electronics

MDPI

Article

Distributed Fog Computing for Internet of Things (IoT) Based Ambient Data Processing and Analysis

Mehreen Ahmed [1], Rafia Mumtaz [1], Syed Mohammad Hassan Zaidi [1], Maryam Hafeez [2,*], Syed Ali Raza Zaidi [3] and Muneer Ahmad [4]

[1] School of Electrical Engineering and Computer Science (SEECS), National University of Sciences and Technology (NUST), Islamabad 44000, Pakistan; mahmed.phdcs17seecs@seecs.edu.pk (M.A.); rafia.mumtaz@seecs.edu.pk (R.M.); drzaidi@seecs.edu.pk (S.M.H.Z.)
[2] Department of Engineering and Technology, School of Computing and Engineering, University of Huddersfield, Queensgate, Huddersfield HD1 3DH, UK
[3] School of Electronic and Electrical Engineering University of Leeds, Leeds L2 9JT, UK; s.a.zaidi@leeds.ac.uk
[4] Department of Information Systems, Faculty of Computer Science & Information Technology, Universiti Malaya, Kaula Lumpur 50603, Malaysia; mmalik@um.edu.my
* Correspondence: m.hafeez@hud.ac.uk

Received: 24 September 2020; Accepted: 19 October 2020; Published: 22 October 2020

Abstract: Urban centers across the globe are under immense environmental distress due to an increase in air pollution, industrialization, and elevated living standards. The unmanageable and mushroom growth of industries and an exponential soar in population has made the ascent of air pollution intractable. To this end, the solutions that are based on the latest technologies, such as the Internet of things (IoT) and Artificial Intelligence (AI) are becoming increasingly popular and they have capabilities to monitor the extent and scale of air contaminants and would be subsequently useful for containing them. With centralized cloud-based IoT platforms, the ubiquitous and continuous monitoring of air quality and data processing can be facilitated for the identification of air pollution hot spots. However, owing to the inherent characteristics of cloud, such as large end-to-end delay and bandwidth constraint, handling the high velocity and large volume of data that are generated by distributed IoT sensors would not be feasible in the longer run. To address these issues, fog computing is a powerful paradigm, where the data are processed and filtered near the end of the IoT nodes and it is useful for improving the quality of service (QoS) of IoT network. To further improve the QoS, a conceptual model of distributed fog computing and a machine learning based data processing and analysis model is proposed for the optimal utilization of cloud resources. The proposed model provides a classification accuracy of 99% while using a Support Vector Machines (SVM) classifier. This model is also simulated in iFogSim toolkit. It affords many advantages, such as reduced load on the central server by locally processing the data and reporting the quality of air. Additionally, it would offer the scalability of the system by integrating more air quality monitoring nodes in the IoT network.

Keywords: air monitoring; internet of things; distributed fog computing; air quality; outlier detection

1. Introduction

Monitoring air quality is important, as it directly influences human health and safety. Air quality management within the populated urban cities of the developing countries presents a challenge for the government and public. According to the World Health Organization (WHO) reports, there have been 59,241 deaths over the years due to air pollution, amongst which 13,683 are children [1]. Table 1 shows the Air Quality Index (AQI) values. Table 2 details the harmful gases, sources, and the effects they

have. For air quality monitoring applications, low cost sensors that measure temperature, humidity, and air pollutants have been a popular choice.

Table 1. AQI values [2].

AQI Values	Levels of Health Concern
0–50	Good
51–100	Moderate
101–150	Unhealthy for Sensitive Groups
151–200	Unhealthy
201–300	Very Unhealthy
301–500	Hazardous

Table 2. Sources and Effects of Pollutant Gases.

Gas	Source	Effect
Carbon Monoxide (CO)	Motor Vehicles	Weakens Heart, Lungs and Brain , Produces Ozone
Sulphur Dioxide (SO_2) and Nitrogen oxides (NO)	Traffic and Transport, Forest Fires	Respiratory and Cardiovascular illnesses, Contributes to Global Warming and Smog, Acid rain
Carbon dioxide (CO_2)	Combustion	Contributes to Global Warming
Ozone (O_3)	UV light sources, Traffic	Respiratory diseases, Contributes to Smog
Ammonia (NH_3)		Produces particulate matter
Particulate Matter (PM)		Haze , Lung problems
Heavy Metals	Traffic and Transport	Brain problems and other toxic diseases

However, most of these applications utilized an architecture where data collected by sensors are fed into cloud. The cloud computing infrastructure provided capabilities to implement inference and visualization capabilities. Sensor readings are forwarded to cloud using either smart phones or dedicated gateways. Cloud technologies involve significant latency and they are not suitable for time sensitive applications. Furthermore, the cloud paradigm has been constrained by data privacy issues, such as for health monitoring and home devices. Therefore, the solutions that Cloud computing approach could provide are becoming limited in the backdrop of growing needs of the present. The integration of Cloud Computing and Internet of Things (IoT) has led to 'Cloud of Things' (CoT) [3–6] that deals with the processing of huge amount of data. For instance, temperature, humidity and Air Quality sensors deployed indoor can contain side information about patterns of life. It helps in managing the IoT resources and produce cost effective services. However, the biggest challenge for CoT is data trimming [7]. Because the IoT devices generate large amounts of unfiltered data, CoT fails when it comes to time sensitive decision making. Additionally, this inevitably raises privacy concerns at the user end, in particular where the shared data originate from residential/work places [8]. These time and latency sensitive real time applications demand rapid response and processing. For these type of services, it is impractical to communicate over a distant cloud while providing the quality of service guarantees. As anything can be connected to the network, this will require the efficient monitoring of the utilization of resources. Additionally, for quick processing, the decision on what amount of data to send over without burdening the cloud and the core network is important. The limitations of Cloud computing are addressed by the emerging Fog computing paradigm. In particular, fog computing addresses the inefficient usage of network bandwidth, unnecessary communication between devices and the cloud, and lastly the reliance on a single point of failure. Fog computing offers benefits of flexible and self-adaptive data analytics near the IoT end node. These capabilities also enable better privacy preservation by employing approaches of data anonymization/randomization and differential privacy [9]. Lightweight and secure data analytics

schemes can be implemented on fog devices, which can ensure security, privacy, and efficiency of the end-to-end operation.

Fog computing [10,11] extends the traditional Cloud Computing by acting as a medium between the IoT user end and the Cloud. In contrast to the centralized cloud, Fog computing follows a distributed approach. It also provides services of preprocessing, including data trimming, by filtering the data to be consumed locally and managing resources. It complements the Cloud to the edge of the network. Fog computing is provisioned over intrinsically heterogeneous physical resources and it supports the distributed deployment of applications. The characteristics of Fog computing include rapid response to time critical data, flexibility of task distribution among large number of nodes, and ability to analyze heterogeneous data in near real time.

An important IoT application that can leverage the power of fog computing is the air quality monitoring pertaining to the growing concern regarding the increasing level of air pollution across the globe. The implications of rising level of air pollution are extremely detrimental in nature and they pose a serious threat to environment and human health. The factors contributing in soaring the air contamination majorly includes rapid urbanization & industrialization, growing number of vehicles, large use of fossil fuel, and elevated living standards. Owing to these reasons, the AQI of the developing countries falls in the "unhealthy" category according to World Health Organization (WHO) standards [1]. The air quality below WHO standards not only affects human health, but also have an adverse impact on the agriculture sector, impeding the growth of staple crops. The traditional air quality monitoring systems comprise of hefty and expensive monitoring stations. Because of the static nature of these monitoring units, the data collection is sparse and limited. However, as of today, the alarming air pollution levels introduce the need for more monitoring station (mobile and stationary) to be placed frequently spatially when considering the extent of a given urban and industrial center. For such a setup, IoT based air quality monitoring nodes can be installed, which can report the air quality status in real time. However, these nodes would generate large volume of data, which would not be feasible to manage locally in term of storage and computation power. Additionally, some percentage of data may contain anomalies and should be removed prior to data analysis. To this end, we propose an IoT enabled air quality monitoring system with distributed Fog computing for data processing and resource optimization. The proposed distributed fog architecture can help in filtering the data and only sending relevant and selective data to the cloud. This arrangement would greatly reduce the data traffic to the cloud, thereby reducing space and bandwidth requirements.

The paper is organized, as follows. Section 2 covers the related work pertaining to various fog computing and allied applications. In the subsequent section, the proposed approach for air quality monitoring is discussed along with the high level architecture. Section 4 discusses the methodology comprising various stages such as data acquisition, processing and analysis. The results generated by the implementation of fog computing algorithm are discussed in Section 5, where, Section 6 describes the conclusion drawn along with potential research directions.

2. Related Work

Fog computing technology can be used for applications which fall under the umbrella of smart cities. Several studies have leveraged Fog computing capabilities for different use cases [12]. In [13] K. Hong et al., discussed Mobile Fog as a programming model of the future large-scale and latency sensitive internet applications. This model processes data at low latency near the edge of the network and performs large scale processing on the Cloud. Mobile Fog model calls event handlers and functions and in turn reduces the network traffic. One example of such large-scale, latency sensitive applications include the Vehicle Tracking System that can be used by the police to track vehicle while using cameras installed on the highways.

J. Zhu et al. [14] applied traditional web optimization methods in a Fog computing architecture in order to improve the performance of web pages. Their approach improved the rendering of web pages beyond that achieved by applying the methods on the web servers. With the proposed approach,

users connect to the Internet through the edge servers. The user requests are passed onto the edge server first and then passed onto the web servers or the core network where files are locally cached. However, security and reliabilit is still a concern for this technology.

The applications of Fog computing, like Smart Traffic Lighting System, Smart City [15] and Smart Homes [16], are the need of the future. With the advancements in transportation and the advent of Connected Vehicles, Smart Traffic Lightning System [17] will prevent accidents and maintain the traffic flow by collecting the information gathered from the sensors that measure the distance, speed, and presence of an object coming from each direction. The key requirements for such real time analytics can be utilized by the Fog computing environment. Therefore, Shi, Weisong, et al. [18] stated that to build a perfect Smart Home, Fog computing is the key, as connecting the floors, walls, pipes, and rooms with cheap wireless sensors will produce a great amount of data that need to be consumed locally with an edge gateway between the home and the cloud. This will ensure the privacy of the data and give a secure environment for living not only for a single home, but also at city level. For a Smart City, Fog computing may be the ideal platform. For public transport, health and safety solutions, a centralized cloud network is unrealistic. Distributed data can be geographically handled with fog nodes or gateways for better processing.

Particularly for time sensitive applications, like Connected Health, which has health care applications like where falls of stroke patients can be detected [19]. Recently, applications for cognitive assistance of mental patients [20] are invented by using sensors and applying fog computing. All of these applications require quick responses. Fog computing technology is very promising, but it may be a bumpy ride to achieve the right outcome, as we have to overcome the challenges of programmability models, power consumption, enabling real time analytics, and providing privacy or security [21,22].

For applications of air quality monitoring, low cost sensors have been popular for quite some time where low cost mobile sensors sample temperature, humidity and air pollutants, such as Carbon Monoxide, Nitrogen Oxide, Methane, Hydrogen Sulfide, Ozone, Ammonia, Particulate Matter, and Sulphur Dioxide [23–25]. However, such applications have mostly utilized cloud computing so far where sensed data are sent directly to the cloud using smart phones. This data is then further analyzed on the cloud to extract further knowledge. Yun Cheng et al. [26] designed a novel air quality monitoring cloud system that gathered the PM values with sensors and the data captured were used to train models on the cloud while using machine learning techniques. They achieved an accuracy of 64.1% with Artificial Neural Network and 81.7% with Gaussian Process (GP) inference model. The data collected are large enough that some studies have applied big data analytics on the sensed data. Pei-Lun Sun et al. [27] stored and processed the sensed data on the cloud using HBase that stores large amounts of data instead of using the traditional relational database. Santos, José, et al. [28] proposed applying fog computing to monitor the air quality. The authors proposed to apply BIRCH algorithm for clustering the collected data and apply Robust Covariance for detecting anomalies in the data. Their approach identified the anomalies as unhealthy locations where the air was polluted during a certain time period. However, the proof of their methodology was supported by the data that were collected for only a month using two Postal service cars.

The techniques discussed above use the fog computing environment. We propose the fog computing paradigm as a solution as it locally stores the relevant information for analysis. It sends data to the cloud for further processing and discards other unimportant information. Next, we shall demonstrate the need of an air monitoring system combined with the distributed fog computing platform.

3. Fog Computing in IoT

With the increase in IoT implementations, problems and challenges that are associated with them are also increasing. IoT enabled devices are an integral part of our businesses but as the IoT data requirements grow, more devices come online. As of today, IoT devices generate huge amount of heterogeneous data at a high rate that needs to be processed rapidly in a very short amount of

time. The cloud has become an indispensable part of that process and IoT devices rely on it for computational, analysis and storage purposes. But it is also a fact that the existing cloud models are not compatible to the large volume, wide variety, and high velocity of data that are generated by IoT devices.

However, the cloud being centrally deployed on a global scale needs to process an enormous amount of data and this creates many challenges for IoT applications. These challenges include insufficient bandwidth, increase in the physical distance between the user, and the cloud that induces transmission latency, in turn increasing the response time, security issues due to failing data protection mechanism, and the requirement of high speed internet connectivity. The IoT gateways alone are not equipped to solve these problems as they offer too narrow range of functionalities. On top of that, the processing speed in an IoT environment is largely dependent on the performance of the user device. To cater for the problems of the completely centralized system, the solution is to somehow distribute the computing power, storage, and networking resource for IoT, to a cloud like infrastructure that is closer to those end points. This cloud like infrastructure is known as Fog computing [10,29], which is a data centric distributed computational platform that descends from the cloud to the Edge.

The fog in Fog computing means centralizing the computing power closer to the Earth where the devices generating the data are located. The data generated by the IoT devices are usually large in volume and much time is lost until the data are transmitted on the Cloud for further processing and analysis. Fog is a layer between the Cloud and the IoT devices (see Figure 1), which provides communication, processing, and an intermediate storage. It acts as a proxy for the Cloud offering minimal latency, as it is more closer to the end user. Fog is faster, as it operates near the network edge rather than on the cloud. The sensor devices and the cloud can have several fog nodes between them. Examples of Fog nodes include switches, routers, industrial controllers, video surveillance cameras, and embedded servers.

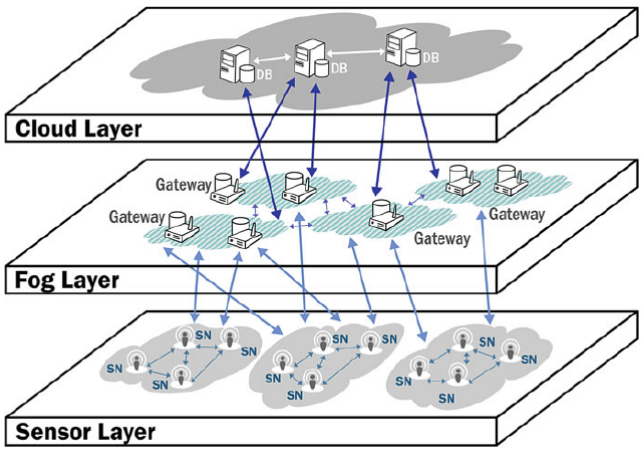

Figure 1. High-level architecture of Fog Computing [30].

3.1. Advantages of Fog Computing

In an IoT environment, it is important to understand the relationship between Fog and cloud computing and assess the impact of fog computing on the IoT service delay and quality of service. Fog computing paradigm offers several features, which make it suitable for IoT enabled systems. These include low latency due to close proximity of the fog services near the network edge, conservation of network bandwidth due to transmission of selective and filtered data, and increased response time due to shorter physical distance between data source and fog nodes relative to the cloud.

- Low Latency: One of the key benefit of Fog is the the low end-to-end latency with edge devices minimizing the service delay for end user IoT applications, in contrast to the cloud, which delivers large scale computation and storage capabilities with greater latency.

- Reduced Data Volume: The load on the central server or the cloud is decreased with the introduction of fog nodes. The fog nodes filter the data, which, in turn, decreases the amount of work the cloud has to perform and also reduces the network traffic received at the cloud.
- Improved Response Time: The data generated from IoT sensors are processed at a much greater pace as compared to sending the data directly to the cloud for centralised processing. Later, the data can be sent by the fog nodes to the cloud for further processing. Moreover, fog computing offers improved response time, which will be useful real-time applications for enhanced user experience.
- Scalability: If the data generated by IoT devices are constantly fed to cloud without any data filtration then the Cloud may limit the scalability of IoT applications, although it has virtually infinite resources. With Fog, the data adjacent to the data source reduce the level of data processing on the cloud and allows more end devices to be integrated into the IoT system.
- Conservation of Network bandwidth: Fog computing lowers the communication between the sensors and cloud by decreasing the bandwidth required. This has a direct impact on the IoT performance. Filtered or selected data reache the cloud for further analysis to preserve the network bandwidth and increase system efficiency.

3.2. Fog Computing in the Context of Air Quality Monitoring Using IoT

The proposed IoT and fog based air quality monitoring system facilitates measuring the levels of different air pollutants and provides filtered data for analysis. The conventional air quality systems have monitoring stations that are static and sparse in nature. However, the proposed system encompasses several IoT nodes that will produce a large volume of data with high frequency. The proposed system will filter the irrelevant data using pre-processing and clustering techniques at the fog layer in order to consume less space on the cloud layer and reduce the bandwidth requirements. This approach will produce a more efficient and effective air quality monitoring system that is computationally more reliable as the superfluous data will be discarded, saving processing time and space at the cloud end.

4. Proposed Distributed Fog Computing Based Air Quality Monitoring System

The proposed air quality monitoring system that is based on fog computing consists of a three tier architecture encompassing sensor layer, fog computing layer, and a cloud layer. Figure 2 shows a high level conceptual overview of this layered paradigm.

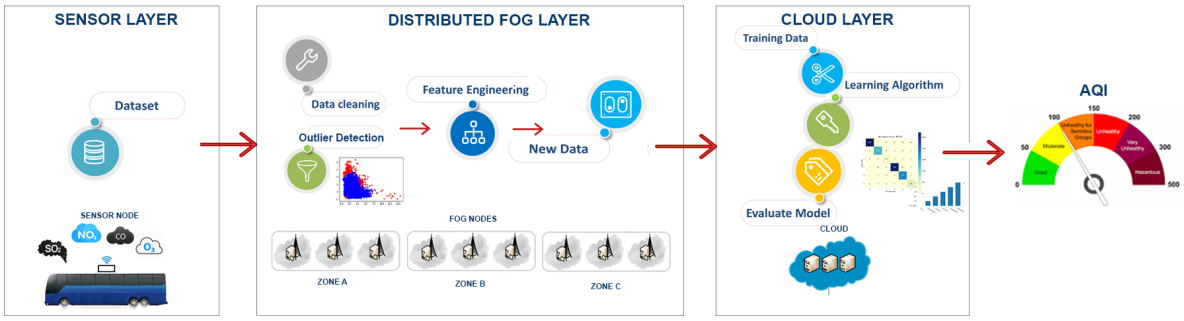

Figure 2. High Level System Architecture.

4.1. Sensor Layer

The sensor layer involves the development of IoT nodes comprising of micro-controller and various sensors for measuring the air pollutants, such as CO_2, NO_2, O_3, SO_2, and CO. The micro-controller is connected to a communication module, such as GSM or Wifi, for sending data to the fog layer. In the sensor layer, the arrangement of IoT nodes can be stationary as well as mobile, depending on the spatial extent of urban center to be monitored and the coverage required.

For measuring the pollutants with high spatio-temporal resolution, sensors can be placed on mobile platforms, such as metro buses. These buses follow a schedule, and run across the city throughout the day and can provide ubiquitous coverage for the air quality data. With sensors placed on these buses, an enormous amount of data can be generated with high frequency. In addition to mobile nodes, stationary nodes can be deployed at those sites that are not covered by the bus routes. The data generated by this layer will be sent to fog layer for pre-processing before it hits the cloud for long-term storage and advanced analysis.

4.2. Distributed Fog Layer

In this layer, the fog nodes will be spatially distributed zone wise to effectively process the air pollutants' data sent by the sensor layer. The reason for zonal placement of fog nodes is primarily the load distribution of processing the data generated by the sensor layer at high frequency. It is important to understand that every fog node is constrained by its power and size limitations and placing too much on it may overwhelm its resources. Additionally, in the case of mobile IoT nodes, the data can be sent to the nearest fog node for data processing and primitive analysis. The zonal fog nodes placed at different selected geographical locations will wirelessly receive the data from the sensor layer either through WiFi or GPRS technology.

The data that are received by fog layer may contain anomalies that are filtered using data pre-processing and clustering techniques and the clean/relevant data is sent to the cloud layer. This topology/arrangement greatly reduces the data traffic to the cloud, thereby reducing bandwidth and space requirements. The raw data that represent the air pollutants' concentration are converted to Air Quality Index (AQI) defined by United States Environmental Protection Agency (EPA) [31] and, subsequently, the hourly average of AQI data is computed. In order to filter the data and detect outliers, clustering techniques like K-means is applied on the samples of the dataset. The outliers can be removed at the fog nodes and only selective and useful air quality data are sent to the cloud for further analysis. The filtered data, which are less in volume than before, eventually reach the cloud with a reduced size, thus optimizing the space requirement. The proposed distributed fog nodes may consume more resources, but will eventually speed up the processing and computation.

4.3. Cloud Layer

The cloud layer receives the data from the distributed fog nodes for long-term storage and advanced data analytic. The historical data can be used for classification, time series analysis, predictive modeling, and identifying hidden trends or patterns. For classification, the air quality data that are based on EPA defined AQI ranges are classified using supervised learning algorithms, such as Artificial Neural Networks (ANN), Multi layer Perceptron (MLP), K Nearest Neighbor (KNN), Naive Bayes (NB), Decision Trees (DT), and Support Vector Machines (SVM), etc. For predictive modeling, machine learning algorithms and regression analysis are used. In the next section, the methodology for the proposed system is discussed in detail.

5. Methodology

This section explains various stages that are involved for processing the data from its source (sensor layer) to the destination (cloud layer) and the intermediary steps performed at the fog layer. Figure 3 describes data processing applied at the fog layer and, thereafter, the classification of air quality data at the cloud layer. The implementation of the data processing and analysis can be found on Github (https://github.com/mahreenahmed/AirQualityMonitoring.git).

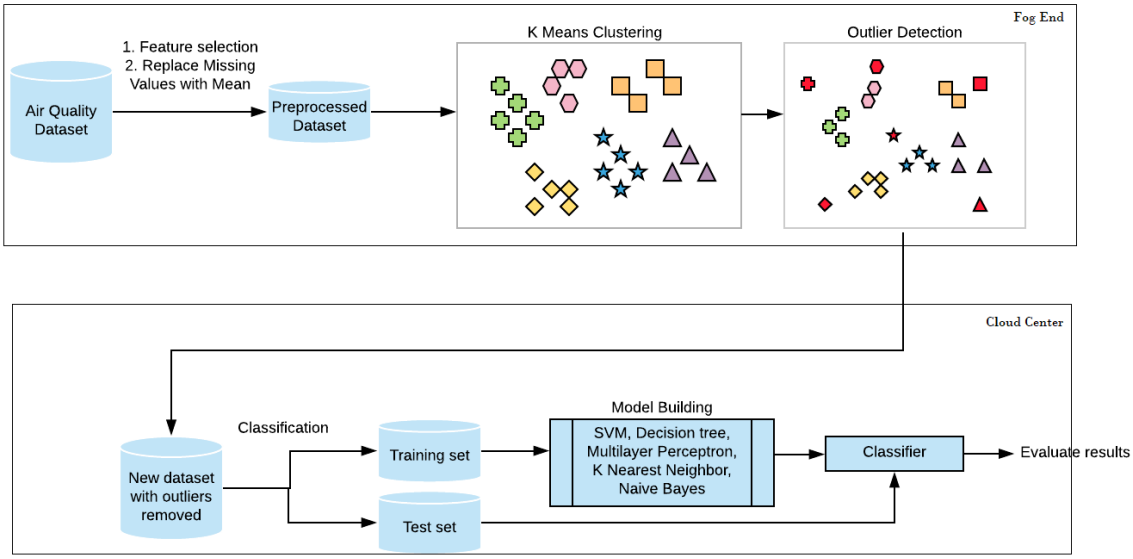

Figure 3. Proposed Methodology for Data Pre-processing at Fog node and Classification at Cloud end.

The proposed data filtering model applied at distributed fog nodes is given in Algorithm 1 and shown in Figure 4. The algorithm takes the air pollutants data as input and generates output that is based on six AQI classes. The three layered architecture has a sensor layer, a distributed fog layer, and a cloud layer, where the sensor layer collects the data samples either through mobile or stationary sensor nodes. The distributed fog layer has these samples, passed on from the sensor layer, filtered, and then preprocessed at different geographically placed fog nodes zones wise. The data are then converted to AQI format with using Equation (1). The sample data contain missing values that are replaced with the interpolated values. Subsequently, the outliers are discarded while using K Means algorithm. This filtered data are then passed on to the cloud layer, where the classification of the filtered data is performed using machine learning algorithms [32,33]. The following sections describe the data processing stages in detail.

5.1. Dataset Collection

In order to demonstrate the proposed approach, we analyzed a publicly available air pollution dataset. The dataset contains data from different sites for the period 2000–2016 [34]. The dataset has a large sample size of around 1.7 million instances. It is based on hourly and mean concentrations of 'NO_2','O_3','SO_2', and 'CO' gases, including other geographical features, like the address, state, and the country. The dataset has no time variable, except the first maximum AQI value recorded at a certain hour of the day. This data are chosen, as they are diverse in nature and have a large sample size that is useful for analyzing our proposed approach.

Conducting irrelevant data filtration can consume a lot of time on the cloud and air quality can produce tremendous data if we want a full coverage of the city. As a proof of concept, we have used this small dataset in order to overcome the constraints of computational resources available to us for data processing.

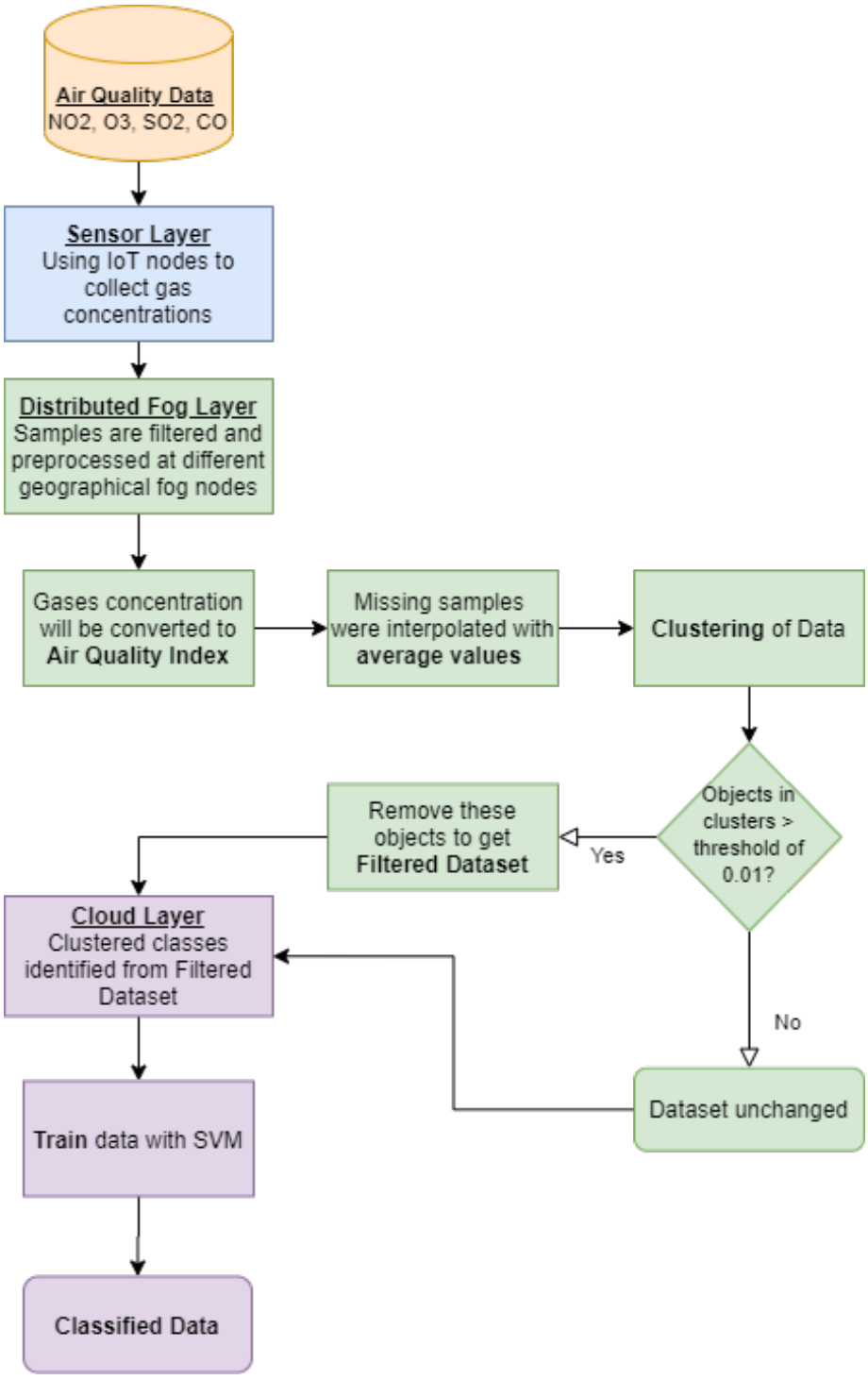

Figure 4. Flowchart of the Proposed Data Filtering Distributed Fog Model.

Algorithm 1: Proposed Data Filtering Distributed Fog Model

Input:
Air Quality Data which includes air pollutants
AQD = x
where, x = (NO_2, O_3, SO_2, CO)
Output:
Air Quality Data Classified as either Good (G), Moderate (M), Unhealthy for sensitive groups (Usg), Unhealthy (U), Very Unhealthy (VU) and Hazardous (H)
Sensor Layer:
Data Samples with concentrations of 'NO_2','O_3','SO_2' and 'CO' gases are collected at the sensor nodes using IoT nodes
Distributed Fog Layer:
Samples are filtered and preprocessed at different geographical fog nodes
a. Gases concentration will be converted to Air Quality Index (AQI) defined by WHO in Equation (1), giving AQD*

$$AQI = \frac{(AQI_{Hi}) - (AQI_{Lo})}{(Conc_{Hi}) - (Conc_{Lo})} X((Conc_i) - (Conc_{Lo})) + (AQI_{Lo}) \quad (1)$$

Here,
$Conc_i$ = Input concentration of a pollutant
$Conc_{Lo}$ = The concentration breakpoint that is less than or equal to $Conc_i$
$Conc_{Hi}$ = The concentration breakpoint that is greater than or equal to $Conc_i$
AQI_{Lo} = The AQI value corresponding to $Conc_i$
AQI_{Hi} = The AQI value corresponding to $Conc_i$
b. Missing samples in AQD* were interpolated with average values giving AQD**
c. Use Elbow Method to determine the value of K giving K*
Here, K* = 6
d. Initial cluster centers C* are selected from AQD**.
Here, C* = K*;
repeat
1. The mean value M in the cluster determines if each K* object is (re)allocated to a given cluster based on its similarity ;
2. Mean value M of each cluster is re calculated giving M*;
until No change;
e. Determine all objects in clusters greater than distance defined by a threshold of 0.01 giving O
f. Remove O from AQD** giving AQD***
Cloud Layer :
Filtered Data AQD*** = x*,y* is received at this layer
Here, x* = filtered (NO_2, O_3, SO_2, CO) samples
a. y* is identified as the clustered classes identified at the Distributed Fog Layer
y* = Cluster Number (N)
b. Train the AQD*** with SVM
Input: x*,y*
Output: Classified AQD***
Identify the right hyperplane using SVM kernels either linear or gaussian
return Classified AQD*** with the 99% classification rate

5.2. Preprocessing of Dataset at Fog Layer

The dataset has a large volume, thus a random selection of 50,000 samples is made due to computational constraints. The AQI features of the four gases i.e., NO_2, O_3, SO_2, and CO are selected for further data analysis. The missing samples in the dataset are interpolated with average values. Once the dataset is preprocessed, K-means clustering is used for the incoming data stream processing and outlier detection. The primary reason for choosing K-means [35,36] is its simplicity, and effectiveness for handling the processing of continuous data stream where a high dimensional data is to be sent to the cloud [37,38]. Secondly, K-means is frequently applied to track environmental changes [39].

In addition to this, the anomaly detection technique using K-means clustering is popular for years and it is applied in order to filter out the outliers or noise in the data [40–42]. Additionally, other statistical approaches, such as computing average, median, or mode, can be explored in order to reduce the amount of irrelevant data, thus allowing only meaningful data to reach the cloud. The data filtering allows for reduced bandwidth for better data transmission as compared to the data directly sent to the cloud layer for processing.

Anomaly Detection with K-Means Clustering

Anomaly Detection [43] is a technique that identifies and filters out the irrelevant data. These irrelevant data points are called outliers and they deviate from the expected behavior. To detect the outliers, we used the K-means clustering technique, where the dataset is divided into K-clusters. We first used the preprocessed data with K in the range of 1:30 in order to decide the number of clusters [44]. The process is then evaluated while using the Elbow method [45], which is the most commonly used method to determine the optimal value of K, at which the cost function stops decreasing (see Figure 5). It can be seen that the optimal score is achieved with the K value equal to 6 (pointed in red in Figure 5), thus dividing the dataset into six clusters, as shown in Figure 6, where the centroids are marked in black dots.

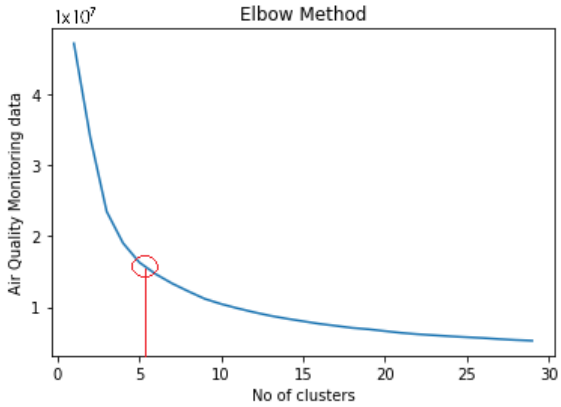

Figure 5. Elbow Method.

After deciding the number of clusters, the subsequent step is to convert the pollutant gases concentrations to Air Quality Index (AQI) and organized as Good (G), Moderate (M), Unhealthy for sensitive groups (Usg), Unhealthy (U), Very Unhealthy (VU), and Hazardous (H) based on the AQI ranges that are defined in Table 1. Figure 7 shows six clusters of the preprocessed dataset, which can be discerned by the different colors.

Further, the outliers are detected by identifying the samples that are distant from the centroid. All of the objects in the clusters with a distance greater than the defined threshold, from the centroid of the respective cluster are considered as anomaly [46,47]. With the threshold of 0.01, around six hundred data samples are marked as outliers. The outliers are highlighted in red color in Figure 8 in order to differentiate them from the normal and relevant samples (in blue).

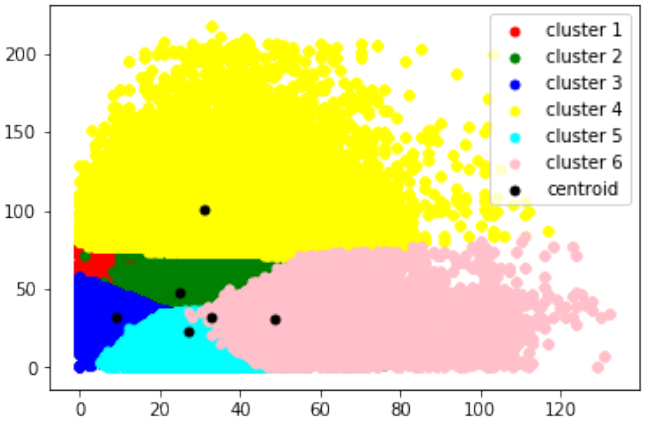

Figure 6. 'NO_2','O_3','SO_2' and 'CO' gases with six clusters.

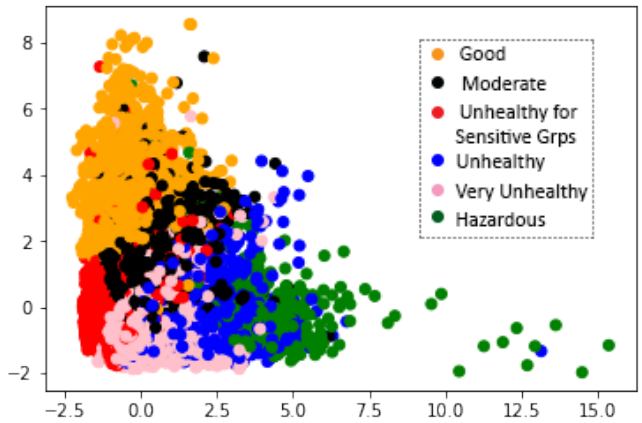

Figure 7. Clusters of preprocessed dataset.

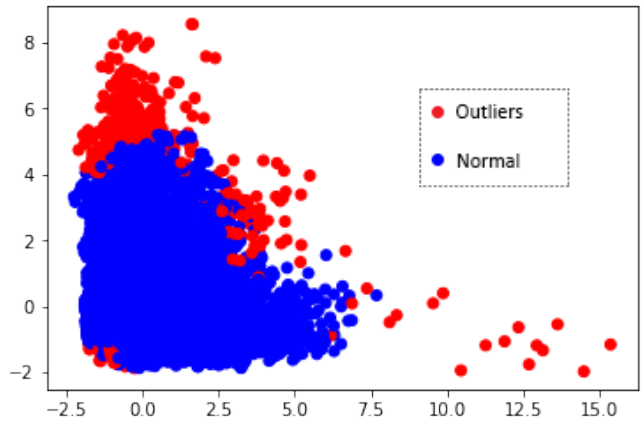

Figure 8. Preprocessed dataset with Outliers (in red).

These detected outliers can be removed from the dataset, thus only allowing useful and meaningful data to reach the cloud.

5.3. Classification of Air Quality Data at the Cloud Layer

Once the air pollutant is converted to AQI, the entire dataset has the uniform format independent of air pollutant measuring unit. Generally, the AQI is calculated for every air pollutant and the air pollutant, which has the highest AQI defines the air quality of that particular instance. Based on this

fact, there is a possibility that the entire dataset may be classified into one class, which may create a class imbalance problem for supervised classification. For this purpose, we devised an alternative approach, and used clustering to group the related data (organized into six classes based on Table 1) and use the resultant clusters as classes, namely Good (G), Moderate (M), Unhealthy for sensitive groups (Usg), Unhealthy (U), Very Unhealthy (VU), and Hazardous (H). These classes can be used in order to train a model for supervised classification.

The different classifiers, including Support Vector Machines (SVM), Multilayer Perceptron (MLP), Decision Tree (DT), K Nearest Neighbor (KNN), and Naive Bayes (NB), are applied on the preprocessed dataset for classification purposes. We selected SVM for its strength to find complex relationships among the data samples without performing any difficult transformations. Moreover, MLP is known for its flexibility and high classification accuracy. However, these classifiers take longer computation time when compared to NB and DT, which take less time and give satisfactory performance.

6. Results and Discussion

In this section, we discussed the results of the data analysis and processing. The dataset is split into 70:30 ratio for the training and test set formation. SVM showed the best performance with an accuracy of 99% on the test set, whereas MLP, KNN, and NB gave satisfactory performance on the clustered dataset. The worst performance is shown by DT with an accuracy of 69%. The metric, such as Accuracy (Acc), Precision (Pre), Recall (Rec), and F1-Score (F1-Sc), are computed in order to evaluate the performance of these models. Equations (2)–(5) show the formulas for these metrics, where TP is True Positive, TN is True Negative, FP is False Positive, and FN is False Negative. The evaluation results of the different classifiers on the dataset are shown in Tables 3–6.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

$$Rec = \frac{TP}{TP + FN} \tag{3}$$

$$Pre = \frac{TP}{TP + FP} \tag{4}$$

$$F1 - Sc = 2 * \frac{Pre * Rec}{Pre + Rec} \tag{5}$$

Table 3. Accuracy on Test Set.

Classifier	Accuracy
NB	89.7
MLP	97
SVM	99.9
DT	69
KNN	98

Table 4. Precision on Test Set.

Classifier	Pre (Class Wise)					
	G	M	Usg	U	VU	H
NB	95	88	91	86	89	90
MLP	100	97	92	99	93	98
SVM	99	99	100	99	99	99
DT	90	-	-	58	67	-
KNN	99	96	98	98	97	99

Table 5. Recall on Test Set.

Classifier	Rec (Class Wise)					
	G	M	Usg	U	VU	H
NB	96	70	93	92	88	99
MLP	98	97	96	97	99	99
SVM	99	100	100	99	99	100
DT	83	0	0	93	90	0
KNN	98	98	95	98	98	98

Table 6. F1-Score on Test Set.

Classifier	F1-Sc (Class Wise)					
	G	M	Usg	U	VU	H
NB	95	78	92	89	88	95
MLP	99	97	94	98	96	98
SVM	99	99	100	99	99	99
DT	86	-	-	71	77	-
KNN	99	97	96	98	98	98

The graph in Figure 9 illustrates the classes predicted by the SVM. Figure 10 shows the confusion matrix for the different classifiers. The Confusion Matrices show the actual and predicted samples for each class. With SVM, we see that 2815 class 'G' samples are correctly predicted out of 2821 samples. Similarly, 1309/1309 class 'M' samples are correctly predicted. For class 'U', two samples are misclassified as class 'G' and 2 as class 'M' while 1717 out of 1721 are correctly classified as class 'U'.

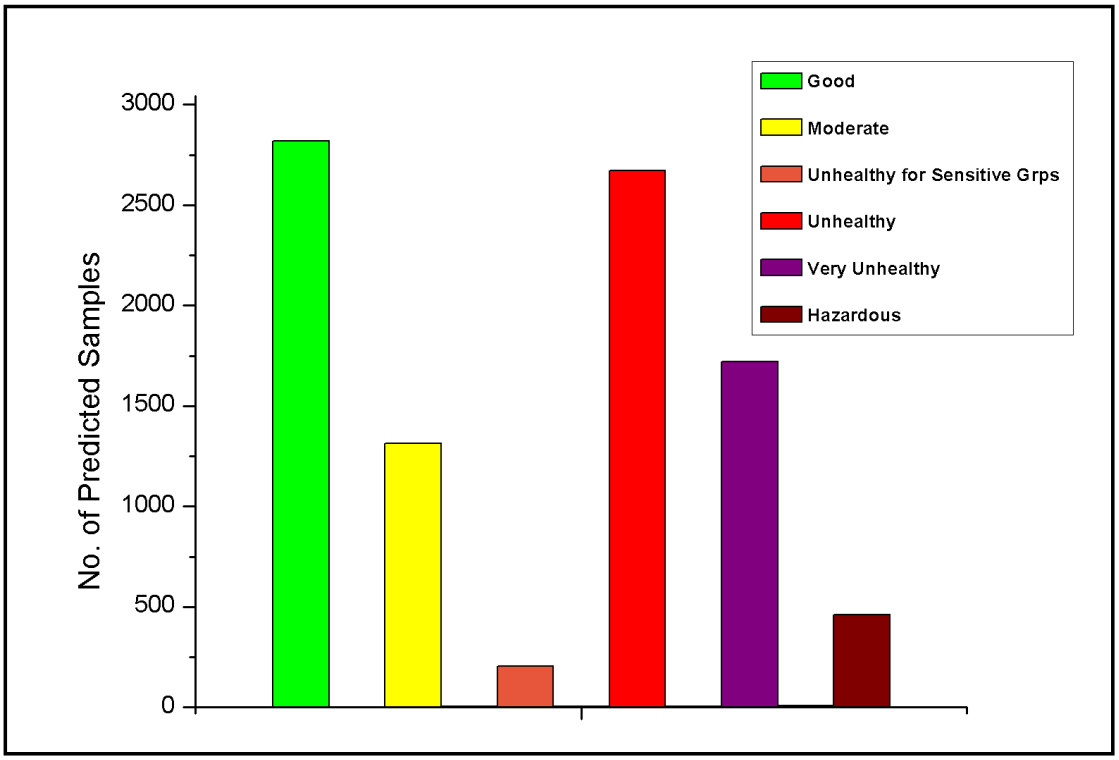

Figure 9. Classification Performance of SVM.

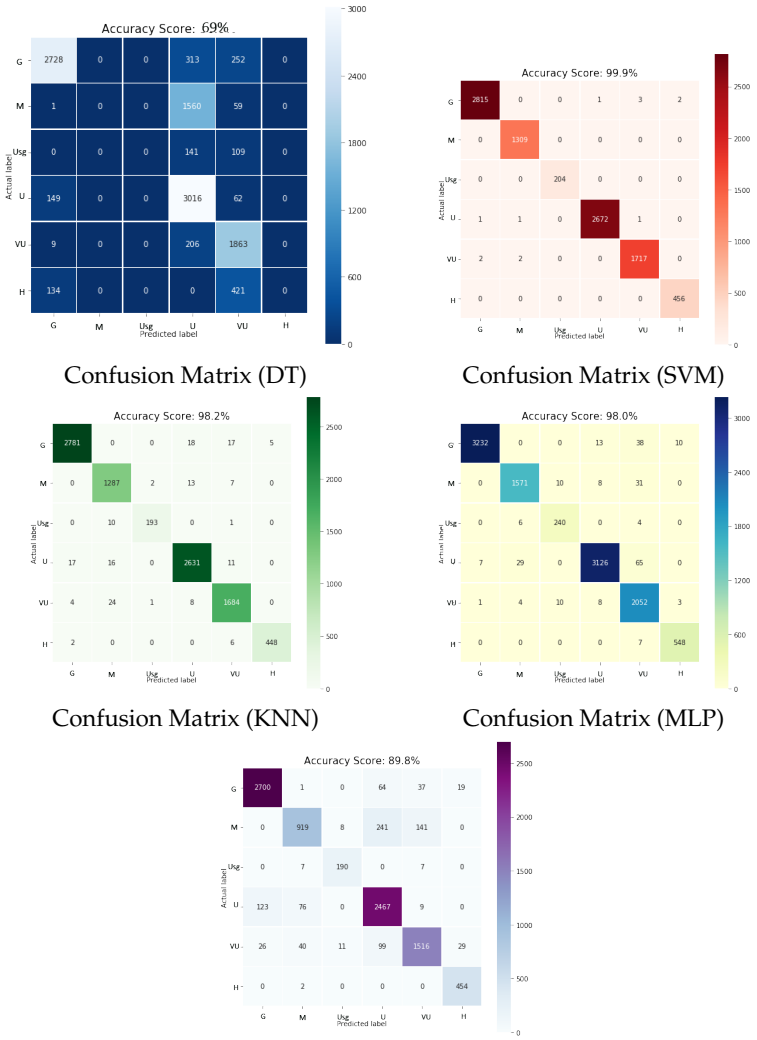

Figure 10. Confusion Matrices of Classifiers.

Simulation of Proposed Distributed Fog Architecture

In order to model the proposed architecture, simulations are performed in software toolkits, such as iFogSim [48], YAFS [49], and CloudSimSDN [50]. The configuration parameters for simulating the cloud, fog, and sensor nodes are given in Table 7. The simulator, iFogSim, has limited definition of physical topologies and due to this the fog devices can only be arranged in a hierarchical tree like fashion, where, the communication is only possible between a parent-child pair as shown in Figure 11. However, comparing this fog based topology to cloud, the latency and network usage values are very low. The latency factor is primarily linked to the efficiency of the system. If the bandwidth of fog network is low, less traffic will pass through the fog node, which makes it more manageable and efficient.

YAFS is a fairly new environment and not commonly used for research works. While working with YAFS, we faced problems with some missing libraries that are needed to run it efficiently. On the other hand, building our proposed architecture topology using the GUI in CloudSimSDN is a challenge, as the packages that support this functionality are not available in the main repository of the simulator. Table 8 summarizes the steps involved to simulate the proposed architecture in these simulators.

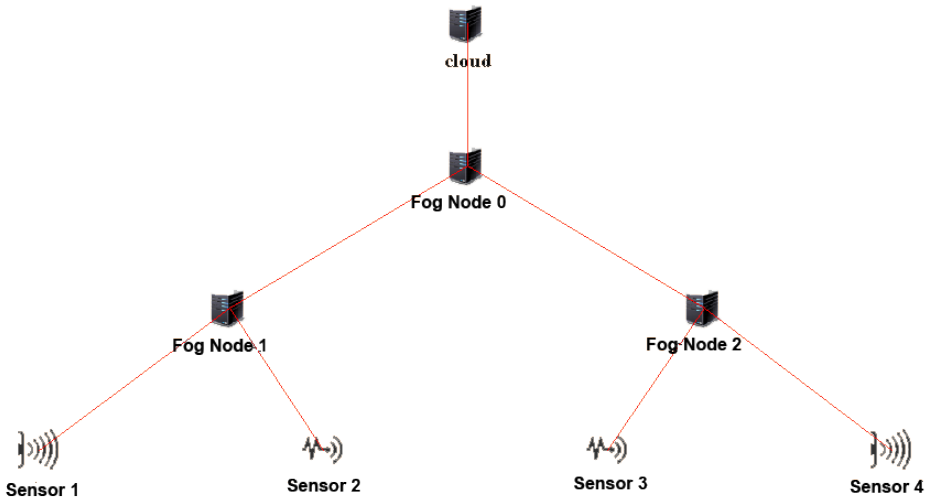

Figure 11. Topology generated in ifogSIM toolkit.

Table 7. Configuration parameters for Cloud, fog and Sensor Nodes.

Parameter	Cloud	Fog	Sensor
Level	0	1	-
Uplink Bandwidth (MB)	1000	1000	-
Downlink Bandwidth (MB)	1200	1100	-
CPU length (MIPS)	5000	4000	-
Random Access Memory (MB)	45,000	4500	-
Rate Per MIPS	1000	500	-
Distribution Type	-	-	Uniform
Min	-	-	20
Max	-	-	100

Table 8. Experimental Procedure.

Simulator	Steps Taken
iFogSim	1. Install and Run iFogSim GUI 2. Create Cloud and Fog Nodes 3. Give values to parameters in Cloud and Fog nodes 4. Attach Fog Nodes to Cloud 5. Create IoT sensors 6. Attach IoT sensors to Fog nodes 7. Run the simulation
YAFS	1. Install Anaconda 2 to run python 2.7 code 2. Install third libraries 3. Clone YAFS project from github 4. Write code to run YAFS 5. Run code
CloudSimSDN	1. Install and Run CloudSimSDN GUI 2. Create Cloud and Fog Nodes 3. Give values to parameters in Cloud and Fog nodes 4. Attach Fog Nodes to Cloud 5. Create IoT sensors 6. Attach IoT sensors to Fog nodes 7. Run the simulation

7. Conclusions and Future Work

In this paper, we have proposed a layered architecture for data processing and analysis for effective air quality monitoring through distributed fog computing. For data collection, the mobile platforms in addition to stationary nodes are proposed for ubiquitous coverage. Cloud is useful for data analytics, but it makes the system slow, whereas the use of fog layer makes the system more effective and efficient. When it comes to security and privacy of the sensitive data sent to the cloud, with fog nodes, it is easy to locally analyze the data instead of on the cloud directly. Protecting the IoT enabled devices requires real time data analysis and analyzing it locally is much easier and accessible than on the cloud. Most of the data processing, such as data filtration, is done near the IoT sensors, which can lead to reduced data processing at the cloud end. We proposed using clustering, such as K-means and other statistical methods, in order to reduce the amount of data to be sent to the cloud. The clustering process can detect the outliers or noise in the dataset, which will only send the meaningful and relevant data to the cloud for advanced analysis. The clusters formed can be used to train different models for supervised classification, such as MLP, SVM, KNN, DT, and NB.

The proposed IoT and fog based air quality monitoring system captures the data at the sensor layer and process it by discarding the irrelevant and superfluous data at the distributed fog layer while using clustering techniques. This ensures less space consumption and processing at the cloud layer. By using the proposed data filtration approach at the fog layer, we achieved 99% accuracy with SVM followed by KNN and MLP classifiers. The limited air quality dataset in the proposed model is used to provide a proof of concept, as we have constrained computational resources that are available and performing filtration of superfluous data can consume significant amount of time and require more computational power owing to the large volume of data produced to monitor the air quality of the entire city. The proposed model is also simulated in iFogSim toolkit and found that the integration of the fog layer in the system would greatly enhance system efficiency. Hence, we conclude that fog computing is an important and useful technology that can easily handle the data locally closer to the IoT sensors, and it is highly suitable for applications that generate a tremendous amount of data that requires preprocessing before performing any basic or advance analysis. However, the fog computing is dependent on many physical links to transfer the information, which can lead to potential failures. Therefore, as a future direction, the Edge computing paradigm [51] can be explored, as it offers data processing at the endpoint as compared to fog computing which processes data near to the endpoint. Edge computing provides low latency by locally processing the data at the network edge and only sends data that were collected to the cloud for long time storage and analysis. This approach facilitates reducing the amount of data sent to the cloud and improves the response time in return.

Author Contributions: Conceptualization, M.A. (Mehreen Ahmed), R.M. and S.M.H.Z.; Methodology, M.A. (Mehreen Ahmed), R.M., S.M.H.Z. and S.A.R.Z.; Validation, M.A. (Mehreen Ahmed), R.M. and M.H.; Investigation, M.A. (Mehreen Ahmed), R.M., S.M.H.Z., S.A.R.Z. and M.A. (Muneer Ahmad); Writing—Original Draft, M.A. (Mehreen Ahmed), R.M., M.H. and S.A.R.Z.; Writing—Review & Editing, M.A. (Mehreen Ahmed), R.M., S.M.H.Z., M.H. and M.A. (Muneer Ahmad); Visualization, M.A. (Mehreen Ahmed) and M.A. (Muneer Ahmad); Validation, R.M., M.H. and S.A.R.Z.; Software, M.H. and S.A.R.Z.; Data Curation, M.H.; Supervision, R.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. BreatheLife. City Data Page-BreatheLife 2030. 2018. Available online: http://breathelife2030.org/city-data-page/?city=1875# (accessed on 15 April 2018).
2. AQI. Air Quality Index (AQI) Basics. 2018. Available online: https://cfpub.epa.gov/airnow/index.cfm?action=aqibasics.aqi (accessed on 21 November 2018).

3. Aazam, M.; Khan, I.; Alsaffar, A.A.; Huh, E.N. Cloud of Things: Integrating Internet of Things and cloud computing and the issues involved. In Proceedings of the 2014 11th International Bhurban Conference on Applied Sciences & Technology (IBCAST), Islamabad, Pakistan, 14–18 January 2014; pp. 414–419.
4. Aazam, M.; Hung, P.P.; Huh, E.N. Smart gateway based communication for cloud of things. In Proceedings of the 2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), Singapore, 21–24 April 2014; pp. 1–6.
5. Huang, D.Y.; Apthorpe, N.; Li, F.; Acar, G.; Feamster, N. Iot inspector: Crowdsourcing labeled network traffic from smart home devices at scale. *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol.* **2020**, *4*, 1–21. [CrossRef]
6. Zhang, F.; Chang, Z.; Niu, K.; Xiong, J.; Jin, B.; Lv, Q.; Zhang, D. Exploring LoRa for Long-range Through-wall Sensing. *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol.* **2020**, *4*, 1–27. [CrossRef]
7. García-Escudero, L.A.; Gordaliza, A.; Matrán, C. Trimming tools in exploratory data analysis. *J. Comput. Graph. Stat.* **2003**, *12*, 434–449. [CrossRef]
8. Liu, D.; Yan, Z.; Ding, W.; Atiquzzaman, M. A survey on secure data analytics in edge computing. *IEEE Internet Things J.* **2019**, *6*, 4946–4967. [CrossRef]
9. Rao, F.Y.; Bertino, E. Privacy Techniques for Edge Computing Systems. *Proc. IEEE* **2019**, *107*, 1632–1654. [CrossRef]
10. Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S. Fog computing and its role in the internet of things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, MCC@SIGCOMM 2012, Helsinki, Finland, 17 August 2012; pp. 13–16.
11. Aazam, M.; Huh, E.N. Fog computing and smart gateway based communication for cloud of things. In Proceedings of the 2014 International Conference on Future Internet of Things and Cloud 2014, Barcelona, Spain, 27–29 August 2014; pp. 464–470.
12. Pan, W.; Dong, W.; Cebrian, M.; Kim, T.; Fowler, J.H.; Pentland, A.S. Modeling dynamical influence in human interaction: Using data to make better inferences about influence within social systems. *IEEE Signal Process. Mag.* **2012**, *29*, 77–86. [CrossRef]
13. Hong, K.; Lillethun, D.; Ramachandran, U.; Ottenwälder, B.; Koldehofe, B. Mobile fog: A programming model for large-scale applications on the internet of things. In Proceedings of the Second ACM SIGCOMM Workshop on Mobile Cloud Computing, MCC@SIGCOMM 2013, Hong Kong, China, 16 August 2013; pp. 15–20.
14. Zhu, J.; Chan, D.S.; Prabhu, M.S.; Natarajan, P.; Hu, H.; Bonomi, F. Improving web sites performance using edge servers in fog computing architecture. In Proceedings of the 2013 IEEE Seventh International Symposium on Service-Oriented System Engineering, Redwood City, CA, USA, 25–28 March 2013; pp. 320–323.
15. Cocchia, A. Smart and digital city: A systematic literature review. In *Smart city*; Springer: New York, NY, USA, 2014; pp. 13–43.
16. Harper, R. *Inside the Smart Home*; Springer Science & Business Media: New York, NY, USA, 2006.
17. Bonomi, F.; Milito, R.; Natarajan, P.; Zhu, J. Fog computing: A platform for internet of things and analytics. In *Big data and internet of things: A roadmap for smart environments*; Springer: New York, NY, USA, 2014; pp. 169–186.
18. Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge computing: Vision and challenges. *IEEE Internet Things J.* **2016**, *3*, 637–646. [CrossRef]
19. Cao, Y.; Chen, S.; Hou, P.; Brown, D. FAST: A fog computing assisted distributed analytics system to monitor fall for stroke mitigation. In Proceedings of the 2015 IEEE International Conference on Networking, Architecture and Storage (NAS), Boston, MA, USA, 6–7 August 2015; pp. 2–11.
20. Ha, K.; Chen, Z.; Hu, W.; Richter, W.; Pillai, P.; Satyanarayanan, M. Towards wearable cognitive assistance. In Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services, Bretton Woods, NH, USA, 16–19 June 2014; pp. 68–81.
21. Stojmenovic, I.; Wen, S. The fog computing paradigm: Scenarios and security issues. In Proceedings of the Computer Science and Information Systems (FedCSIS), Warsaw, Poland, 7–10 September 2014; pp. 1–8.
22. Dastjerdi, A.V.; Buyya, R. Fog computing: Helping the Internet of Things realize its potential. *Computer* **2016**, *49*, 112–116. [CrossRef]

23. Sevusu, P. Real-Time Air Quality Measurements Using Mobile Platforms. Ph.D. Thesis, Rutgers University-Graduate School-New Brunswick, New Brunswick, NJ, USA, 2015.
24. Fioccola, G.B.; Sommese, R.; Tufano, I.; Canonico, R.; Ventre, G. Polluino: An efficient cloud-based management of IoT devices for air quality monitoring. In Proceedings of the Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI), Bologna, Italy, 7–9 September 2016; pp. 1–6.
25. Aazam, M.; Huh, E.N. Fog computing: The cloud-iot\/ioe middleware paradigm. *IEEE Potentials* **2016**, *35*, 40–44. [CrossRef]
26. Cheng, Y.; Li, X.; Li, Z.; Jiang, S.; Li, Y.; Jia, J.; Jiang, X. AirCloud: A cloud-based air-quality monitoring system for everyone. In Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems, Memphis, TN, USA, 3–6 November 2014; pp. 251–265.
27. Sun, P.L.; Weng, J.Y.; Yang, C.T.; Chen, S.T.; Liu, J.C. The Implementation of Air Pollution Monitoring Service Using Hybrid Database Converter. In Proceedings of the Cloud Computing and Big Data (CCBD), 2016 7th International Conference, Macau, China, 16–18 November 2016; pp. 269–274.
28. Santos, J.; Leroux, P.; Wauters, T.; Volckaert, B.; De Turck, F. Anomaly detection for Smart City applications over 5G low power wide area networks. In Proceedings of the NOMS 2018–2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, Taiwan, 23–27 April 2018; pp. 1–9.
29. Atlam, H.F.; Walters, R.J.; Wills, G.B. Fog computing and the internet of things: A review. *Big Data Cogn. Comput.* **2018**, *2*, 10. [CrossRef]
30. Rahmani, A.M.; Liljeberg, P.; Preden, J.S.; Jantsch, A. *Fog Computing in the Internet of Things: Intelligence at the Edge*; Springer: New York, NY, USA, 2017.
31. Dennis, R.L.; Byun, D.W.; Novak, J.H.; Galluppi, K.J.; Coats, C.J.; Vouk, M.A. The next generation of integrated air quality modeling: EPA's Models-3. *Atmos. Environ.* **1996**, *30*, 1925–1938. [CrossRef]
32. Ruan, S.; Bao, J.; Liang, Y.; Li, R.; He, T.; Meng, C.; Li, Y.; Wu, Y.; Zheng, Y. Dynamic Public Resource Allocation Based on Human Mobility Prediction. *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol.* **2020**, *4*, 1–22. [CrossRef]
33. Wang, Q.; Zhang, J.; Guo, B.; Hao, Z.; Zhou, Y.; Sun, J.; Yu, Z.; Zheng, Y. CityGuard: Citywide Fire Risk Forecasting Using A Machine Learning Approach. *Proc. ACM Interactive Mobile Wearable Ubiquitous Technol.* **2019**, *3*, 1–21. [CrossRef]
34. Pollution. U.S. Pollution Data | Kaggle. 2018. Available online: https://www.kaggle.com/sogun3/uspollution (accessed on 18 August 2018).
35. Kanungo, T.; Mount, D.M.; Netanyahu, N.S.; Piatko, C.D.; Silverman, R.; Wu, A.Y. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 881–892. [CrossRef]
36. Niu, Z.; Shi, S.; Sun, J.; He, X. A survey of outlier detection methodologies and their applications. In *International Conference on Artificial Intelligence and Computational Intelligence*; Springer: New York, NY, USA, 2011; pp. 380–387.
37. Valsamis, A.; Tserpes, K.; Zissis, D.; Anagnostopoulos, D.; Varvarigou, T. Employing traditional machine learning algorithms for big data streams analysis: The case of object trajectory prediction. *J. Syst. Softw.* **2017**, *127*, 249–257. [CrossRef]
38. Puschmann, D.; Barnaghi, P.; Tafazolli, R. Adaptive clustering for dynamic IoT data streams. *IEEE Internet Things J.* **2017**, *4*, 64–74. [CrossRef]
39. Elangasinghe, M.; Singhal, N.; Dirks, K.; Salmond, J.; Samarasinghe, S. Complex time series analysis of PM10 and PM2.5 for a coastal site using artificial neural network modelling and k-means clustering. *Atmos. Environ.* **2014**, *94*, 106–116. [CrossRef]
40. Münz, G.; Li, S.; Carle, G. Traffic anomaly detection using k-means clustering. In Proceedings of the GI/ITG Workshop MMBnet 2007, Hamburg, Germany, 13–14 September 2007.
41. Chawla, S.; Gionis, A. k-means–: A unified approach to clustering and outlier detection. In Proceedings of the 2013 SIAM International Conference on Data Mining, Austin, TX, USA, 2–4 May 2013; SIAM: Bangkok, Thailand, 2013; pp. 189–197.
42. Rao, A.R.; Garai, S.; Clarke, D.; Dey, S. A system for exploring big data: an iterative k-means searchlight for outlier detection on open health data. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.

43. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly detection: A survey. *ACM Comput. Surv. (CSUR)* **2009**, *41*, 15. [CrossRef]
44. Jain, A.K. Data clustering: 50 years beyond K-means. *Pattern Recognit. Lett.* **2010**, *31*, 651–666. [CrossRef]
45. Coates, A.; Ng, A. Learning feature representations with k-means. In *Neural Networks: Tricks of the Trade*; Springer: Berlin, Germany, 2012.
46. Hodge, V.; Austin, J. A survey of outlier detection methodologies. *Artif. Intell. Rev.* **2004**, *22*, 85–126. [CrossRef]
47. He, Y.; Zhu, C.; He, Z.; Gu, C.; Cui, J. Big data oriented root cause identification approach based on Axiomatic domain mapping and weighted association rule mining for product infant failure. *Comput. Ind. Eng.* **2017**, *109*, 253–265. [CrossRef]
48. Gupta, H.; Vahid Dastjerdi, A.; Ghosh, S.K.; Buyya, R. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. *Softw. Pract. Exp.* **2017**, *47*, 1275–1296. [CrossRef]
49. Lera, I.; Guerrero, C.; Juiz, C. YAFS: A simulator for IoT scenarios in fog computing. *IEEE Access* **2019**, *7*, 91745–91758. [CrossRef]
50. Son, J.; Dastjerdi, A.V.; Calheiros, R.N.; Ji, X.; Yoon, Y.; Buyya, R. Cloudsimsdn: Modeling and simulation of software-defined cloud data centers. In Proceedings of the 2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Shenzhen, China, 4–7 May 2015; pp. 475–484.
51. Satyanarayanan, M.; Bahl, P.; Caceres, R.; Davies, N. The case for vm-based cloudlets in mobile computing. *IEEE Pervasive Comput.* **2009**, *8*, 14–23. [CrossRef]