Enabling Edge Computing in an IoT-Based Weather Monitoring Application

Chapter · April 2020

DOI: 10.1007/978-981-15-2329-8_18

Sudhir Kumar Sharma
Institute of Information Technology & Management New Delhi

Enabling Edge Computing in an IoT based Weather Monitoring Application

Kavita Srivastava[1] and Sudhir Kumar Sharma[2]

[1, 2] Institute of Information Technology & Management, Janakpuri, Delhi, India
kavita332211@gmail.com
Sharmasudhir08@gmail.com

Abstract. Internet of Things (IoT) applications employ several sensors for gathering data. These sensors are often placed at hard to reach locations since they need to be left unmonitored. Sensors collect data in real time and send it to a cloud server for further processing. Often the sensors generate a large volume of data which is redundant in nature. Transferring sensor data to a cloud server for processing leads to high bandwidth consumption, delay, increase in operational cost and data security issues. Edge computing allows the sensor data to be stored and analyzed on an edge device and only the data summary is sent to the cloud server. In this paper, an edge computing approach for managing and analyzing data in a weather monitoring application is proposed. The application has been built using a Raspberry Pi system. The computation has been performed by creating Microsoft Azure IoT Hub resource and Microsoft Azure IoT Edge solution. Results have been compared with AzureML Cloud platform.

Keywords: Edge Computing, IoT Hub, IoT Edge, Internet of Things, Device Provisioning System, Edge Runtime, Edge Modules.

1 Introduction

Edge computing is an enabling technology that leverage Internet of Things (IoT) applications by providing a wide range of services. An IoT application solution is built by using one or more sensor nodes comprising of microcontroller units (MCU) and some provisioning of data storage, analysis and prediction. Often these facilities are provided by a cloud computing service. This makes the centralization of resources in IoT application. This scheme has certain drawbacks. It is not suitable for IoT applications which require quick response. For instance suppose a weather monitoring application is designed to collect temperature and humidity data detects sudden rise in temperature and causes an alarm to ring. If the data has been sent to cloud service for analysis it would cause delays making the system ineffective.

Edge computing performs the analysis of data near to device location. It means the analysis has been done at the place where data is collected. Edge computing benefits the IoT applications in several ways.

- It makes application distributed in nature. We can aggregate data of several nodes which are placed in neighborhood of each other on a single edge device. Hence, for one IoT application consisting of thousands of nodes we can use several edge devices.

- Employing edge devices results in less data to be transferred to the cloud thereby reducing the requirement of data storage on cloud and less utilization of network bandwidth.
- The edge computing makes application more secure since data is stored only on the edge device and can be processed offline.
- The application becomes more scalable since we can easily employ more sensor nodes targeted to an existing edge device or to a new edge device.

In this study we have proposed an Edge Computing enabled weather monitoring application built from Raspberry Pi board and DHT11 Temperature and Humidity sensor. We have used Microsoft Azure IoT Edge for stream analysis and Microsoft AzureML for comparative analysis.

Rest of the paper is organized as follows. Section 2 provides the related research in edge computing. Section 3 discusses the architecture of edge computing enabled IoT application. Section 4 provides workflow of implementation and section 5 provides experiments performed and section 6 provides results and discussion.

2 Related Work

In the context of IoT, Edge Computing has not been explored much by the research community. Dupont et. al, 2017 presented Cloud4IoT platform for IoT functions migration horizontally and vertically [1]. They explained two use cases for healthcare and remote engine diagnostic for roaming and offloading respectively. Higashino et. al, 2017 [2] have pointed out a number of challenges in Edge Computing like creation technology, networking technology, processing technology and content curation technology. They discussed their ongoing research on disaster mitigation using edge computing concepts. Pan et. al, 2017 discussed IoT applications benefited from Edge Cloud. They discussed several benefits and challenges of using edge cloud infrastructure [3]. Babou et. al, 2018 proposed a Home Edge Computing (HEC) architecture for providing data storage and processing capability near user locations [4]. They demonstrated their work on EdgeCloudSim simulator. Ramljak et. al, 2018 presented Baysian Reasoning based framework for minimizing data transfer between edge and cloud [5]. They provided BelifeCache modular framework to support their work. Zeng et. al, 2018 proposed a system for edge-based computing on visual sensors for face recognition application [6]. Hsieh et. al, 2018 presented a Docker container based managed platform for smart city applications [7]. They described deployment of three different applications of air quality monitoring, image recognition and sound classification. Hu et. al, 2018 proposed a retail Point-of-Sale system based on blockchain technology [8]. Here a node in blockchain network works as edge computing server. Sanchez et. al, 2018 proposed a system of performing computation through Convolution Neural Networks (CNN) on edge devices which are placed near the camera sensors [9]. Yu et. al, 2018 described edge computing based on containers which provide lightweight virtualization [10]. They have used Docker container and deployed the container through Kubernetes orchestration tool. Merlino et. al, 2019 suggested an OpenStack based middleware platform [11]. They described how containers at Edge, Fog, and Cloud level can be discovered and combined.

3 Architecture

An IoT solution is basically comprised of three parts – things, insight and action. Things include devices, sensors, industrial PCs, gateways, and everything collecting the data. Insights include real time analysis of data, historical insight, and making predictions. IoT projects are still complex to start with. They are incompatible with existing infrastructure and difficult to scale. Azure IoT provides secure, fast, open, and scalable solutions. With azure IoT we can enhance and scale our IoT applications effortlessly. Azure IoT provides a way to establish a bidirectional communication between cloud to device and device to cloud messages. Fig. 1 shows the layered architecture of edge computing enabled IoT application.

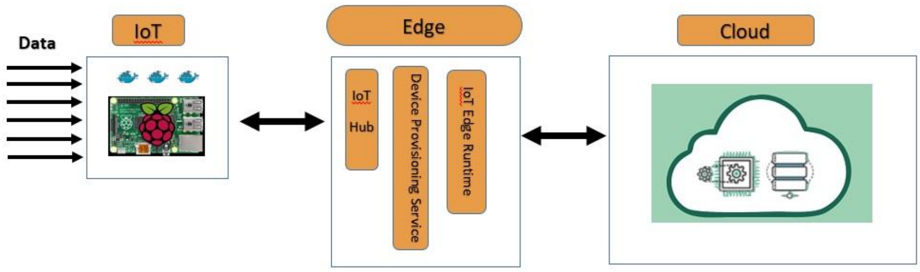


Fig. 1. Architecture of Edge Computing enabled IoT Application.

The architecture of application has three layers – Device layer, Edge layer and Cloud layer. These layers are explained below in detail.

3.1 Device Layer

Device layer is composed of sensors and microcontrollers. Sensors directly collect data from the environment. The sensors are employed for a wide range of use cases. We need to install Docker container on Device node.

3.2 Edge Layer

Edge layer has three components – Iot Hub, Device Provisioning Service and IoT Edge Runtime. IoT hub provides services to devices and applications. It is responsible for establishing communication among sensor devices and backend application. We can monitor and control devices using IoT hub, determine device status and detect device failure from cloud. Device Provisioning Service is a way to register devices on cloud with zero touch. Device Provisioning Service can provision up to millions of devices. Azure Device Provisioning Service eliminates the need of human intervention completely. It also provides security and scalability. Device Provisioning Service provides automatic configuration and load balancing. Edge runtime is responsible for providing basic services such as management, connectivity and security for devices which may be operating offline. Capabilities of runtime are provided in the form of modules which perform specific actions such as stream analytics or storage solution. Edge runtime modules can be chained together for pipeline processing of data. Modules are implemented as Docker containers. Custom modules can be written in the language of your choice such as C#, Java or Python. All devices connected through IoT hub run IoT Edge runtime. Modules in Edge runtime are deployed according to application use case.

3.3 Cloud Layer

Edge layer can provision millions of devices and filter the streaming data before it reaches the cloud layer. However, cloud layer is required for further processing of data. Cloud layer provides Big Data storage, business analytics and machine learning.

In the next section, implementation of the application has been explained.

4 Implementation

An IoT solution requires IoT sensor nodes and a backend application running on cloud. Microsoft Azure IoT solution provides the facility for controlling and monitoring IoT devices. The communication between device and backend need to be bidirectional. For instance, in case of Weather Monitoring Application, sensor nodes collect data and send to the cloud. If any abnormal behavior is detected like sudden rise in temperature along with sudden fall in humidity, it may be an indication of fire outbreak. The backend cloud application must send a notification to the device causing an alarm to ring as well as a notification on user's mobile phone should also be sent.

Microsoft Azure provides a number of IoT related services including IoT Central (a SaaS solution), IoT solution accelerator (PaaS solution), IoT hub, Device Provisioning Service, IoT Edge, Azure Digital Twins, Time Series Insights and Azure Maps for providing geographic information to applications. IoT Edge is suitable when we need to analyze data on the IoT devices rather than using cloud service to analyze data. By doing this, we can perform the computation on edge and only few messages need to be send to the cloud for storage. In our Weather Monitoring application we have utilized the service of Azure IoT Edge. The hardware part of our application consists of Raspberry Pi board and DHT 11 Temperature and Humidity Sensor. We have connected the sensor with Raspberry Pi board and installed the Raspbian Operating System. Then, we have installed Docker container on Raspberry Pi followed by IoT Edge Runtime. After that we can login into Azure account and create IoT Hub resource, create a resource group, select a region nearest to the device location and specify size and scale. We can obtain Connection String to be used in the application code. Then we registered the device by specifying Device Id. Then we have linked Device Provisioning Service to IoT Hub. For that, we have created a C# console application and installed Microsoft.Azure.Devices.Provisioning.Service library. Since we want to analyze data on the device edge rather than on cloud, we have configured IoT Edge. The workflow of application configuration and deployment is shown in fig. 2. The application starts by fetching data from device. It computes the average of both temperature and humidity values. If the difference between current value and average more than 10 a message is generated and sent to the cloud. Next, the application plots the data fetched. Fig. 3 shows the computation flowchart.

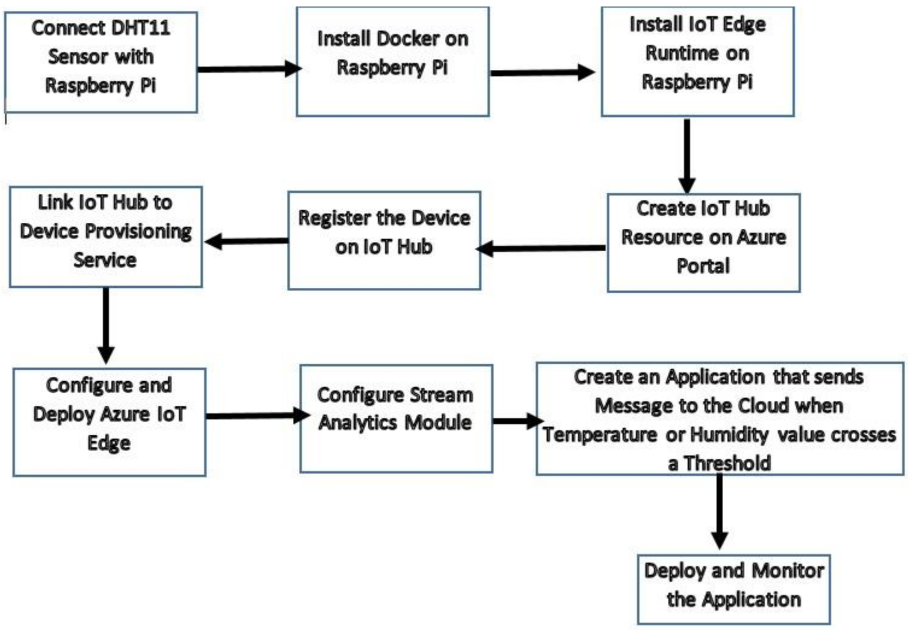


Fig. 2. Application Workflow.

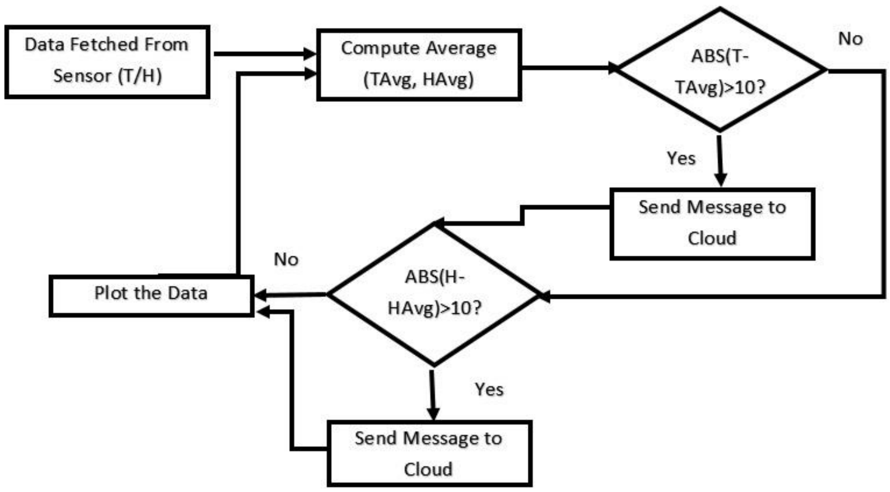


Fig. 3. Computation Flowchart.

5 Experiments

The device data has been collected for a period of one month in winter season of January 12, 2019 to February 10, 2019. It has been observed that variation in temperature and humidity values is rare for a particular day. This results in very few messages being sent to cloud for further processing. Plot of Day 1 shows the effect of turning on a heating device which resulted in some increase in temperature and decrease in humidity value. The same pattern followed for each day.

The results have shown that in spite of collecting large amount of data by sensor node, only few messages have been send to the cloud backend. The data has been analyzed at IoT Edge before sending to the cloud. In order to perform a comparative analysis of computing on Edge vs. computing on Cloud, we have performed regression analysis on cloud by utilizing AzureML platform or machine learning. The cloud experiment model is shown in fig. 4.

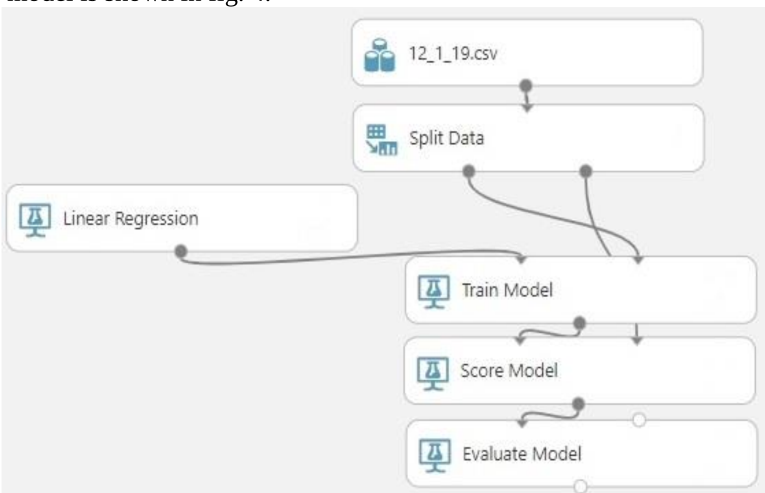


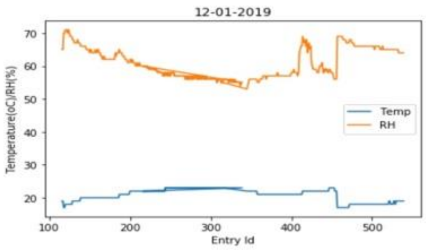
Fig. 4. Regression Analysis with AzureML

The data has been uploaded using Datasets module of AzureML Studio. Next, in Experiment module, this dataset is divided into training set containing 80% of total data and test set containing remaining 20% of data using Split Data function. A Linear Regression Model is added to the experiment with Online Gradient Descent method with learning rate=0.1 and number of training epochs=10. A Train Model function is added and Temperature column is selected from dataset. A Score Model function is added to the experiment to find predicted values in test dataset. Finally, an Evaluation Model function is added to measure the accuracy of trained dataset. The experiment is executed and results are shown in fig. 5.

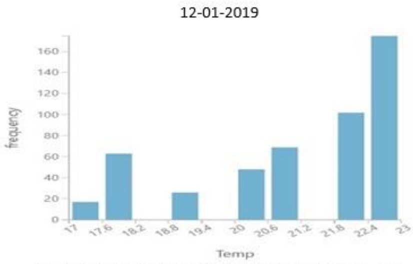
6 Results and Discussion

Fig. 5 (a) shows the plot of telemetry data obtained on Edge device. Fig. 5(b) shows the data of the same day visualized on the cloud server. Fig. 5(c) indicates that Scored Labels have shown very small variation in data with standard deviation of 0.1675. Fig. 5(d) shows the Evaluation Metrics.

The results indicated that analysis on edge and cloud are mostly similar however Edge computing provides several benefits over cloud computing. Edge computing results in substantial reduction of bandwidth utilization and latency. It makes our application fast and cost effective.



(a) Plot of Temperature and Humidity Data Fetched by Edge Device.



(b) Temperature data visualization on cloud server.

Mean	15.0058
Median	15.0147
Min	14.6496
Max	15.2909
Standard Deviation	0.1675
Unique Values	96
Missing Values	0
Feature Type	Numeric Score

(c) Statistics obtained from Scored Labels

Mean Absolute Error	6.114204
Root Mean Squared Error	6.406236
Relative Absolute Error	3.755654
Relative Squared Error	11.509944
Coefficient of Determination	-10.509944

(d) Metrics obtained from Evaluation Model

Fig. 5. Comparative Analysis on Edge and Cloud

7 Conclusion and Future Work

The Edge Computing has emerged as a technology that can be utilized in between the sensors node collecting data from environment and the cloud computing services. Edge computing has tremendous potential of making our IoT applications work efficiently by causing the resource utilization efficiently. Edge computing results in making IoT application distributed in nature by managing the data near the location where it is generated.

In this study we have used Microsoft Azure IoT Edge for edge computing. We have employed temperature and humidity sensor with Raspberry Pi board and collected the data. The data has been processed by the Edge Device which can be a laptop or PC. The results have shown that very few messages have been transferred to the cloud service indicating abnormal conditions. The Edge device has successfully analysed the data. The resulting application is fast and cost effective.

Our future work will involve integrating more sensor nodes for collecting data and employing other edge modules.

References

1. Dupont, C., Giaffreda, R. and Capra, L., 2017, June. Edge computing in IoT context: Horizontal and vertical Linux container migration. In 2017 Global Internet of Things Summit (GIoTS) (pp. 1-4). IEEE.
2. Higashino, T., Yamaguchi, H., Hiromori, A., Uchiyama, A. and Yasumoto, K., 2017, June. Edge computing and iot based research for building safe smart cities resistant to disasters. In 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS) (pp. 1729-1737). IEEE.
3. Pan, J. and McElhannon, J., 2017. Future edge cloud and edge computing for internet of things applications. IEEE Internet of Things Journal, 5(1), pp.439-449.
4. Babou, C.S.M., Fall, D., Kashihara, S., Niang, I. and Kadobayashi, Y., 2018, June. Home Edge Computing (HEC): Design of a New Edge Computing Technology for Achieving Ultra-Low Latency. In International Conference on Edge Computing (pp. 3-17). Springer, Cham.
5. Ramljak, D., Tom, D.A., Voigt, D. and Kant, K., 2018, June. Modular Framework for Data Prefetching and Replacement at the Edge. In International Conference on Edge Computing (pp. 18-33). Springer, Cham.
6. Zeng, J., Li, C. and Zhang, L.J., 2018, June. A Face Recognition System Based on Cloud Computing and AI Edge for IOT. In International Conference on Edge Computing (pp. 91-98). Springer, Cham.
7. Hsieh, Y.C., Hong, H.J., Tsai, P.H., Wang, Y.R., Zhu, Q., Uddin, M.Y.S., Venkatasubramanian, N. and Hsu, C.H., 2018, April. Managed edge computing on Internet-of-Things devices for smart city applications. In NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium (pp. 1-2). IEEE.
8. Hu, B., Xie, H., Ma, Y., Wang, J. and Zhang, L.J., 2018, June. A Robust Retail POS System Based on Blockchain and Edge Computing. In International Conference on Edge Computing (pp. 99-110). Springer, Cham.
9. Sanchez, J., Soltani, N., Kulkarni, P., Chamarthi, R.V. and Tabkhi, H., 2018, June. A reconfigurable streaming processor for real-time low-power execution of convolutional neural networks at the edge. In International Conference on Edge Computing (pp. 49-64). Springer, Cham.
10. Yu, Z., Wang, J., Qi, Q., Liao, J. and Xu, J., 2018, June. Boundless Application and Resource Based on Container Technology. In International Conference on Edge Computing (pp. 34-48). Springer, Cham.
11. Merlino, G., Dautov, R., Distefano, S. and Bruneo, D., 2019. Enabling Workload Engineering in Edge, Fog, and Cloud Computing through OpenStack-based Middleware. ACM Transactions on Internet Technology (TOIT), 19(2), p.28.