DOI 10.12694/scpe.v20i4.1561
Scalable Computing: Practice and Experience ISSN 1895-1767
Volume 20, Number 4, pp. 599–606. http://www.scpe.org

IOT BASED AIR QUALITY MONITORING SYSTEM USING MQ135 AND MQ7 WITH MACHINE LEARNING ANALYSIS

KINNERA BHARATH KUMAR SAI*, SOMULA RAMASUBBAREDDY†, AND ASHISH KR. LUHACH‡

Abstract. This paper deals with measuring the Air Quality using MQ135 sensor along with Carbon Monoxide CO using MQ7 sensor. Measuring Air Quality is an important element for bringing awareness to take care of the future generations and for a healthier life. Based on this, Government of India has already taken certain measures to ban Single Stroke and Two Stroke Engine based motorcycles which are emitting high pollution. We are trying to implement a system using IoT platforms like Thingspeak or Cayenne in order to bring awareness to every individual about the harm we are doing to our environment. Already, New Delhi is remarked as the most pollution city in the world recording Air Quality above 300 PPM. We have used easiest platform like Thingspeak and set the dashboard to public such that everyone can come to know the Air Quality at the location where the system is installed. Machine Learning analysis brings us a lot of depth in understanding the information that we obtained from the data. Moreover, we are proviing a reducement of the cost of components versus the state of the art.

Key words: IoT, MQ135, MQ7, Thingspeak, Machine Learning

AMS subject classifications. 68M10, 92B20

1. Introduction.

Air is getting polluted because of release of toxic gases by industries, vehicle emissions and increased concentration of harmful gases and particulate matter in the atmosphere. The level of pollution is increasing rapidly due to factors like industries, urbanization, increasing in population, vehicle use which can affect human health. Particulate matter has the most significant contribution to the increase in air pollution [2]. This creates a need for measurement and analysis of real-time air quality monitoring so that appropriate decisions can be taken in a timely period. This paper presents a real-time standalone air quality monitoring.

Internet of Things (IoT) is nowadays finding profound use in each and every sector and plays a key role in our air quality monitoring system too. Our setup will show the air quality in PPM (Parts per Million) in a web page so that we can monitor it very easily. In this IoT project, one can monitor the pollution level from anywhere using a computer or a mobile [1].

Air Pollution is increasing heavily these days due to the many important factors like Vehicle Emissions, Deforestation, Industrialization [21]. Old vehicles could produce more smoke and hence those vehicles could be banned from using. But we need a proper metric that tells us whats the intensity level of air pollution, whether it is low, medium or high. So, we can take help of various sensors that could detect the air quality and from the corresponding values, we do mathematical calculations and from that results, we could fix the threshold value from which we can say whether the desired air quality is below the threshold value or not.

Every sensor that helps us in this experiment will come with a manufacturer sheet which helps us for the mathematical calculations. Now, Internet of Things helps us to connect these sensors to the Cloud where we are able to access the system from anywhere using login credentials [22]-[27]. There are many online free sources available targeting Internet of things (IoT) that could give a ready dashboard and a means of connecting the account to the physical system. We can use the above said free sources or we could build our own website maintaining a background database and access to the physical system and this could be a tedious job. If one's requirement is something other than available options being provided in the open source websites, then we could develop our own website. Moreover, as the sensors are analog in nature, it would continuously project the values from time to time. We need an enough amount of space that could handle this real time data being generated. Also, making an Android application that could be actively projecting these changing values accurately and fastly is challenging. Open source websites like mydevices.cayyene.com provides both web interface as well as mobile application for both Android and iOS.

Finally, we could apply machine learning for detailed analysis like which are the areas having more air

*School of Computer Science and Engineering (SCOPE), VIT University, Vellore, India (yourfriend9014@gmail.com).
†Information Technology, VNRVJIET, Hyderabad,India.(svramasubbareddy1219@gmail.com)
‡Department of Electrical and Communication Engineering,The PNG University of Technology, Papua New Guinea.(ashishluhach@acm.org)

TABLE 2.1
Air Quality index

Range(PPM)	Status
0-50	Good
51-100	Moderate
100-150	Unhealthy for sensitive groups
151-200	Unhealthy
201-300	Very Unhealthy
301-500	Hazardous

quality than the desired threshold and also at what time we are getting more values from the sensors. If we could integrate these values at respective places in the maps, more awareness can be targeted such that people will take necessary steps for improvising the air quality like afforestation, growing pot plants inside, upgrading engine quality for reducing bad emissions from vehicles, encouraging electric vehicles etc. Calibrating the sensors before installing at a location matters a lot. Also, almost all the electronic components suffer from aging effect.

2. Related Work. Table 1 [11] explains the Air Quality Index ranges. Firstly, 0-50 PPM can be considered completely safe. 51-100 PPM can be considered as Moderate – this could be usually observed at traffic areas [4]. 100-150 PPM can be considered as Unhealthy, but only for sensitive groups. Above 151 PPM [11] is completely unsafe or unhealthy – India's capital New Delhi falls in this range. Its very rare to record 300 PPM and above, which can be considered as Hazardous, possibly due to Coal gas in mines [10].

The paper [11] implemented the idea of [1] with less cost i.e., while pushing the data to the Cloud, ther is no need to see the output on LCD which adds more cost to the project [14, 18, 28, 29]. Targeting IoT as a platform, our intension should be to present the data on Internet using the platforms like thinger.io or thingspeak or Cayenne website which are beautifully designed to present the output and even able to download the dataset. When doing an experiment for air quality monitoring, there is no need to use LPG (Liquified Petroleum Gas) or methane detecting sensors as it is used for home/office safety. This paper used WiFi to push the data onto the Cloud rather using GSM or GPRS module as in [2]. Note that in [3, 30] hasn't been calibrated the sensor and not converted the sensor output value into PPM. As per the guidelines by UN Data, 0-50 PPM is SAFE value, 51-100 is moderate as shown in Table 1. New Delhi, the capital of India is the most polluted city in the world recorded around 250PPM [15]. As this paper uses two sensors, both of them have internal heater element, it draws more power (P=V*I), so though the both sensors are turned ON, its output voltage levels varies and shows unpredictable values due to insufficient power drive. So we used a 9 Volts battery and a 7805 family LM7805 Regulator for the CO sensor MQ7 as the power from Arduino alone is not sufficient to drive two sensors.

In the paper [4] is not clear with the components used and the cloud used. This paper also aims to implement the machine learning on the real time dataset collected from the Thingspeak website and try to convey the information about the adverse affects on one's health to the people and government if the same pollution continues [31, 32]. This paper also aims to extend by adding three more sensors related to air quality, i.e, ozone sensor, PM 2.5 laser dust sensor, MG811 (CO_2) sensor that acts as an all-in-one setup giving in depth monitoring of the air quality [4]. The paper [5] had completely taken wrong assumption where they have showed the output 997 PPM as the fresh air, while Delhi which is the most polluted city recording 250 PPM. Its clear understanding that they have not calibrated the sensor and did not even convert the raw sensor data into PPM [16] using derivations we did. They have used a Local Host which is limited where they are able to see the output only on the laptop within the area where experimental setup is connected. But this paper targets using standard IoT platform which is highly secured and open source [6].

3. Numerical Evaluation. We have used Arduino Uno Development kit that comes with ATMega 328P microcontroller. In order to provide WiFi Support for it, we have used cost effective ESP-01 WiFi module which helps us to connect to the ThingSpeak Platform. Figure 3.1 represents the connections between the components used like Arduino Uno, MQ135, MQ7, ESP-01 WiFi Module, 9 Volts Battery, LM7805 Regulator. As shown in

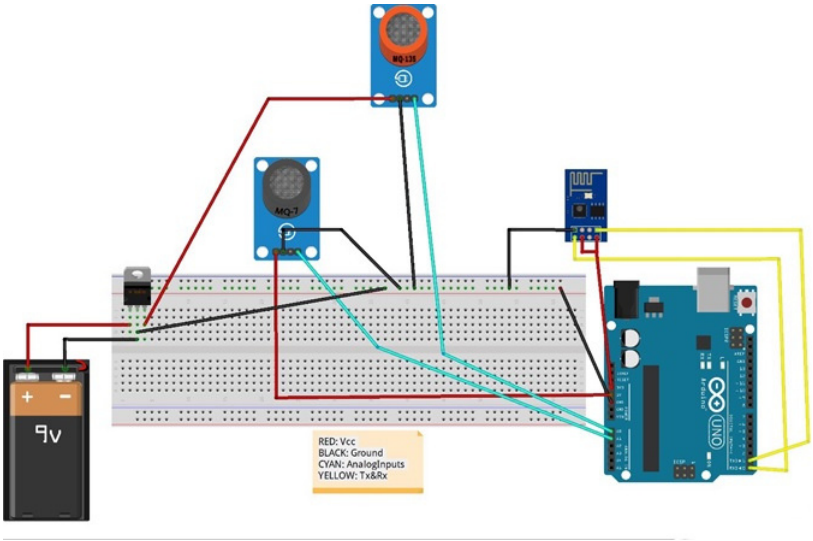

FIG. 3.1. *Connections diagram*

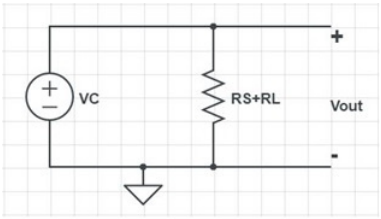

FIG. 3.2. *Internal circuit diagram of MQ135*

Figure 3.1, ESP-01 is connected to 3.3 Volts pin of Arduino Uno. MQ135 is connected to 5 Volts pin of Arduino Uno. As power wont be sufficient to drive one more sensor, MQ7 is connected to 9 Volts Battery via 5 Volts LM7805 Regulator. ESP-01 is connected to the Local Hotspot by giving corresponding SSID and Password. The reason for using LM7805 Regulator is that 9 Volts supply should not be directly given to MQ7 sensor [17] where it needs only 5 Volts input at maximum, so the regulator does the job of stepping 9 Volts to 5 Volts [7, 9].

The most important step is to calibrate the sensor in fresh air and then draws an equation that converts the sensor output voltage value into our convenient units PPM. Here are the mathematical calculations derived from [1].

From Ohm's Law, at constant temperature, we can derive I as follows:

$$I = \frac{V}{R} \tag{3.1}$$

Equation 3.1 is equivalent according to [12] with

$$I = \frac{V_c}{R_s + R_l} \tag{3.2}$$

From Figure 3.2, we can obtain the output voltage at the load resistor using the value obtained for I and Ohm's Law at constant temperature ($V = I \cdot R$):

$$V_{R_l} = \left[\frac{V_c}{R_s + R_l}\right] \cdot R_L \tag{3.3}$$

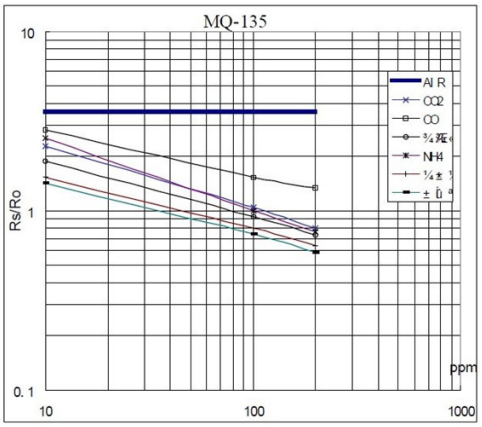

FIG. 3.3. *MQ135 Datasheet-Change in Resistance vs change in PPM*

$$V_{R_l} = \left[\frac{V_c * R_l}{R_s + R_l}\right] \tag{3.4}$$

So now we solve for R_s:

$$V_{R_l} * (R_s + R_l) = V_c * R_L \tag{3.5}$$

$$(V_{R_l} * R_s) + (V_{R_l} * R_l) = V_c * R_L \tag{3.6}$$

$$V_{R_l} * R_s = (V_c * R_l) - (V_{R_l} * R_l) \tag{3.7}$$

$$R_s = \frac{(V_c * R_l) - (V_{R_l} * R_l)}{V_{R_l}} \tag{3.8}$$

$$R_s = \frac{(V_c * R_l)}{V_{R_l}} - R_l \tag{3.9}$$

Equation 3.9 help us to find the internal sensor resistance for fresh air [14].

From the graph shown in Figure 3.3, we can see that the resistance ratio in fresh air is a constant. Figure 3.3 is taken from the MQ135 datasheet where X axis represents the PPM and Y axis represents R_s/R_0 ratio. Now, the sensor is subjected to the respective gases alone and on increasing the PPMs of the gas say methane or Carbon Dioxide, then the corresponding resitance ratio is plotted. If we are interested to calculate the Carbon Monoxide, we have to consider the respective line in the graph and calculate the resistance ratio:

$$\frac{R_s}{R_0} = 3.6 \tag{3.10}$$

Value 3.6 which is mentioned in Equation 3.10 is derived from the datasheet mentioned in Figure 3.3. To calculate R_0, we will need to find the value of the R_s in fresh air. This will be done by taking the analog average readings from the sensor and converting it to voltage [12]. Then we will use the R_s formula to find R_0. First of all, we will treat the lines as if they were linear. This way we can use one formula that linearly relates the ratio and the concentration [19, 20]. By doing so, we can find the concentration of a gas at any ratio value even outside of the graphs boundaries. The formula we will be using is the equation for a line, but for a log-log scale. The formula for a line is [21]:

$$y = mx + b \tag{3.11}$$

From Figure 3.3, we try to derive the following calculations. For a log-log scale, the formula looks like this:

$$\log y = m \log x + b \tag{3.12}$$

Lets find the slope. To do so, we need to choose 2 points from the graph. In our case, we chose the points (200,2.6) and (10000,0.75) from the LPG (Liquified Petroleum Gas) line and we can choose any line shown in Figure 3.3. The formula to calculate m is the following:

$$m = \frac{\log y - \log y_0}{\log x - \log x_0} \tag{3.13}$$

If we apply the logarithmic quotient rule we get the following:

$$m = \frac{\log(y/y_0)}{\log(x/(x_0)} \tag{3.14}$$

Now we substitute the values for x, x_0, y, and y_0:

$$m = \frac{\log(0.75/2.6)}{\log(10000/200)} \tag{3.15}$$

$$m = -0.318 \tag{3.16}$$

Now that we have m, we can calculate the y intercept. To do so, we need to choose one point from the graph (once again from the CO2 line). In our case, we chose (5000,0.9),

$$\log(y) = m \log(x) + b \tag{3.17}$$

$$b = \log(0.9) - (-0.318) * \log(5000) \tag{3.18}$$

$$b = 1.13 \tag{3.19}$$

Now that we have m and b, we can find the gas concentration for any ratio with the following formula:

$$\log(x) = \frac{\log(y) - b}{m} \tag{3.20}$$

However, in order to get the real value of the gas concentration according to the log-log plot we need to find the inverse log of x:

$$x = 10^{\frac{\log(y)-b}{m}} \tag{3.21}$$

Using equations 3.9 and 3.21, we will be able to convert the sensor output values into PPM [12].

We developed the code and flashed it into the Arduino Uno giving proper connections as mentioned in Figure 3.1.

4. Implementation. After connecting the ESP-01 successfully to the hotspot, it gets established with Thingspeak wesbite and the account API Key is written in Arduino Code which helps to save the data only to our account bearing the given API key. Thingspeak needs 15 seconds of refresh interval to push to the data. Figure 4.1 shows the field charts of MQ135 and MQ7 sensor values for the location where the experiment is conducted in PPM [7, 8]. Also Figure 4.1 shows the visualization charts for corresponding sensors.

Figure 4.2 shows the graphical analysis of the values collected with timestamping on X axis and Air Quality PPM on Y axis. It represents Linear Regression applied on the dataset collected from Thingspeak account in CSV format (Comma separated values). Using Python and Anaconda, linear regression is applied for the dataset and the graph as shown in Figure 4.2.

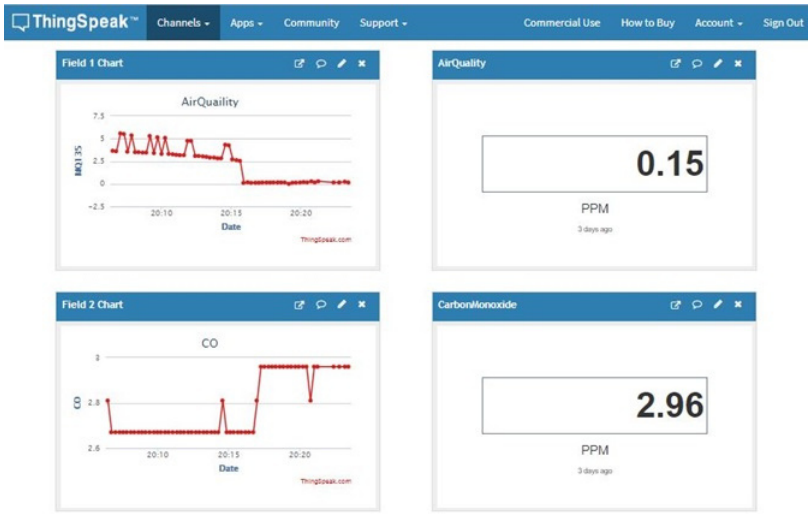

FIG. 4.1. *Output on Thingspeak*

FIG. 4.2. *Graph showing AirQuality*

- Mean Absolute Error: 1.238542891142161
- Mean Squared Error: 4.252972617980345
- Root Mean Squared Error: 2.0622736525447696

In Figure 4.2, X axis is Air Quality in PPM, Y axis is Time stamping in 24 Hour clock format and if the slope of the green color line is very high, AirQuality is very good, if the slope is 45 degrees, Air Quality is moderate, if the slope is very low towards X axis, then the Air Quality is very bad and needs to be concerned.

5. Conclusion. From all the above derivations, figures mentioned, connections diagram, we are able to calculate AirQuality in PPM. The problem with MQ135 sensor is that specifically it cannot tell the Carbon Monoxide or Carbon Dioxide level in the atmosphere, but the pros of MQ135 is that it is able to detect smoke, CO, CO_2, NH_4 as mentioned in Figure 3.3. So, just to tell the individual gases level particularly, we have used CO (Carbon Monoxide) MQ7 sensor. This paper also corrects the PPM calculations mentioned at Literature Survey. This project can be used both for indoor as well as outdoor. For indoor, we can make this kit as a compact device such that if every home started using the device, we can monitor the indoor air quality of a particular targeted area. Due to increasing air pollution, there is necessity to keep an eye on Indoor air quality too. But for outdoor purpose, certainly one sensor is not sufficient because one sensor has a sensitivity range of around 1 meter, so a network of sensors has to be deployed to monitor the outdoor air quality. Enough care is taken while calibrating the sensors. This paper also targets the Machine Learning analysis on the dataset collected.

6. Future Work. We can use one more sensor that tells the ozone layer status, but it costs very high. Also we can use PM2.5 laser dust sensor helpful for exclusively for vehicle and factory emissions sensing. Thingspeak

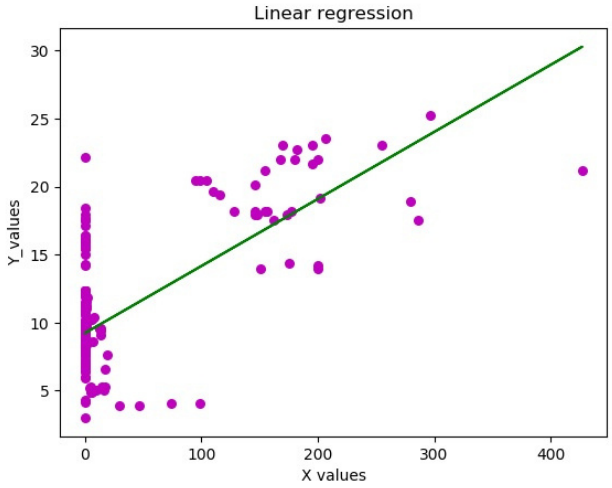

FIG. 4.3. *Linear Regression*

has a limitation that it requires 15-20 seconds for every push of the values which is not reliable. We plan to use another IoT platform mydevices cayenne which is very fast in showing the values from the Arduino that helps us to collect more values in the dataset. Cayenne also comes with a ready Android/iOS application. But it doesnt work with Arduino Uno rather works with only NodeMCU or Raspberry Pi. If we use NodeMCU, even the cost becomes less than the current setup. But the limitation in NodeMCU is, it has only one analog input pin, so we will use ADS1115 I2C 16Bit ADC as an analog extender for NodeMCU or a simple CD4051 8 to 1 Analog Multiplexer could easily overcome the problem of having only one analog input pin of NodeMCU (ESP8266). CD4051 Multiplexer is highly recommended as it is very cheap to purchase and easy to handle many sensors for NodeMCU. NodeMCU (ES8266) has inbuilt Wi-fi support (ESP-12E) and microcontroller. We can link this to the Facebook API using IFTTT, Webhooks and adafruit platform collectively, such that user can request the airquality via facebook messenger chat application and get the output on the screen using chatbot. Machine learning can also be implemented on the dataset such that we can predict the harmfulness of the airquality on people if the same bad airquality continues.

REFERENCES

[1] Tragos, E. Z., Angelakis, V., Fragkiadakis, A., Gundlegard, D., Nechifor, C. S., Oikonomou, G., Gavras, A. (2014). Enabling reliable and secure IoT-based smart city applications. In 2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS) (pp. 111-116). IEEE.

[2] Shah, J., Mishra, B. (2016). IoT enabled environmental monitoring system for smart cities. In 2016 International Conference on Internet of Things and Applications (IOTA) (pp. 383-388). IEEE.

[3] Pasha, S. (2016). ThingSpeak based sensing and monitoring system for IoT with Matlab Analysis. International Journal of New Technology and Research, 2(6).

[4] Khan, R., Khan, S. U., Zaheer, R., Khan, S. (2012). Future internet: the internet of things architecture, possible applications and key challenges. In 2012 10th international conference on frontiers of information technology (pp. 257-260). IEEE.

[5] Ranjan M., Rai Kumar, R. (2009), Understanding Parts per million in real time air quality index, Journal of Mathematics and advanced sciences, pp. 23-29.

[6] Kumar, N. S., Vuayalakshmi, B., Prarthana, R. J., Shankar, A. (2016, November). IOT based smart garbage alert system using Arduino UNO. In 2016 IEEE Region 10 Conference (TENCON) (pp. 1028-1034). IEEE.

[7] Kumar, S., Jasuja, A. (2017, May). Air quality monitoring system based on IoT using Raspberry Pi. In 2017 International Conference on Computing, Communication and Automation (ICCCA) (pp. 1341-1346). IEEE.

[8] Talari, S., Shafie-Khah, M., Siano, P., Loia, V., Tommasetti, A., Catalao, J. (2017). A review of smart cities based on the internet of things concept. Energies, 10(4), 421.

[9] Ma, Y., Yang, S., Huang, Z., Hou, Y., Cui, L., Yang, D. (2014, December). Hierarchical air quality monitoring system design. In 2014 International Symposium on Integrated Circuits (ISIC) (pp. 284-287). IEEE.

[10] Ahlgren, B., Hidell, M., Ngai, E. C. H. (2016). Internet of things for smart cities: Interoperability and open data. IEEE

Internet Computing, 20(6), 52-56.
[11] Cho, R. (2018, June 26). What you should know about air quality alerts. Retrieved from https://phys.org/news/2018-06-air-quality.html
[12] Systems, J. (2016, May 09). Understanding a Gas Sensor. Retrieved from https://jayconsystems.com/blog/understanding-a-gas-sensor
[13] Xiaojun, C., Xianpeng, L., Peng, X. (2015, January). IOT-based air pollution monitoring and forecasting system. In 2015 International Conference on Computer and Computational Sciences (ICCCS) (pp. 257-260). IEEE.
[14] Fang, S., Da Xu, L., Zhu, Y., Ahati, J., Pei, H., Yan, J., Liu, Z. (2014). An integrated system for regional environmental monitoring and management based on internet of things. IEEE Transactions on Industrial Informatics, 10(2), 1596-1605.
[15] Zheng, K., Zhao, S., Yang, Z., Xiong, X., Xiang, W. (2016). Design and implementation of LPWA-based air quality monitoring system. IEEE Access, 4, 3238-3245.
[16] Rushikesh, R., Sivappagari, C. M. R. (2015). Development of IoT based vehicular pollution monitoring system. In 2015 International Conference on Green Computing and Internet of Things (ICGCIoT) (pp. 779-783). IEEE.
[17] Marques, G., Pitarma, R. (2016). An indoor monitoring system for ambient assisted living based on internet of things architecture. International journal of environmental research and public health, 13(11), 1152.
[18] Manna, S., Bhunia, S. S., Mukherjee, N. (2014). Vehicular pollution monitoring using IoT. In International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014) (pp. 1-5). IEEE.
[19] Kang, B., Park, S., Lee, T., Park, S. (2015). IoT-based monitoring system using tri-level context making model for smart home services. In 2015 IEEE International Conference on Consumer Electronics (ICCE) (pp. 198-199). IEEE.
[20] Ibrahim, M., Elgamri, A., Babiker, S., Mohamed, A. (2015). Internet of things based smart environmental monitoring using the Raspberry-Pi computer. In 2015 Fifth International Conference on Digital Information Processing and Communications (ICDIPC) (pp. 159-164). IEEE.
[21] Xiaojun, C., Xianpeng, L., Peng, X. (2015). IOT-based air pollution monitoring and forecasting system. In 2015 International Conference on Computer and Computational Sciences (ICCCS) (pp. 257-260). IEEE.
[22] Atzori, L., Iera, A., Morabito, G. (2010). The internet of things: A survey. Computer networks, 54(15), 2787-2805.
[23] Somula, R., Sasikala, R. (2018). Round robin with load degree: An algorithm for optimal cloudlet discovery in mobile cloud computing. Scalable Computing: Practice and Experience, 19(1), 39-52.
[24] Somula, R. S., Sasikala, R. (2018). A survey on mobile cloud computing: mobile computing+ cloud computing (MCC= MC+ CC). Scalable Computing: Practice and Experience, 19(4), 309-337.
[25] Somula, R., Sasikala, R. (2019). A load and distance aware cloudlet selection strategy in multi-cloudlet environment. International Journal of Grid and High Performance Computing (IJGHPC), 11(2), 85-102.
[26] Somula, R., Sasikala, R. (2019). A honey bee inspired cloudlet selection for resource allocation. In Smart Intelligent Computing and Applications (pp. 335-343). Springer, Singapore.
[27] Somula, R., Sasikala, R. (2019). A research review on energy consumption of different frameworks in mobile cloud computing. In Innovations in Computer Science and Engineering (pp. 129-142). Springer, Singapore.
[28] Ramasubbareddy, S., Sasikala, R. (2019). RTTSMCE: a response time aware task scheduling in multi-cloudlet environment. International Journal of Computers and Applications, 1-6.
[29] Ramasubbareddy, S., Vedavasu, G., Krishna, G., KB, N., Savithri, A. (2019). PIOCM: Properly Identifying Optimized Cloudlet in Mobile Cloud Computing. Journal of Computational and Theoretical Nanoscience, 16(5-6), 1967-1971.
[30] Basu, S., Kannayaram, G., Ramasubbareddy, S., Venkatasubbaiah, C. (2019). Improved Genetic Algorithm for Monitoring of Virtual Machines in Cloud Environment. In Smart Intelligent Computing and Applications (pp. 319-326). Springer, Singapore.
[31] Ellaji, C. H., Jayasri, P., Ramasubbareddy, S., Kannayaram, G. (2019). Cypher Query Processing for Secure Data Provenance in Cloud. Journal of Computational and Theoretical Nanoscience, 16(5-6), 2517-2522.
[32] Nalluri, S., Ramasubbareddy, S., Kannayaram, G. (2019). Weather Prediction Using Clustering Strategies in Machine Learning. Journal of Computational and Theoretical Nanoscience, 16(5-6), 1977-1981.

Edited by: Dharm Singh Jat
Received: May 11, 2019
Accepted: Aug 20, 2019