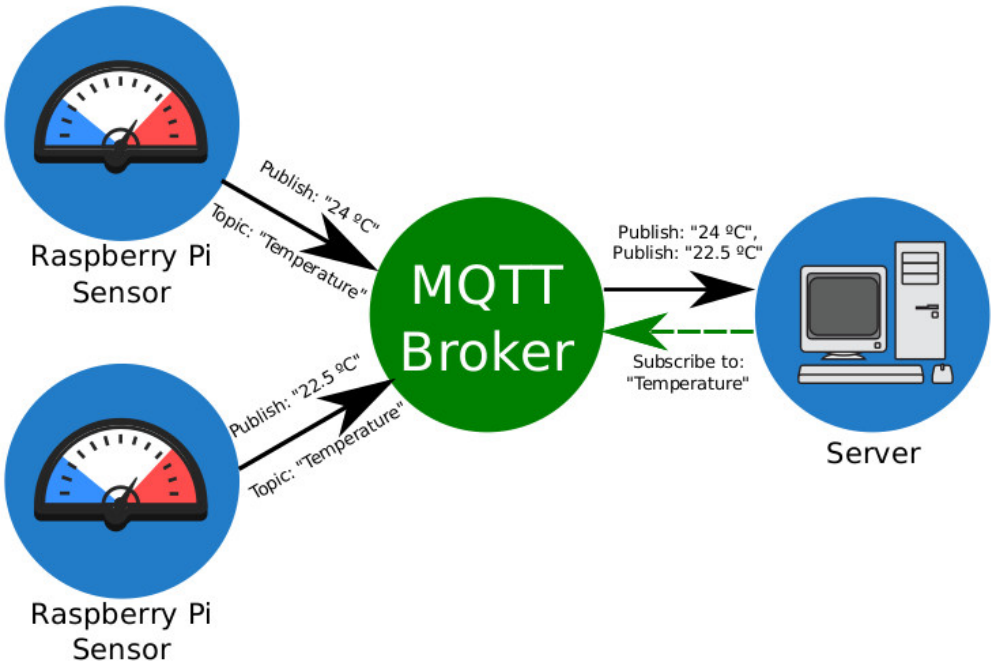


Figure 5: Basic structure of MQTT

the device. The web page provides a list of available devices (Figure 6(a)), from which the user can obtain data. When a device is selected, its data is loaded into the page (Figure 6(b)).

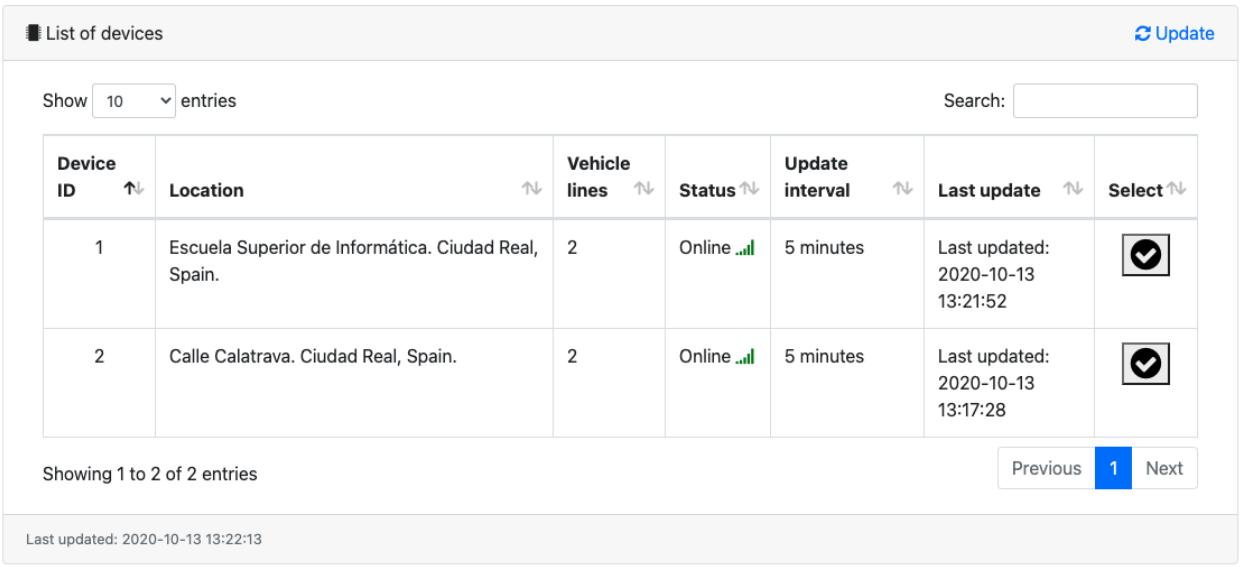
Finally, a reporting system is put in place. Notifications allow users to be alerted by e-mail if a certain parameter (estimated AQI, PM_{10}, $PM_{2.5}$ or vehicle flow) exceeds a certain threshold. Based on this information, which is located in the Cloud, an Intelligent Transport System (ITS) can be deployed to process the information and to take real-time traffic control actions of variable duration in a given area when the pollution levels are excessive. Now that the whole architecture has been presented, the next section will introduce the results from the different experiments that were made.

4. Experimental results

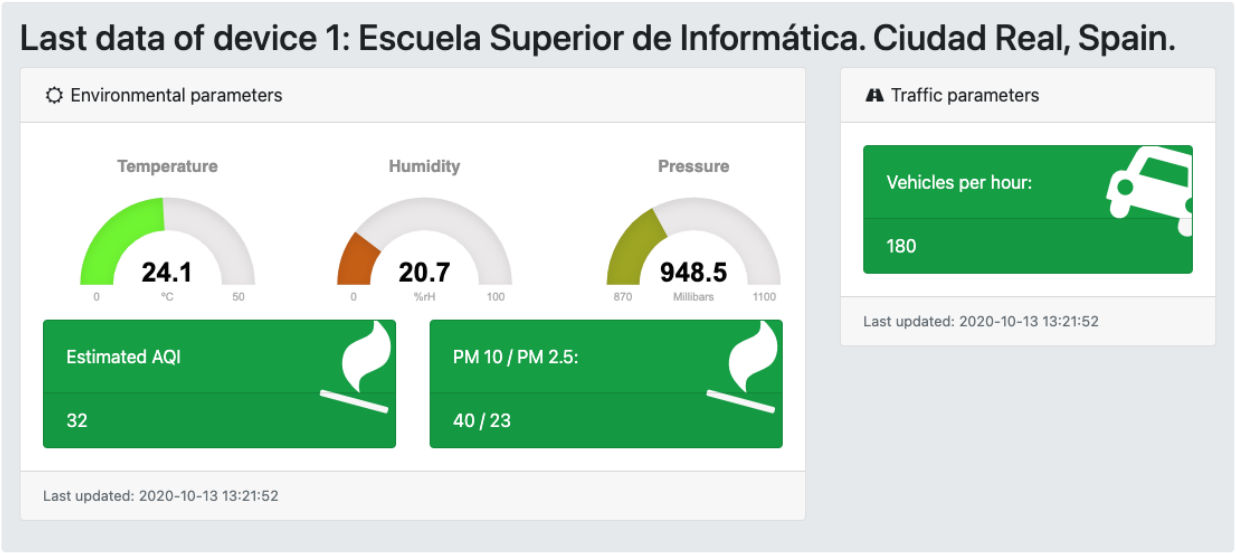
The evaluation of the developed system is through the validation of each of its layers (See Figure 7). The physical layer, dedicated to the acquisition of the raw data, is validated mainly by the configuration process for the correct recovery of data in the Raspberry Pi from the sensors. The cloud layer has been validated through the construction and implementation of the system and the collection of the elaborated data in the Cloud. The fog layer is where the most innovative algorithmic proposals of this work are found, and that is the reason why this section mainly focuses on studying the results of the algorithms for the detection of the vehicle flows (Section 4.1) and for the estimation of the AQI (Section 4.2).

4.1. Determining the flow of the vehicles

Algorithm 1 has been tested using different videos. A set of 23 one minute long videos recorded with the IoT device using different camera angles and distances from the road has been used. The video resolution for all the experiments is fixed to 1080×720 pixels and the frame rate set to 30 frames per second. It must be remarked that the algorithm operates in real time on the Raspberry Pi device, completing each iteration of the main loop in 0.0024 seconds. This is about 14 times faster than the period between frames, which is



(a) Device selection table



(b) Device most recent data

Figure 6: Realtime data subpage

$\frac{1}{30} = 0.033$ seconds. This fact is particularly relevant, since it allows the algorithm to be executed in real time at 30 frames per second on the device.

The results of this experiment are shown in Table 5, where 89.57% of the vehicles were correctly identified by the algorithm. Moreover, it is shown that the algorithm performs with a very low error if the camera is centred, i.e. it looks perpendicularly to the road, as in Figure 8. The main goal is not to obtain the exact link flow at a location, but to obtain a mapping of traffic congestion and to monitor its evolution over time, by comparing the level and distribution of congestion at different points in time. This would allow the identification of the main sources of traffic-related air pollution in the road network. The ITS assesses the level of congestion of the road network using the so-called level of service (LOS), which is measured in

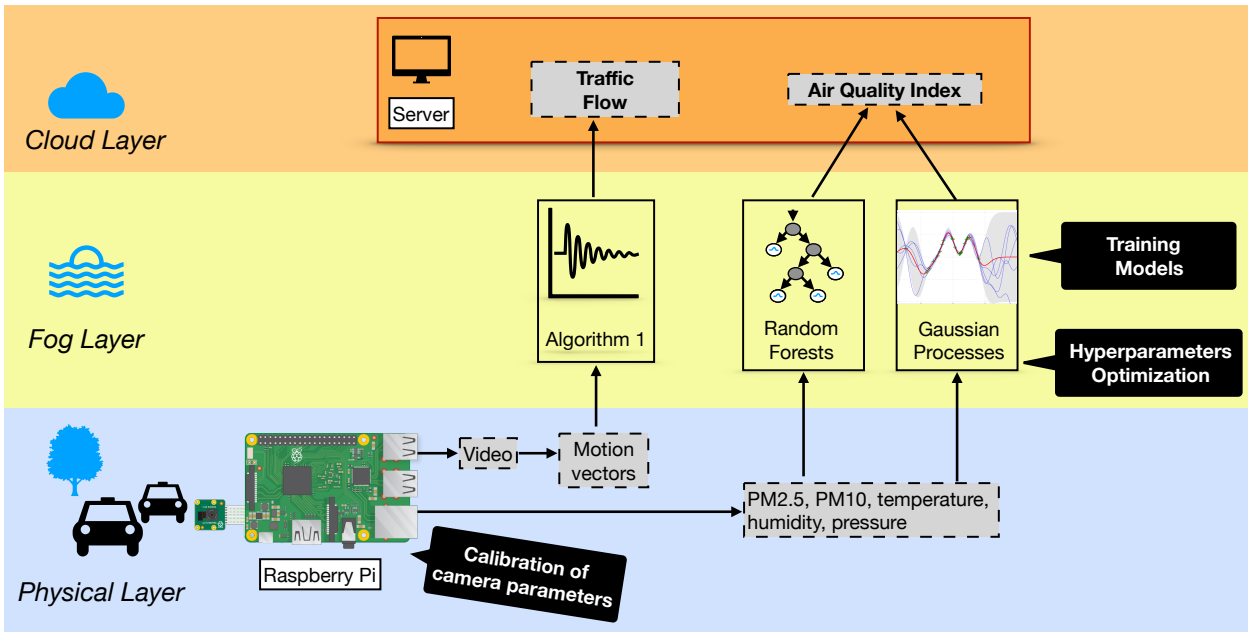


Figure 7: Architecture of the system

a qualitative scale (a grade from A to F). The algorithm obtains an accuracy of more than 90%, and it is sufficient to determine when the threshold values for which congestion is important have been achieved and to obtain dynamically these congestion levels.

As a consequence, the obtained results are successful for the objectives sought. Moreover, the small amount of data provided by the motion vectors allows the algorithm to be executed in real time at low cost with little consumption.

4.2. Validation of the AQI regression models

In this experiment we have validated the AQI regression models that have been implemented on the IoT device. Assuming the reliability of the SDS011 sensor as demonstrated by several studies like [62] and needing large amounts of data to train the model, five one-year sets with data collected from different environmental stations have been employed for the validation. These data-sets use information obtained from the Spanish cities of Madrid, Albacete, and Puertollano, which represent scenarios with different climate and traffic conditions: Madrid is an urban area of more than six million inhabitants, Albacete is a city of about 170,000 inhabitants, and Puertollano is a small industrial city, with a refinery and associated industries of about 50,000 inhabitants. All data-sets consist of environmental parameters, such as temperature, humidity, and pressure, and different pollutant gases, such as O_3, $PM_{2.5}$, PM_{10}, CO SO_2, and NO_2. These gases may vary slightly between the data-sets depending on the availability of the data. All these data-sets contain hourly data for the whole year, which constitutes approximately 8,760 data entries. Using all this information, AQI has been computed for each entry (hour) using Equation 7.

The first data-set corresponds to data collected from downtown Madrid during the year 2019. This data-set has been generated from a data fusion process taking as a source the data provided by the Madrid City Council's open data portal[1]. Two data-sets correspond to data collected in the city of Albacete during the years 2017 and 2018. The other two data-sets correspond to data collected in the city of Puertollano

[1]Madrid air quality and meteorological data: `https://datos.madrid.es/`

Table 5: Computing vehicle flow experiment results

Experiment	Camera angle	Distance to road (m)	Percentage of hits (%)
1	centre	1	**100**
2	centre	1	**100**
3	centre	1	**90**
4	centre	2	**90.91**
5	centre	2	**100**
6	centre	2	**100**
7	centre	2	**94.74**
8	centre	3	**100**
9	centre	3	**90.91**
10	30º left	1	**86.67**
11	30º left	1	**100**
12	30º left	1	**100**
13	30º left	2	**100**
14	30º left	2	**55.56**
15	30º left	2	**73.33**
16	30º left	2	**88.89**
17	30º right	1	**66.67**
18	30º right	1	**78.57**
19	30º right	1	**90**
20	30º right	2	**83.33**
21	30º right	2	**91.67**
22	30º right	2	**88.89**
23	30º right	2	**90**
			Average: 89.57

during the years 2017 and 2018. The data of Albacete and Puertollano have been obtained from the official website of the Castilla-La Mancha regional government[2].

In the first stage, each individual data-set was pre-procesed, removing some entries as they contained errors or non-validated data from the environmental or pollution sensors. Then, to reduce the sensor noise, a 12-hour smoothing was applied to each of the environmental and pollution parameters. Subsequently, all the features of the data-sets were standardised using a standard scaler by setting the mean to zero and scaling to unit variance. This is necessary to later apply GPR, as these methods assume that all features are centred around 0 and have variance in the same order. Once the data have been pre-processed, we proceed to estimate the models.

The standard way of assessing the quality of machine learning models using a single dataset is based on partitioning-based measures such as cross-validation, leave-one-out, random resampling or separated sets. These methods pestered by the biased variance estimations due to dependencies between the samples of examples drawn from the dataset. In this context, in [56], five statistical tests were analysed and authors recommended a cross-validation (CV) technique and a t-test that overcomes the problem of underestimated variance and the consequently elevated Type I error (the probability of incorrectly detecting a difference when no difference exists). This methodology has been applied in this paper but adapted to a time series problem. Hence, a time series cross-validator procedure is used to divide the time series data samples into a train and a test set, but keeping both sets of observations temporally independent from each other. This limits the time-dependent correlations between the two data sets. Therefore, a CV procedure ensures that all data is used both for the training set and for the test set. Specifically, we have used the Scikit-learn function *TimeSeriesSplit* with 12 folds and a time separation of 24 hour between each fold and limiting

[2]Castilla-La Mancha air quality network data: `https://www.castillalamancha.es/node/289605`

Figure 8: Device location for Algorithm 1 testing

the train set to a random subset of 672 training instances (equivalent to one month of data), to reduce the overfitting of the models.

The coefficient of determination R^2 has been used in order to measure the goodness-of-fit of the regression models on training data. This coefficient measures the proportion of the variance in the dependent variable that is predictable from the independent variables. The definition of the coefficient of determination is

$$R^2 = 1 - \frac{\sum_{i=1}^{m} (y_i - f_i)^2}{\sum_{i=1}^{m} (y_i - \bar{\mathbf{y}})^2}, \tag{9}$$

where m is the number of samples, $\mathbf{y}$ is the vector that contains the actual values of the AQI, $\bar{\mathbf{y}}$ is its average, and $\mathbf{f}$ is the vector of predicted values of the AQI. In the best case, the modelled values exactly match the observed values, resulting in $R^2 = 1$. However, a baseline model, which always predicts $\bar{\mathbf{y}}$, will have $R^2 = 0$. Models that have worse predictions than this baseline will have a negative R^2.

To obtain a complementary information on the regression model's performance on the test set on each fold of the cross-validation, the following measures will be used:

- Mean square error (MSE) which is defined as

$$MSE = \frac{1}{m} \sum_{i=1}^{m} (y_i - f_i)^2 \tag{10}$$

- Mean absolute error (MAE) which is defined as

$$MAE = \frac{1}{m} \sum_{i=1}^{m} |y_i - f_i| \tag{11}$$

- Mean relative error (MRE) which is defined as

$$MRE = \frac{1}{m}\sum_{i=1}^{m}\left|\frac{y_i - f_i}{y_i}\right| \tag{12}$$

MSE can be interpreted geometrically as the average fit of points to a regression model. MSE weigh errors proportionally to their magnitude, whereas MAE weighs all errors equally. This makes MSE more sensitive to outliers. MRE is a non-dimensional measure and it is useful for expressing how far estimated values are from the reference values.

The aim of this experiment is to compare the performance of LR, RF and GPR with respect to not having any model, the AQI of each of the test sets has been obtained using only the pollution parameters that can be collected by the IoT device, i.e. applying Equation 7 using exclusively the PM_{10} and $PM_{2.5}$ as parameters.

The first task has been the optimisation of the hyperparameters. For RF, five hyperparameters have been adjusted by using a random search with 100 iterations over a hyperparameter grid: number of trees B = [200:25:2000]; maximum number of features to consider [$\sqrt{\text{`n\_features'}}$, `n_features']; maximum depth D_{max} = [5:10:100]; minimum number of samples required to split an internal node [2, 5, 10, 20]; and minimum number of samples required to be at a leaf node $N_{min} = [1, 2, 4]$. In the case of GPR, it has been estimated using a Matérn kernel function, defined as

$$k(\mathbf{x}_i, \mathbf{x}_j) := \frac{1}{\Gamma(\nu)2^{\nu-1}}\left(\frac{\sqrt{2\nu}}{l}d(\mathbf{x}_i, \mathbf{x}_j)\right)^{\nu} K_\nu\left(\frac{\sqrt{2\nu}}{l}d(\mathbf{x}_i, \mathbf{x}_j)\right), \tag{13}$$

where $d(\cdot,\cdot)$ is the Euclidean distance, $K_\nu(\cdot)$ is a modified Bessel function, and $\Gamma(\cdot)$ is the gamma function. This kernel is parametrised by a length-scale parameter $l > 0$, and a smoothness parameter ν. These hyperparameters have been estimated for each data-set using the Python scikit-learn package [55]. Afterwards, a LR, RF and GPR regressors have been estimated using the same package.

Table 6: Experimental results (R^2)

	Linear Regression		Random Forest		Gaussian Process		IoT Device Sensors	
R^2	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Madrid (2019)	0.5134	0.0655	0.9437	0.0215	0.8722	0.0195	-7.8525	3.5015
Albacete (2017)	0.6451	0.1060	0.8298	0.0800	0.8838	0.0444	-2.2392	1.4914
Albacete (2018)	0.5654	0.1432	0.7918	0.0719	0.8657	0.0345	-3.9551	1.8051
Puertollano (2017)	0.6730	0.1375	0.8471	0.0809	0.8907	0.0444	-3.9875	3.1041
Puertollano (2018)	0.9425	0.0221	0.9724	0.0109	0.9532	0.0093	-1.6406	0.9927

Table 6 shows the coefficient of determination R^2 that measures the goodness-of-fit of the regression models after applying the time series cross-validation procedure on each dataset. For each model, it has been reported the mean and the standard deviation of the R^2 metric. As can be observed, all the values of R^2 are negative, which means that, using only the pollution parameters collected by the IoT device, the results obtained are worse than a model which always predicts the average value of AQI ($\bar{\mathbf{y}}$). It is observed that the machine learning models have R^2 coefficients on training data higher than 0.79, which is higher than the results achieved by LR models. Next, we compare their performance on out-of-sample data.

Tables 7 to 9 show the MSE, MAE, and MRE metrics of each regression model after applying the cross-validation procedure on each dataset. It is observed that all the regression models improve the results of the basic IoT model. A t-test has been applied to all metrics and it is corroborated that this difference is significant in all datasets (p-value < 0.001). There is only one exception, which is in the case of the city of Madrid, where for the MSE metric, LR is not significantly better than the IoT Device Sensors. This is possibly due to the presence of outliers derived from the high variability of pollution in this city. In the case of Madrid, those algorithms are able to reduce the relative error of the IoT device sensors up to 70%.

Moreover, we observe that the performance of LR is comparable to RF and GPR on out-of-sample data. This statement has also been verified using a t-test, and no method has been found to be statistically significantly better than the rest. The only exception is in Puertollano (2018) where GPR is worse than LR or RF (p-value < 0.01). If we compare the results of all the datasets, we see that the worst results are obtained in the city of Madrid and correspond to a worst relative error (MRE) of about 22%, whereas in absolute errors (MAE) is 12 AQI points. Hence, these results shows that this methodology allows to improve the spatial resolution of pollution monitoring system.

Table 7: Experimental results (MSE)

	Linear Regression		Random Forest		Gaussian Process		IoT Device Sensors	
MSE	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Madrid (2019)	1607.5270	4462.8448	246.8756	191.5817	247.8186	215.7923	2081.3605	669.6900
Albacete (2017)	52.9540	51.4298	51.8522	49.7620	60.7418	62.7254	199.4227	114.3951
Albacete (2018)	56.5605	120.7219	53.6083	74.6906	54.7514	94.7250	175.0988	83.9274
Puertollano (2017)	74.5614	55.2234	76.1541	51.6302	106.1074	110.6991	378.0860	117.9598
Puertollano (2018)	7.3283	2.6282	11.1395	8.0552	53.0269	49.0126	391.3366	506.6034

Table 8: Experimental results (MAE)

	Linear Regression		Random Forest		Gaussian Process		IoT Device Sensors	
MAE	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Madrid (2019)	15.9747	10.0048	12.5342	5.0670	12.3677	5.8085	42.8073	7.3094
Albacete (2017)	5.4392	2.6013	5.1297	2.4186	5.7230	2.6381	11.5296	3.0776
Albacete (2018)	4.7366	3.7610	4.9905	3.0186	4.9385	3.2336	10.9142	2.2541
Puertollano (2017)	6.6893	2.6391	6.8821	2.6029	7.8857	3.2465	16.2717	2.6227
Puertollano (2018)	2.1534	0.4692	2.3800	0.8544	5.0678	2.6180	12.2728	3.1883

Table 9: Experimental results (MRE)

	Linear Regression		Random Forest		Gaussian Process		IoT Device Sensors	
MRE	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Madrid (2019)	0.2958	0.2088	0.2252	0.0878	0.2240	0.1074	0.7267	0.0707
Albacete (2017)	0.1515	0.0593	0.1415	0.0514	0.1596	0.0493	0.3180	0.0521
Albacete (2018)	0.1402	0.0683	0.1574	0.0778	0.1497	0.0537	0.3446	0.0537
Puertollano (2017)	0.2163	0.0771	0.2280	0.0718	0.2607	0.1037	0.5010	0.0900
Puertollano (2018)	0.0643	0.0214	0.0700	0.0317	0.1350	0.0551	0.3382	0.0793

To obtain a deeper insight of the behaviour of these regressors, we have trained all the algorithms into a subset of the dataset and used the rest of the data to visualize and compare the predictions of the algorithms with the actual values of the AQI. Hence, for each of the data-sets, we have divided the year in several partitions, as shown in Figure 9. In this procedure, the following pattern was employed: four weeks of training (blue), one week that is discarded (light turquoise) and three weeks of testing (green). This cycle covers a total of eight weeks, so it repeats approximately every two months. This leads to six smaller sets of training and test data for the entire year. However, all six training sets have been combined in order to train each algorithm with data from the whole year. We have chosen this way of systematically sampling and testing throughout the year, as pollution has a seasonal dimension.

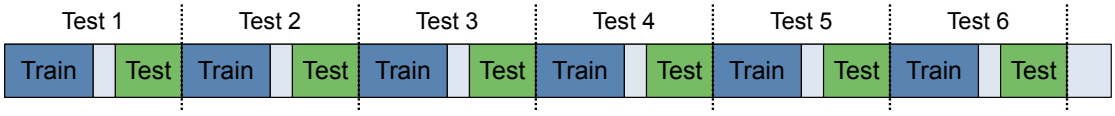


Figure 9: Division of each dataset into training and testing sets

We have shown for each city, the results achieved by the best regressor according to Table 9. Therefore, Figure 10 show the estimation of the GPR on each of the six test sets corresponding to Madrid in the year

2019. Each vertical dotted line represents the end of each of the generated test sets for this year. Note that the training data and the data which were not used have been omitted. The black line represents the actual AQI of the air quality station, whereas the green line represents the AQI value computed using only the pollution parameters that can be collected by the IoT device sensors (Dataset AQI), and the blue line is the AQI value estimated by GPR. The results shows how the algorithm is able to understand the general pollution trend throughout the year and significantly improves the results of the IoT device sensors .

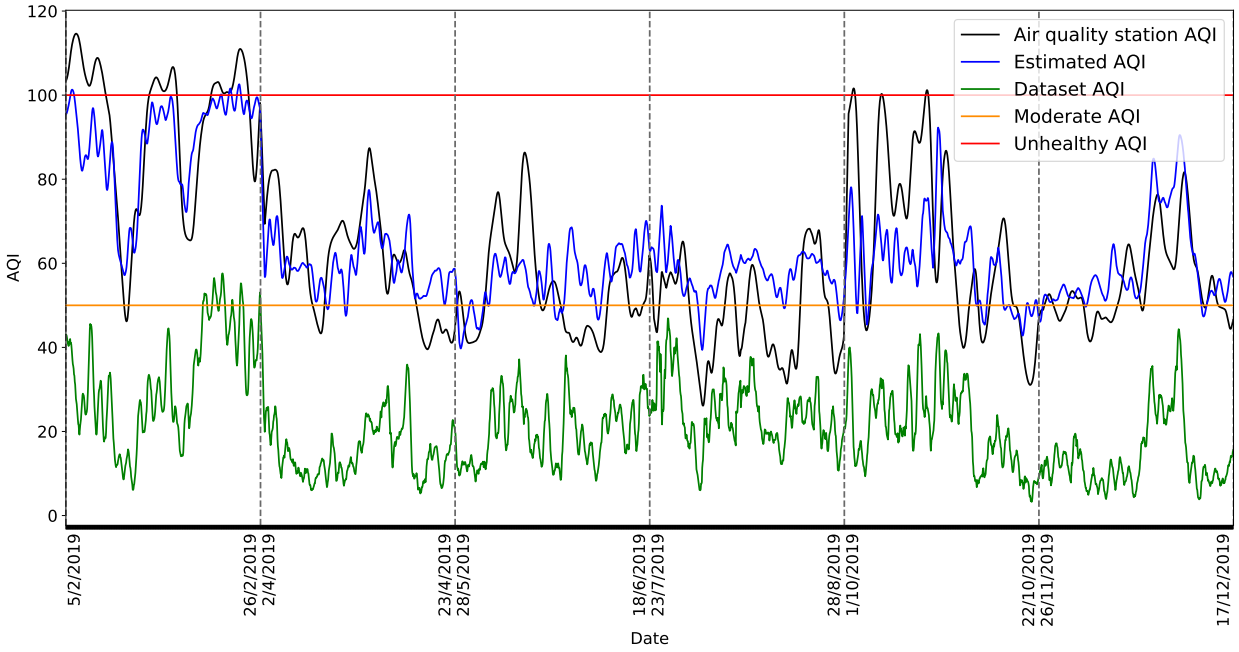


Figure 10: GPR estimation for Madrid 2019

Next, Figures 11 and 12 show the result of applying LR to the datasets Albacete 2018 and Puertollano 2018, respectively. It can be seen that in both cases the LR significantly improve the estimation of the AQI with respect to having no model at all. Although this is not shown, the results obtained with RF and GPR are very similar, revealing the general trend of the AQI value. However, observing the graph, it can be seen that the improvement achieved by the regressors is less significant than in the case of Madrid, because there is significantly less road traffic in these cities, and therefore the pollution is more stable throughout the whole year, producing less peaks or imbalances to be estimated. The case of Puertollano is noteworthy, because it is an industrial city and therefore several pollution peaks are produced that the models are capable of detecting.

It can be concluded from these results that there is the need to incorporate a regression model to the IoT devices because if we limit ourselves to calculating the AQI for those pollutants that can be measured by these devices, we underestimate the levels of contamination (it can be seen that the green line of the graphs is always below the actual value of the AQI). Regarding which of the regression techniques is to be most recommended, no conclusive result has been obtained, noting that RF and GPR models work better for Madrid (a city with much more unstable pollution value), while simple LR seems to be enough for obtaining a good prediction for Albacete and Puertollano.

Now that the core of the experiments have been introduced, we want to clarify two aspects related to the validity or extrapolation of the models in Sections 4.2.1 and 4.2.2.

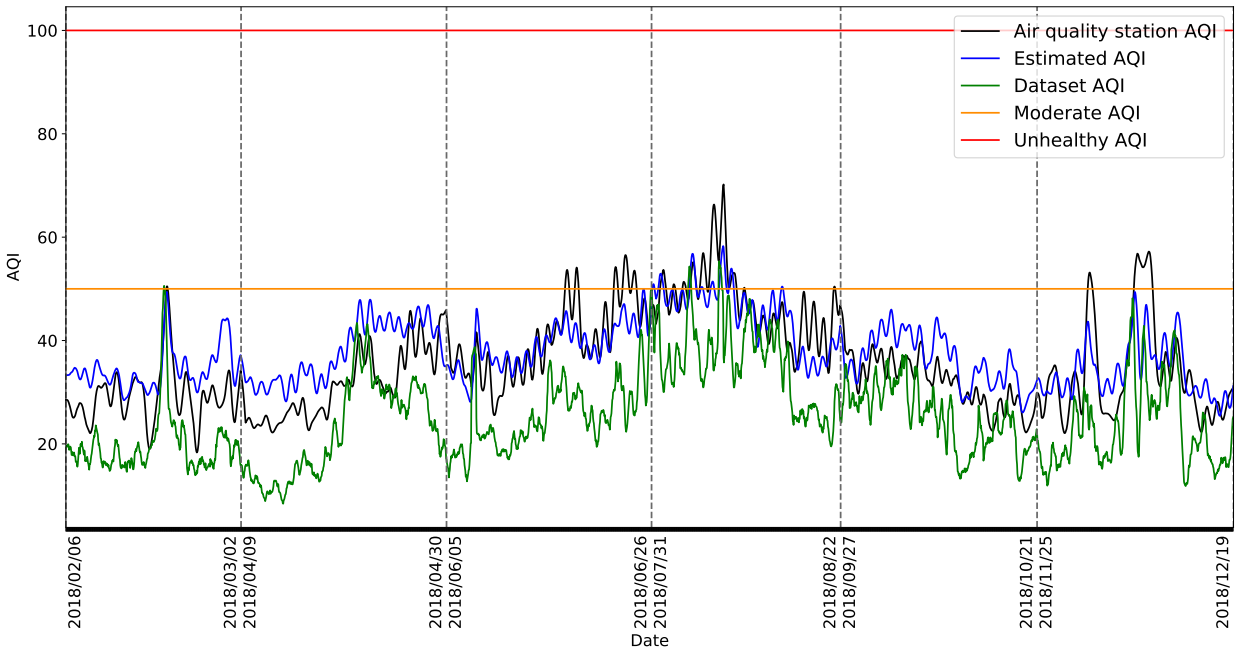


Figure 11: LR estimation for Albacete 2018

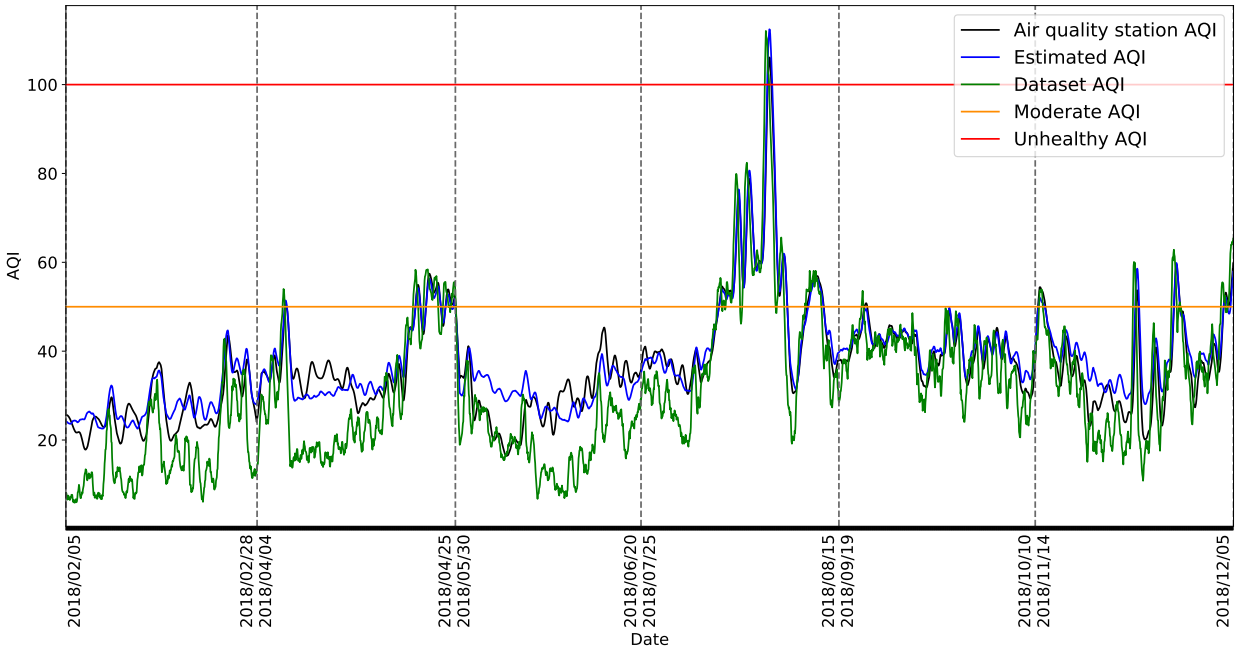


Figure 12: LR estimation for Puertollano 2018

4.2.1. Comparison of the values of AQI obtained from IoT devices with the estimations from environmental stations

The main objective of the system is to establish a monitoring network at a neighbourhood scale. In this experiment, we analysed this question by testing whether the estimates that would be obtained at a specific location of the city of Madrid using an IoT device is better than extrapolating the value of AQI obtained at another environmental station. We have considered two locations of Madrid: the first is located in *Escuelas Aguirre*, near the *Retiro* park. Then, we selected the nearest sensor, located in the *Cuatro Caminos* neighborhood, less than three kilometres away. The results are shown in Figure 13. The true and estimated values of AQI were determined at the first location with the IoT device (black and blue lines, respectively). Then, the data of the second location has been used to estimate the value of AQI for the first location (green line). It can be concluded from these experiments that a higher accuracy is observed in the measurements made by the local IoT device than when considering the value of the adjacent environmental station.

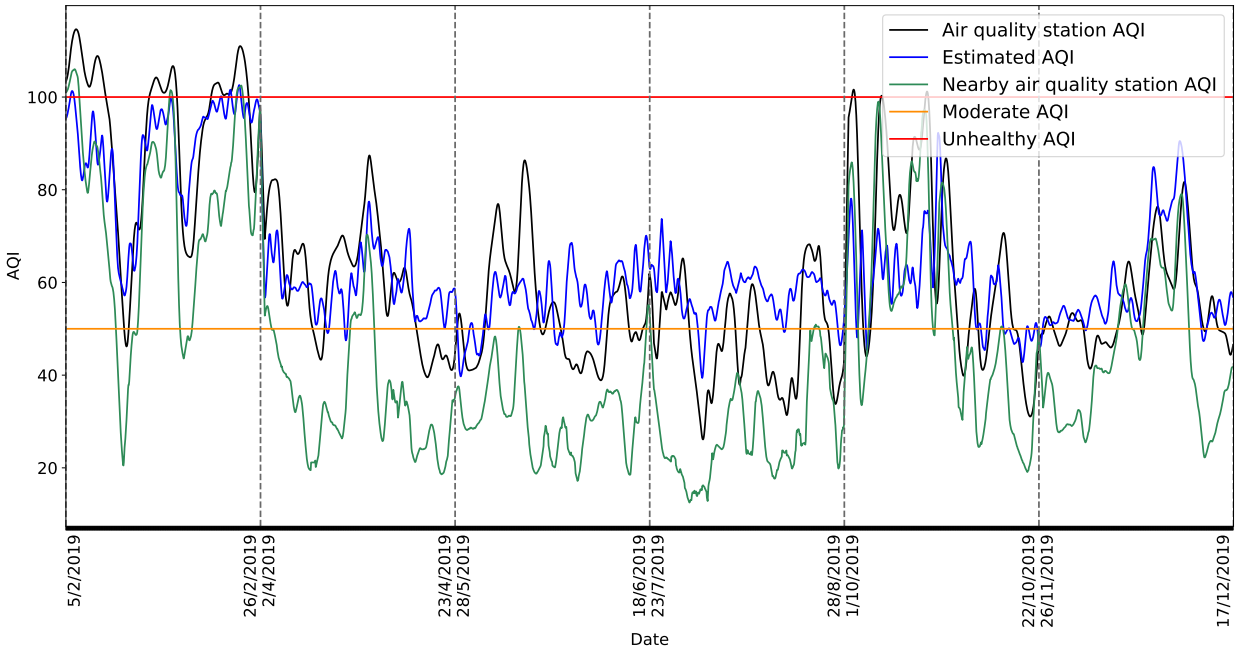


Figure 13: GPR estimation against the AQI obtained by a nearby station for Madrid 2019

4.2.2. A note about the model training

A limitation of the proposed methodology is the need of training the models in their respective locations, requiring mobile environmental stations to collect all the pollutants involved in the calculation of the AQI and the environmental parameters in order to generate the training data. This is reflected in Figures 14 and 15, where the GPR and RF models have been trained using data from the city of Madrid, and then they are used to estimate the pollution levels of Albacete and Puertollano. From these figures it can be concluded that as the environmental conditions of each city are different, and, more concretely, for the concrete case of Madrid, a city with higher pollution levels, when this is applied to other cities one obtains what is clearly an overestimation of AQI.

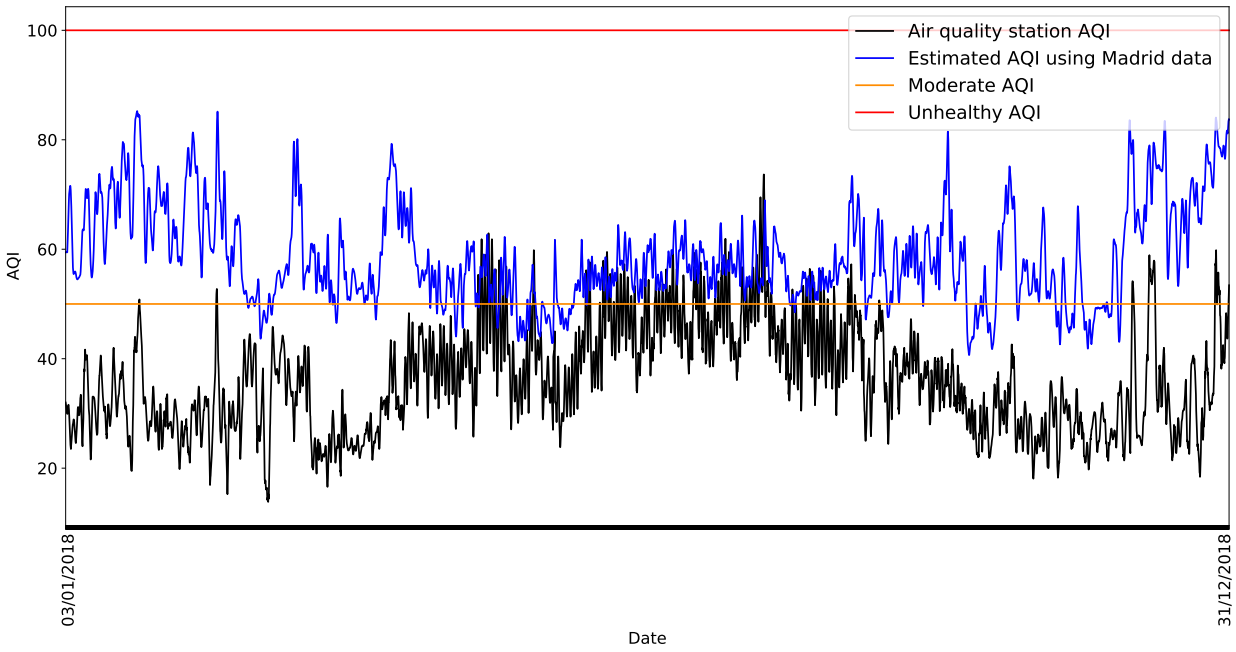


Figure 14: GPR estimation for Albacete 2018 using a model estimated on Madrid 2019

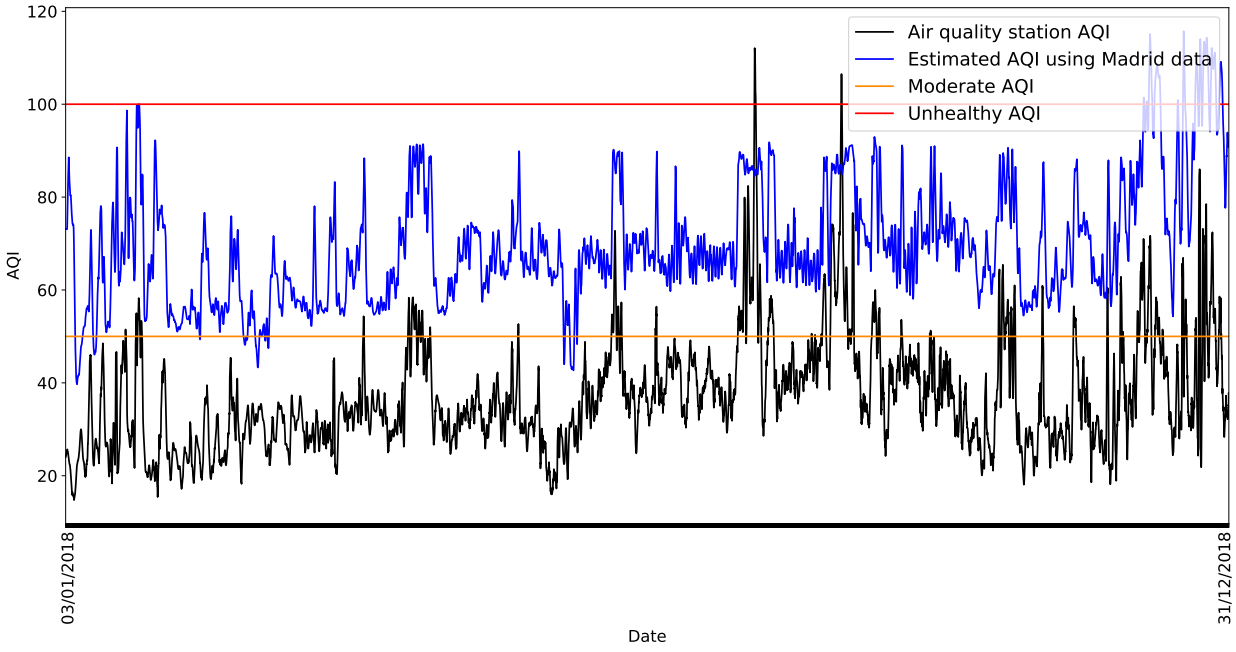


Figure 15: RF regressor estimation for Puertollano 2019 using a model estimated on Madrid 2019

5. Conclusions and Future Work

The main objective of this paper was the design and development of an IoT prototype for pollution and traffic analysis based on a three-layered hierarchical distributed architecture. With its construction, the technical viability of the system has been demonstrated. With respect to the economical viability, the cost of each individual device and its components, including its assembly, is around 150 euros, which makes it feasible to deploy a monitoring network in medium and large cities according to their needs. It must be taken into account that the cost of an unique environmental monitoring stations could vary between 5,000 to 30,000 euros as stated in [58, 60] Furthermore, a great challenge was solved in this paper (more concretely in the Fog Layer): the transformation of the raw data obtained from the devices into the variables AQI and traffic flow. To do this, two tasks have been undertaken:

1. An efficient algorithm to determine the traffic flow in a road link was developed. This algorithm detects correctly about 90% of the vehicles, which makes this solution compatible with the objective pursued in this work. Thanks to the use of motion vectors, which are generated by the H.264/AVC video encoder on the GPU, the number of vehicles that are travelling on a street was counted in real time, while consuming little CPU resources (about 10% of the CPU). The algorithm is executed approximately 14 times faster than the time available between frames. Therefore, the use of this approach allows using an inexpensive embedded system such as the Raspberry Pi.
2. Numerical tests show that direct estimation of the AQI index from the contaminants that Raspberry Pi is capable of monitoring significantly underestimates the value of the actual AQI index. Regression models based on LR, RF and GPR have been proposed to correct this limitation. These models estimate AQI on the basis of temperature, pressure, humidity, PM_{10}, and $PM_{2.5}$. The computational results show that the use of these regression models yields errors of less than 12 points on the estimation of the AQI value, which makes it possible to estimate the AQI using these IoT devices. There is no evidence that one technique outperforms the other, since RF yields the best results for the city of Albacete (2017), GPR for Madrid and LR for Albacete (2018) and Puertollano (2017, 2018). However, we observe that in data with high variability, as in the case of Madrid, GPR and RF obtain the best results.

As conclusions with respect to the cloud layer through the use of cloud tools, such as the IBM Watson IoT Platform, scalability is provided to the solution developed in this paper, which allows using the device massively in urban areas. This is a key goal of such IoT systems in general. Moreover, a web page was built in order to allow the visualization of the data generated by every connected device.

Finally, from our point of view, three lines of future research are necessary, the first one motivated by the fact that the network can include a massive amount of devices, so research should be done on methods that facilitate the calibration of these devices and re-training the calibration function to account for changes in the properties of the sensors, in particular, an automatic parameter tuning for the vehicle detection algorithm should be developed. Using different machine learning and video analysis techniques, the different thresholds and variables of the vehicle counting algorithm could be estimated. Therefore, these values should be calculated automatically when the device is placed in a new location, saving the time needed for trial and error estimation. In addition, techniques must be studied to overcome the limitations of methods based on motion vectors in two specific circumstances, one of them related to changes in lighting and weather conditions [61], and on the other hand, in situations of congestion in which the motion field disappears and segmentation and tracking techniques that work at the pixel level must be used, always choosing those that require less computational load. Another option is to add noise sensors, which would make it possible, when there is a situation where the motion field disappears, to determine whether it is indeed due to a lack of vehicles or to traffic congestion. Furthermore, in order to have much richer information available for the evaluation of this algorithm, it is necessary to introduce more precise metrics that will allow us to identify more directly the situations in which the algorithm does not produce the expected results.

Secondly, research should be done on methods to move from a one-to-one training of regression models to methods that allow all of the underlying regression models of the device network to be trained simultaneously.

This is not only for the sake of computational efficiency, but also for the use of the data and to reduce the costs and time needed for data generation.

Finally, the third line of research that would be interesting to consider is the integration of this monitoring module within ITS. For this integration, methods based on soft-computing and h-step-ahead predictions would be developed to predict the thresholds related to AQI and traffic flows. The fact of being outside these limits would entail the design of future traffic control measures, and more specifically, it would be possible to determine time intervals and the geographical areas over which to implement such measures.

Acknowledgments

Grants TRA2016-76914-C3-2-P and PID2020-112967GB-C32 funded by MCIN/AEI/ 10.13039/501100011033 and by *ERDF A way of making Europe*.

The research of Martín-Baos has been supported by the FPU Predoctoral Program of the Spanish Ministry of Universities with reference FPU18/00802.

References

[1] U.S. Environmental Protection Agency, Office of Air Quality Planning and Standards, Air Quality Index (AQI) Basics, EPA (2016).

[2] T. L. Octaviani, Z. Rustam, Random forest for breast cancer prediction, in: AIP Conference Proceedings, Vol. 2168, 2019. doi:10.1063/1.5132477.

[3] G. Iannace, G. Ciaburro, A. Trematerra, Wind turbine noise prediction using random forest regression, Machines 7 (4) (2019). doi:10.3390/machines7040069.

[4] J. Shen, J. Wan, S. J. Lim, L. Yu, Random-forest-based failure prediction for hard disk drives, International Journal of Distributed Sensor Networks 14 (11) (2018). doi:10.1177/1550147718806480.

[5] M. Z. Joharestani, C. Cao, X. Ni, B. Bashir, S. Talebiesfandarani, PM2.5 prediction based on random forest, XGBoost, and deep learning using multisource remote sensing data, Atmosphere 10 (7) (2019). doi:10.3390/atmos10070373.

[6] A. Dahl, E. V. Bonilla, Grouped Gaussian processes for solar power prediction, Machine Learning 108 (8-9) (2019). doi:10.1007/s10994-019-05808-z.

[7] W. K. Tsang, D. F. Benoit, Gaussian Processes for Daily Demand Prediction, Journal of Forecasting (2019). doi:10.1002/for.2644.

[8] N. Chen, Z. Qian, I. T. Nabney, X. Meng, Wind power forecasts using Gaussian processes and numerical weather prediction, IEEE Transactions on Power Systems 29 (2) (2014). doi:10.1109/TPWRS.2013.2282366.

[9] G. Rafiq, B. Talha, M. Patzold, J. G. Luis, G. Ripa, I. Carreras, C. Coviello, S. Marzorati, G. P. Rodriguez, G. G. Herrero, M. Desaeger, What's new in intelligent transportation systems?: An overview of European projects and initiatives, IEEE Vehicular Technology Magazine 8 (4) (2013) 45–69. doi:10.1109/MVT.2013.2281660.

[10] M. Carrier, P. Apparicio, A. M. Séguin, D. Crouse, The cumulative effect of nuisances from road transportation in residential sectors on the Island of Montreal - Identification of the most exposed groups and areas, Transportation Research Part D: Transport and Environment 46 (2016). doi:10.1016/j.trd.2016.03.005.

[11] D. Jandacka, D. Durcanska, M. Bujdos, The contribution of road traffic to particulate matter and metals in air pollution in the vicinity of an urban road, Transportation Research Part D: Transport and Environment 50 (2017). doi:10.1016/j.trd.2016.11.024.

[12] M. Péres, G. Ruiz, S. Nesmachnow, A. C. Olivera, Multiobjective evolutionary optimization of traffic flow and pollution in Montevideo, Uruguay, Applied Soft Computing Journal 70 (2018). doi:10.1016/j.asoc.2018.05.044.

[13] CITEAIR Project, http://citeair.rec.org/home.html. [Last accessed: 13/10/2021]

[14] J. J. Chung, C. Rebhuhn, C. Yates, G. A. Hollinger, K. Tumer, A multiagent framework for learning dynamic traffic management strategies, Autonomous Robots 43 (6) (2019). doi:10.1007/s10514-018-9800-z.

[15] F. Köster, M. W. Ulmer, D. C. Mattfeld, G. Hasle, Anticipating emission-sensitive traffic management strategies for dynamic delivery routing, Transportation Research Part D: Transport and Environment 62 (2018). doi:10.1016/j.trd.2018.03.002.

[16] R. García-Ródenas, M. L. López-García, M. T. Sánchez-Rico, An Approach to Dynamical Classification of Daily Traffic Patterns, Computer-Aided Civil and Infrastructure Engineering 32 (3) (2017) 191–212. doi:https://doi.org/10.1111/mice.12226.

[17] M. Fellendorf, K. Hirschmann, A toolbox to quantify emission reductions due to signal control, in: Transportation Research Board 89th Annual Meeting, 2010.

[18] S. K. Zegeye, B. De Schutter, H. Hellendoorn, E. Breunesse, Reduction of travel times and traffic emissions using model predictive control, in: Proceedings of the American Control Conference, 2009.

[19] S. Elshout, M. Mahmod, B. Arem, Decision making on short term traffic measures to influence traffic related air pollution, in: the 17th Transport and Air Pollution Symposium and the 3rd Environment and Transport Symposium, 2009.

[20] L. Ntziachristos, Z. Samaras, EMEP/EEA air pollutant emission inventory guidebook 2013, Tech. rep. (2014). doi:10.2800/92722.

[21] A. Zaldei, F. Camilli, T. De Filippis, F. Di Gennaro, S. Di Lonardo, F. Dini, B. Gioli, G. Gualtieri, A. Matese, W. Nunziati, L. Rocchi, P. Toscano, C. Vagnoli, An integrated low-cost road traffic and air pollution monitoring platform for next citizen observatories, in: Transportation Research Procedia, Vol. 24, 2017, pp. 531–538. doi:10.1016/j.trpro.2017.06.002.

[22] A. K. Agarwal, N. N. Mustafi, Real-world automotive emissions: Monitoring methodologies, and control measures, Renewable and Sustainable Energy Reviews 137 (2021). doi:10.1016/j.rser.2020.110624.

[23] A. York Bigazzi, M. Rouleau, Can traffic management strategies improve urban air quality? A review of the evidence, Journal of Transport and Health 7 (2017) 111–124. doi:10.1016/j.jth.2017.08.001.

[24] I. Laña, J. Del Ser, A. Padró, M. Vélez, C. Casanova-Mateo, The role of local urban traffic and meteorological conditions in air pollution: A data-based case study in Madrid, Spain, Atmospheric Environment 145 (2016) 424–438. doi:10.1016/j.atmosenv.2016.09.052.

[25] G. Hoek, R. Beelen, K. de Hoogh, D. Vienneau, J. Gulliver, P. Fischer, D. Briggs, A review of land-use regression models to assess spatial variation of outdoor air pollution, Atmospheric Environment 42 (33) (2008) 7561–7578. doi:10.1016/j.atmosenv.2008.05.057.

[26] K. Karroum, Y. Lin, Y.-Y. Chiang, Y. Ben Maissa, M. El Haziti, A. Sokolov, H. Delbarre, A Review of Air Quality Modeling, Mapan - Journal of Metrology Society of India 35 (2) (2020) 287–300. doi:10.1007/s12647-020-00371-8.

[27] X. Xie, I. Semanjski, S. Gautama, E. Tsiligianni, N. Deligiannis, R. T. Rajan, F. Pasveer, W. Philips, A review of urban air pollution monitoring and exposure assessment methods, ISPRS International Journal of Geo-Information 6 (12) (2017). doi:10.3390/ijgi6120389.

[28] J. P. Lahti, P. Helo, A. Shamsuzzoha, K. Phusavat, IoT in electricity supply chain: Review and evaluation, in: International Conference on ICT and Knowledge Engineering, 2018. doi:10.1109/ICTKE.2017.8259615.

[29] M. Shamshiri, C. K. Gan, K. A. Baharin, M. A. M. Azman, IoT-based electricity energy monitoring system at Universiti Teknikal Malaysia Melaka, Bulletin of Electrical Engineering and Informatics 8 (2) (2019). doi:10.11591/eei.v8i2.1281.

[30] C. Badii, P. Bellini, A. Difino, P. Nesi, Sii-mobility: An IoT/IoE architecture to enhance smart city mobility and transportation services, Sensors (Switzerland) 19 (1) (2019). doi:10.3390/s19010001.

[31] V. A. Memos, K. E. Psannis, Y. Ishibashi, B. G. Kim, B. B. Gupta, An Efficient Algorithm for Media-based Surveillance System (EAMSuS) in IoT Smart City Framework, Future Generation Computer Systems 83 (2018). doi:10.1016/j.future.2017.04.039.

[32] Y. S. Jeong, J. J. Park, IoT and smart city technology: Challenges, opportunities, and solutions, Journal of Information Processing Systems 15 (2) (2019). doi:10.3745/JIPS.04.0113.

[33] P. Gurani, M. Sharma, S. Nigam, N. Soni, K. Kumar, IOT smart city: Introduction and challenges, International Journal of Recent Technology and Engineering 8 (3) (2019). doi:10.35940/ijrte.C5245.098319.

[34] A. Botta, W. De Donato, V. Persico, A. Pescapé, Integration of Cloud computing and Internet of Things: A survey, Future Generation Computer Systems (2016). doi:10.1016/j.future.2015.09.021.

[35] M. Díaz, C. Martín, B. Rubio, State-of-the-art, challenges, and open issues in the integration of Internet of things and cloud computing, Journal of Network and Computer Applications 67 (2016) 99–117. doi:10.1016/J.JNCA.2016.01.010.

[36] J. Shah, B. Mishra, IoT-enabled Low Power Environment Monitoring System for prediction of PM2.5, Pervasive and Mobile Computing 67 (2020). doi:10.1016/j.pmcj.2020.101175.

[37] P. Kalia, M. A. Ansari, IOT based air quality and particulate matter concentration monitoring system, Materials Today: Proceedings 32 (2020). doi:10.1016/j.matpr.2020.02.179.

[38] R. Senthilkumar, P. Venkatakrishnan, N. Balaji, Intelligent based novel embedded system based IoT enabled air pollution monitoring system, Microprocessors and Microsystems 77 (2020). doi:10.1016/j.micpro.2020.103172.

[39] T. H. Nasution, M. A. Muchtar, A. Simon, Designing an IoT-based air quality monitoring system, in: IOP Conference Series: Materials Science and Engineering, Vol. 648, 2019. doi:10.1088/1757-899X/648/1/012037.

[40] H. F. Hawari, A. A. Zainal, M. R. Ahmad, Development of real time internet of things (IoT) based air quality monitoring system, Indonesian Journal of Electrical Engineering and Computer Science 13 (3) (2019). doi:10.11591/ijeecs.v13.i3.pp1039-1047.

[41] R. Sreevas, R. Shanmughasundaram, V. Swami Vadali, Development of an IoT based air quality monitoring system, International Journal of Innovative Technology and Exploring Engineering 8 (10 Special Issue) (2019). doi:10.35940/ijitee.J1004.08810S19.

[42] W. J. Ng, Z. Dahari, Enhancement of real-time IoT-based air quality monitoring system, International Journal of Power Electronics and Drive Systems 11 (1) (2020). doi:10.11591/ijpeds.v11.i1.pp390-397.

[43] L. Rodriguez-Benitez, E. Lara, J. Giralt, J. Moreno-Garcia, J. Dondo, L. Jimenez-Linares, An IoT approach for efficient overtake detection of vehicles using H264/AVC video data, Internet of Things 11 (2020). doi:10.1016/j.iot.2020.100272.

[44] T. Ujiie, M. Hiromoto, T. Sato, Interpolation-based object detection using motion vectors for embedded real-time tracking systems, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Vol. June, 2018. doi:10.1109/CVPRW.2018.00104.

[45] H. G. Al-Majhad, A. Bramantoro, I. Syamsuddin, A. Yunianta, A. H. Basori, A. S. Prabuwono, O. M. Barukab, A traffic congestion framework for smart Riyadh city based on IoT services, International Journal of Advanced Computer Science and Applications 9 (4) (2018). doi:10.14569/IJACSA.2018.090443.

[46] M. Razavi, M. Hamidkhani, R. Sadeghi, Smart Traffic Light Scheduling in Smart City Using Image and Video Processing, in: Proceedings of 3rd International Conference on Internet of Things and Applications (IoT), 2019. doi:10.1109/IICITA.2019.8808836.

[47] A. Riaz, S. A. Khan, Traffic congestion classification using motion vector statistical features, in: B. Vuksanovic, J. Zhou, A. Verikas (Eds.), Sixth International Conference on Machine Vision (ICMV 2013), Vol. 9067, International Society for Optics and Photonics, SPIE, (2013), 245–251. doi:10.1117/12.2051463.

[48] M. Kochlan, M. Hodon, L. Cechovic, J. Kapitulik, M. Jurecka, WSN for traffic monitoring using Raspberry Pi board, 2014 Federated Conference on Computer Science and Information Systems, FedCSIS2014 2 (2014) 1023–1026. doi:10.15439/2014F310.

[49] P. Wei, P. Brimblecombe, F. Yang, A. Anand, Y. Xing, L. Sun, Y. Sun, M. Chu, Z. Ning, Determination of local traffic emission and non-local background source contribution to on-road air pollution using fixed-route mobile air sensor network, Environmental Pollution 290 (2021). doi:10.1016/j.envpol.2021.118055.

[50] N. Hilker, J. Wang, C.-H. Jeong, R. Healy, U. Sofowote, J. Debosz, Y. Su, M. Noble, A. Munoz, G. Doerksen, J. Brook, G. Evans, Traffic-related air pollution near roadways: Discerning local impacts from background, Atmospheric Measurement Techniques 12 (10) (2019) 5247–5261. doi:10.5194/amt-12-5247-2019.

[51] S. Kimbrough, R. Baldauf, G. Hagler, R. Shores, W. Mitchell, D. Whitaker, C. Croghan, D. Vallero, Long-term continuous measurement of near-road air pollution in Las Vegas: Seasonal variability in traffic emissions impact on local air quality, Air Quality, Atmosphere and Health 6 (1) (2013) 295–305. doi:10.1007/s11869-012-0171-x.

[52] L. Zwack, C. Paciorek, J. Spengler, J. Levy, Characterizing local traffic contributions to particulate air pollution in street canyons using mobile monitoring techniques, Atmospheric Environment 45 (15) (2011) 2507–2514. doi:10.1016/j.atmosenv.2011.02.035.

[53] Sense HAT, http://pythonhosted.org/sense-hat/.

[54] C. Hultquist, G. Chen, K. Zhao, A comparison of Gaussian process regression, random forests and support vector regression for burn severity assessment in diseased forests, Remote Sensing Letters 5 (8) (2014). doi:10.1080/2150704X.2014.963733.

[55] J. Hao, T. K. Ho, Machine Learning Made Easy: A Review of Scikit-learn Package in Python Programming Language (2019). doi:10.3102/1076998619832248.

[56] T. G. Dietterich, Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms, Neural Computation 10 (7) (1998) 1895–1923.

[57] J. Mirthubashini and V. Santhi, Video Based Vehicle Counting Using Deep Learning Algorithms, 6th International Conference on Advanced Computing and Communication Systems (ICACCS) (2020). doi:10.1109/ICACCS48705.2020.9074280.

[58] , N. Castell et al., Can commercial low-cost sensor platforms contribute to air quality monitoring and exposure estimates?, Environment International 99 (2017), doi:10.1016/j.envint.2016.12.007.

[59] F. Concas, J. Mineraud, E. Lagerspetz, S. Varjonen, X. Liu, K. Puolamäki, P. Nurmi, S. Tarkoma, Low-Cost Outdoor Air Quality Monitoring and Sensor Calibration, ACM Transactions on Sensor Networks 17 (2) (2021). doi:10.1145/3446005.

[60] E. Lagerspetz et al., MegaSense: Feasibility of Low-Cost Sensors for Pollution Hot-spot Detection IEEE International Conference on Industrial Informatics (INDIN) (2019), doi:10.1109/INDIN41052.2019.8971963.

[61] X. Dai, X. Yuan, J. Zhang and L. Zhang, Improving the performance of vehicle detection system in bad weathers, IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC) (2016), doi:10.1109/IMCEC.2016.7867322.

[62] , H.-Y Liu, P. Schneider, R. Haugen and M. Vogt, Performance assessment of a low-cost $PM_{2.5}$ sensor for a near four-month period in Oslo, Norway, Atmosphere (2019), doi:10.3390/atmos10020041