[1] Mrs. Swapnaja Rajesh Hiray

[2]Dr. Bhramaramba Ravi

A Novel Data Analysis Architectural Framework for Data Reduction on Edges for Smart City Healthcare

Abstract: - The most difficult task is managing and digesting the massive amount of heterogeneous data created by IoT. It questioned the efficiency and computational storage capacity of today's infrastructure resources. Furthermore, because there is so much data, it becomes difficult to give the user enough information to make a decision. A technology known as Complex Event Processing (CEP) has been created to glean important insights from massive real-time data streams. The goal of this research is to create a system that can extract complicated events from large amounts of sensor data and only transmit those events to a cloud server. Global healthcare services have been expanding at an exponential rate because of the vast amount of data that clinical and medical organizations generate daily. Healthcare systems can use e-health services to meet the medical and assistive requirements of people. Edge technology is proven in this direction. Mixing the old classic concepts with the innovative we have proposed the Novel data analysis architecture for smart heart disease prediction. We have created the fuzzy-based inference system at the edges for the data analysis and used the resulting Complex Events at the end predictions.

Keywords: CEP, Edge Computing, Smart Healthcare, Fuzzy logic, Inference Engine, CNN.

I. Introduction

The Internet of Things (IoT) is a new network and information technology that can automatically monitor and manage a network of smart IoT devices [5]. IoT is pervasive in our environment and has gradually grown to be a significant part of our lives. Since its start, the Internet of Things market has experienced remarkable growth. exceptional forecasting abilities, cheap cost, and capacity to save time. Physical items are connected by the Internet of Things (IoT), and data can be sent and received over it. Sensors, machine learning, real-time analysis, embedded systems, and other technologies have all evolved into and out of the concept of the Internet of Things.

A significant amount of data can be produced by the rapidly expanding number of smart devices, including wearables, actuators, and sensors [7] that are connected to the Internet of Things (IoT). The inference of precise and practical conclusions from sensor data is challenging because of the rapid transmission of complex data. Due to a variety of variables, including deterioration, the machine's underlying dynamic structures change with time, making data processing one of the main problems. The enormous data set contains both organized and unstructured data, making it challenging to handle using traditional techniques. However, to obtain insightful knowledge that is useful for making strategic business decisions, business data analysis is a must.

The study of Complex Event Processing (CEP) is a new field that analyzes data to find behaviors that are not visible to the human eye. It helps provide real-time information by reflecting more pertinent events from more sensors than just the one. For instance, if the heart rate surpasses the threshold limit within a specific time frame and the breathing rate rises, this may be identified as an arrhythmia event.

Cloud computing offers information access from anywhere in the world, automated backups, and quick recovery for the healthcare sector. Like any new technology, cloud computing has its share of difficulties and obstacles; in particular, it presents issues when dealing with a lack of IT resources, bandwidth, and technical know-how. Concurrently, the cloud computing sector lacks legitimacy and support due to lax laws and regulations.

Nevertheless, current complex event-processing systems are slow because they do not take redundancy and similarity in sensor data into consideration. This system tries to upload all of the fundamental data to the cloud, which makes it more complicated and slows down the upload process. Numerous methods exist in the literature that provides prediction solutions in different domains. For example, [3] uses rule-based machine learning strategies to detect

[1] PhD Scholar CSE GITAM Visakhapatnam and PICT Pune. India

swapnajahiray@gmail.com

2Professor, Dept. of CSE, Visakhapatnam, Andhra

bravi@gitam.edu

Complex Event Processing (CEP). Moreover, earlier attempts have been made to use Esper to generate CEP, as demonstrated by Boubeta et al. [4] who combine technologies like Mule ESB and ESPER engine using the Xively IoT platform. It does not, however, benefit from multi-cloud setups or offer the dynamic communication protocol required for event-driven architectures due to the centralized framework.

This research is driven primarily by the necessity of excellent quality of service (QoS) at low latency for Healthcare Internet of Things (IoT) applications under time constraints. None of these objectives can be fulfilled by the cloud. Agile thinking and fast decisions are crucial because remote patients need to be closely watched and because a patient's physiological state can change over time. Latency is more erratic in unstable network conditions. Patient health data (PHD) is given slowly due to a significant amount of delay. As a result, the record is no longer credible, adequate, or valuable. The problem deteriorates when cascading-based data processing (such as signal processing of electroencephalogram (EEG) or electrocardiogram (ECG) signals) is required [16]. Milliseconds to microseconds could be the range of service delays in healthcare IoTs. warnings to doctors. Creating complicated events on a mobile device and delivering that data to the cloud rather than streams of big data from several sensors may assist in saving data transfer time and storage space. The suggested architecture with the CEP computing strategy for data analysis in healthcare applications addresses outstanding issues in this regard. We have suggested three components at the different levels i)Filtering algorithm at the device level to filter out the atomic events. ii)Inference Engine at the Edge level iii)A machine learning method is used to enhance the prediction of a patient's health (Cloud-based CNN)

II. LITERATURE REVIEW

A. Smart City Architecture in Healthcare

The application of information and communication technologies to improve the accessibility and caliber of healthcare is known as "smart health." Because of their overload, the current healthcare systems are unable to satisfy the demands of an expanding population. In this way, smart health uses AI to give doctors better diagnostic support and to reach as many people as possible with access to healthcare. It is now possible to analyze this data in the cloud to make better healthcare decisions because mobile phones and health trackers are widely available [3][8]. These devices are capable of collecting information on people's health in real-time (ECGs, temperature, body oxygen saturation, and other biosensors), as well as monitoring daily activity and identifying anomalous motions using inertial sensors. Consequently, overall costs and the strain on medical facilities are decreased. As a result, total expenses and the stress on healthcare institutions are reduced. The smart healthcare architecture is worked in 4 phases, roughly.

TABLE I: MAIN ARCHITECTURAL LAYERS IN A MODERN HEALTHCARE DELIVERY SYSTEM

Sr.NO	Layer Name	Comments
01	Level 1: Obtaining Information	This level stores raw data acquired from sensors for subsequent processing. CSV, tweets, database schemas, and text messages are just a few of the forms used to capture diverse data.
02	Level 2: Data processing	Semantic web technologies are used to summarise the information obtained at the data collection level before transmission, analysis, and fusion at subsequent levels.
03	Level 3: Data integration and reasoning	When the data has been classified, data may be supplemented using domain professionals and undefined reasoning. This strategy will aid in the acquisition of new information through uncertain

		reasoning, hence assisting in the development of an intelligent smart system.
04	Level 4: Device control and alerts	Numerous online applications may make use of Level 3 data to establish intelligent working conditions. The derived data can be utilized for a variety of purposes, including input/output, messages, alerts, and warnings.

B. *Challenges in Smart City Healthcare Application*

Designers need to know what people expect and how smart healthcare systems are used in the real world, to accomplish the aims and solve the issues of smart healthcare. Smart City IoT system implementations must deal with concerns such as security and privacy, smart sensors, networking, and big data analytics. In addition to that huge data collection and Big data itself pose Volume as the major Challenge and this is affecting real-time analysis which is a basic need nowadays.

The processing of measured data by sensors on sensor devices or devices next to sensors, as opposed to cloud servers or distant healthcare facilities, is known as fog computing [14] [17]. Thus, an enhanced fog computing method can be thought of as a low-latency alert or diagnosis system. Unlike cloud frameworks, which are narrowly focused, fog computing enables an organization to leverage local computational resources at the organization level in addition to cloud administrations at the organization level. The fog nodes also assist in saving energy by limiting long-distance transmissions and selecting neighbors based on a time restriction calculation.

C. *ML and AI in IoT healthcare*

When integrated, artificial intelligence and the Internet of Things (IoT) should improve operational efficiency in the healthcare industry. Modern healthcare systems are heavily dependent on artificial intelligence (AI) and machine learning (ML), which have a significant impact on data management and provide accurate, low-latency results. AI has greatly impacted the diagnosis and forecasting of problems that call for more sophisticated clinical testing, and machine learning (ML) can help with ongoing problem detection and personalized healthcare. The diagnosis of illnesses and health monitoring systems have been significantly impacted by the Internet of Things (IoT). Below is a summary of a few of the survey items

TABLE 2: SURVEY SUMMARY ON AI AND ML LITERATURE IN IOT HEALTHCARE

Study	Objectives	Methodology
Azimi et al. [58]	Classification of ECG signal anomalies	Distributed implementation of a deep learning algorithm and a partitioned linear machine learning technique (linear SVM)
Kaur and Jasuja [59]	Pulse rate and body temperature monitoring	Remote health monitoring with the use of the Bluemix cloud.
Magaña Espinoza et al. [60]	Detection and notification of specialists when individuals fall to the ground; event-based monitoring	Encryption technique for communication between wireless sensors. For simple access and push notifications of anomalous occurrences,

	to report tachycardia and bradycardia	a mobile application and a web page are available.
Orha and Oniga [61]	Automatic recording of the human body's major physiological characteristics with the use of an Arduino microcontroller.	Data is transferred to a personal computer for processing.
Yakut et al. [62]	ECG signal measurement through an E-Health Sensor Platform coupled to a Raspberry Pi.	The Raspberry Pi writes data to a text file that may be processed further in the Matlab computer environment.

D. Complex Event Generation

IoT CEP solution (IHS) is proposed by Dhillonx et al. [68] using a mobile edge device and a cloud-based IoT Hospital Server. Wearable health sensors (WHS), a mobile device, and a distant IoT Hospital Server (HIS) are the main components of the system. WHS may communicate sensor data through Bluetooth or wifi to a smartphone, which connects to the IHS over a WiFi network. The edge uses the MQTT protocol to transfer complex events to the hospital server. Like security surveillance and healthcare systems in real-time environments. Numerous particular problems are presented by cloud computing. Regarding IoT in healthcare, security, and privacy are important considerations. The primary issue here is that data processing is commensurate with the amount of data created and moved across the network. Therefore, the only way to lessen this would be to consider reducing the data resources. So subsequent routing of data over the web will be avoided. So here, we found the research gap, and then we developed the innovative rule-based complex event-generated engine on edge devices. We did the comparative analysis for data reduction with the help of various Machine learning algorithms. We found that time and space complexity is the main problem with these resource-constrained edge devices. And then come up with a Complex event generation module.

III. RESEARCH PROBLEM AND METHODOLOGIES

A. Identifying the GAP :

The vast amount of data created by IoT sensors leads to a great deal of data traffic, which clogs the network and causes delays. End users are not provided with enough useful healthcare data due to increased round-trip delays resulting from huge data transfers and high hop counts between IoT and cloud servers. network congestion. Applications in healthcare that need to respond quickly require real-time data. The low latency needs of end users and healthcare IoT devices cannot be satisfied by traditional cloud servers. Therefore, for IoT data transfer, it is necessary to decrease computation latency, network latency, and communication delay. Basically when we are working with the Smart city applications and talking about Fog architecture. However, privacy is viewed as the main objective of security; it involves enforcing specific rules and principles that limit the amount of data that can be accessed, collected, or provided to a second or third party about an individual or organization. Data privacy and ownership are more directly related to each other than to data security. While using information systems, people and organizations may claim a moral right to privacy; however, computer security is not a moral right in and of itself.

Generally, when we are transferring data to edge devices from end users we are using a LAN-like protocol where communication delay is considered negligible. Also while modeling the network for getting performance evaluation parameters Unprocessed data in smart applications is forwarded to cloud servers via the internet for processing, so communication delay from edge devices to cloud servers needs to be considered.

B. Problem Statement

We need an effective framework to address the time and space complexity problems with the data provided by IoT devices. As a result of this issue statement that follows has been developed:

In smart city healthcare, where complex event generation has occurred on edge devices, to design a novel computing approach for data analysis. To analyze the data and gain real-time insights to intensely anticipate patient health (arrhythmia). Alerts are generated by the system and sent to the doctor's registered mobile number.

The problem statement is broken down into the following subproblems:

• Creating a hardware interface for measuring physiological characteristics.

• Create a model for data compression.

• Creating a better CEP generation model for cloud-based data transfer in Internet of Things-based smart city applications.

Based on the above We come up with the following objectives:

1. To design a data analysis block on edges of cloud-based smart healthcare services incorporating domain-specific Complex events generated with real-time patient-cited health sensor values. This rule-based Complex event generation reduces the vast amount of data generated and transferred from IoT devices over the cloud.

2. To devise an efficient and novel hybrid data analysis framework on edges of smart city healthcare applications, which will correlate the atomic real-time events for generating Complex events, which will be used for further analysis in the cloud for disease severity prediction. To perform experimental research with a simulated soft sensor-generated dataset and calculate the computational complexity of proposed/developed frameworks on throughput, CPU utilization, and RAM required with increased rules and compare the values.

3. To devise an efficient and novel architectural framework and fine-tuned stacked data analysis approaches for smart healthcare applications, which can further be integrated with any smart city application.

The first and third are already explained in our previous publications so we are focusing more on the second one.

3. Mathematical Model

When we talk about fog computing architectures for Internet of Things smart city applications, we make the following assumptions:

1. The network's terminal nodes (TNs), which include cell phones and body sensors, can communicate their precise geographic location. This is also applied to the modeling of events.

2. The fog computing tier, which is made up of "intelligent" devices that can compute, process, and store data in addition to forwarding and routing data packets to the cloud layer, is where edge devices like routers and data collector nodes are located.

3. Depending on what's needed, the networking devices in the fog computing layer can divide up the network, compute, and storage load among themselves.

4. All the communication network layer delays are handled by standard protocols.

The tier nearest to the ground in our smart healthcare application creates a network of several body sensor-equipped, Internet-connected end devices, also known as IoT. The edge gateways located at the margins of the fog tier receive the data transmitted by these sensors, also referred to as the Terminal nodes. In fog computing, not every data packet is routed to the main cloud computing module for processing, in contrast to the typical cloud architecture. Rather, the fog layer itself is used to perform all real-time analysis and latency-sensitive applications.

This layer's fog computing devices have limited semi-permanent storage, which enables them to temporarily store received data for analysis before sending the necessary feedback back to the source devices.

Now with the focus on our third objective:

Two factors primarily affect the end-to-end delay in the results when we attempt to model the smart healthcare application. Traditionally, the following factors have a major impact on data processing delay or computational overhead in smart city healthcare applications: 1. Communication latency for data flow from devices at the edge to a cloud server. 2. Machine learning computational overhead at cloud server network protocols, the speed, and the kind of data we are sending are among the things that we take for granted. So, there will be an end-to-end delay caused by these factors; nevertheless, the analytical engine we are developing primarily aims to minimize the events data transfer to and from the cloud. The problem can be represented graphically as follows (Ref. Fig.1.)

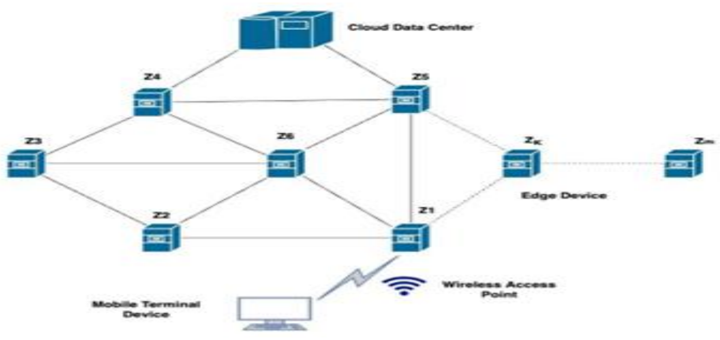


Figure 1 Diagrammatically Representation of Fog architecture(Curtsy: Authors Previous Publication)

We can model graphically the above network as follows (Ref. Fig2.)

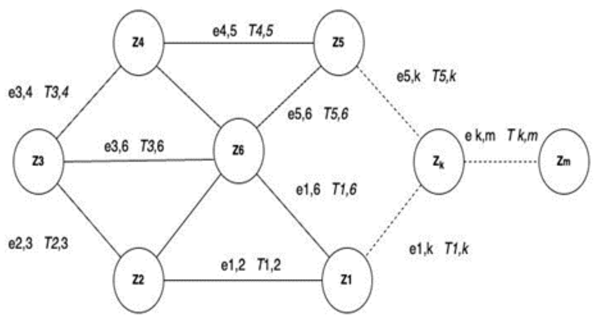


Figure 2. Graphical representation for Fog network(Curtsy: Authors Previous Publication)

With the above-said architecture, the performance metrics of the Smart healthcare applications can be modeled or benchmarked on the following criteria:

Service latency: The total of the transmission and processing latency for a request is the service latency, which is essentially the response time for a request sent by an application instance operating within an IoT device. We presume that there is very little latency due to high bandwidth communication between the cloud server and several edge devices.

We assume that high-bandwidth channels are used for communication between various FIs at tier 2 and across various cloud data centers at tier 3, resulting in very little latency. Furthermore, the TNs' communication latency is considered negligible.

DEdge = min∑mi=1 (1/ Vzi) * (aixi2) Where xi is a function of the amount of data generated at the edge layer.

delay in the cloud computing layer includes processing delay plus delay in transmitting data from the fog layer to the Cloud server layer.

For the cloud server j if the amount of data it needs to deal with is yj and the computing capability is Vj its computational delay can be expressed as: Dj = yj / vj (j= 1, 2…. n). So for n cloud servers with total data to process is Y we can have

The system delay in Smart healthcare applications with Fog architecture mainly consists of

Delay System = D Edge + D Cloud + D edge-cloud Where D Edge = Computational Delay at Edges, D Cloud = Computational Delay at cloud, and D edge-cloud = Communication Delay From edge to cloud.

So our Objective function parameters mainly focus on the following: 1. Data amount transferred from Terminal Devices to Edge 2. Data amount transferred from edge to cloud 3. Edge layer data analytics strategies and algorithms that have better Computational performance.

With this as the GAP analysis and Objective function, our Research Objectives are as follows:

- To reduce transmission data packets by introducing suitable methodology on the edge devices.

- To do the performance analysis of the new suggested algorithm As it will be designed at the fog layer which has limited resources.

- To devise an efficient and novel architectural framework and fine-tuned stacked data analysis approaches for smart healthcare applications which further be integrated with any smart city application

B. Methodologies

We have suggested the use of Complex Event Processing algorithms on the Edge devices to reduce the amount of data transferred from Edge devices to the cloud server.

Proposed System: We have proposed the following components at different layers along with standard fog architecture in the Smart healthcare application. To discuss the feasibility and analyze the different design parameters we have used the prototyping approach. As a case study, we describe a unique health analysis technique for heart failure prediction that tackles the performance challenge of evaluating and processing the massive data supplied by sensors such as ECG signal values, SPO2, and temperature. It is based on the usage of Complex Event Processing (CEP) technique mixed with statistical methodologies. A CEP engine analyses incoming health data using threshold-based rules.

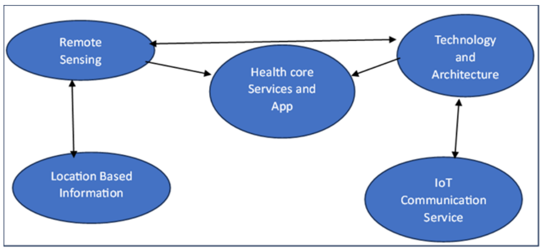

Remote Sensing
Health core Services and App
Technology and Architecture
Location Based Information
IoT Communication Service

Figure 3. Proposed architectural Design for smart healthcare

In Objective One, we have explained how our Complex event processor can be used in the Data analytics block on the Gateway of the Edge computing IoT healthcare applications. CEP offers abstractions (event processing logic) for event operations that are distinct from the application logic (event producers and event consumers). This can help to cut development and maintenance costs. Event processing logic is frequently written in domain-specific languages known as event processing languages (EPLs).

Coming again to our hypothesis, the Computational cost of the data analytics block, i.e. CEP in our case is essential. So, we should focus on the design issues of the Complex event Processor or suggest a Novel approach to design the Complex event Processor so that the overall latency of our Smart Healthcare Application will be reduced.

Recommended algorithms at the terminal devices: The diverse environment of smart city applications is the primary source of worry while gathering input. As an illustration, we have recorded ECG signals, SpO2, and body temperature using a thermometer. We have proposed a generic Event model that may be used for any kind of sensor. When the input model is being run on terminal devices with sensors connected, the input port will receive values from each sensor. Since each sensor has a standard Device ID and location, the event object will have the format shown below:

e = (ID, DeviceID, DeviceType, Location, StartTime, EndTime, KeyValueList) where,

ID: Event No, Device ID: 1,2,3,……..I (For I different sensors), Location: Geographical location, Start time: time stamp for start of reading input port, End time: time stamp for start of reading input port, Key values list: Input port data (It depends on the Sensor) This standard Object format for storing event format helped us to reduce the heterogeneity. Further, this Object oriented event data is used for further analysis.

Event Collection system: The data collector node is where this Block is proposed. Now, it depends on the WSN topologies you are using and your fundamental network architecture. Sometimes it is present at your router, access point, or terminal device (such as a mobile phone or Fit Band), or it may be a separate data collector node. Preprocessing is similar to this when it comes to input data. It makes use of event filtering, for which common methods like thresholding or windowing can be applied. Once more, we have a variety of events here.

1. Simple atomic events or 2. Composite events.

We have created a hierarchical event model in the suggested event model that aligns with our goals and only pushes important events that are service- and application-oriented. Experts in the field recommend the business guidelines here. In our instance, we have spoken with the on-call cardiologist to obtain the appropriate protocols for noteworthy occurrences.

Our Algorithms for the first and third objects have already been published in previous publications. Here we are discussing the second component algorithm

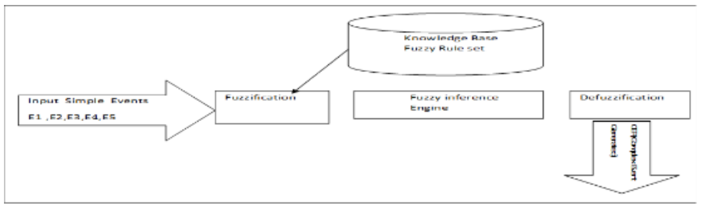


Figure 4: Complex Event Processor Model

This Fuzzy-based Complex event processor(Ref. Fig 4) mainly consists of the above-shown blocks. We are pipelining the output of our simple event generation block as input to our complex event generator. Fuzzification is mainly responsible for converting simple input events to fuzzy input data sets. The inference engine is primarily using this fuzzy input dataset. For the inference engine, we are using association rules. The knowledge base is taken and confirmed by the domain experts. The defuzzification block will give us the action selected as the result of inference and will convert to the corresponding output dataset. The output will be a Complex event generated.

Mainly our Complex Event Processor consists of the following modules :

Inference Engine 2. Simple event Processing (Fuzzification) 3. Complex event Generator (Defuzzification) 4. Inference Knowledge Base or Rule Engine

Input Simple Events :

Data Resource: From our Pre-processing Block.

Event Format: E1 (1, l, t, Value): Distorted ECG

E2(2,l,t,Value) : SPO2 below 95

E3(3,l,t,value) temperature above 96.6

In our pre-processing unit we are collecting Raw events (explained in Eq 4.3.1.1) and with our pre-processing analyzer converting them into Simple events as shown in the above format. It has 4 parameters

1: Sensor No (1. ECG,2 SPo2, and 3 Te

2: Location

3: Timestamp

4. Value of sensor

The example is given below :

{ E2(2, Icu bed 1,10.00am, 85)

E3(3,icu bed 1,10.00am,34)

E1(1,Icu bed 1, 10.00am, 389404297

Total We can say about the dataset :

Dataset Resource: Collected Sensor values.

Number of Attributes : 3 (ECG, SPo2. Temp)

Sampling rate : ECG: 300Hz for 3 mins on 10 patients

SPo2: 30 mins We have collected a total of 169 values

Temp: 1 hr. We have a total of 169 values

Dataset Characteristics: Real-time sensor data (We have used the generator of time data for the demonstration but it is real-time data)

Output data will be a complex event. The format for this output event will be as shown below :

CE1: fix OPD appointment and suggest lab testing.

CE2: Get admitted to patient to ICU in emergency suggest lab test

CE3: Push ECG signals, suggest lab testing, and admit the patient to the cardiology dept

CE4: admit the patient immediately, and do lab testing.

Here we are using a heart arrhythmia-type detection web service for demonstration purposes so only checking for the CE3 generation rules and generating those Complex events. The output complex events have having following format

CE3 (Patient ID, Event ID, Timestamp, ECG signal values)

Here ECG value we are forwarding is what we got for that time stamp. This will be consumed by our web service which uses CNN to detect the type and severity of heart disease.

Complex Event Processing Systems CEP: In comparison to a simple event, it represents more relevant events from several sensors, which aids in the provision of real-time information. Context-awareness is one of the key characteristics of any smart-city application since, as we all know, they are ubiquitous. We have recommended a complex event processor—a crucial component—for additional data reductions. The system looks for anomalies in the patient's health status using complex event processing (CEP). The fuzzy-based technique is used to generate complex events (CE). We have incorporated a third novelty into our component design for the edge gadget. Real-time alerts are generated as complex events, and they are sent to the cloud together with ECG data so that the model can be further trained.

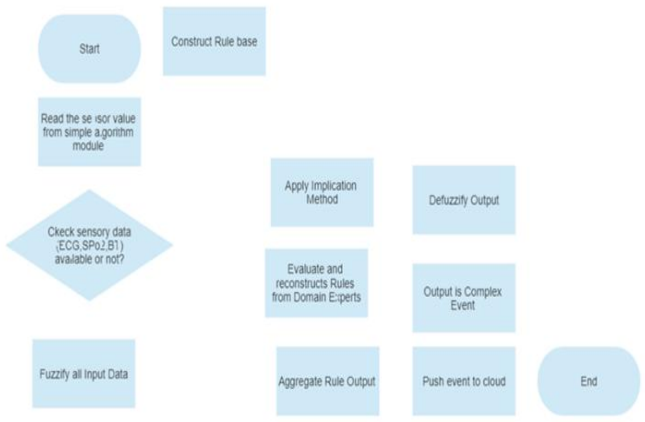


Figure 5 Inference engine at the heart of the CEP

Fuzzy-based approach for the design of CEP: As we can see in the smart-healthcare problem statement ECG and Spo2, Temperature are analyzed and a thresholding technique is used for filtering and creating the critical events, data transfer in bytes is considerably reduced. This we can do offline also with low internet connectivity. But still, the set of values (sensor reading) is sharp boundary values. With a fuzzy rule base here we are allowing smooth boundaries and then applying a rule-based inference engine with simple and /or operation to generate new complex events which will carry only necessary information used for cloud-based analysis. This complex event is pushed further. The fuzzy set concept is applied to atomic events which we are getting as the output from our first algorithm and as per rule complex event will be generated. In this fuzzy representation, this limitation of a sharp boundary set is further analyzed by allowing membership in a set to be a matter of degree. the degree of membership in a set is expressed by a number between 0 and 1 ; 0 means entirely not in the set, and 1 means completely in the set.in between means partially in the set. From Fig 6 It is clear that I put to our block Critical events. The fuzzification functions we are using are triangular and trapezoidal. Following is the CEP Algorithm :

Input data set : SE(d,l,t,value)// Where d=(1,2,3,4,5), l = location(bedno),t=time stamp, (Atomic events after filtration)

Output dataset : EE1,EE2,…………

1. If SE.d==1 then(call fuzzification function for ECG) or
2. If SE.d==2 then (call fuzzification function for SPo2) or
3. If SE.d==3 then (call fuzzification function for BT) or
4. For all events with the same l and t Search for patterns in Ruleset
5. Generate complex events when Rule Matches
6. Push Events for the next analysis to Cloud

Performance of Complex Event Processor on the Edge Devices

Input: Events generated from our first Simple event model :

E1 (1, l, t, Value): Distorted ECG

E2(2,l,t,Value) : SPO2 below 95

E3(3,l,t,value) temperature above 96.6

Output : Complex Events CE1(Event id,Patient id , location , Timestamp , SpO2 values, action 1)

CE2(Event id,Patient id , location , Timestamp , SpO2 values, action 2)

Process: A fuzzy inference system (FIS) maps inputs (in our use instance, sensor data) using fuzzy set theory. We are getting SPo2, ECG, and Temp in the form of Simple Events generated) to outputs (Complex events we want to generate CE1, CE2, CE3, CE4). This mechanism involves

the following steps: (1) Definition of the fuzzy sets and fuzzification of each input, (2) Definition of fuzzy if-then rules, (3) Aggregation of the consequent by applying the rules, and (4) Defuzzification of the output.

Fuzzification Stage:

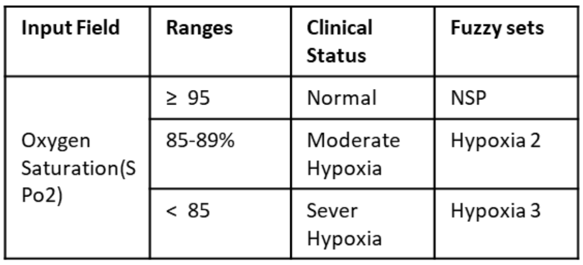
The parameters (inputs) were modeled as linguistic variables before fuzzification. Heart rate was denoted as HR, body temperature as BT, and oxygen saturation level as SP. The linguistic variable CE is also used to indicate the output parameter. The representation of the activity was ACT.

Inputs with fuzziness: This procedure, which serves as the initial phase, uses membership functions to characterize the inputs and establish the degree to which each is a member of a specific fuzzy set.

Fuzzy sets and membership functions' input variables: The list includes the possible ranges for each input value and the implications (clinical status).

Fuzzy SpO2 set: Per experts' guidance, we use three fuzzy sets for this input field (NSP, Hypoxia2, and Hypoxia3). The membership requirement for this fuzziest is the Trapezoidal function.

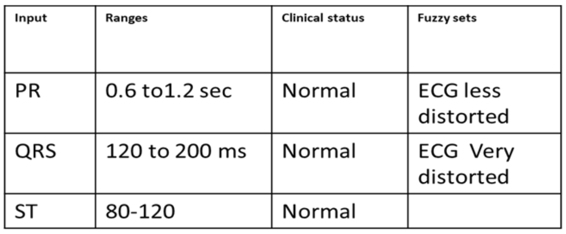
Table 4: SpO2 Fuzzy set and membership function



Input Field	Ranges	Clinical Status	Fuzzy sets
Oxygen Saturation(SPo2)	≥ 95	Normal	NSP
	85-89%	Moderate Hypoxia	Hypoxia 2
	< 85	Sever Hypoxia	Hypoxia 3

ECG Fuzzy input set: We employ three fuzzy sets for this input field (ECG less distorted, ECG very distorted, and ECG Normal) per the experts' recommendations. The PR, QRS, and ST values in the ECG waveform we plotted on the pre-processing module will determine this.

Table 5: ECG Fuzzy set and membership function



Input	Ranges	Clinical status	Fuzzy sets
PR	0.6 to1.2 sec	Normal	ECG less distorted
QRS	120 to 200 ms	Normal	ECG Very distorted
ST	80-120	Normal	

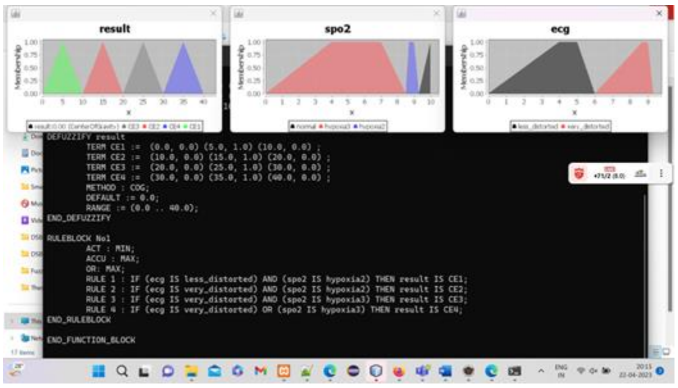
Temperature Fuzzy input set: Per experts' recommendations, we use three fuzzy sets for this input file (Normal, Moderate, and Fever).

Fuzzy sets and membership functions as output variables: The Complex Event created is referred to by the output variable "clinical status". The linguistic term "CE" is used in this system to indicate the output variable.

The rules for the Complex event generation for our prototype we have taken are as follows (Ref: table 6)

The fuzzy logic engine's "clinical status" output variable has four fuzzy sets in our study. These sets' membership functions are triangular. The fuzzy set details are shown in Table 5. Fuzzification Method: Fuzzy Values can't be clearly distinguished from one another. For instance, the patient is experiencing hypoxia. The SPo2 in this statement, denoted by the letter C, could be 93, 85, or 35, which can be anything in a given range and cannot be distinguished from each other in the statement. Fuzzy sets represent these variables because they cannot take crisp values in any other way.

Rules	Fuzzy sets Output	Clinical status	Action performed
1.IF ecg IS less_distorted AND spo2 IS hypoxia2 THEN result IS	CE1	Moderately bad	Push ECG and SPo2 Events with our event format to the corresponding web service (admit patient immediately, do lab testing.)
IF ecg IS very_distorted AND spo2 IS hypoxia2 THEN	CE2	Urgent Attention required	Push ECG and SPo2 Events with our event format to the corresponding web service (In our case, our service is what we are demonstrating for heart disease severity checking)
IF ecg IS very_distorted AND spo2 IS hypoxia3	CE3	Emergency	Push ECG and SPo2 Events with our event format to corresponding web service(Can be called ambulance and get admitted to Emergency)
IF ecg IS very_distorted OR spo2 IS hypoxia3	CE4	Emergency	Push ECG and SPo2 Events with our event format to the corresponding web service(It may be calling specialist)


result
Membership
x
result:0.00 (CenterOfGravity) ● CE3 ● CE2 ● CE4 ● CE1
spo2
Membership
x
● normal ● hypoxia3 ● hypoxia2
ecg
Membership
x
● less_distorted ● very_distorted

```
DEFUZZIFY result
        TERM CE1 :=  (0.0, 0.0) (5.0, 1.0) (10.0, 0.0) ;
        TERM CE2 :=  (10.0, 0.0) (15.0, 1.0) (20.0, 0.0) ;
        TERM CE3 :=  (20.0, 0.0) (25.0, 1.0) (30.0, 0.0) ;
        TERM CE4 :=  (30.0, 0.0) (35.0, 1.0) (40.0, 0.0) ;
        METHOD : COG;
        DEFAULT := 0.0;
        RANGE := (0.0 .. 40.0);
END_DEFUZZIFY

RULEBLOCK No1
        ACT : MIN;
        ACCU : MAX;
        OR: MAX;
        RULE 1 : IF (ecg IS less_distorted) AND (spo2 IS hypoxia2) THEN result IS CE1;
        RULE 2 : IF (ecg IS very_distorted) AND (spo2 IS hypoxia2) THEN result IS CE2;
        RULE 3 : IF (ecg IS very_distorted) AND (spo2 IS hypoxia3) THEN result IS CE3;
        RULE 4 : IF (ecg IS very_distorted) OR (spo2 IS hypoxia3) THEN result IS CE4;
END_RULEBLOCK

END_FUNCTION_BLOCK
```

Figure :2 Membership function for ECG and SPo2 and Output

A triangular MF is specified by three parameters {a, b, c} as follows:

$$\text{triangle}(x; a, b, c) = \begin{cases} 0, & x \le a. \\ \dfrac{x-a}{b-a}, & a \le x \le b. \\ \dfrac{c-x}{c-b}, & b \le x \le c. \\ 0, & c \le x. \end{cases}$$

By using min and max, we have an alternative expression for the preceding equation:

$$\mathrm{triangle}(x; a, b, c) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right)$$

Membership function for ECG and SPo2 and Output, when we have 90 to 100 as SPo2, it is treated as Normal, and from 85 to 90 is treated as hypoxia2 and 0 to 80 as hypoxia3. Similarly, we are considering the ECG's PR and RR interval ranges. ["RRInterval"]>=0.6

and data["RRInterval"]<=1.2) and (data["PRinterval"]>=0.05 and data["PRinterval"]<=0.2):

After the fuzzification with these ranges, we decided on the four rules for the demonstration purpose with discussion with physicians.

Defuzzification Mathematical Model: The weighted average approach, centroid method, and maximum membership principle are the three primary techniques for defuzzification.

We have used the center of gravity Defuzzification function. A defuzzification technique in which the coordinates of the level's section contained between the graph of the membership function involved and the OX axis are transferred to corresponding levels.

IV RESULTS AND EXPERIMENTATION

We have used simple and/or functions from the CEP operators for our rule base.

We aim to design a complex event generator using the fuzzy rule-based inference engine on edges in smart healthcare applications and optimize it on Complex event generation time. We have suggested the Architectural design and algorithm for the Complex Event Generator. Now, as stated in the objective, we have to analyze this Design on the following parameters :

1. The time complexity for the Complex event generation.
2. CPU usage or processor time required.
3. RAM usage for CEP Engine.

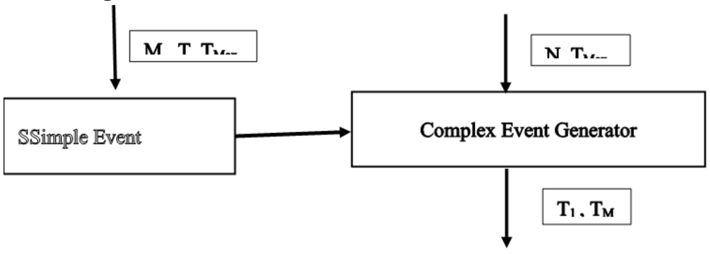


Figure 4 Complex Event Generator Performance Analysis Model

The success of CEP is crucial for real-time IoT monitoring.

Fig. 4 depicts the experimental configuration of the CEP system.

Specify M, T, and Tmax. The time interval is T. The largest time window is Tmax. To send to the CEP system, M simple events must be generated in each T period. The actual system is called CEP. It receives sensing input and goes through a series of steps to produce complicated events. The CEP system has a few parameters. The complex event definition number, N, and the largest temporal frame, Tmax, are input parameters. The processing time Tc is calculated using the output parameters T1 and TM. The system time at which the first event in a T-minute period was received is T1. The computing method of Tc is Formula (1) as follows. In 1, i is the number of CEP system runs.

Tc = 1/n $\sum_{i=5}^{n}(TM - T1)$ ---------------(1)

To get the time and space complexity, we run this module separately before integrating this as a middleware architectural component. The experiments are carried out on an Intel Core i5-6300HQ CPU 2.30GHz PC, with 8GB of RAM running on Windows 10.

1) THE PROCESSING TIME PERFORMANCE The number of complex event definitions N is :1, 2, 5, 10, 20, the input simple event flow M is increased from 1 × 104 to 10 × 104, and the processing time Tc is shown in Figure 6. The experimental results show that:(1) The processing time Tc increases as the number of complex event definitions N increases.

2)) THE RAM AND CPU USAGE: We assess CEP's use of computational resources. Figure 5.10 displays the CPU use of the CEP for cases where N = 20 and M = 200 ×104. The H/W Specifications are Intel i5 PC with 8GB RAM. If the required throughput is thousands levels, such as 1000–9000 events/second, or tens of thousands level but less than 3 ×104 events/second, it is adequate to sustain the CEP system's operation.

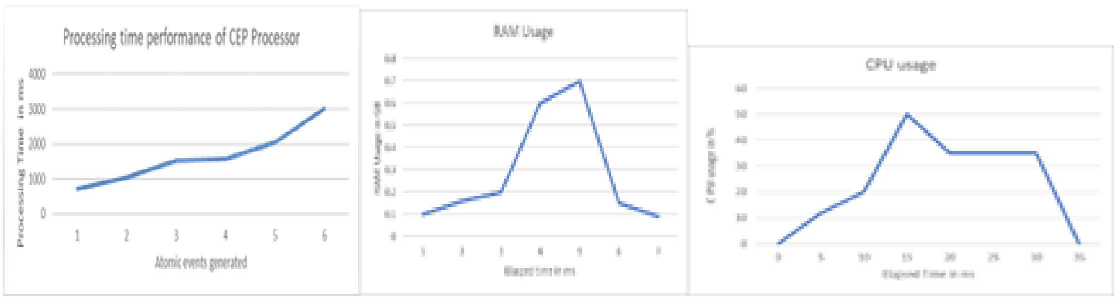


Figure.3 CEP Performance Analysis Graphs

The processing time Tc increases as the number of complex event definitions N increases for all of the two kinds of complex events. Elapsed time is the amount of time that passes from the start of an event to its finish. Here We have M which is Raw Events and the last complex event we received.

A complex event (CE2) is generated if every sensor value is high (according to the rule set). These events are transmitted to the cloud and Critical alarms are given to the doctor or family members when heart failure is predicted using machine learning techniques on the cloud with CNN. The cloud is being used to generate the analytics reports. For the patients for whom a CE2 has been generated, we have uploaded the entire data stream to the cloud. After compression, the data is kept in the cloud. We utilized a typical compressor algorithm, which is beyond the parameters of the problem statement.

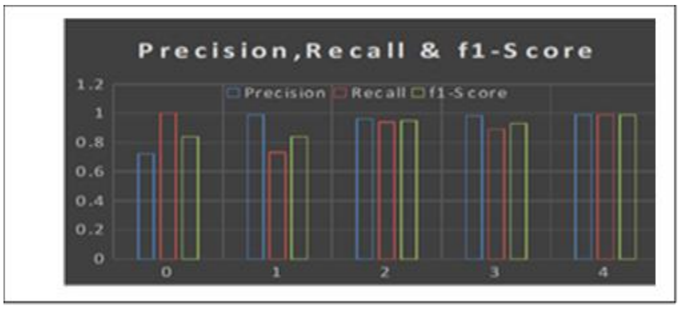


Figure 7; -Graphical representation of the Patient's Data files transfer On the cloud server (AWS)

The overall accuracy obtained for the Classification is 98.76% using CNN.

V. CONCLUSION

We started our journey by developing an effective algorithm on the edges of intelligent healthcare applications, mainly focused on the total end-to-end data transfer on the cloud. The main motive is to create an algorithm that will take all due advantages of the edge computing architecture and suit the agile framework. As we know that Old is Gold and simplicity is always better, we focused on the Fuzzy inference system and used that theory in our algorithm development. This acts smartly and looks at CPU and RAM utilization to help us improve the algorithmic performance parameters. This thesis defined the problem of massive healthcare data storage in IoT frameworks. With vast amounts of IoT data, manually storing and archiving health information becomes increasingly difficult. It's difficult to locate a patient's data in a huge record room containing a high quantity of medical files if all data is stored manually or on paper. It takes a long time and a lot of effort to locate a patient's medical records from the cloud. Patients medical information is critical for further decision-making, and a lack of access to medical information may jeopardize the delivery of the best possible care to patients. Storing patient records electronically simplifies the exchange and availability of information for healthcare processes, increasing the productivity of any patient care system that takes a central position and provides easy access and usage.

REFERENCES

[1] Dalia Rizka, Hoda Hosnya, El-Sayed El-Horbaty b, and Abdel-Badeeh Salemb, "A Study on Cloud Computing Architectures for Smart Healthcare Services",3rd International Conference on Informatics & Data-Driven Medicine, November 19–21, 2020.

[2] Ayub Manya1, Petter Nielsen1, Raphael Pundo2, "Cloud Computing as a Catalyst for Integrated Health Information Systems in Developing Countries", Association for Information Systems AIS Electronic Library (AISeL),2016.

[3] Omar Alia, Anup Shrestha, Jeffrey Soara, Samuel Fosso Wambab," Cloud computing-enabled healthcare opportunities, issues, and applications: A systematic review", International Journal of Information Management 43 (2018) 146–158.

[4] Sachin Kumar1*, Prayag Tiwari2 and Mikhail Zymbler1, "Internet of Things is a revolutionary approach for future technology enhancement: a review",https://doi.org/10.1186/s40537-019-0268-2,Springer,2019.

[5] JAGADEESWARI1, V. SUBRAMANIYASWAMY1*, R. LOGESH1 AND V. VIJAYAKUMAR2, "A STUDY ON MEDICAL INTERNET OF THINGS AND BIG DATA IN THE PERSONALIZED HEALTHCARE SYSTEM", JAGADEESWARI ET AL. HEALTH INF SCI SYST (2018) 6

[6] Peng H, Tian Y, Li L, Yang Y, Wang D. Secure and energy-efficient data transmission system based on chaotic compressive sensing in body-to-body networks. IEEE Trans Biomed Circuits Syst. 2017;11(3):1–16

[7] Xiao Chu,1 Shah Nazir,2 Kunhao Wang,3 Zeqi Leng,3 and Wajeeha Khalil4, "Big Data and Its V's with IoT to Develop Sustainability", Hindawi Scientific Programming Volume 2021, Article ID 3780594, 16 pages.

[8] Sreekanth Rallapallia,*, Gondkar RRb, Uma Pavan Kumar Ketavarapu, "Impact of Processing and Analyzing Healthcare Big Data on Cloud Computing Environment by Implementing Hadoop Cluster", International Conference on Computational Modeling and Security (CMS2016).

[9] Ahmad, Mahmood, et al. "Health fog: a novel framework for health and wellness applications." The Journal of Supercomputing 72.10 (2016): 3677-3695.

[10] Shukla, Saurabh, et al. "Architecture for latency reduction in healthcare internet-of-things using reinforcement learning and fuzzy based fog computing." International Conference of Reliable Information and Communication Technology. Springer, Cham, 2018.

[11] Greco, Luca, et al. "Trends in IoT based solutions for health care: moving AI to the Edge." Pattern Recognition Letters (2020).

[12] Alli, Adam A., and Muhammad Mahbub Alam. "The fog cloud of things: A survey on concepts, architecture, standards, tools, and applications." Internet of Things 9 (2020): 100177.

[13] Abdelmoneem, Randa M., Abderrahim Benslimane, and Eman Shaaban. "Mobility-Aware Task Scheduling in Cloud-Fog IoT-Based Healthcare Architectures." Computer Networks (2020): 107348.