Intelligent and Scalable IoT Edge-Cloud System

Shifa Manihar[1], Ravindra Patel[3]
Department of Computer Science
UIT RGPV, Bhopal, India

Tasneem Bano Rehman[2]
Department of Computer Science
Sage University, Bhopal, India

Sanjay Agrawal[4]
Department of Computer Science
NITTTR, Bhopal, India

***Abstract*—Scalability is an utter compulsory for the success of the IoT's unprecedentedly growing network. The operational and financial bottlenecks allied with growth can be overwhelming for those peeping to integrate IoT solutions. As the IoT technology proceeds, so is the scale of operations desired to arrive at a wider target region. Breakdown may take place not because of device's ability to scale, but due to data scale. As more devices are being incorporated, more data/information will be amassed, stored, processed, and scrutinized. The volume of this collection simply cannot be managed from a single edge device by deploying vertical approach. When starting small, it's important to peep into the future and anticipate growth. Companies that can't adapt to unpredictable market changes will fold without the right IoT architecture in place. Therefore, a scalable IOT framework has been proposed in the paper, which will provide load balancing or scalability by deploying the provisions of horizontal scalability for the system. The framework will be utilizing SOM for the purpose of classifying applications (whether delay sensitive or delay insensitive), so that proper decisions can be made based on the incoming data (typically signals) and if edge gets over flooded with the data, then edge is scaled to instigate the other edge for computing additional requests . The proposed system is termed as intelligent because its algorithm empowers the edge to take decision and classify applications based on the type of requirement of the application.**

Keywords*—Scalability; internet of things; self organizing map; edge; horizontal scalability

I. Introduction

Internet of Things (IoT) was first introduced to the community in 1999 for supply chain management [1], and then the concept of "making computer sense information without the aid of human intervention" was widely adapted to other fields such as healthcare, home, environment, and transports [2], [3]. A prediction has been made by Analysts Firms that there will be around 2020 billion of active connected products. Now with IoT, we will arrive in the post-cloud era, where there will be a large quality of data generated by things that are immersed in our daily life, along with lots of applications deployed at the edge to consume these data.

IoT is an environment that encompasses the objects (living and non-living) communicating with each other by means of Internet. The basic purpose of IoT is to induce intelligence into any object and providing it with the decision making capability. Here we lay emphasis on the capacity building capability of any device. We do not need to have storage within the object itself. The whole system relies on offloading the computing and storage on to the network by the IoT devices.

Edge computing refers to the technologies empowering computation to be performed at the edge of the network. Here we define "edge" as any computing and network resources along the path between cloud data centers and data sources. For example, a smart phone is the edge between body things and cloud, a gateway in a smart home is the edge between home things and cloud, a micro data center and a cloudlet [4] is the edge between a mobile device and cloud. The main logic behind edge computing is that computing should happen in close proximity of mobile devices or sensors. Data is increasingly produced at the edge of the network; therefore, it would be more efficient to also process the data at the edge of the network. Previous work such as micro datacenter [5], [4], cloudlet [6], and fog computing [7] has been introduced to the community because cloud computing is not always efficient for data processing when the data is produced at the edge of the network.

Acting as a common interface or middleware for unprecedentedly increasing number of disparate data from variety of IoT devices, edge computing has gained a tremendous impetus by many researchers from the last decade. It not only provide a platform for heterogeneous and unpredictable data from n number of IoT devices while communicating with the cloud but also act as a computational hub which encompasses intelligence for decision making and encounters the bottleneck of latency and power consumption. Performing computation at the end devices had the ramifying effect on the global power consumption by the IoT devices. Now the computational burden of IoT devices has been offloaded to edge servers lying to the vicinity of the IoT devices and hence alleviating the energy conservation. As per the survey conducted by IEA based on 4E Agreement, the standby power consumption by the IoT devices and their respective edges globally is estimated to be 46 TWh by 2025 [8]. Thus recent researches are laying great emphasis on conserving this standby or idle power consumption by IoT devices by enhancing new technological developments. We often encounter the problem of limited storage on edge servers as contradictory to unlimited storage at the cloud. This bottleneck of edge has attracted the users towards the proliferating use of cloud. But this issues of limited storage at edge can be tackled if data approaching edge server is compressed beforehand, hence enhancing the capacity of the edge to accommodate more number of applications. Edge computing incorporates the intelligence, processing power and communication capabilities of an edge gateway or appliance directly into devices [19].

Scale, by definition, refers to "the capability of a system, network, or process to handle a growing amount of work

(service requests), or its potential to be enlarged in order to accommodate that growth" [9]. The Scalability is the phenomenon which means accommodating and servicing the ever increasing network traffic without creating a burden on an edge server. Scalability can be two dimensional. One dimension involves deploying more and more resources to partition the incoming service requests among the multiple servers to handle the requests. This is termed as horizontal scalability. This dimension involves partitioning the tasks based on the number of incoming requests. The second dimension involves partitioning the incoming requests based on action without deploying extra resources. This is termed as vertical partitioning. This paper emphasizes on the first dimension of the scalability.

The rest of the article is organized as follows: Section I briefly discuss about some of the existing scalable based edge IoT systems. In Section II, we describe our proposal of an intelligent edge system in details. In Section III, implementation and simulation of the algorithm is discussed. Finally, the paper is concluded in Section IV.

II. Related Works

Many researchers have been working on the proliferation of edge computing and have proposed various fog computing layered architectures and paradigms. Shreshth Tuli et al. [20] proposed a framework, known as EdgeLens, adapts itself to the application or user requirements to provide high accuracy or low latency modes of services. They tested the performance of the software in terms of accuracy, response time, jitter, network bandwidth and power consumption and show how EdgeLens adapts to different service requirements. Gusev [10] has suggested the concept of 'Everything as a Service', and has emphasized on the significance of the distribution of smaller servers in front of the central server and in the vicinity of the user, hence promoting the concept of edge computing. Wei Yu [11] has classified various edge computing architectures and compared their influence on the performance of various parameters of IoT networks such as network latency, storage, bandwidth occupation, energy expenditure, and the overhead incurred and various security issues such as availability, integrity, availability and confidentiality.

Jalali [8] has compared various literatures which have been proposed to curtail energy consumption in the IoT devices and has proposed various fog computing techniques which can alleviate power consumption in various IoT devices. Various factors such as access network technologies; idle power consumption of IoT devices, application type, virtualization and network management which lead to higher power consumption has been discussed. Case study on the effect of using Fog computing along with microgrid to reduce energy consumption has been proposed. Fei Li [12] in his work has proposed IoT PaaS architecture promoting vertical scalability by utilizing computing resources and middleware services on the cloud. Domain mediation which is deployed for domain specific control application by the solution providers has been suggested. Two use cases have been put forward in building management domain to show the results.

Sarkar [13] has proposed a layered architecture for distributed environment that utilize scalability which is coupled with cognitive capabilities that promoted intelligent decision making. In his work, an usage control policy model has been proposed to endorse security and privacy in the distributed environment. Sarkar [14] has also proposed a distributed architecture which emphasized interoperability, heterogeneity and scalability, security and privacy of the IoT devices, and named such architecture as Distributed Internet like Architecture (DIAT). In this layered architectures, each layer is classified based on the similar function, forming the hierarchical structure of functionalities.

Onoriode [15] has compared various existing architectural frameworks for integrating various heterogeneous IoT devices and has put forward a viable solution based on micro service. The proposed IoT integration framework takes advantage from an cognitive API layer that makes use of an external service assembler, service auditor, service monitor and service router module to direct service publishing, subscription, decoupling and service amalgamation within the architecture. Ju Ren [16] has proposed a Transparent Computing based IoT architecture to build scalable IoT platforms. With the help of case study, he has build scalable lightweight wearables deploying the proposed architecture.

Kajal [17] has worked for compressing the data especially video frames using Self Organizing Map (SOM) and stored the output feature using Hopfield networks. The frames are preprocessed by making it to pass through the SOM that outputs the helpful features diminishing redundant and irrelevant tenets [17]. Hopfield network is deployed in turn, to store various output patterns. The compressed data can be restored by increasing the dimensions of the frame.

III. Learning based Scalable IoT System (LSI)

Taking into account the issue of idle power consumption by IoT devices, a framework has to be proposed that can be deployed to reduce the idle time of edge servers. Also as mentioned above, one of the basic tasks of deploying the edge server is to minimize the latency when addressing several simultaneous requests by IoT devices. Delay sensitive applications are time sensitive thus possess high priority, need to be addressed at first instance. This paper proposes a mechanism that provides intelligence (a learning mechanism) to the edge server by exploiting which the type of application can be classified whether delay sensitive or delay insensitive. This paper also addresses the issue of scalability when edge server gets flooded with delay sensitive applications taking into account horizontal scalability.

Objects (e.g. sensors) are provided with IP address, which aid communication with each other. Various sensors send numerous and variety of incompatible data to the edge for processing. In this paper, the proposed system employ Self organizing map for classifying the delay sensitive data and the delay insensitive data. The delay sensitive data is immediately forwarded to the edge server for further processing. The SOM not only clusters the type of data but also compresses the incoming data, hence enabling the edge server to accommodate more data as edge is limited by its storage capacity, and therefore data compression is of vital importance to edge computing. Whenever the edge gets enormous delay sensitive data, its throughput declines, therefore this paper laid

emphasis on providing additional edge servers, which at sleep initially. It activates only when data overflow at the server 1. This additional server can also be deployed to handle delay sensitive data. This server comes to play only at high peak times, but at normal times, it is just set to sleep. This can be accomplishes through the use of Aneka Auto-scaling provision. Through this mechanism load balancing has been achieved without much affecting the energy consumption rate. The comparison of LSI with the already existing researches and algorithms has been listed in the Table I.

TABLE I. COMPARISON OF VARIOUS IOT ARCHITECTURES

IoT Features	DIAT [14]	Transparent Computing Architecture [16]	IoT PAAS [12]	LSI
Horizontal Scalability	Yes	Yes	Not Known	Yes
Vertical Scalability	No	No	Yes	No
Energy Consumption	Not Known	Yes	Not Known	Yes
Response Delay	Not Known	Yes	Not Known	Yes
Context Aware Service Support	Yes	Yes	Not Known	Yes
Intelligence	Yes	Not Known	Not Known	Yes

A. Self Organizing Map (SOM)

With the purpose of data visualization which in turn endorse in realizing high dimensional data by short sizing the dimensions of the data to a map, SOM is of crucial weight and has grabbed the attention of many scholars and researchers. SOM deploys competitive learning paradigm to cluster data by amassing similar data altogether. Thus SOM reduces data dimensions and displays similarities among data. With the aid of SOM, clustering is accomplished through having several units compete or race for the current element. Once the SOM is provided with the data, the network of artificial neurons goes through competitive training. The weight vector of the unit that is found nearest to the current element is proclaimed to be the winning or active unit. While undergoing training, SOM preserves the neighborhood relationship that persists among the input data sets. As it approaches near to vicinity of the input object, the weights of the winning unit are put to adjustment as well as its neighbor's nodes.

In contradiction to other learning technique, SOM do not employ target vector. A SOM learns to classify the training data with no external interference. Normally, Euclidean distance is the most rapidly used method for determining the distances of the input vector from the weight vector. The input vector whose distance is shortest from the weight vector is proclaimed to be the winner. SOM algorithm can be summarized as follows:

1) Initial steps involve the weights to be provided with arbitrary weights.

2) Next we choose any random input vector.

3) We examine each node by evaluating which node's weights are found to be closest to the input vector. We call this winning node as Best matching node.

4) Next we find out the neighbors of the winning node.

5) The winning weight is rewarded with becoming more like the sample vector. The neighbors tend to be more like the sample vector. The closer a node is to the winner node, the more its weights adjusted and the farther away the neighbor is from the winning node, the less it learns.

6) Go to step 2 for next samples.

B. LSI Working

Raw data and/or information in the form of signals are amassed based on whatever is sensed by the Sensors and actuators. The most clear-cut demonstration of a time series signal is based on its time-domain form, and then distances between time series relate to differences between the time-ordered measurements themselves. A traditional approach to the classification of time series problem used in data mining domain, focus on classifying short time series which encode meaningful patterns. This is termed to as instance based classification. Here new time series are classified by matching them to similar instances of time series with a known classification. These incompatible data has to be preprocessed and normalized before being used as input for any further action. Collected sensed data is forwarded to the edge device, where it first encounters Self organizing map (SOM) where the learning is basically relied upon only the input data and such unlabelled data is independent of the desired output. We call such learning mechanism as unsupervised learning. Since SOM tend to respond to several output categories after performing training. Hence an additional mechanism is added, so that SOM can respond to which one unit will respond (this is termed as Competition, often called Winner take all strategy). The number of output unit has been limited to two, either cluster into delay insensitive data or to delay sensitive data. During training, SOM identifies which output unit best matches the input sensed data. The sensed data is preprocessed and input to the SOM situated at the edge. Kohonen feature map is created, which determine the winner unit (whether delay insensitive or delay sensitive data unit). Kohonen feature map can also be trained to compress the sensed input data, so that more number of data can be accommodated at the edge, hence compensating the limited size of the edge server to a limit. Based on the winner unit, data is forwarded to either Queue 1 or Queue 2. Queue 1 is maintained for delay insensitive data, whereas Queue 2 is maintained for delay sensitive data. Kohonen network will undergo two phases: training and testing. During the training phase, the SOM is trained to form the cluster of inputs based on the similarity (or minimum Euclidean distance). During the test phase, we evaluate the performance by measuring to what extent our network classifies delay sensitive and delay insensitive data.

Data stored in the Queue 2 is basically stored for processing at the edge device. A variable counter is maintained on the number of the requests arriving at a time and being served by the edge server. If the counter exceeds 30, then additional edge is initialized and any further request is forwarded to this edge 2 provided Aneka auto-scaling.

Queue 2 is provided with a threshold (T) on its size. Whenever overflow occurs i.e. Queue 2 reaches its threshold (T), an alternative Queue 3 is maintained which will store the further data. With the instigation of Queue 3, an additional edge which is maintained in order to handle the traffic at the edge device 1, is also provoked. This edge server 2 will process the further data arriving after the overflow at the Queue 2. Initially, additional edge device is at sleep, and provoked only after the initialization of the Queue 3. After processing the data at Queue 3, the edge server 2 can be disabled by the Aneka and again goes to sleep. This is done to curtail the extra energy consumption, which would have been expended due to the idle state of the edge server 2 during its free time. Thus, this feature ensures horizontal scalability without much energy expenditure.

Queue 1 is maintained for the delay insensitive data, hence data in the Queue 1 can either be directly forwarded to the cloud for processing, or additionally we provide a mechanism called cycle steal. Sometimes, while processing delay sensitive data at edge server 1 and edge server 2, the processor may be found in an idle state. During such time, data in Queue 1 or training at the SOM can be processed at these edge servers. This mechanism not only enhances CPU throughput, but also saves energy consumption due to idle state at the edge servers.

C. Inputs

This proposed research encompasses the following application areas: Hospital, Surveillance, Organization, Inventory, and Smart Home. Each of these applications is facilitated with the following types of Sensor: Smoke, Temperature, Proximity, Thumb, and Smoke. The utility of each sensor varies according to the application areas, for example the data sensed by smoke sensors is of highest priority and has to be processed immediately; the thumb sensors finds importance for the Organization but is of less importance if found in the Inventory; the temperature sensor is critical for the applications like Hospital and Inventory, etc. It is proposed to process the data sensed by the higher priority sensor (Priority vary according to the application area) in the edge, forwarding the rest data to the cloud. The inputs are basically signals which can be represented in time series.

D. LSI Algorithm

The LSI algorithm is explained in detail as follows:

```
Here T (threshold) = 30     /*Threshold T is set to be 30 keeping in view the processing capacity of the edge & IoT sensors are low power devices and require less processing power and an Edge can easily process at least 30 requests at a time without causing Overloading. */
If (Sensed_Data == True)
{
Initialize SOM;
SOM← Sensed_Data;           /*SOM evaluates the Euclidean distance of the signals with the already stored ones.*/
If Output(SOM)==1
        Queue1←Sensed_Data;     //Send data to the Cloud.
If Output(SOM)==2
        Counter++;
        If (Counter<=T) /*Check whether counter value <=Threshold */
                Queue2←Sensed_Data; /*Send data to the Edge Processor 1 */
        else
                Queue3←Sensed_Data; /* Send data to the Edge Processor 2 */
If Processing(Sensed_Data)==True /*If the processing of the request is completed.*/
                Counter--;
If (Queue3==Empty)//Check if no task at Edge processor 3
        Dissolve(Queue3);       /*Disable the Edge processor 3 if under load.*/
        }
```

This particular system and algorithm can be well illustrated in Fig. 1.

E. Multi-Application use Case

To validate architecture, a composite use case is described in this section. The instances consist of various IoT applications like smart city, smart transportation, smart home, smart healthcare, smart retail and supply chain system, smart biometric system, smart agriculture, smart logistics, smart fitness system etc. The proposed LSI architecture overcomes the barrier of separate heterogeneous IoT applications, and thus achieves a global IoT system. When multiple such sensed data is collected by the sensors and sent to the LSI, the SOM classifies real time data such as smart healthcare, smart transportation, etc. and non-real time data such as smart biometric, logistics, fitness, etc., and pass them either to the edge or to the cloud respectively for further processing.

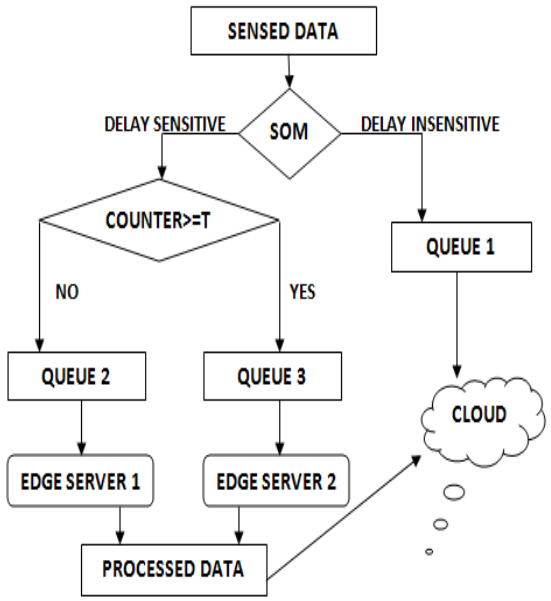

Fig. 1. LSI Flowchart.

IV. Simulation and Result Analysis

This research is simulated in the Aneka platform, which is a software platform for developing cloud computing applications [18]. It is a Pure PaaS solution for cloud computing. It is a framework which provides both middleware for managing and scaling distributed applications and an extensible set of APIs for developing them. Aneka 5.0 can be installed on the Virtual machine which may be configured as per the user requirements and agreed SLA. Aneka is interfaced with Matlab 8.0 which contains the code for implementing the scheduling of the delay sensitive and delay sensitive data. An Aneka server was developed locally with a single master and multiple workers, which create the environment of an edge, and have installed Aneka on the virtual machine created on the AWS cloud. Hence, creating edge-cloud environment. Aneka 5.0 has the provision of Autoscaling, which varies according to our algorithm.

In order to create the edge environment, a master and two workers have been created. The algorithm for SOM is run on the Aneka master computer which installed with Matlab 8.0 in which algorithm for classification implemented, which decides whether to forward the data to its worker computers or to the cloud. Initially only single worker computer is in the processing state, while the other inactive. Whenever the number of requests crosses 30 (the threshold T) to be served at the edge, the Aneka transfers the workload to its other worker along with its first worker. Here in this scenario, only two workers have been created, but many more worker computers can be created for processing the unprecedentedly increasing incoming requests. This research has been performed at a small level. Alternatively a virtual machine has been created with the Aneka PaaS installed in it on the AWS cloud. The classified delay insensitive applications are forwarded to this particular cloud. This cloud is also provisioned with the Autoscaling of VMs as the number of requests gets flooded. But in our research, any such requests flood at the cloud was not witnessed, since performed at the small level. When the workload at the edge gets low, the edge master automatically disables its other worker computer. The master does not perform any task on its own except classification and load distribution.

Initially, the research is simulated by taking number of incoming requests as: 5, 15, 25, 35, 45, 55, 65, and 75. Firstly these entire requests were forwarded to the cloud and their performance was evaluated. Then these requests were processed on the LSI system as proposed above. Then comparison on the following parameters were made and the proposed algorithm is found to prone to succeed not only in achieving the primary characteristics of IoT like heterogeneity, scalability, etc. but also attained the following metrics:

1) Throughput: The amount of service requests processed by the device per unit time. Since LSI incorporates the feature of scalability by providing the facility of additional edge server (Aneka worker), the network throughput is prone to enhance tremendously as multiple requests are being processed simultaneously at edge worker 1, another additional edge worker, and the cloud. Thus number of requests processed per unit time improves.

The average number of requests per second was found to be 63 in case of LSI, and 48 in case of Cloud.

2) Response time/delay: The duration of time required taking delivery of a response to a request. It can also be seen as the average time the client has to wait to get its job done. Delay sensitive applications are to be responded immediately without any delay in processing. By setting the priority for the delay sensitive data, LSI improves the response time tremendously. The observations are manifested in Fig. 2.

3) Packet loss: It is termed as the number of packet drops due to traffic or network congestion during a specified duration of time. LSI reduces the number of packet loss since it endorses the factor of horizontal scalability by deploying the provision of additional edge device whenever edge device 1 exceeds the threshold. Thus LSI is least prone to network congestion and improves the packet delivery factor for the measured duration of time. The use of mirror server aid in processing the majority of the requests at the edge server level, and not at the cloud level, therefore the packets are least faced with the network congestions. Hence reducing the probability of packet loss as shown in Fig. 3.

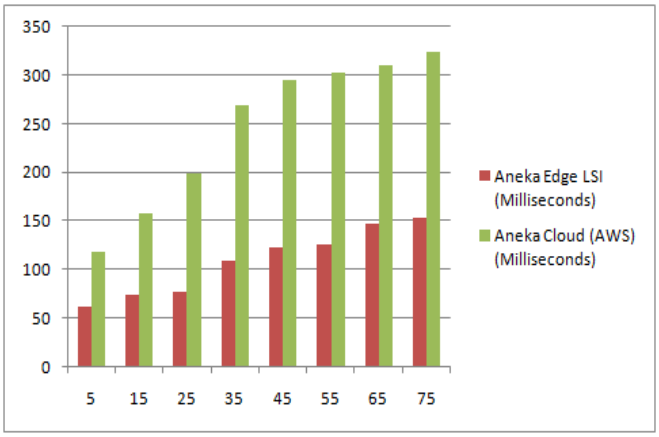

Fig. 2. Comparison of Response Delays wrt Cloud and LSI.

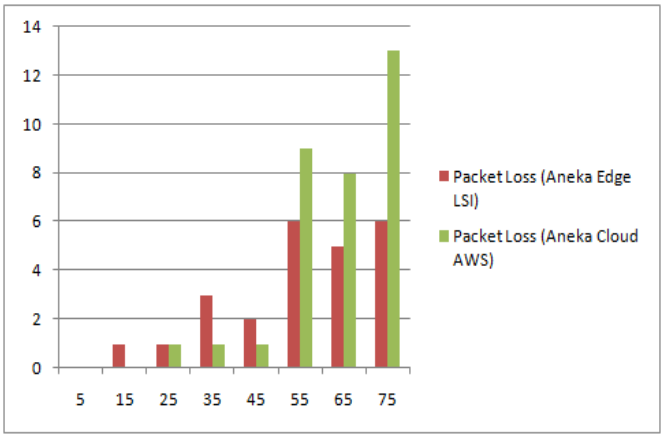

Fig. 3. Comparison of Packet Loss wrt LSI and Cloud.

V. Conclusion and Future Work

As can be clearly manifested by the current scenario, the unprecedented growth of IoT devices has generated the urgency of deploying such edge devices which are scalable and can accommodate the varying and consistently increasing number and types of IoT devices. Whether accurate or not, we must prepare for it or risk becoming the victims of our own success. In the paper an intelligent IOT framework has been proposed where in Self Organizing Map is deployed, in order to differentiate between delay sensitive data and delay insensitive data. This leads to the improvement in response time of the critical applications such as related to healthcare which to be immediately addressed, promoted scalability by utilizing the additional edge device, in turn improving the throughput. It reduced the idle time of the edge server 2 by promoting disabling phenomenon supported by Aneka Auto-scaling provision, which in turn reduced energy expenditure that may have incurred due to the idle state of the edge processor 2. As compared to the existing work, LSI not only reduces latency but also work in enhancing resource utilization, throughput and reduction in packet loss as manifested by our results. This research is done at small scale, but can be extended to incorporate various other applications. This research improves the performance to an extent as compared to the cloud because classification of delay sensitive and delay insensitive data is accomplished on the pattern of the data itself (time series), which may generate inaccurate results sometimes (i.e. may not classify data correctly) due to overlapping range of values of data of various sensors, but it shown good results specially in terms of response delay, because we are bifurcating the sensed data to the edge as well as cloud, as compared to performance when processing done at cloud only. Furthermore pre-processing can be deployed on the data to give better classification results. Also, scalability can be achieved at higher level by deploying several more additional edge servers. Further, in order to increase the scalability and to have better performance other algorithms like deep learning or machine learning can also be applied in future.

References

[1] K. Ashton, “That Internet of Things thing,” RFiD J., vol. 22, no. 7, pp. 97–114, 2009.

[2] H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelfflé, “Vision and challenges for realising the Internet of things,” vol. 20, no. 10, 2010.

[3] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of Things (IoT): A vision, architectural elements, and future directions,” Future Gener. Comput. Syst., vol. 29, no. 7, pp. 1645–1660, 2013.

[4] E. Cuervo et al., “MAUI: Making smartphones last longer with code offload,” in Proc. 8th Int. Conf. Mobile Syst. Appl. Services, San Francisco, CA, USA, 2010, pp. 49–62.

[5] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, “The cost of a cloud: Research problems in data center networks,” ACM SIGCOMM Comput. Commun. Rev., vol. 39, no. 1, pp. 68–73, 2008.

[6] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, “The case for VM-based cloudlets in mobile computing,” IEEE Pervasive Comput., vol. 8, no. 4, pp. 14–23, Oct./Dec. 2009.

[7] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, “Fog computing and its role in the Internet of things,” in Proc. 1st Edition MCC Workshop Mobile Cloud Comput., Helsinki, Finland, 2012, pp. 13–16.

[8] Fatemeh Jalali ; Safieh Khodadustan, et.al, “Greening IoT with Fog: A Survey”, IEEE International Conference on Edge Computing (EDGE), IEEE, 11 September 2017.

[9] https://en.wikipedia.org/wiki/Scalability.

[10] Marjan Gusev, Schahram Dustdar, “Going Back to the Roots—the Evolution of Edge Computing, an IoT Perspective” IEEE Computer Society, March/April 2018.

[11] Wei Yu, Fan Liang, Xiaofei He, et.al, “A Survey on the Edge Computing for the Internet of Things” IEEE Access, Volume 6, 9 March 2018.

[12] Fei Li, Michael Vogler, et.al, “Efficient and scalable IoT service delivery on Cloud” , 2013 IEEE Sixth International Conference on Cloud Computing.

[13] Chayan Sarkar, Akshay Uttama Nambi S. N., et.al, “A Scalable Distributed Architecture Towards Unifying IoT Applications”, 2014 IEEE World Forum on Internet of Things (WF-IoT), March-2014.

[14] Chayan Sarkar, et.al, “DIAT: A Scalable Distributed Architecture for IoT” IEEE INTERNET OF THINGS JOURNAL, VOL. X, NO. X.

[15] Onoriode Uviase, et.al, “IoT Architectural Framework: Connection and Integration Framework for IoT Systems”, EPTCS 264, 2018, arXiv preprint arXiv:1803.04780.

[16] Ju Ren, Hul Guo, et.al, “ Serving at the edge: A Scalable IoT Architecture Based on Transparent Computing”, IEEE Networks, September/October 2017.

[17] Dr. Manu Pratap Singh, Kajal Sharma, “Video Compression using Self Organizing Map and pattern storage using Hopfield Neural Network”, International Conference on Industrial and Information Systems (ICIIS), IEEE, 11 March 2010.

[18] Vecchiola, Christian, Xingchen Chu, and Rajkumar Buyya. "Aneka: a software platform for .NET-based cloud computing." High Speed and Large Scale Scientific Computing 18 (2009): 267-295.

[19] Shreshth Tuli, Nipam Basumatary, Rajkumar Buyya.

[20] EdgeLens: Deep Learning based Object Detection in Integrated IoT, Fog and Cloud Computing Environments” 4th IEEE ISCON (November 21-22, 2019).