TASK SCHEDULING AND PORTIONING FOR SOFTWARE DEFINED NETWORK USING REINFORCEMENT LEARNING

A PROJECT REPORT

Submitted by

DEEPIKA.M	**[613019205006]**
GOKILA.S	**[613019205012]**
MOHANAPRIYA.P	**[613019205029]**

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

VIVEKANANDHA COLLEGE OF TECHNOLOGYFOR WOMEN

ELAYAMPALAYAM, TIRUCHENGODE – 637 205

ANNA UNIVERSITY:CHENNAI 600 025

MAY 2023

ANNA UNIVERSITY:CHENNAI – 600 025

BONAFIDE CERTIFICATE

Certified that this project report "**TASK SCHEDULING AND PORTIONING FOR SOFTWARE DEFINED USING REINFORCEMENT LEARNING**" is the bonafide work of "**DEEPIKA.M(613019205006), GOKILA.S(613019205012),MOHANAPRIYA.P(613019205029)**"who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.P.PRABAHARAN, M.E., Ph.D.,	Mr.S.VIGNESH, M.E.,
HEAD OF THE DEPARTMENT,	**SUPERVISOR,**
Information Technology,	Assistant Professor,
Vivekanandha College of	Information Technology,
Technology for Women,	Vivekanandha College of
Elayampalayam,	Technology for Women,
Tiruchengode-637205.	Elayampalayam,
	Tiruchengode-637205.

Submitted to project Viva-voce examination held on _______________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We wish to convey our thanks to our honorable Chairman and Secretary **'Vidhya Rathna' Prof.Dr.M.KARUNANITHI, B.Pharm.,M.S.,Ph.D., D.Litt.,** and the trust members who have provided all the facilities to develop our projects successfully.

We are extremely grateful to our beloved Executive Director **Dr.S.KUPPUSWAMI, B.E., M.Sc., (Engg).,Dr.Ing.(France)** who gave opportunity to frame the project to the full satisfaction.

We are extremely thankful to our beloved Chief Executive **Mr.M.CHOCKALINGAM, M.Sc., B.Ed.,** for allowing us to have maximum use of facilities to do this project.

We are eternally grateful to our grateful Principal **Dr.M.DEVI, M.E., Ph.D.,** for providing us an opportunity to carry out this project work.

We extend our sincere thanks to our Head of the Department of Information Technology **Dr.P.PRABAHARAN, M.E.,Ph.D.,** and Project Coordinator **Mrs.M.MEENA, M.E.,** for their valuable help and suggestion imparted us.

We are indebted to our Project Guide **Mr.S.VIGNESH, M.E.,** who had motivated us and provided guidance throughout the execution of our project.

Finally, we are very thankful to all the teaching, non-teaching staff members, parents, sibling and friends for their support, encouragement and effective ideas during this project.

ABSTRACT

The Internet of Things (IoT) connect millions of devices in diverse areas such as smart cities, e-health, transportation and defence to meet a wide range of human needs. To provide these services, a large amount of data needs to be transmitted to the IoT network servers. Currently emergency data packets do not get any special priority while routing through the Internet of Things networks.

These data packets flow through routers using conventional QoS process which does not guarantee that an emergency data packet traveling in congested IoT network will actually be routed to control room on time. A major challenge in packet scheduling is that the behaviour of each traffic class may not be known in advance, and can vary dynamically.

Therefore, innovative packet prioritization techniques, e.g.,queue management approach needs to be developed to overcome prioritization problemsin IoT networks. This project proposes an Artificial Intelligence Packet Priority Queuing Model P^2 Queue based emergency data packet classification with a prioritization algorithm to provide a required transmission priority for emergency data.

In this project, **Long Short Term Memory (**LSTM) is used to classify the emergency data packet. Then, according to the model characteristics, a deep intelligent scheduling algorithm based on a **Deep Q Network** (DQN) framework is proposed to make scheduling decisions for communication. Effective simulation experiments demonstrate that the proposed P^{2}Queue can effectively create scheduling strategies for emergency data packet flows. Results confirmed the machine learning module achieved 91.5% of accuracy when identifying the emergency data and assigning the expected priority.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

S.NO	ABBREVIATION	EXPANSION
1	IOT	Internet Of Things
2	QOS	Quality Of Service
3	LSTM	Long Short Term Memory
4	DQN	Deep Queuing Network
5	MQTT	Message Queuing Telemetry Protocol
6	FCFS	First Come First Served
7	EDF	Earliest Deadline First
8	PQS	Priority Queues
9	DRL	Deep Inforcement Learning
10	ANN	Artificial Neural Network
11	CNN	Convolution Neural Network
12	RNN	Recurrent Neural Network
13	NLP	Natural Language Processing
14	CAE	Computer Aided Engineering
15	SARSA	State Action Reward State Action
16	PFMAC	Proposes A Priority Aware Fasmac
17	IIOT	Industrial Internet Of Things
18	DWG	Dynamic Wavelength Grouping
19	MAC	Media Access Control Address
20	HPQS	High Priority Queuing System
21	DMP	Dynamic Multilevel Priority
22	SJF	Shortest Job First
23	EDF	Earliest Deadline First

CHAPTER 1

INTRODUCTION

1.1. OVERVIEW

The Internet of Things of IoT refers to the billions of physical devices around the world that are now connected to the internet, all collecting and sharing data. Thanks to the arrival of super-cheap computer chips and the ubiquity of wireless networks, it's possible to turn anything, from something as small as a pill to something as big as an aeroplane, into a part of the IoT. Connecting up all these different objects and adding sensors to them adds a level of digital intelligence to devices that would be otherwise dumb, enabling them to communicate real-time data without involving a human being. The Internet of Things is making the fabric of the world around us smarter and more responsive, merging the digital and physical universes.

Any Device
Anybody
Anywhere
Any Business
Any Network
Anytime

Figure1.1.IoT

1.1.1. IoT Works

IoT devices are empowered to be our eyes and ears when we can't physically be there. Equipped with sensors, devices capture the data that we mightsee, hear, or sense. They then share that data as directed, and we analyse it to help us inform and automate our subsequent actions or decisions. There are fourkey stages in this process:

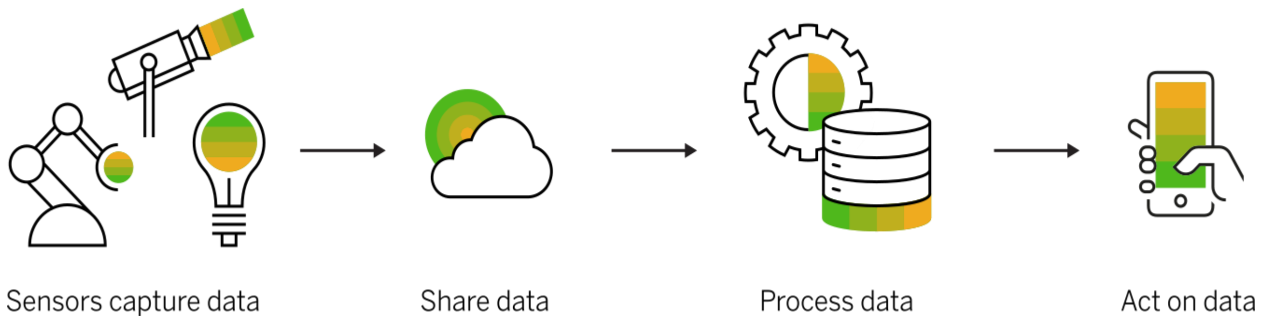


Figure1.2.A Schematic View of IoT

Capture the data - Through sensors, IoT devices capture data from their environments. This could be as simple as the temperature or as complex a real- time video feed.

Share the data - Using available network connections, IoT devices make this data accessible through a public or private cloud, as directed.

Process the data - At this point, software is programmed to do something based on that data such as turn on a fan or send a warning.

Act on the data - Accumulated data from all devices within an IoT network is analysed. This delivers powerful insights to inform confident actions and business decisions.

1.1.2. Communication models in IoT

In general, the Internet of Things is about connecting devices to the Internet, but how they connect is not always obvious. IoT devices connect and communicate through their technical communication models. An effective communication model shows how the process works and helps one understand how communication can be done. The Internet of Things (IoT) enables people and things (devices) to be connected wherever they are, using any network or service they like. Types of communication models:

Request-Response Model

This communication model is based on the client (IoT Device) making requests and the server responding to those requests. Upon receiving a request, the server decides what response to provide, fetches the requested data, prepares the response, and then sends it back to the client. This model is stateless because thedata between requests is not retained, therefore each request is handled independently.

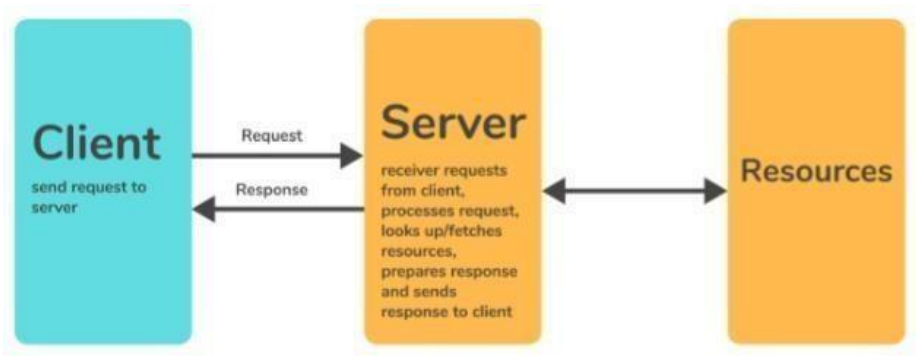

Client
send request to server
Request
Response
Server
receiver requests from client, processes request, looks up/fetches resources, prepares response and sends response to client
Resources

Figure1.3.Request Response Communication Model

Publisher-Subscriber Model

Publishers, brokers, and consumers are all involved in this communication model. Publishes are the sources of data that send data to topics. The broker manages the topics, and consumers (consume data from topics) subscribe to the topics. Publishers and consumers are unaware of each other. Upon receiving data for a topic from the publisher, the broker forwards it to all subscribed consumers. As a result, brokers are responsible for receiving data from publishers and sending it to the appropriate consumers.

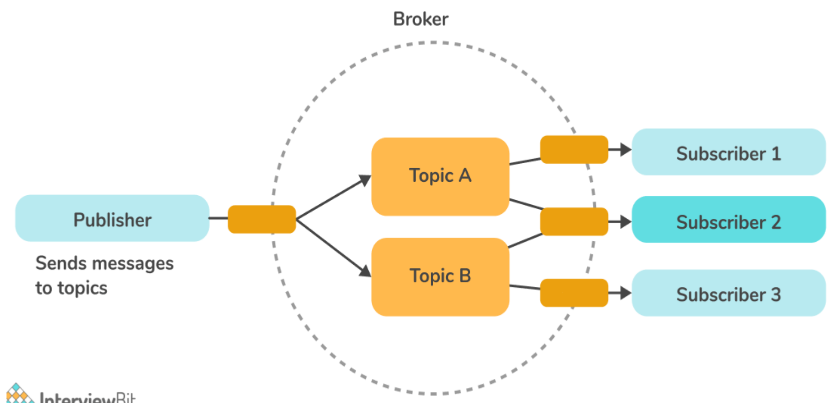


Figure1.4.Publisher Subscriber Model

Push-Pull Model

This communication model entails data producers pushing the data into queues, while data consumers pull the data from the queues. Neither producer nor consumer needs to know about each other. The queues help decouple the messages between the consumers and the producers. Also, queues act as a buffer when there is a mis match between the rate at which producers push data and the rate at which consumers pull it.

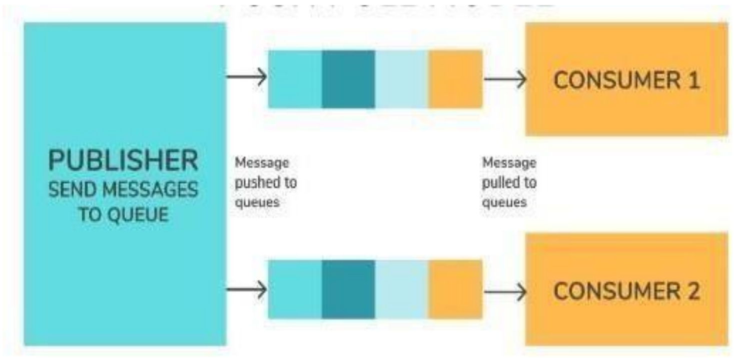


Figure1.5.Push Pull Model

Exclusive-Pair Model

Exclusive pairs are full-duplex, bidirectional communication models developed for constant/continuous connections between a client and server. Aftera connection is established, clients and servers can exchange messages. As long as a client doesn't send a request to close the connection, the connection remains open. The server is aware of every open connection.

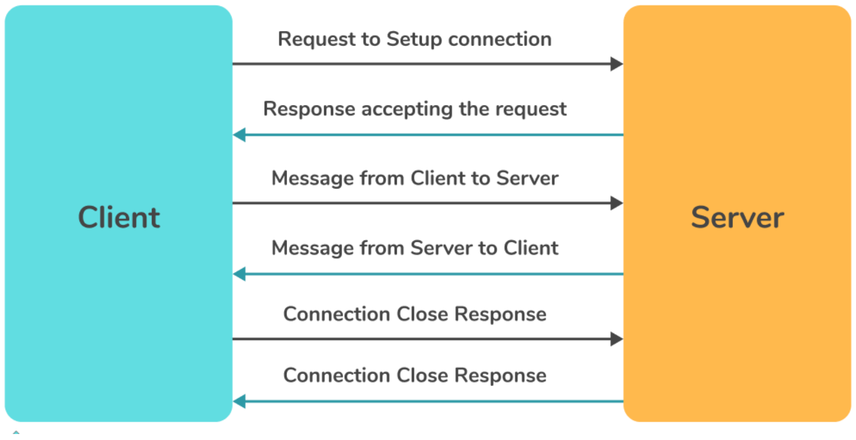

Client
Server
Request to Setup connection
Response accepting the request
Message from Client to Server
Message from Server to Client
Connection Close Response
Connection Close Response

Figure1.6.Exclusive-Pair Model

MQTT (Message Queue Telemetry Transport Protocol)

The Message Queuing Telemetry Transport protocol (MQTT) is a publish/subscribe message protocol designed for networks with limited bandwidth and IoT devices with extremely high latency (delay in data transmission). This messaging protocol is simple and lightweight, suited to devices and networks with low bandwidth, high latency, or insecure networks. Ithas been designed to reduce network bandwidth and resource requirements of devices and to ensure supply security. Further more, these principles are beneficial for IoT or M2M devices, since battery life and bandwidth are very important. Because MQTT is efficient and lightweight, it can be used to monitor or control a large amount of data. Nowadays, MQTT is used in a variety of industries, including automotive, manufacturing, telephony, oil and gas.

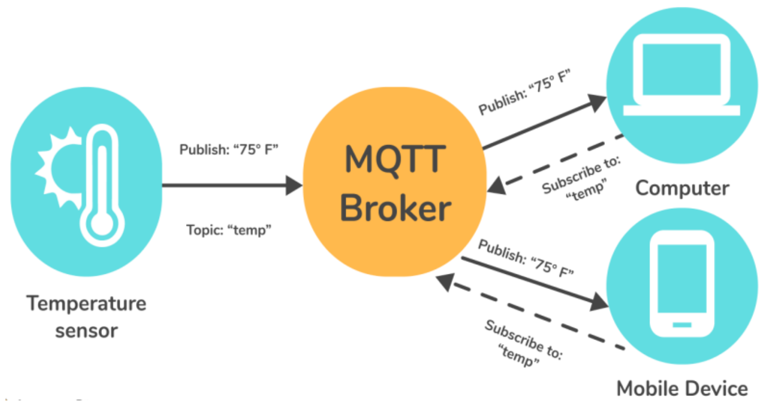


Figure1.7.MQTT

Publishes are the sources of data that send data to topics. The broker manages the topics, and consumers subscribe to the topics. Upon receiving data for a topic from the publisher, the broker forwards it to all subscribed consumers.As a result, brokers are responsible for receiving data from publishers and sending it to the appropriate consumers.

1.1.3. IoT Use cases

Optimize industrial operation

Create rich and scalable industrial IoT applications to remotely monitor operations, improve quality, and reduce unplanned down time.

Build differentiated consumer products

Develop connected consumer applications for home automation, home security and monitoring, and home networking.

Reinvent smart buildings and cities

Build commercial IoT applications that solve challenges in infrastructure, health, and the environment.

Transform mobility

Deliver IoT applications that gather, process, analyse, and act on connectedvehicle data, without having to manage any infrastructure.

1.2. PROBLEM IDENTIFIED

The Internet of Things (IoT) for smart cities is a large integrated system that consists of sensor networks, wireless mesh networks, mobile communications, social networks, smart transportation, and intelligent transportation. It has become a hot topic in many applications at present. In most of the existing works in IoT for smart cities, the researchers mainly focused on the improvement of the routing algorithm and node energy saving management, to improve the network performance. However, with the expansion of network scale and applications, the data in network become so large or complex that traditional data scheduling schemes are inadequate to deal with them. In addition,there are a lot of data packets that need to be sent to the sink node as soon as possible, called emergency packets, such as the alarm information in medical rescue service, the fire information in forest fire monitoring service, etc. Thus, data packets emergency response has become a serious challenge, especially in the disaster recovery system. Lots of packet scheduling algorithms for the large- scale networks with big data are proposed to reduce the end-to-end packet transmission delays and make the emergency packets reach the sink node before the expiration of deadline. In recent years, the research works for data packet scheduling are mainly divided into three types: First-Come First-Served (FCFS), Earliest Deadline First (EDF), and emergency task first rate monotonic. Here, FCFS is widely used. This algorithm is generally used in multilevel Priority Queues (PQs). These algorithms do not consider the impact between different nodes, so there will be some extra time when the emergency data packets are sent to the sink node. Thus, we need an efficient packet scheduling algorithm that can allocate network resources rationally. The packets can be scheduled based on their emergency information, which can ensure timeliness of emergency data packets and the effectiveness of non emergency data packets. When the actual states of the network change and the transmission strategies need to be adjusted, the existing network protocols lack intelligence. Therefore, with the growth in network scale and the explosion in data, Artificial Intelligence (AI) has been drastically promoted in recent years. The AI technique has already made break throughs in a variety of areas, such as robot control, automatic drive, and speech recognition. Compared with conventional methods, Deep Reinforcement Learning (DRL) that emerged from AI has shown great advantages in large-scale networks.

1.3. DEEP LEARNING

Deep Learning is a field which comes under Machine Learning and is related to the use of algorithms in artificial neural networks. It is majorly used to create a predictive model to solve the problems with just a few lines of coding.A Deep Learning system is an extensive neural network which is inspired by the function and structure of the brain. Deep learning models can be supervised, semi-supervised, or unsupervised (or a combination of any or all of the three). They are advanced machine learning algorithms used by tech giants, like Google, Microsoft, and Amazon to run entire systems and power things, like self-driving cars and smart assistants.

First, machine learning had to get developed. ML is a framework to automate (through algorithms) statistical models, like a linear regression model, to get better at making predictions. A model is a single model that makes predictions about something. Those predictions are made with some accuracy.

A model that learns machine learning takes all its bad predictions and tweaks the weights inside the model to create a model that makes fewer mistakes.

The learning portion of creating models spawned the development of artificial neural networks. ANNs utilize the hidden layer as a place to store and evaluate how significant one of the inputs is to the output. The hidden layer stores information regarding the input's importance, and it also makes associations between the importance of combinations of inputs.

1.3.1. Deep Learning Process

A deep neural network provides state-of-the-art accuracy in many tasks, from object detection to speech recognition. They can learn automatically, without predefined knowledge explicitly coded by the programmers.tg6rewqtwer5u80p

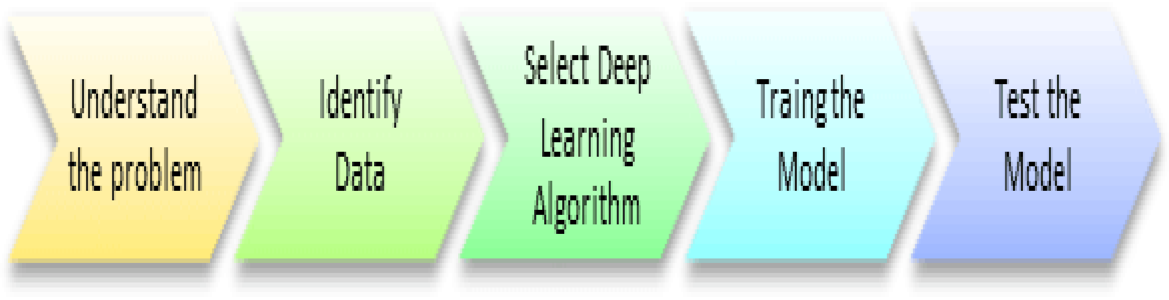


Figure1.8.Deep learning Process

1.3.2. Classification of Neural Networks

- **Shallow neural network**: The Shallow neural network has only one hidden layer between the input and output.
- **Deep neural network**: Deep neural networks have more than one layer. For instance, Google LeNet model for image recognition counts 22 layers.

1.3.3. Types of Deep Learning Networks

Now in this Deep Neural network tutorial, we will learn about types of Deep Learning Networks:

Multilayer Perceptron

The simplest kind of neural network is known as a perceptron, which usually takes inputs from two or more input nodes directly into an output node. There is no hidden layer involved in this type of neural network.

Convolutional Neural Networks

Convolutional Neural Networks (CNN) are a type of neural network that can assign importance to the given data on its own, taking the load off human labour to accurately label and classify the data. CNNs can drastically reduce the amount of time it takes to pre-process data, as it can label the data on its own. Recurrent Neural Networks (RNN) utilize many inputs, outputs, and hidden layers in series, with the network being able to memorize past inputs and outputs. These networks make future decisions based not only on the data given to them at that point in time but also based on the past data that they had ingested.

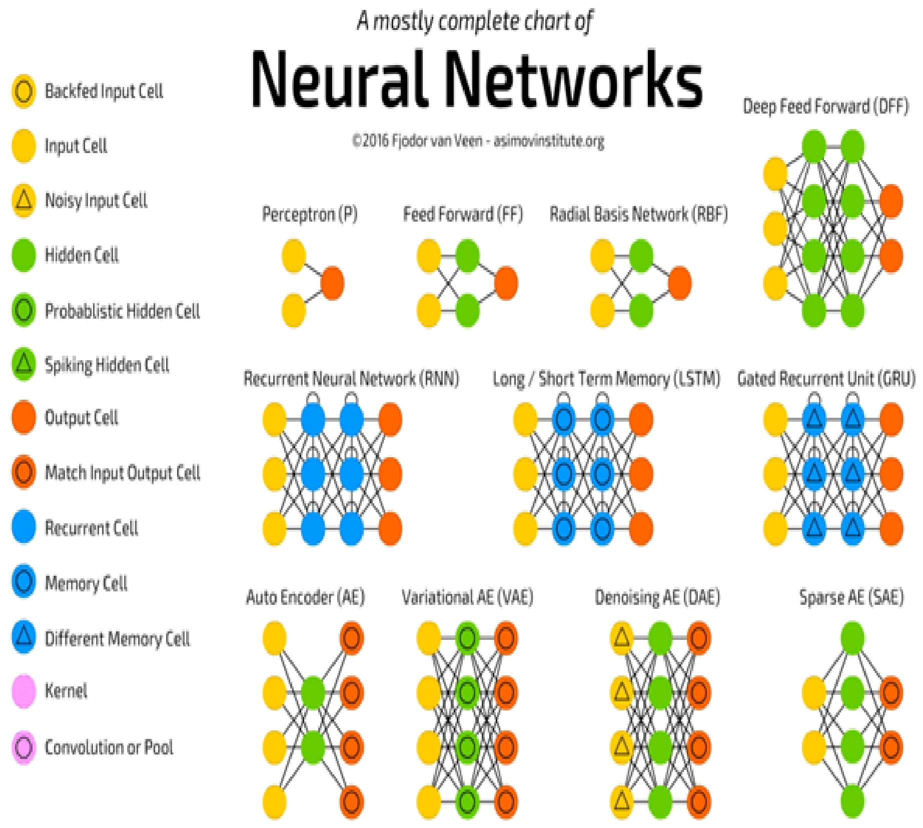


Figure1.9.Deep Learning Networks

1.3.4. Use cases of Deep Learning

Social Media

Deep learning can be used to analyse a large number of images, which canhelp social networks find out more about their users. This improves targeted ads and follow suggestions.

Finance

Neural networks in deep learning can be used to predict stock values and develop trading strategies, and can also spot security threats and protect against fraud.

Healthcare

Deep learning can play a pivotal role in the field of healthcare by analysing trends and behaviours to predict illnesses in patients. Healthcare workers can also employ deep learning algorithms to decide the optimal tests and treatments for their patients.

Cyber Security

Deep learning can detect advanced threats better than traditional malware solutions by recognizing new, suspicious activities rather than responding to a database of known threats.

Digital Assistants

Digital assistants represent some of the most common examples of deep learning. With the help of Natural Language Processing (NLP), Siri, Cortana, Google, and Alexa can respond to questions and adapt to user habits.

Manufacturing

Manufacturers need to deliver higher quality products and services faster and with lower costs. Many companies are adopting Computer-Aided Engineering(CAE) to reduce the time, expense, and materials needed to develop physical prototypes to test new products. Deep learning can be used to model very complexpatterns in multi dimensional data and improve the analytics accuracy of testing data.

1.4. REINFORCEMENT LEARNING

Reinforcement Learning is a subfield of machine learning that teaches an agent how to choose an action from its action space, within a particular environment, in order to maximize rewards over time.

Reinforcement Learning has four essential elements:

1. **Agent -** The program you train, with the aim of doing a job you specify.

2. **Environment** - The world, real or virtual, in which the agent performs actions.
3. **Action** - A move made by the agent, which causes a status change in the environment.
4. **Rewards** - The evaluation of an action, which can be positive or negative.

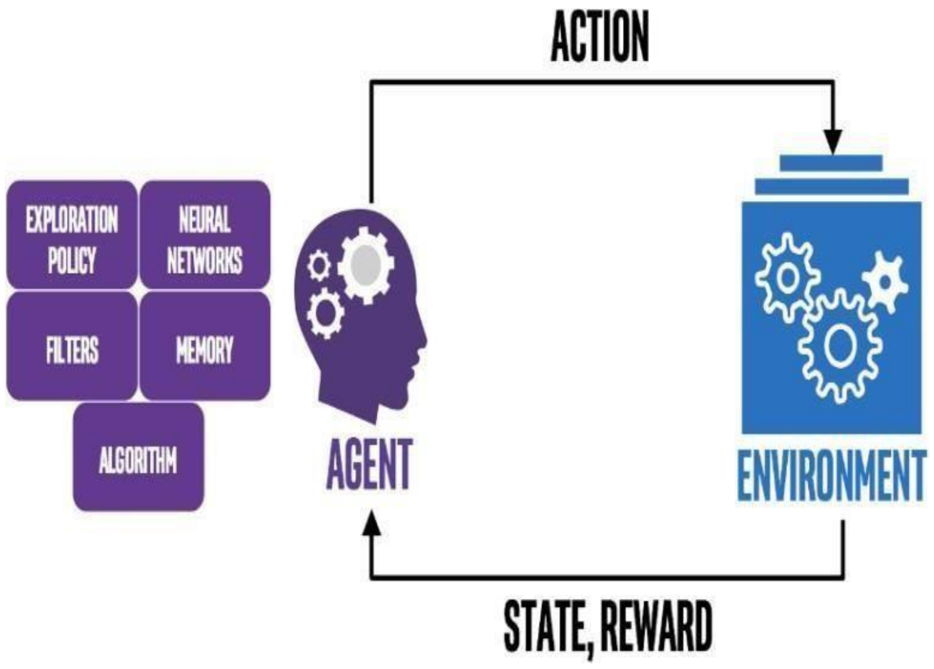


Figure1.10.Reinforcement Learning

1.4.1. Reinforcement Learning Algorithms

Rather than referring to a specific algorithm, the field of reinforcement learning is made up of several algorithms that take somewhat different approaches. The differences are mainly due to their strategies for exploring theirenvironments.

State-Action-Reward-State-Action (SARSA)

This reinforcement learning algorithm starts by giving the agent what's known as a policy. The policy is essentially a probability that tells it the odds of certain actions resulting in rewards, or beneficial states.

Q-Learning

This approach to reinforcement learning takes the opposite approach. The agent receives no policy, meaning its exploration of its environment is more self-directed.

Deep Q-Networks

These algorithms utilize neural networks in addition to reinforcement learning techniques. They utilize the self-directed environment exploration of reinforcement learning. Future actions are based on a random sample of past beneficial actions learned by the neural network.

1.5. OBJECTIVE OF THE PROJECT

The objective of the project is to propose an Artificial Intelligence based Packet Priority Queuing model for emergency data packet classification with a prioritization algorithm to provide a required transmission priority for emergency data.

CHAPTER 2

LITERATURE SURVEY

1. Priority-Aware Fast MAC Protocol for UAV-Assisted Industrial IoT Systems

Author: Shreya Khisa;Sangman Moh

Year: 2021

Link: https://ieeexplore.ieee.org/document/9400366

Objective:

This article proposes a Priority-Aware Fast Mac (pf-mac) protocol for uav-assisted iiot systems, ensuring fast and robust data delivery. The hybrid pf-mac protocol integrates the benefits of both contention-based and contention-free protocols for a remote iot scenario.

Methodology:

This paper proposes a Priority-Aware Fast MAC (PF-MAC) Protocol for UAV-assisted industrial Internet-of-Things (IIoT) systems, ensuring fast and robust data delivery. At first, the IoT devices under the UAV communication range transmit a reservation frame to the UAV to catch transmission opportunities using CSMA/CA. The devices utilize static traffic priority and a novel adaptive back off mechanism during CSMA/CA. After receiving the reservation frames from the IoT devices, the UAV calculates the dynamic device priority based on their static traffic priority, communication duration, sampling frequency, and remaining energy. Then, time slot is assigned by the UAV to each device for data transmission. To ensure fairness, if a device fails in contention during the CSMA/CA period, the static traffic priority is raised in the next re transmission. There is no prior work in the literature that considers both the traffic priority andthe device priority to ensure Quality of Service in IIoT.

Advantages:

- Uav-assisted Industrial Internet-of-Things (IIoT) systems, ensuring fast and robust data delivery.

Disadvantages:

- Developments in wireless and mobile networking technology have affected every aspect of our everyday lives.

2. Dynamic Wavelength Grouping for Quality of Service in Optical Packet Switching

Author: Hafsa Bibi;Farrukh Zeeshan Khan

Year: 2021

Link: https://ieeexplore.ieee.org/document/9405790

Objective:

This article aimed to analyze DWG with more functions like plr, pdv, bit error rate and response time. Also, design a framework provisioning of resources in more adverse load situations at edge ops switches with assurance of same impact on these factors.

Methodology:

This paper proposed and analyzed an advanced technique DWG for absolute qos in an op, which makes dynamic and most appropriate partitions of available wavelength resources and allocate them to traffic of each class of service in rapidly varying traffic loads by tracking the current link load status. The wavelength grouping changes over time with change in load share of each class, and results in efficient utilization of wavelength resources at each link.We described a baseline ops model to express the expected performance, and the briefly examined the proposed technique over various qos metrics to evaluate itsperformance and effectiveness.

Advantages:

- Flexible approach, efficiently functional even when the share of the high-priority load is temporarily low.

Disadvantage:

- Ops have higher Packet Delay Variation (PDV), Packet Loss Ratio (PLR) and transmission delay than ocs.

3. An Efficient Reservation-Based MAC Protocol for Multi- Priority Traffic in Slotted Multi-Channel Distributed Cognitive Radio Networks

Author: Jaehwoon Lee

Year: 2020

Link: https://ieeexplore.ieee.org/document/9218918

Objective:

This article proposes protocol and the analytical results show that higher priority traffic can be transmitted first ahead of the lower priority traffic.

Methodology:

This article having data packets with different class of priorities transmits its control packet containing the priority value through the control channel. The order of access to primary channels is determined based on the priority of the data packet and the position of the non-colliding control packet. The access order determines the idle primary channel that an SU uses to transmit its data packet. In this protocol, there is no performance degradation either from SU's choosing a busy primary channel or multiple SU's choosing the same idle primary channel. Moreover, even though the SU cannot transmit its data packet because there is no idle primary channel that the SU can utilize, it can re- transmit its control packet without having concern over additional collision.

Advantages:

- Data packet arrives at an SU in idle state at the beginning of a slot, the SU changes its state to the sensing state.

Disadvantages:

- Traffic can be transmitted first ahead of the lower priority traffic.

4. Queue and Priority-Aware Adaptive Duty Cycle Scheme for Energy Efficient Wireless Sensor Networks.

Author: Bashir A. Muzakkari;Mohamad A.

Year: 2020

Link: https://ieeexplore.ieee.org/document/8963893

Objective:

This article proposed scheme is validated using numerous experiments conducted under different network conditions to evaluate the significance of energy preservation, extended network lifetime and minimize packet delay.

Methodology:

This paper proposes a scheduling algorithm that uses an optimized randomearly detection algorithm to provide low queuing delay for priority packets with an exponential weighted moving average to solve the problem of starvation suffered by low priority classes by keeping the value of the average queue length below the minimum threshold.

Advantages:

- Duty cycling coordinates the sleep/wake-up time of sensor nodes to maximize the network lifetime.

Disadvantages:

- Energy consumption is considered as the most fundamental issue of wsn, and it is widely affected by the communication.

5. Priority access for QoS support in distributed wireless networks

Author: Zhou Xin; Zheng Changwen

Year: 2019

Link: https://ieeexplore.ieee.org/document/8945571

Objective:

This article proposes scheme significantly improves the real-time traffic capacity, throughput, delay, fairness and packet loss rate.

Methodology:

This paper proposes a scheme called the Priority Access Based on Busy Tone (PABT) for distributed wireless networks. It uses an in-band slot-length Busy Tone (BT) signal, i.e., sine signal, to reserve the channel for absolute priority access. Based on that, it also improves the real-time traffic capacity and the aggregated through put significantly through the optimal CW tuning for each traffic type individually.

Advantages:

- The priority channel access, this scheme uses an in-band busy tone to limit the transmission of lower-priority traffic when higher-priority traffic as packets to send.

Disadvantages:

- It is not so easy to support QoS in wireless networks, because the signal transmission may suffer from the interference.

6. Performance Analysis of D2D Underlying Cellular Networks Based on Dynamic Priority Queuing Model.

Author: Jianfang Xin;Qi Zhu;Guangjun Liang

Year: 2019

Link: https://ieeexplore.ieee.org/document/8624403

Objective:

This article simulation results show the validity of the theoretical analysis. Moreover, by comparing the dropping probability of the priority queuing model with and without the jump strategy, the rationality of the introduced model is confirmed.

Methodology:

This article develops a spatio temporal mathematical model for analyzing the performance of prioritized data transmissions in the Device-to-Device (D2D) underlying cellular network. A dynamic interference model of a D2D user is constructed by exploiting thinned poison point process to model the D2D user location with data stored in the buffer. A dynamic priority queuing model is adapted to analyze the performance of multiple types of traffic, in which the priority jump strategy is proposed to provide increased transmission opportunity for low-priority user packets. Then, we employ a two-dimensional geo/g/1 markov chain to describe a queue model with priority jump and evaluate it using quasi-birth-and-death process approach.

Advantages:

- Provide increased transmission opportunity for low-priority user packets.

Disadvantages:

- Delay outage performance was analyzed with random arrival of traffic in heterogeneous cellular networks.

7. HPQS: A Fast, High-Capacity, Hybrid Priority Queuing System for High-Speed Networking Devices.

Author: Imad Benacer;François-Raymond Boyer

Year: 2019

Link: https://ieeexplore.ieee.org/document/8842569

Objective:

This article proposes a Hybrid Priority Queuing System (HPQS) with two distinct configurations. The first configuration is intended for strict priority scheduling with distinct queues.

Methodology:

This article presents placement and routing results of the HPQS in a ZC706 Field Programmable Gate Array (FPGA) board and xcvu440 device, the total capacity can reach 1/2 million packet tags of 16-bit priority keys in a single FPGA. The HPQS is proposed for high-speed networking devices operating in constant 1-cycle latency per packet (queue operation) targeting 10 to 40 gb/s network links. Moreover, the performance of the proposed HPQS is independent of the PQ capacity. Also, the proposed HPQS architecture is entirely coded in C++,providing easier implementation and more flexibility than some reported works.

Advantages:

- Reducing latency to best support the upcoming 5g wireless standards.

Disadvantages:

- Increasing traffic and tight requirements of high-speed networking devices, a high capacity priority queue.

8. Research on Multi-Level Priority Polling MAC Protocol in FPGA Tactical Data Chain.

Author: Hongwei Ding;Chao Li;

Year: 2019

Link: https://ieeexplore.ieee.org/document/8656465

Objective:

This article proposed a multi-level priority polling mac protocol mppmp, which controlled the arrival rate of each user with a priority weight matrix according to the priority level of different users in the tactical data link system.

Methodology:

This article ensures the urgency of priority site transmission in tactical datalink system and better serve information warfare, we use the idea of multi- priority service to improve the traditional gated polling system and propose a multi-priority polling control protocol. Depending on the priority, multiple priorities are set. The protocol adds a priority control field to the ieee 802.11 macframe, and uses the priority control field to distinguish the user's priority. The smaller the value of the priority control field, the higher the priority.

Advantages:

- Important impact on system performance and has always been a hot issue in tactical data link research.

Disadvantages:

- Multiple users will co-exist in the tactical data link system, and each user's privilege level will not be the same. The importance and urgency of the messages transmitted by each user are also different.

9. Achieve Privacy-Preserving Priority Classification on Patient Health Data in Remote e-Healthcare System.

Author: Guoming Wang; Rongxing Lu

Year:2019

Link: https://ieeexplore.ieee.org/document/8606116

Objective:

This article aims at solving the above challenges; we propose an efficient and Preserving Priority Classification (PPC) on patient health data in remote e-healthcare system, which allows authenticated users to periodically send medical packets to the healthcare center through WBAN-gateways.

Methodology:

This article proposes an efficient and privacy-preserving priority classification scheme, named PPC, for classifying patients' encrypted data at theWBAN -gateway in a remote e-healthcare system. Specifically, to reduce the system latency, we design a non-interactive privacy-preserving priority classification algorithm, which allows the WBAN-gateway to conduct the privacy-preserving priority classification for the received users' medical packetsby itself and to relay these packets according to their priorities (criticalities). A detailed security analysis shows that the PPC scheme can achieve the priority classification and packets relay without disclosing the privacy of the users' personal information and the confidentiality of the healthcare center's disease models.

Advantages:

- The PPC scheme, an efficient privacy preserving non-interactive priority classification scheme for users' medical packets in WBAN-gateways.

Disadvantages:

- The attacker knows some users' information or some disease models. Even in this context, the attacker cannot recover other plaintext of the corresponding encrypted data.

10. A Priority-Based Reservation MAC Protocol in Multi-Channel Cognitive Radio Networks

Author: Jaehwoon Lee

Year:2018

Link: https://ieeexplore.ieee.org/document/8481351

Objective:

This article proposed protocol in an independent and distributed manner. Also, any new SU's can enter the network at any time. It is because a new SU checks the control channel during one slot and can know the current status ofthe network.

Methodology:

This article has presented a priority-based reservation mac protocol in slotted multi-channel cognitive radio networks. In this protocol, a common control channel is exclusively used by SU's to determine the access priority of the SU's. A slot of the common control channel is further divided into the time period to sense all primary channels and number of mini slots. Moreover, mini slots can be divided into two parts: reservation part and contention part. When a data packet arrives at an SU, the SU counts the number of successful control packets by monitoring one slot of the control channel. And then, the SU randomly chooses one of the mini slots belonging to the contention part. If the control packet is successfully transmitted without collision, then the SU gets the priority based on the position of the control packet.

Advantages:

- SU's to transmit control packets in order to determine the priority for accessing the primary channels.

Disadvantages:

- Results show that the number of Su's affects the performance due the collision possibility of random access.

CHAPTER 3

SYSTEM ANALYSIS

3.1. EXISTING SYSTEM

Existing works for data packet scheduling are mainly divided into three types: First-Come First-Served (FCFS), Earliest Deadline First (EDF), and Emergency Task First Rate Monotonic.

Priority Queuing

Packets are sorted into different priority queues. Higher priority packets are transmitted ahead of lower priority packets. The priority-based scheduling algorithms include two types: preemptive scheduling and non-preemptive scheduling. In the non-preemptive scheduling algorithm, a packet once started to be processed, other packets, even though the priorities are higher, have to wait until the packet processing is completed. Differently, preemptive scheduling algorithm is more flexible. At each node, the packets with higher priorities can preempt packets with lower priorities. Evaluate preemptive scheduling and non preemptive scheduling through a lot of experiments.

Deadline-Based Scheduling

In this type of packet scheduling, the deadlines of packets are considered. In the Rate Monotonic Scheduling (RMS) algorithm, the priorities of packets are static priorities, which are assigned based on the deadlines of packets. The priorities of the data packets forRMS are static. Thus, the RMS scheme is limited and inapplicable. In the EDF algorithm, the priority of each packet is determined according to the deadline. Thus, the overhead is relatively large.

Packet-Type-Based Scheduling

There are different types of packets in this scheduling scheme, such as real-time packets and non-real-time packets. The priorities of the packets are determined based on the packet types.

Class-Based Queuing

Packets are sorted into queues, one queue per class. Packets within each queue are transmitted FCFS. Different queues are served in round-robin fashion.

Dynamic Multilevel Priority

Packet scheduling technique exists for WSNs in which the sensor nodes are arranged into a hierarchical structure.

Shortest Job First (SJF)

This technique ensures the packet delivery on the basis of their priority with lesser time.

Earliest Deadline First (EDF)

Which involves the arrival time and deadline as their basis of data processing respectively.

3.2. DISADVANTAGES

- Lack of efficient scheduling algorithms
- Inappropriate delivery of emergency packets
- Real time data mismanagement
- Packet dropping
- Unwanted delays
- Lesser network lifetime
- Large waiting time and inefficient energy utilization

3.3. PROPOSED SYSTEM

The proposed system prioritizes data on IoT devices on the basis of data importance extracted from the deep learning based LSTM model for prediction, which enables it to reduce the total data traffic for real-time prediction while maintaining prediction accuracy. The proposed system consists of two main components: IoT devices and an edge server. IoT devices (such as probe vehicles, WBAN, smartphones, and UAVs) prioritize collected data and send high importance input data for prediction to the edge server. The edge server aggregates the data received from IoT devices, complements the missing parts ofthe data, and performs prediction. Predicted packets are then scheduled using Deep Q Learning.

- **LSTM – Long Short Term Memory**
- **LSTM – Long Short-Term Memory**

LSTMs are a special kind of RNN which is capable of learning long-term dependencies. LSTMs are designed to dodge long-term dependency problem as they are capable of remembering information for longer periods of time. Long Short-Term Memory (LSTM) units (or blocks) are a building unit for layers of a recurrent Neural Network (RNN). A RNN composed of LSTM units is often calledan LSTM network. A common LSTM unit is composed of a cell, an input gate, anoutput gate and a forget gate. The cell is responsible for "remembering" values over arbitrary time intervals; hence the word "memory" in LSTM. Each of the three gates can be thought of as a "conventional" artificial neuron, as in a multi- layer (or feed forward) neural network: that is, they compute an activation (using an activation function) of a weighted sum. Intuitively, they can be thought as regulators of the flow of values that goes through the connections of the LSTM; hence the denotation "gate". There are connections between these gates and the cell. The expression long short-term refers to the fact that LSTM is a model for the short-term memory which can last for a long period of time. An LSTM is well-suited to classify, process and predict time series given time lags of unknownsize and duration between important events. LSTMs were developed to deal withthe exploding and vanishing gradient problem when training traditional RNNs.

The popularity of LSTM is due to the Getting mechanism involved with each LSTM cell. In a normal RNN cell, the input at the time stamp and hidden state from the previous time step is passed through the activation layer to obtain a new state. Whereas in LSTM the process is slightly complex, as you can see in the above architecture at each time it takes input from three different states like the current input state, the short-term memory from the previous cell and lastly the long-term memory.

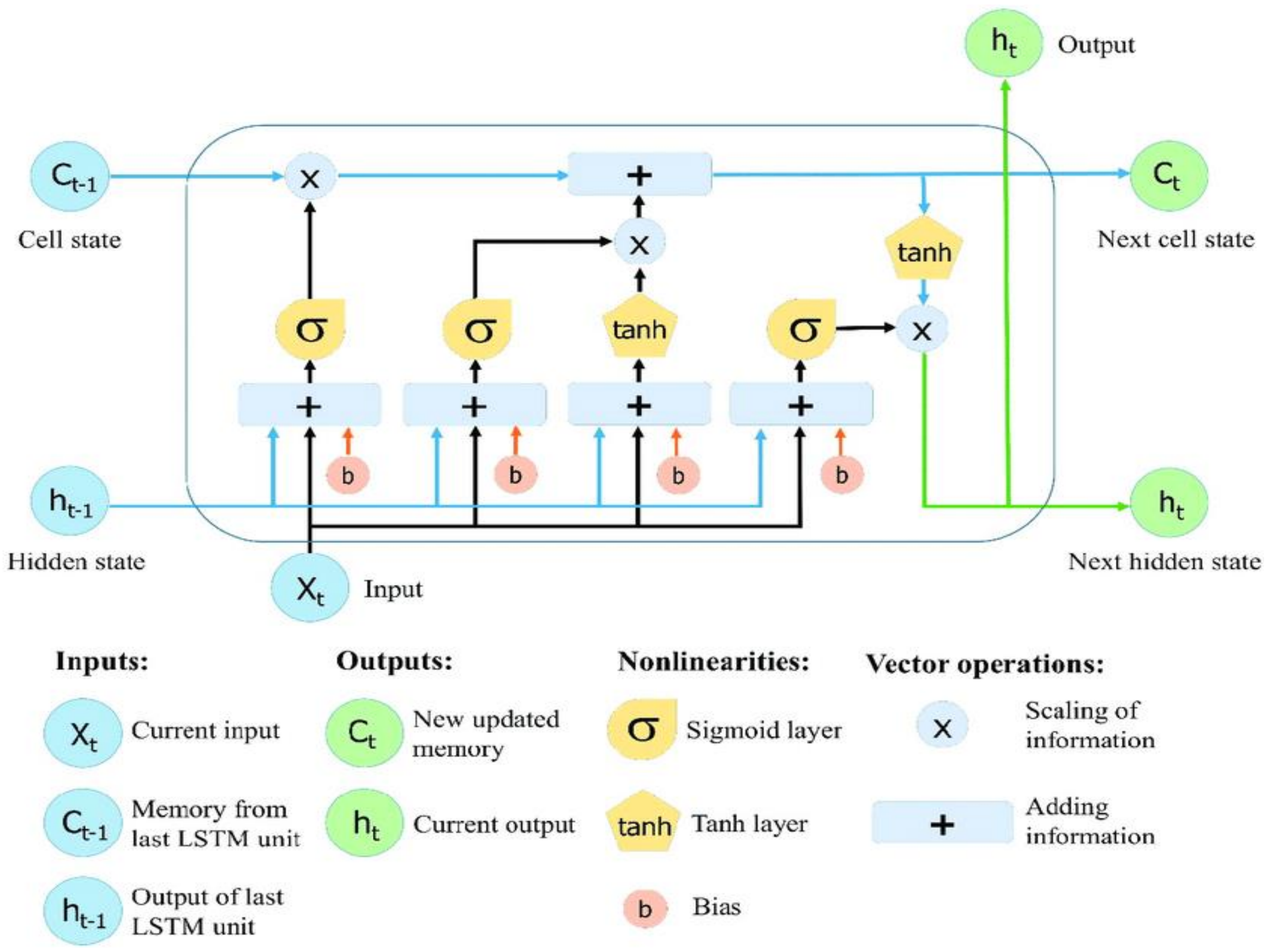


Figure3.1.LSTM

These cells use the gates to regulate the information to be kept or discarded at loop operation before passing on the long term and short-term information to the next cell. We can imagine these gates as Filters that remove unwanted selected and irrelevant information. There are a total of three gates that LSTM uses as Input Gate, Forget Gate, and Output Gate.

Input Gate

The input gate decides what information will be stored in long term memory. It only works with the information from the current input and short- term memory from the previous step. At this gate, it filters out the information from variables that are not useful.

Forget Gate

The forget decides which information from long term memory be kept or discarded and this is done by multiplying the incoming long-term memory by a forget vector generated by the current input and incoming short memory.

Output Gate

The output gate will take the current input, the previous short-term memory and newly computed long-term memory to produce new short-term memory which will be passed on to the cell in the next time step. The output of the current time step can also be drawn from this hidden state.

Deep Q-Learning Algorithm

As one of the main reinforcement learning algorithms, Q learning is a model-free learning method which provides the intelligent system with the abilityto select the optimal action according to the action sequences from experience in the Markov environment. A key assumption of Q-learning is that the interaction between the agents and the environment can be treated as a Markov Decision Process (MDP), i.e., the current state and action of the agent will determine the state transfer probability distribution and the next state with an immediate reward.

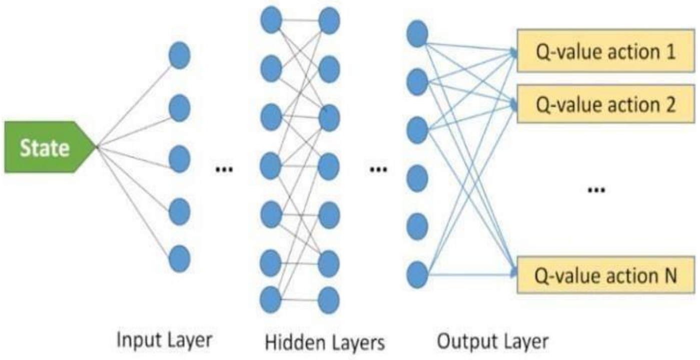


Figure 3.2. Deep Q-Learning

The goal of Q-learning is to find a policy that can maximize the reward. The Q-value is an important parameter in Q-learning. It is defined as the sum of rewards for executing the current related actions and those to be performed sub sequently in accordance with a certain strategy. A given state s and action a correspond to a given Qvalue Q(s,a). Q-value is used in the learning process to select the action. If the sub sequent actions are performed according to the optimal polices the corresponding Q-value is referred to as the optimal Q value Q*,

$$Q^*(s,a) = r(s,a) + \gamma \sum T(s,a,s') max Q^*(s',a').$$

where T (s, a, s') represents the transfer probability from states to state s' via action a, r(s,a) represents the reward for executing action a from states, γ∈(0,1) is the discount factor, which indicates the degree of farsightedness. If the γ value is small, the system pays attention to only the recent actions. If γ is large the actions during a relatively long period of time are involved. An agent learning process can be viewed as selecting an action from a random state using a strategy. The value of Q(s,a) is updated according to

$$Q_{t+1}(s,a) = (1-a) + Q_t(s,a) + a[\gamma max(s',a')]$$

where α∈ (0,1) is the learning factor used to control the speed of learning: the greater the value of α, the faster the convergence speed. After performing the selected action, the agent observes the new state and the reward obtained, and then updates the Q-value of the state and action based on the maximum Q- valueof the new state. In this way the agent continually updates the action according to the new state until it arrives at the terminal state with an optimalQ-value Q*.

3.4. ADVANTAGES

- Accurately predicting the emergency data packet.
- Emergency data will get higher priority and less delay over normal data;
- Data aggregation will result in less energy consumption and longer network lifetime.
- Control the network congestion effectively and increase the network throughput.
- Minimization of end to end delay
- Reduced waiting
- Time and delivery of data before expiration of deadline

CHAPTER 4

SYSTEM SPECIFICATION

4.1. HARDWARE REQUIREMENTS

- Processors: Intel® Core™ i5 processor 4300M at 2.60 GHz or 2.59GHz (1 socket, 2 cores, 2 threads per core), 8 GB of DRAM
- Disk space: 320 GB
- Operating systems: Windows® 10/ macOS*/and Linux*

4.2. SOFTWARE REQUIREMENTS

• Server Side	:	Python 3.7.4(64-bit) or (32-bit) and PHP
• Client Side	:	HTML, CSS, Bootstrap
• IDE	:	Tkinter, Dreamweaver
• Back end	:	MySQL 5
• Server	:	WampServer 2i

CHAPTER 5

SYSTEM DESCRIPTION

5.1. FRONT END : DEVELOPMENT

5.1.1. Python 3.7.4

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido Van Rossum during 1985-1990. Like Perl, Python source code is also available under the General Public License (GPL). This tutorial gives enough understanding on Python programming language.

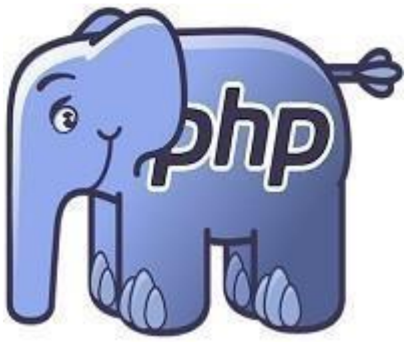
Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages. Python is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain.

Python is currently the most widely used multi-purpose, high-level programming language. Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc. The biggest strength of Python is huge collection of standard library which can be used for the following:

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like OpenCV, Pillow)
- Web scraping (like Scrapy, Beautiful Soup, Selenium)
- Test frameworks
- Multimedia
- Scientific computing
- Text processing and many more.

5.1.2. PHP 8.1

The PHP Hypertext Preprocessor (PHP) is a programming language that allows web developers to create dynamic content that interacts with databases. PHP is basically used for developing web-based software applications. This tutorial helps you to build your base with PHP.



PHP is a flexible, dynamic language that supports a variety of programming techniques. It has evolved dramatically over the years, notably adding a solid object-oriented model in PHP 5.0 (2004), anonymous functions and name spaces in PHP 5.3 (2009), and traits in PHP 5.4 (2012).

Object Oriented Programming

PHP has a very complete set of object-oriented programming features including support for classes, abstract classes, interfaces, inheritance, constructors, cloning, exceptions, and more.

Functional Programming

PHP supports first-class functions, meaning that a function can be assigned to a variable. Both user-defined and built-in functions can be referenced by a variable and invoked dynamically. Functions can be passed as arguments to other functions (a feature called Higher-order Functions) and functions can return other functions.

Standard PHP Library

The Standard PHP Library (SPL) is packaged with PHP and provides a collection of classes and interfaces. It is made up primarily of commonly needed data structure classes (stack, queue, heap, and so on), and iterators which can traverse over these data structures or your own classes which implement SPL interfaces.

Command Line Interface

PHP was created to write web applications, but is also useful for scripting Command Line Interface (CLI) programs. Command line PHP programs can help automate common tasks like testing, deployment, and application administration.

5.2. FRONT END : DESIGN

5.2.1. Bootstrap 4

Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.

It solves many problems which we had once, one of which is the cross-browser compatibility issue. Nowadays, the websites are perfect for all the browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets, Phablets, and Phones). All thanks to Bootstrap developers -Mark Otto and Jacob Thornton of Twitter, though it was later declared to be an open-source project.

Easy to use: Anybody with just basic knowledge of HTML and CSS can start using Bootstrap

Responsive features: Bootstrap's responsive CSS adjusts to phones, tablets, and desktops

Mobile-first approach: In Bootstrap, mobile-first styles are part of the core framework

Browser compatibility: Bootstrap 4 is compatible with all modern browsers (Chrome, Firefox, Internet Explorer 10+, Edge, Safari, and Opera)

Using an IDE

As good as dedicated program editors can be for your programming productivity, their utility pales into in significance when compared to Integrated Developing Environments (IDEs), which offer many additional features such as in-editor debugging and program testing, as well as function descriptions and much more.

5.2.2. Tkinter

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications.

Tkinter

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets. There are currently 15 types of widgets in Tkinter.

5.2.3. Macromedia Dreamweaver

Macromedia® Dreamweaver® 8 from Adobe is the industry-leading web development tool that lets you efficiently design, develop and maintain standards-based websites and applications. Dreamweaver 8 provides a powerful combination of visual layout tools, application development features, and code editing support.

Dreamweaver 8 includes many new features to help you create and maintain websites that range from basic home pages to advanced applications thatsupport best practices and the latest technologies.

5.3. BACK END : MYSQL

MySQL tutorial provides basic and advanced concepts of MySQL. Our MySQL tutorial is designed for beginners and professionals. MySQL is a relational database management system based on the Structured Query Language, which is the popular language for accessing and managing the records in the database. MySQL is open-source and free software under the GNU license. It is supported by Oracle Company. MySQL database that provides for how to manage database and to manipulate data with the help of various SQL queries. These queries are: insert records, update records, delete records, select records, create tables, drop tables, etc. There are also given MySQL interview questions to help you better understand the MySQL database.

MySQL is currently the most popular database management system software used for managing the relational database. It is open-source database software, which is supported by Oracle Company. It is fast, scalable, and easy to use database management system in comparison with Microsoft SQL Server and Oracle Database. It is commonly used in conjunction with PHP scripts for creating powerful and dynamic server-side or web-based enterprise applications.

5.4. WAMPSERVER

WampServer is a Windows web development environment. It allows you to create web applications with Apache2, PHP and a MySQL database. Alongside, PhpMyAdmin allows you to manage easily your database.

WampServer is a reliable web development software program that lets you create web apps with MYSQL database and PHP Apache2. With an intuitive interface, the application features numerous functionalities and make sit the preferred choice of developers from around the world. The software is free to useand doesn’t require a payment or subscription.

CHAPTER 6

SYSTEM DESIGN

6.1. DATA FLOW DIAGRAM

6.1.1. Level 0

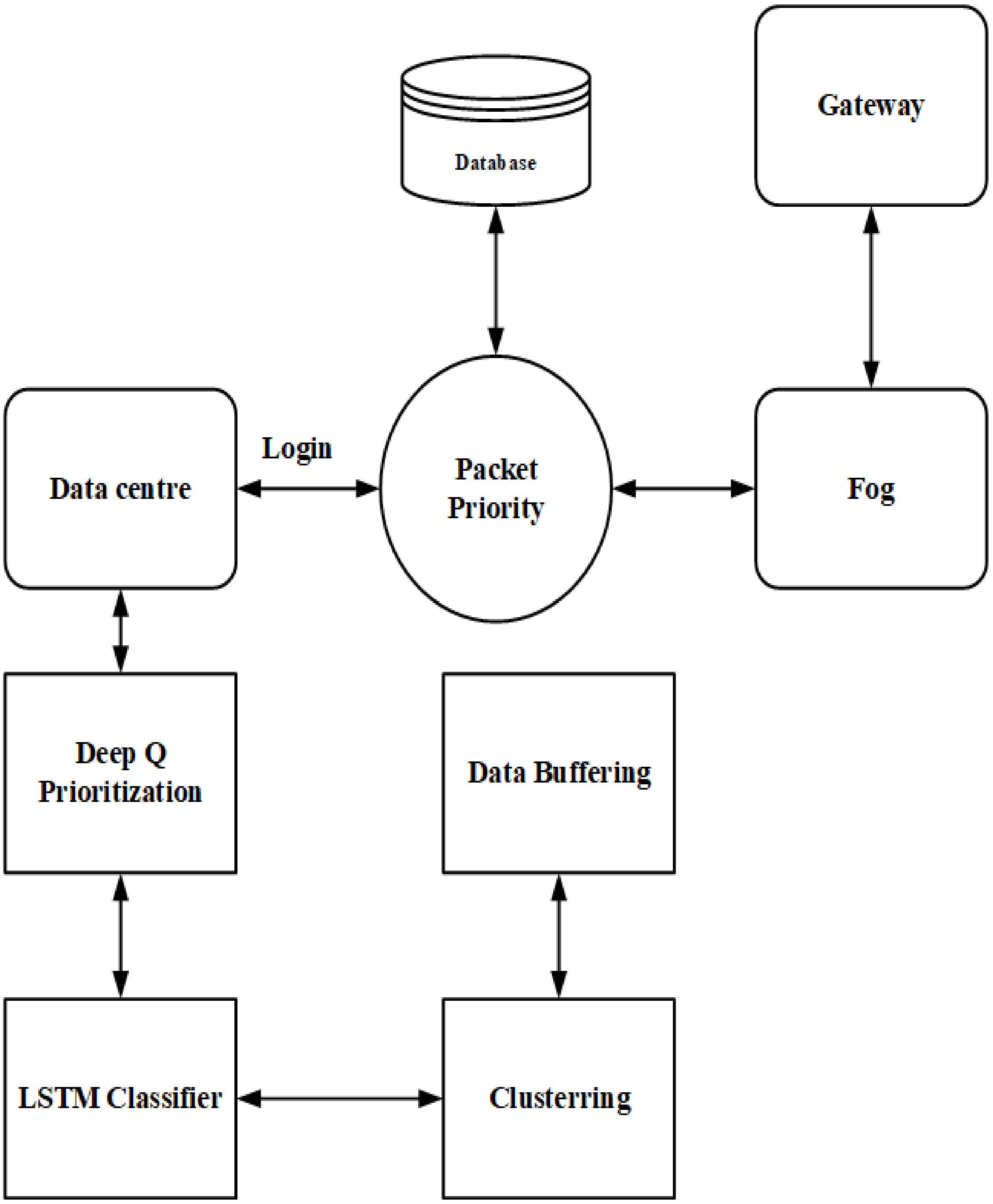


Figure6.1.Packet Priority Classification

6.1.2. Level 1

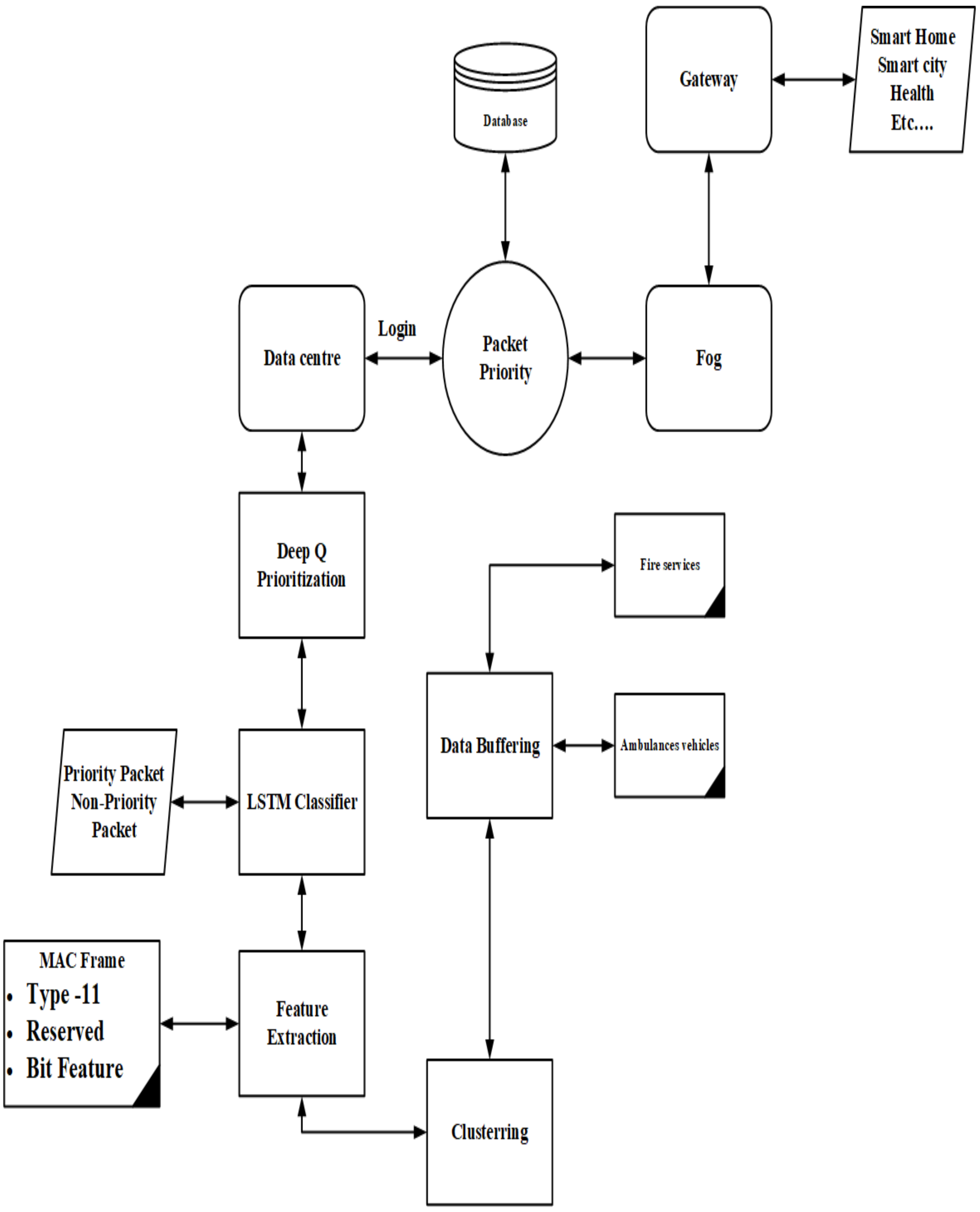


Figure6.2.Packet Priority Processing

6.1.3. Level 2

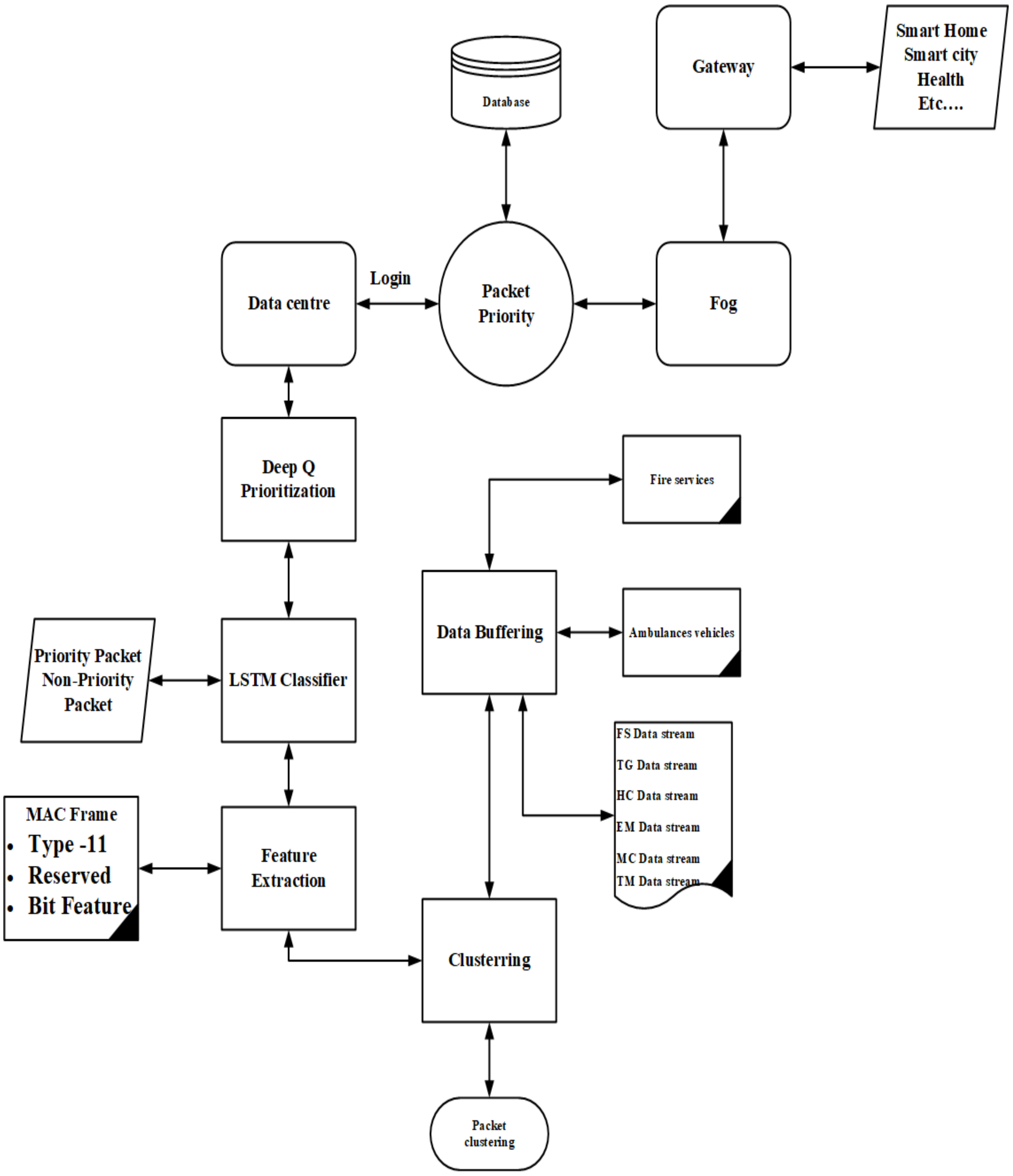


Figure 6.3.Packet Clustering

6.2. DATABASE DESIGN

DataBase : Packet Priority

6.2.1. Table Structure for Table Details

Field	Type	Key
Monitor_category	varchar(30)	Primary Key
Sensor_name	varchar(30)	

6.2.2. Table Structure for Table node_data

Field	Type	Key
id	int(11)	
monitor	varchar(30)	Foreign Key
sensor	varchar(30)	
Packet_energy	int(11)	
Rss	int(11)	
location	varchar(30)	
Deliver_to	int(11)	
Receive_seconds	int(11)	
Delay_seconds	int(11)	

6.2.3. Table Structure for Table Register

Field	Type	Null
id	int(11)	
monitor	varchar(30)	Foreign Key
sensor	varchar(30)	
Transfer_count	int(11)	

6.3. UML DIAGRAM FOR PACKETS CLASSIFICATION

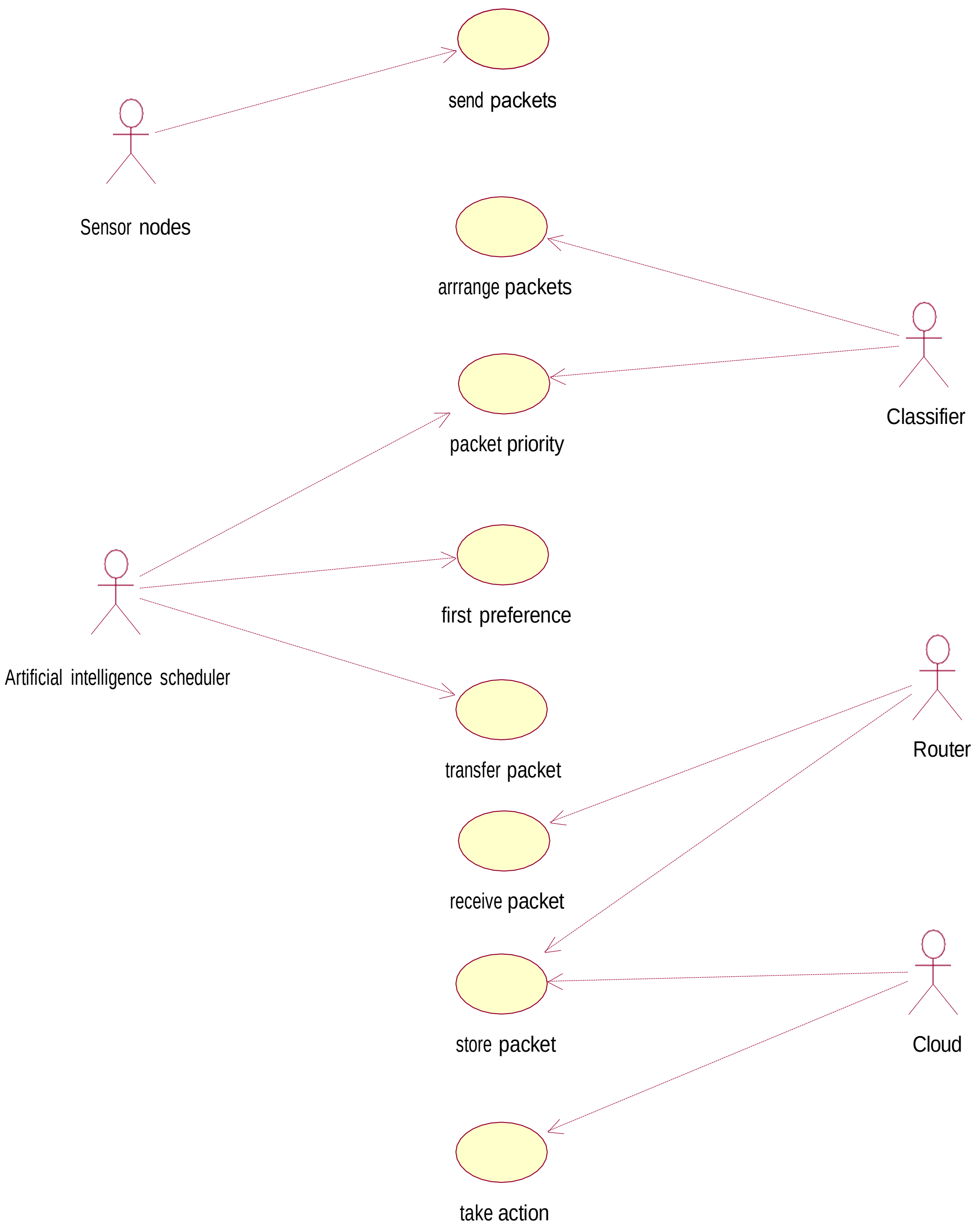


Figure6.4. Priority based packets classification

6.3.2. Class Diagram

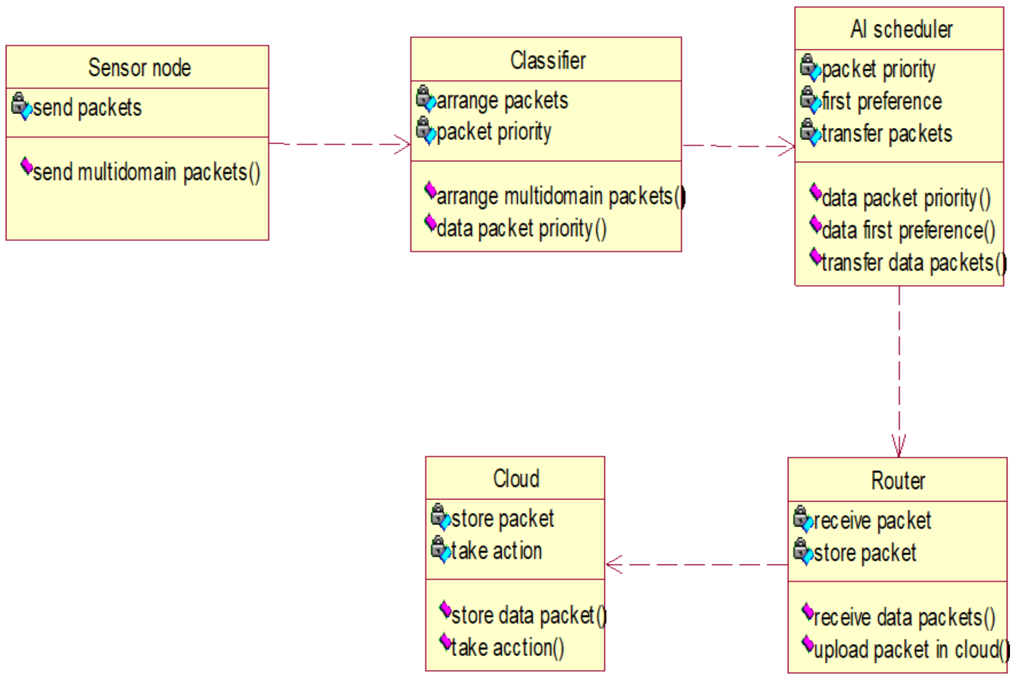


Figure6.5.Class Diagram for Packet Priority Classification

6.3.3. Activity Diagram

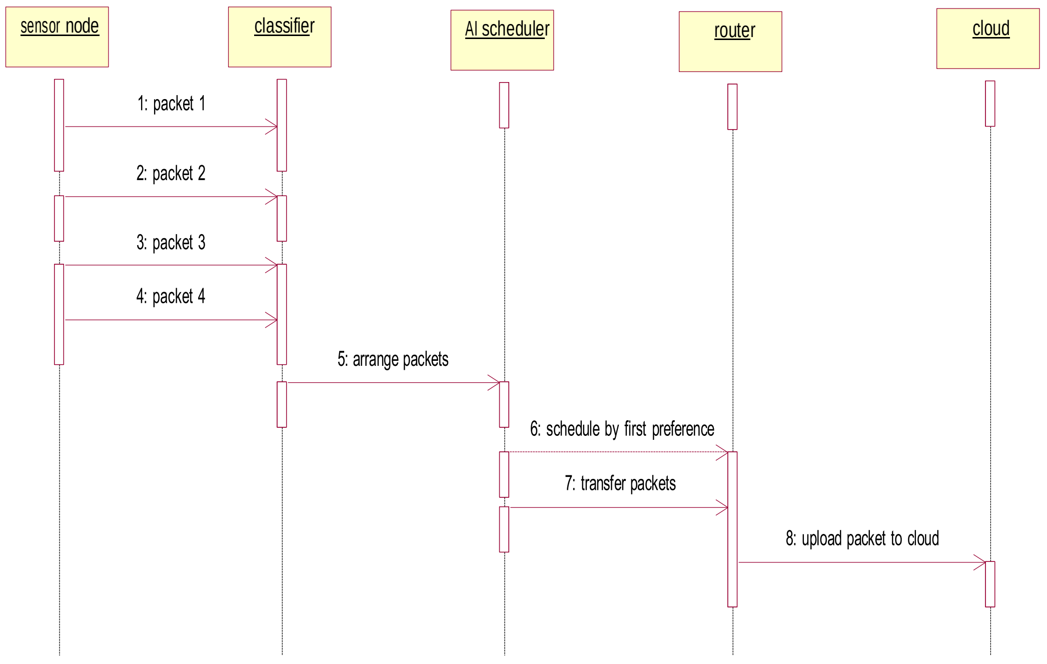


Figure6.6. Activity Diagram of Packet Priority Classification

6.5.4. Sequence Diagram

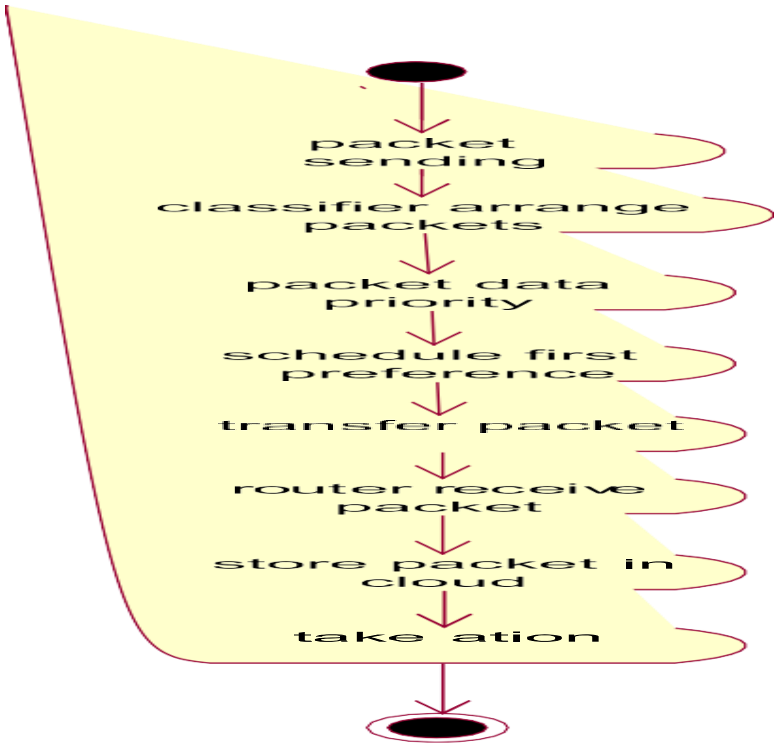


Figure 6.7. Sequence Diagram for Packet Priority Classification

6.5.5. Collaboration Diagram

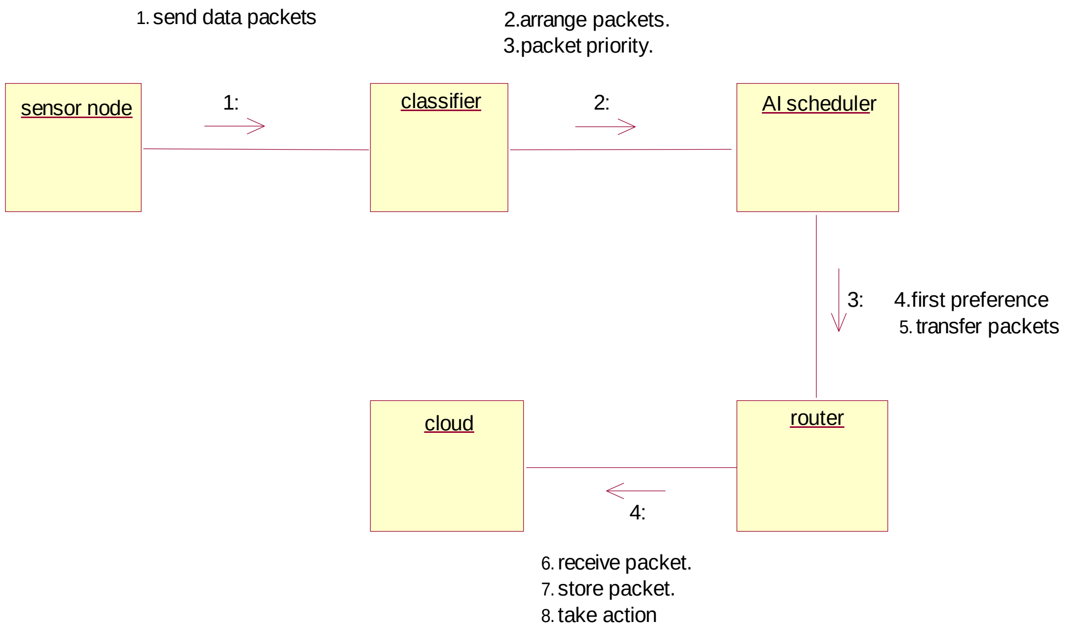


Figure 6.8.Collaboration Diagram for Packet Priority Classification

CHAPTER 7

SYSTEM TESTING

Software testing is an important part of any software development process, including the development of emergency packet classification using LSTM and packet scheduling using Deep Q Learning techniques.

7.1. TYPES OF SOFTWARE TESTING

1. **Unit testing**: Unit testing involves testing individual components of the system in isolation to ensure that they function correctly. For emergency packet classification using LSTM, unit testing can be used to test the LSTM model to ensure that it is correctly classifying packets as emergency or regular. For packet scheduling using Deep Q Learning, unit testing can be used to test the Deep Q Learning algorithm to ensure that it is making the correct decisions based on the reward signal.
2. **Integration testing**: Integration testing involves testing how different components of the system work together. For emergency packet classification using LSTM, integration testing can be used to test how the LSTM model interacts with other components of the system, such as the packet classifier or the packet prioritizer. For packet scheduling using Deep Q Learning, integration testing can be used to test how the Deep Q Learning algorithm interacts with other components of the system, such as the packet scheduler or the network traffic simulator.
3. **Functional testing**: Functional testing involves testing the system against the functional requirements. For emergency packet classification using LSTM, functional testing can be used to test that emergency packets are correctly classified and prioritized over regular packets. For packet scheduling using Deep Q Learning, functional testing can be used to test that packets are prioritized according to the reward signal provided by the network administrator.

4. **Performance testing**: Performance testing involves testing the system's performance under different loads and conditions. For emergency packet classification using LSTM, performance testing can be used to test the model's performance under high traffic conditions to ensure that it can handle the load and classify packets in a timely manner. For packet scheduling using Deep Q Learning, performance testing can be used to test the algorithm's performance under different network traffic conditions to ensure that it can make decisions in a timely manner and optimize network performance.

7.2. TEST CASES

Here are some example test cases for IoT Network Emergency packet classification using LSTM and Packet Scheduling using Deep Q Learning.

1.Unit testing for IoT Network Emergency packet classification using LSTM:

- Test that the LSTM model is able to correctly classify packets as emergency or regular for IoT devices.
- Test that the model can handle different types of IoT traffic (e.g., sensor data, control messages) and classify them correctly.
- Test that the model can handle different sizes of packets from IoT devices and classify them correctly.
- Test that the model can handle high traffic conditions from IoT devices and classify packets in a timely manner.

2.Integration testing for IoT Network Emergency packet Classification using LSTM:

- Test that the LSTM model integrates correctly with other components of the IoT network, such as the IoT device classifier or the packet prioritizer.
- Test that emergency packets from IoT devices are correctly prioritized over regular packets after classification.

3.Functional testing for IoT Network Emergency packet classification using LSTM:

- Test that emergency packets from IoT devices are correctly classified and prioritized over regular packets according to the requirements.
- Test that the system can handle different types of emergency packets from IoT devices (e.g., medical, security) and prioritize them correctly.
- Test that the system can handle different types of regular packets from IoT devices (e.g., sensor data, control messages) and prioritize them correctly.

4.Performance testing for IoT Network Emergency packet Classification using LSTM:

- Test the model's performance under different loads and conditions from IoT devices to ensure that it can handle the load and classify packets in a timely manner.
- Test the system's ability to handle high traffic conditions from IoT devices and prioritize emergency packets correctly.
- Test the system's ability to handle different types of IoT devices and prioritize packets according to their importance.
- For Packet Scheduling using Deep Q Learning in IoT networks, the test cases will be similar but with a focus on testing the Deep Q Learning algorithm's ability to prioritize packets from IoT devices according to the reward signal provided by the network administrator. The test casesshould cover unit testing, integration testing, functional testing, performance testing, and security testing. Additionally, the testing should account for the unique challenges of IoT networks, such as limited bandwidth, low-power devices, and variable network conditions.

7.3. TEST RESULT

In general, the test results for IoT Network Emergency packet classification using LSTM and Packet Scheduling using Deep Q Learning should show that the system can effectively classify packets from IoT devices as emergency or regular and prioritize them accordingly. The results should also demonstrate that the system can handle different types of IoT traffic and prioritize packets in a timely manner.

- **Unit testing:** The test results should show that the LSTM model is able to correctly classify packets as emergency or regular for IoT devices. The Deep Q Learning algorithm should also be able to prioritize packets based on the reward signal provided by the network administrator. The unit testing should also test the functionality of individual components and identify any issues that may arise.
- **Integration testing:** The test results should demonstrate that the LSTM model integrates correctly with other components of the IoT network, suchas the IoT device classifier and packet prioritizer. The emergency packets from IoT devices should be correctly prioritized over regular packets after classification. The integration testing should also ensure that the system can handle different types of IoT devices and classify and prioritize packets correctly.
- **Functional testing:** The test results should show that emergency packets from IoT devices are correctly classified and prioritized over regular packets according to the requirements. The system should be able to handle different types of emergency packets from IoT devices (e.g., medical, security) and prioritize them correctly. The system should also be able to handle different types of regular packets from IoT devices (e.g., sensor data, control messages) and prioritize them correctly.
- **Performance testing:** The test results should demonstrate that the system can handle high traffic conditions from IoT devices and prioritize packets in a timely manner. The system should be able to handle different types of IoT devices and prioritize packets according to their importance. The performance testing should also test the system's scalability and identify any bottlenecks that may arise under heavy loads.

Thus, the test results should demonstrate that the IoT Network

Emergency packet classification using LSTM and Packet Scheduling using Deep Q Learning system can effectively classify and prioritize packets from IoT devices in emergency situations while maintaining the security and reliability of the network.

7.4. TEST REPORT

- **Introduction:** This section should provide an overview of the test report, including the purpose of the tests, the scope of the testing, and the system being tested.
- **Test Environment:** This section should describe the testing environment, including the hardware and software used for testing, as well as any tools and technologies used to conduct the tests.
- **Test Strategy:** This section should describe the testing strategy, including the types of testing conducted (e.g., unit testing, integration testing, functional testing, performance testing, security testing), the testing techniques used (e.g., black-box testing, white-box testing), and the testing objectives.
- **Test Results:** This section should provide a detailed account of the test results for each type of testing conducted. The results should be presented in a clear and concise manner, and should include any issues identified, as well as any recommendations for resolving them.
- **Conclusion:** This section should provide a summary of the test results, highlighting the strengths and weaknesses of the system being tested. It should also provide any recommendations for improving the system, based on the test results.

Thus the test report for IoT Network Emergency packet classification using LSTM and Packet Scheduling using Deep Q Learning should provide a comprehensive account of the testing process, including the results of each type of testing conducted, as well as any issues identified and recommendations for resolving them.

CHAPTER 8

SYSTEM IMPLEMENTATION

8.1. SYSTEM ARCHITECTURE

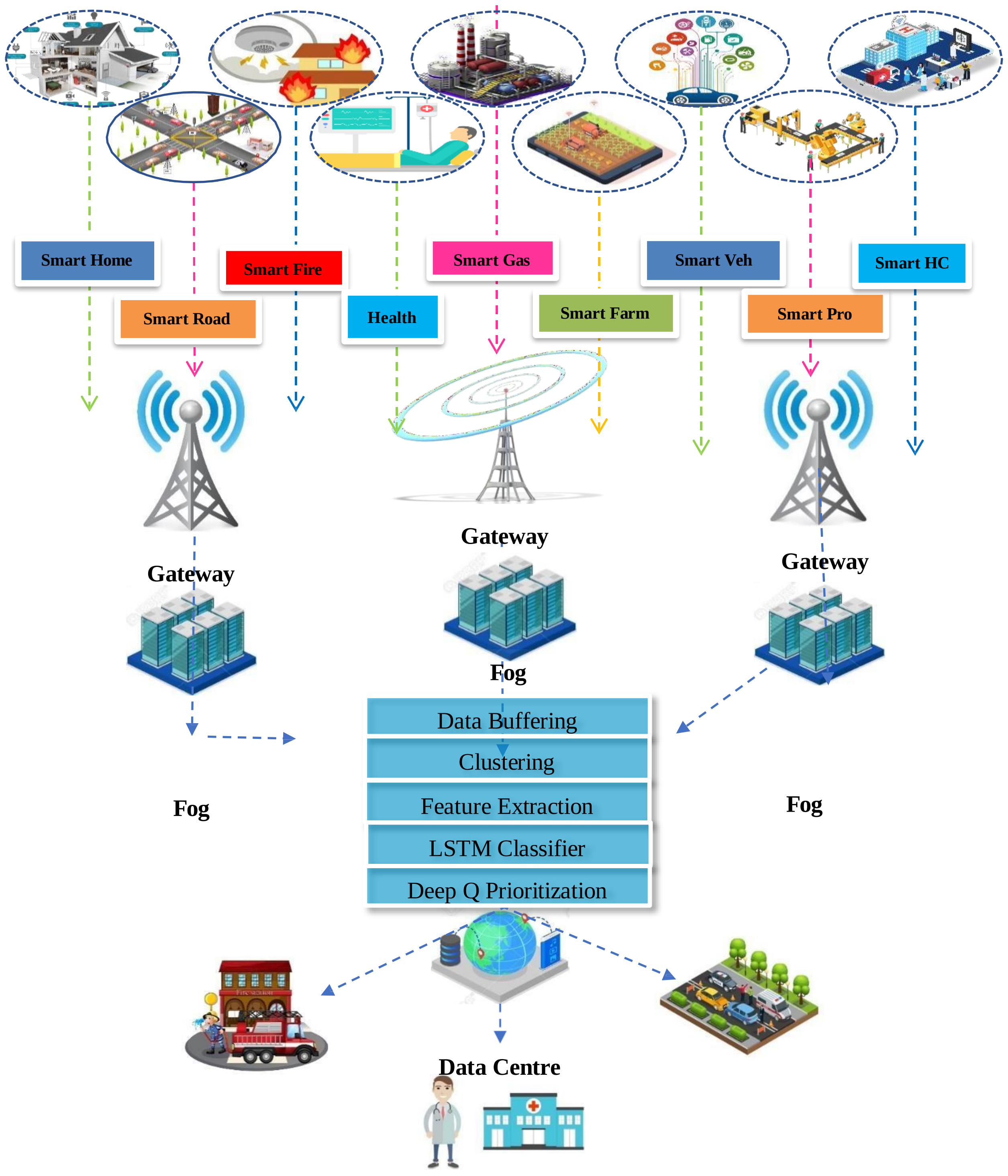


Figure8.1.Emerging and Non-Emerging Classification

8.2. EMERGENCY PACKET CLASSIFICATION

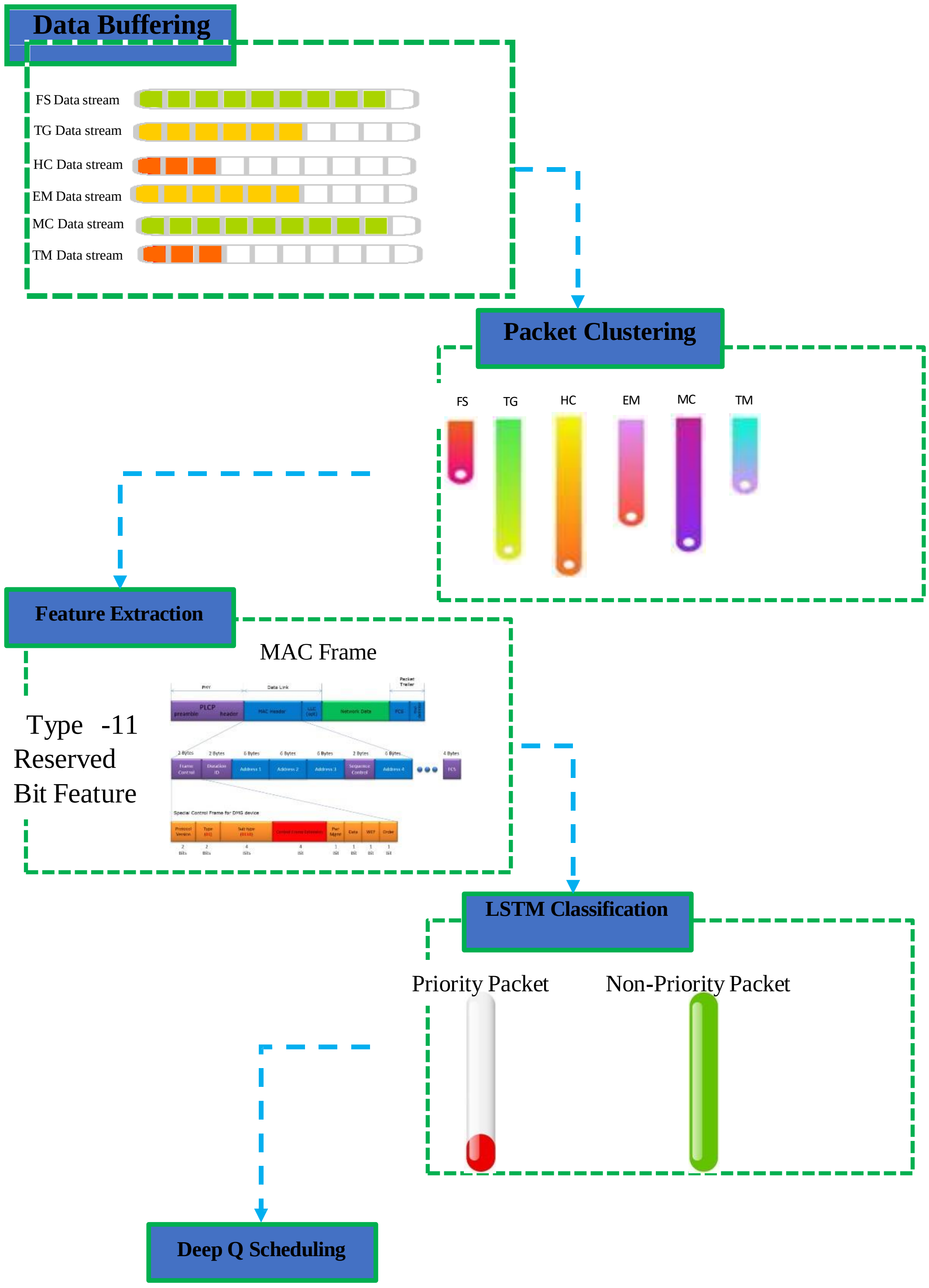


Figure8.2.Emergency Queuing

8.3. PROJECT DESCRIPTION

IoT devices (such as probe vehicles, smart phones, and UAVs) continuously collect data at a specific sampling interval. Part of the collected data is sent to an edge server for prediction. To decide whether a data should be PP or NPP, the controller fetches the importance of the block where the IoT device is currently located. The controller orders the transmitter to send the data or not in priority based on the importance of the current block received from the fog server.The fog server receives collected data from IoT devices, aggregates and pre- process the data, and predicts the real-time spatial information of the next time slot using LSTM. The aggregator and pre-processor on the fog server receive data from the transmitters of several IoT devices, pre-process the data, and complete the missing parts of the data. The predictor predicts real-time spatial information.The importance extractor extracts the importance of blocks from the pre-trained LSTM model for prediction. The responder sends the importance of blocks to each IoT device. In each time slot, Deep Q Learning decide whether to transmit the data observed in the time slot based on the importance of the data. The importanceof data corresponds to the importance of the block where the data was observed.

8.4. DATA PACKET DESCRIPTION

For test purposes, we chose to insert the identifier within the MAC layer Any packet data has two major parts, the header and the information. We are focusing on the packet headers. All packet headers contain the same features in the IEEE 802.15.4 standard. The headers are described in the MAC frame format(refer to Table 1). This format consists of the reserved bits that can be used for any specific application.

We are using this specific reserved bit to save an identifier at the sensor level. Anidentifier is only two bits which means it will not take a lot of memory space.

Hence, the identifier is included in the header part rather than taking up space inthe memory of the application defined information.

- **Type:**

It is a 2-bit long field which determines the function of frame i.e management (00), control (01) or data (10). The value 11 is reserved.

8.5. MODULES DESCRIPTION

8.5.1. IoT Network Simulator

In this module, design a hierarchical structure to generalize all the contents in the IoT network. There are three layers in this structure: the device layer, the fog intelligence layer, and the centralized intelligence layer. The device layer is composed of all objectives, IoT Device, users, and smart terminals that can collect sensed data from live environments. The fog intelligence layer provides distributed, low-latency, and limited computing resources between the device layer and the higher layer. The centralized intelligence layer consists of cloud data centers that aggregate data from lower layers.

8.5.2. Fog Application and Service Layer

This layer applies services for IoT data analytics and periodically processes sensor data to make IoT based decisions using Publish/Subscribe Machines (PSMs). The output of this layer is the data packets containing decisions. In a scalable fog architecture, multiple PSMs may perform simultaneous operations and generate a low-rate periodic IoT decision traffic pattern independently Context Broker (CB), PAS - A Publish/Subscribe Machine (PSM), Fog Connector (FC).

8.5.3. Priority Aware Scheduling

The purpose of this module is to recognize the received information packets and process them according to their priority.

8.5.3.1. LSTM Packet Priority Classification

The LSTM based deep learning model is trained on the fog server in advance before the first time slot begins. The proposed system considers the prediction of future real-time spatial information as a regression task. Regression,in general, is a type of task that estimates a numerical value given some input. To solve the task, the learning algorithm is asked to learn a function that maps an input variable to an output variable.

In the proposed system, the LSTM model for prediction receives the aggregated past sensor data collected from IoT devices in each block as an inputand calculates the future real-time spatial information as output. The deep learning model for prediction receives an input variable X, which consists of aggregated sensing data collected in the last few time slots, and predict output variable y, which is the real-time spatial information of each block in the next time slot. The input variable X of the machine learning model for prediction is a T × NB matrix, where T is the number of time slots used in one prediction and NBis the number of blocks. The output y of the machine learning model for prediction is a vector of NB elements.Each element of the output vector represents the real- time spatial information ofeach block in the next time slot.

Calculation of Importance of Blocks

The importance of a block is calculated from the pre-trained machine learning model for prediction on the edge server using a feature selection method. The importance of block I_b is defined as $I_b = \sum_{t=1}^{T} F_{t,b}$, where $F_{t,b}$ is the feature importance of the (t, b) element of the input matrix. $F_{t,b}$ is calculated from the pre-trained model using the feature selection method.

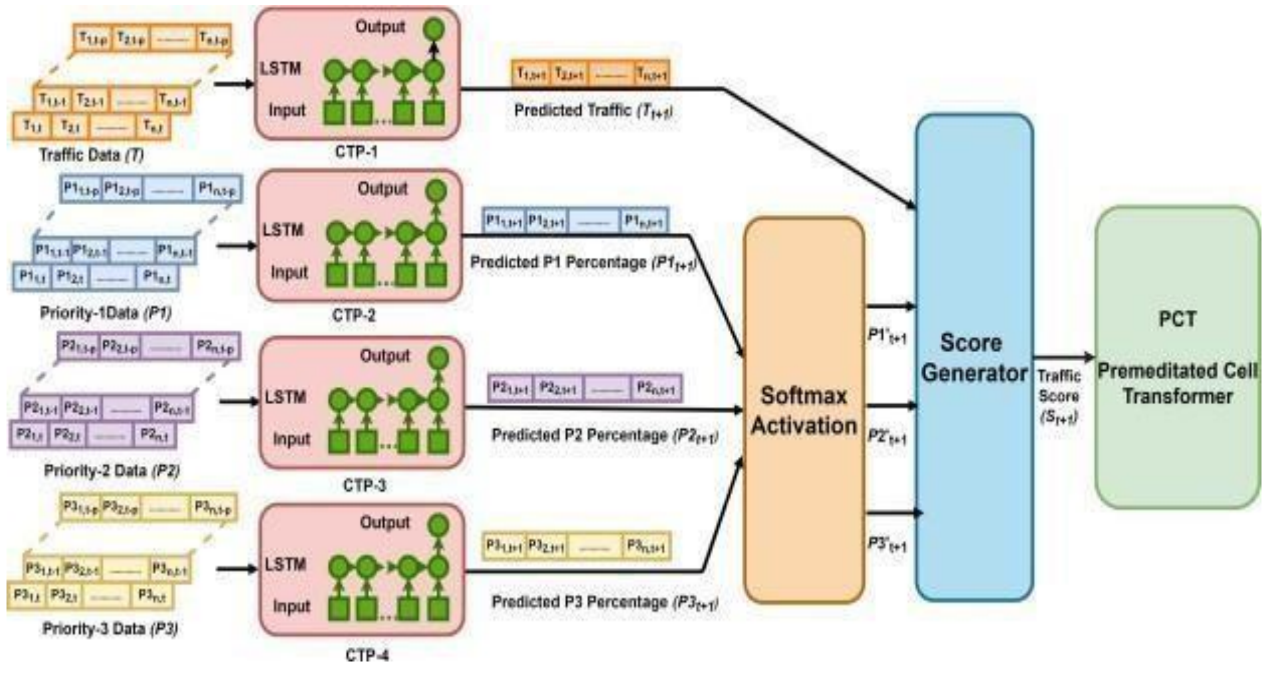


Figure8.3.LSTM Packet Priority Classification

Data Buffering - Then PAS will receive the MAC address packet from the sender node.

Clustering – Received Packet are clustered according to their applications.

Feature Extraction - The incoming packets not only have data packets, but also have emergency information packets and MAC-address packets. Extractedfeatures of Type 11 – reserved bit feature can distinguish the incoming packets from MAC frame.

Classification - Based on the extracted type 11 in the data packet header mac frame the LSTM classifier classified as Emergency and Non-Emergency.

In this phase, the data packets are processed, scheduled, and forwarded based on their emergency information. The data packets are divided into three types according to their priorities and deadlines.

1) **Emergency data packets (PP)** - This type of packets needs the quickest response. Thus, these packets' end-to-end delays must be reduced as much as possible.

2) **Normal data packets (NP)** - The most of data packets in network are of this type. The emergency data packets can preempt this type of data packets.

3) **Nonemergency data packets (NPP)** - They have the lowest requirement on delay. Compared with other data packets. Their deadlines are longer.

8.5.3.2. Deep Q Learning – Packet Scheduling

To allocate channels and time slots efficiently, we use the deep Q network (DQN) architecture with experience replay and a greedy policy to solve the reinforcement learning problem. This architecture not only yields more accurate value estimations but also leads to much higher learning stability.

Queuing Network Model - Based on the traffic mix and flows, a queuing network model is developed for the ingress and egress queues. The queuing model is used to study the effects of changing multiple parameters, such as queuing discipline, drop rates, and priority.

RL Agent - The effect of changes on queue configurations are fed into the RL model for training. It has actions spanning multiple queue configuration parameters.

Observations on throughput, latency, and packet drop changes are used to provide rewards to the agent.

Configuration Deployment - The policy generated by the RL agent is then deployed on the router port. This can be directly run on the router operating system or be configured via SDN controllers. Based on the observed traffic mix, the agent's policy is deployed for ingress traffic policing and egress traffic shaping.

State- It is composed of the source node, destination node, number of transmission time slots, and time slot occupancy.

Action- In the approach, the agent determines which channel and time slot combination is available to assign to the current network state.

Reward - The reward is the objective of the algorithm. The agent relies on rewards to evaluate the effectiveness of the action and further improve the policies.

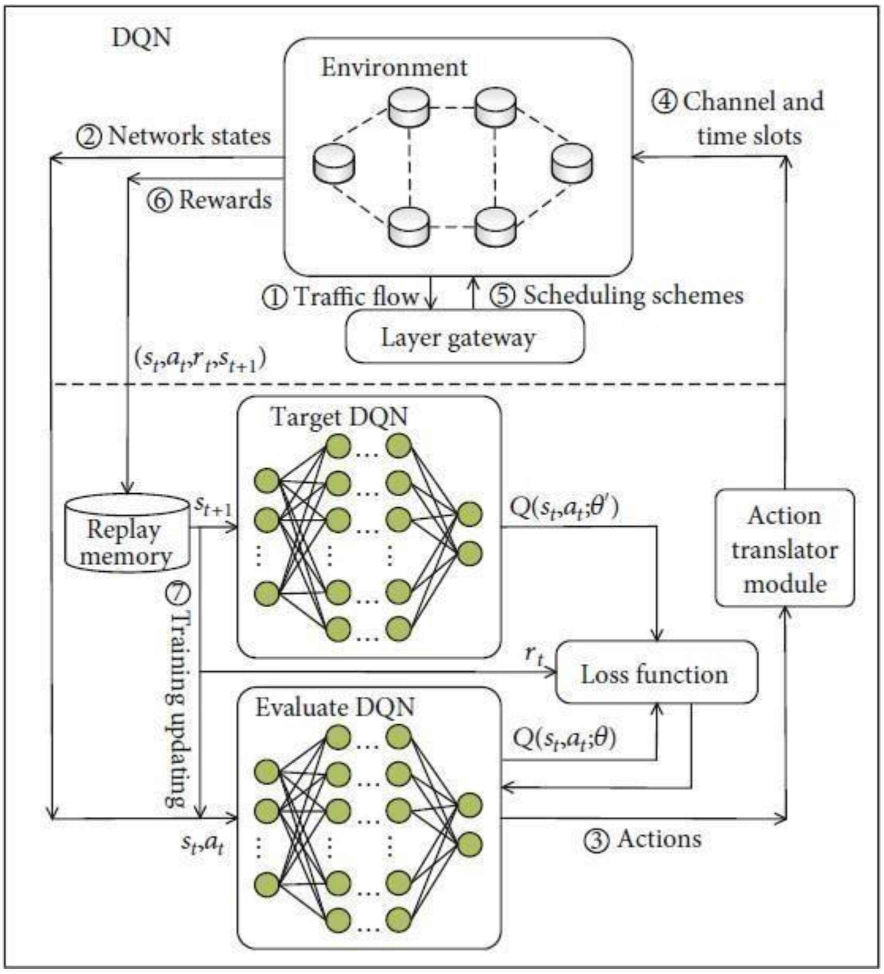


Figure8.4.Deep Q Learning-Packet Sheduling

8.5.4. Packet Forward

Packet entering the router port is assigned an internal priority level and internal drop precedence. This is determined based on a class map linked to the packet header information. As the packet enters the router, it is subjected to a classification filter. Packets belonging to each class can be rate-limited or marked. The per - class traffic can be treated as follows:

a) The per-class rate limits may be set. By default, conforming traffic is marked green; exceeding traffic yellow; and violating traffic red. Violating traffic can be marked red (if the violate red command is configured) or dropped immediately (if the violate drop command is configured).

b) Packets that are not dropped because of rate limiting can have their drop precedence values modified. The router transports the packet to the egress port. Egress scheduling: Each outgoing packet is assigned to an egress queue basedon the destination information from the forwarding table.

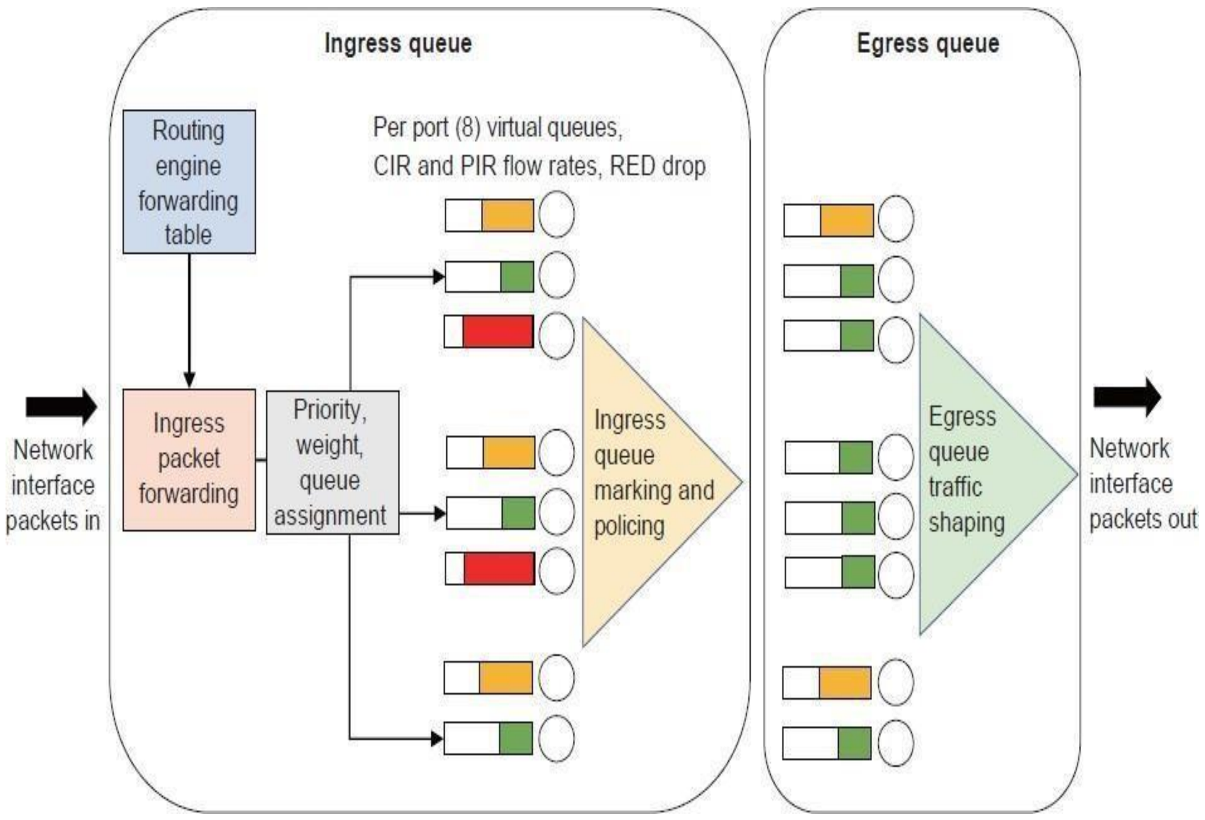

Ingress queue
Egress queue
Routing engine forwarding table
Per port (8) virtual queues, CIR and PIR flow rates, RED drop
Network interface packets in
Ingress packet forwarding
Priority, weight, queue assignment
Ingress queue marking and policing
Egress queue traffic shaping
Network interface packets out

Figure8.5.Packet Forward

8.5.4.1. Control Plane

This plane maintains a routing table that lists the routes to be taken by a data packet. This may be statically defined or learned based on the traffic mix. The control plane builds the Forwarding Information Base (FIB) that is fed to the forwarding plane.

8.5.4.2. Forwarding Plane

The router forwards data packets between incoming and outgoing port connections. Using information from the packet header, the accurate FIB is used to map the outgoing packet. Both the switch and router ports have ingress (inbound) queues and egress (outbound) queues. In this packets scheduling scheme, there are three units: access control unit (ACU), emergency-aware unit (EAU), and packet forward unit (PFU). The incoming packets not only have data packets, but also have emergency information packets and MAC-address packets. The packet analysis (PA) can distinguish the incoming packets. Through the analysis of PA, the data packets are sent to conditional access control to check whether the deadlines expire, the emergency information packets are sent to EAU. Then, the packets within their deadlines are placed into priority queue. Corresponding to three different priorities, each node has three priority queues.

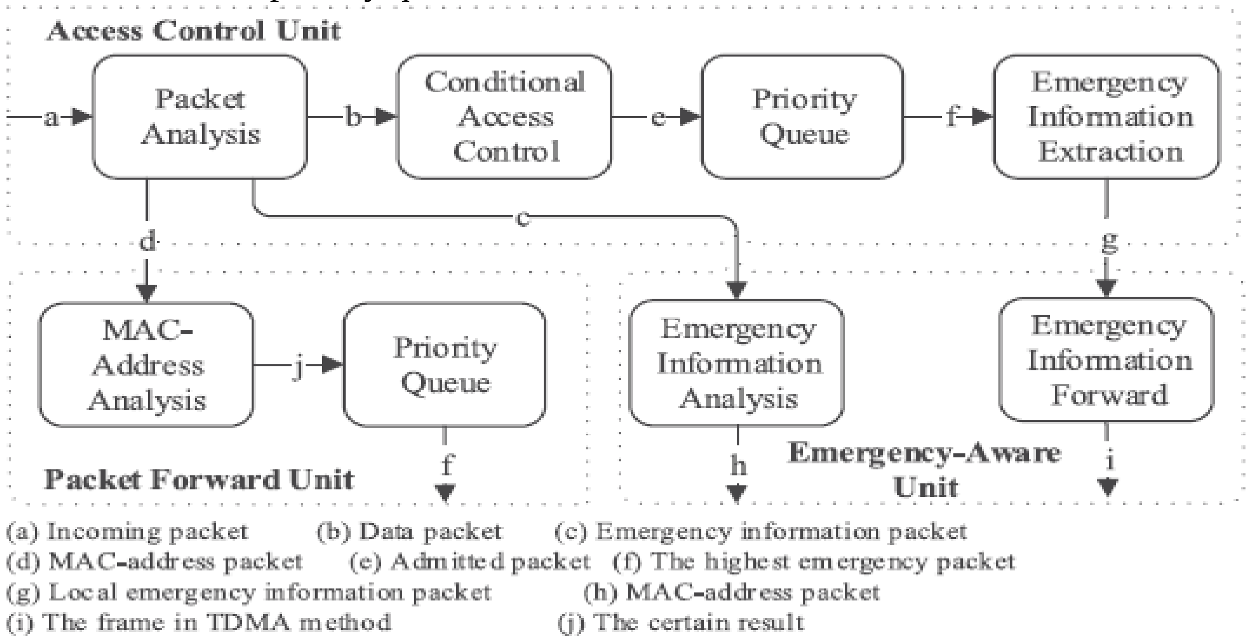
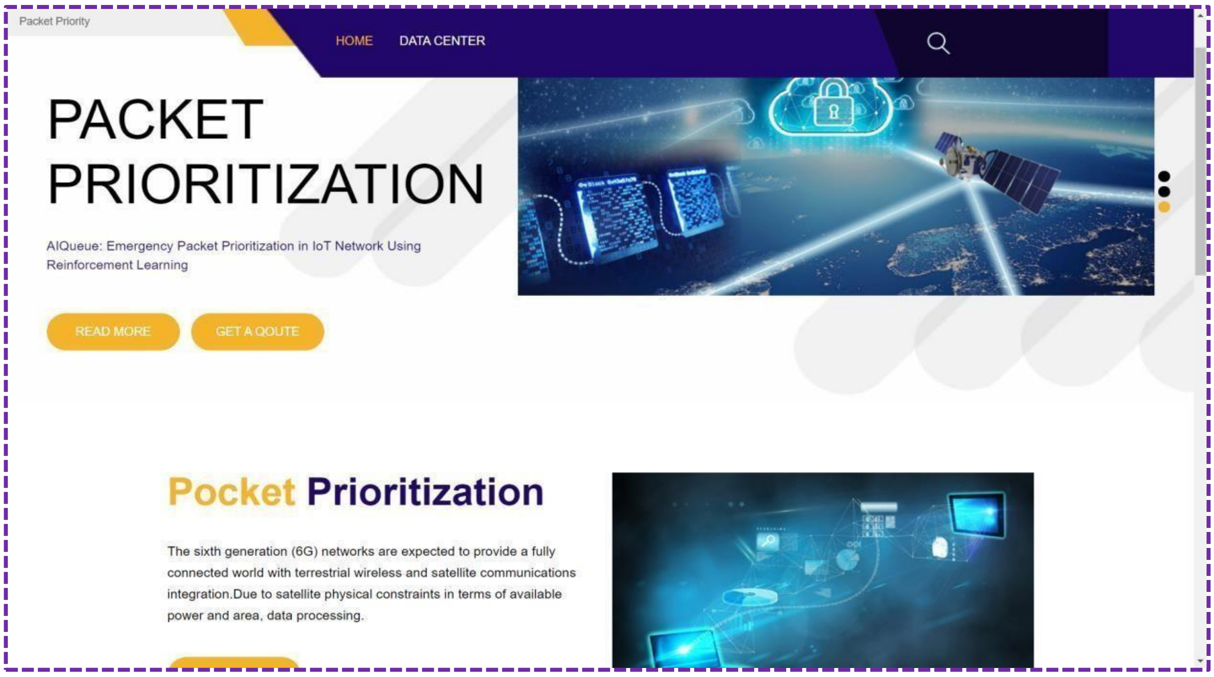


Figure8.6.Forwarding Plane

CHAPTER 9

APPENDICES

9.1. OUTPUT SCREENSHOTS



The sixth generation (6G) networks are expected to provide a fully connected world with terrestrial wireless and satellite communications integration.Due to satellite physical constraints in terms of available power and area, data processing.

Figure9.1.Home Page for WampServer

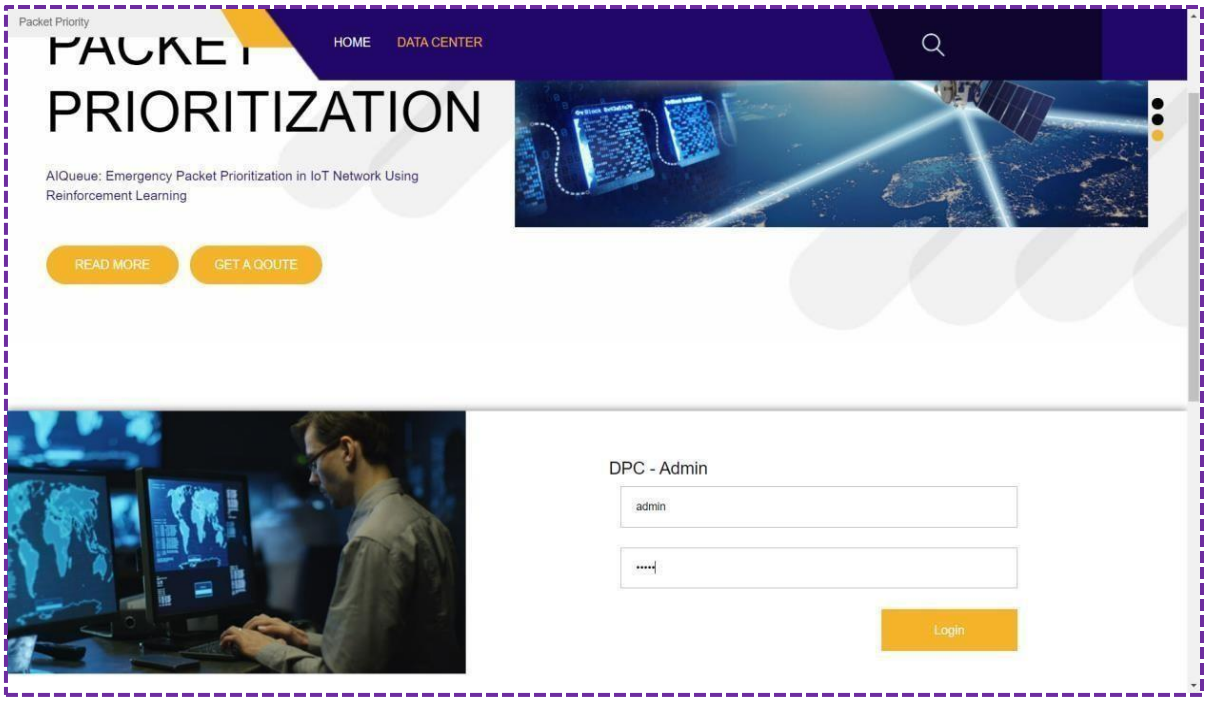
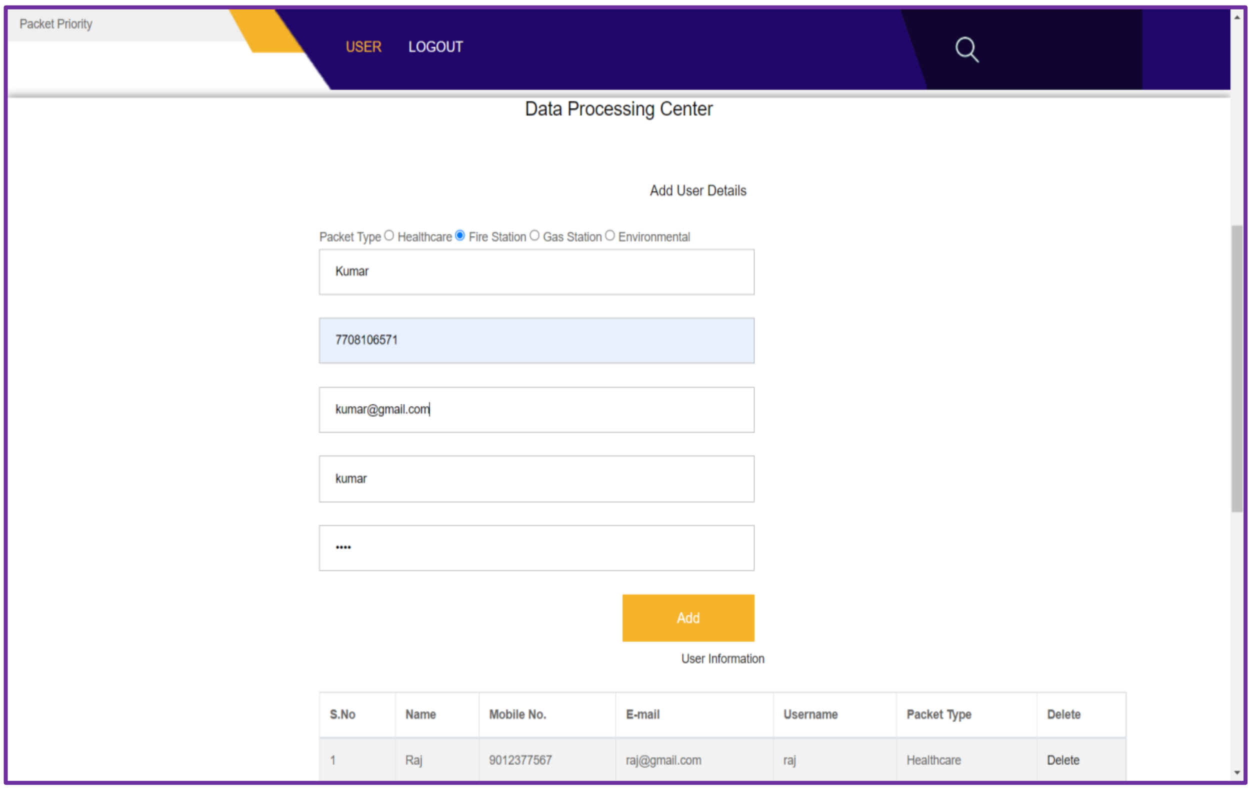


Figure9.2.Admin Page



Data Processing Center

Add User Details

Packet Type ○ Healthcare [✓ Fire Station] ○ Gas Station ○ Environmental

Kumar

7708106571

kumar@gmail.com

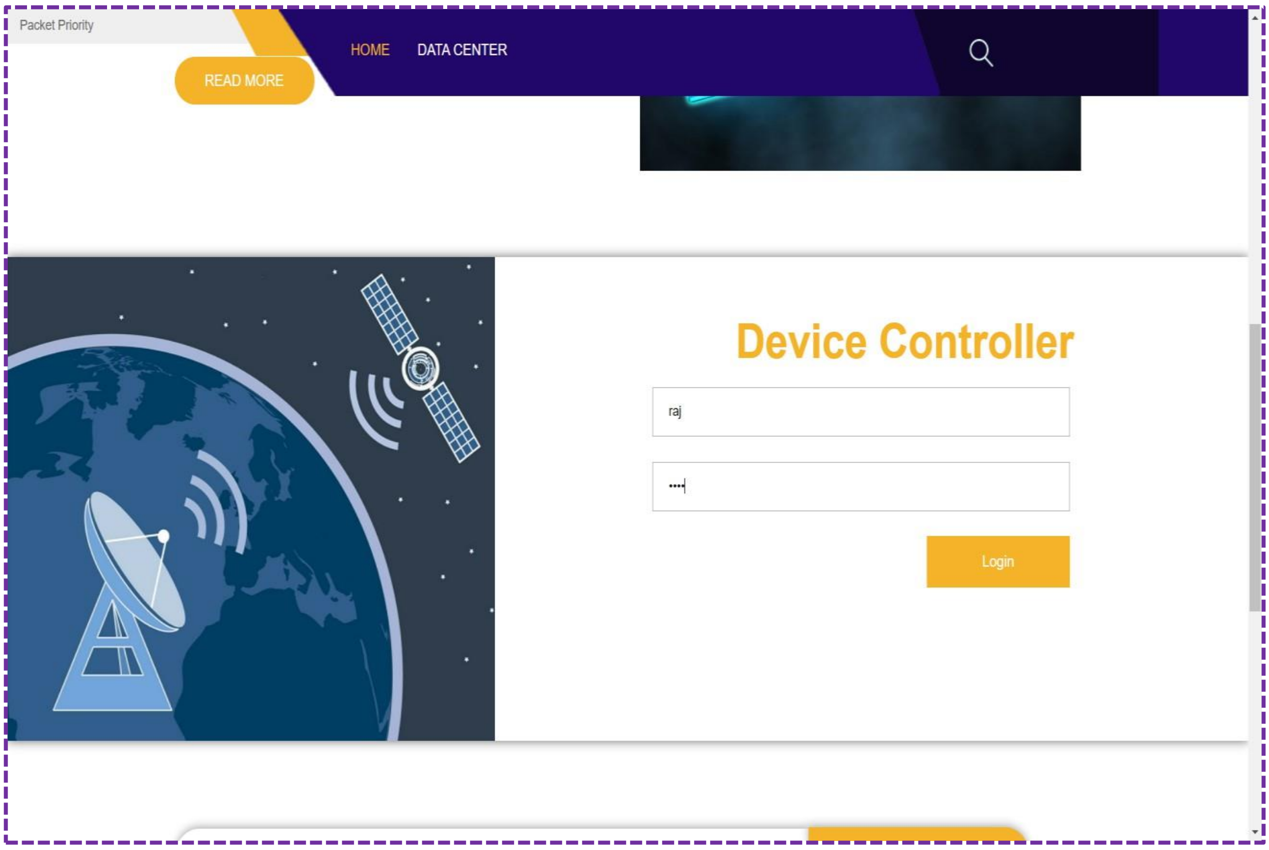
kumar

••••

User Information

S.No	Name	Mobile No.	E-mail	Username	Packet Type	Delete
1	Raj	9012377567	raj@gmail.com	raj	Healthcare	Delete

Figure9.3.User Informtion



raj

••••

Figure9.4.Login for Device Controller

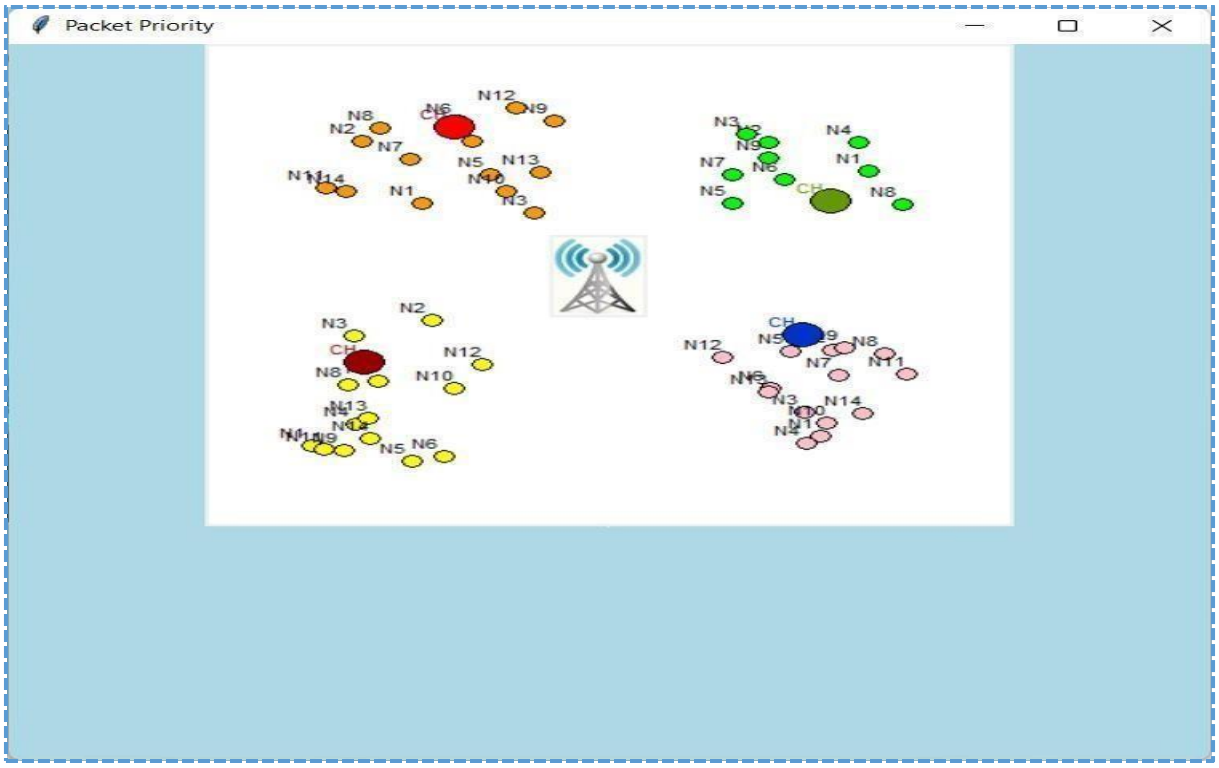
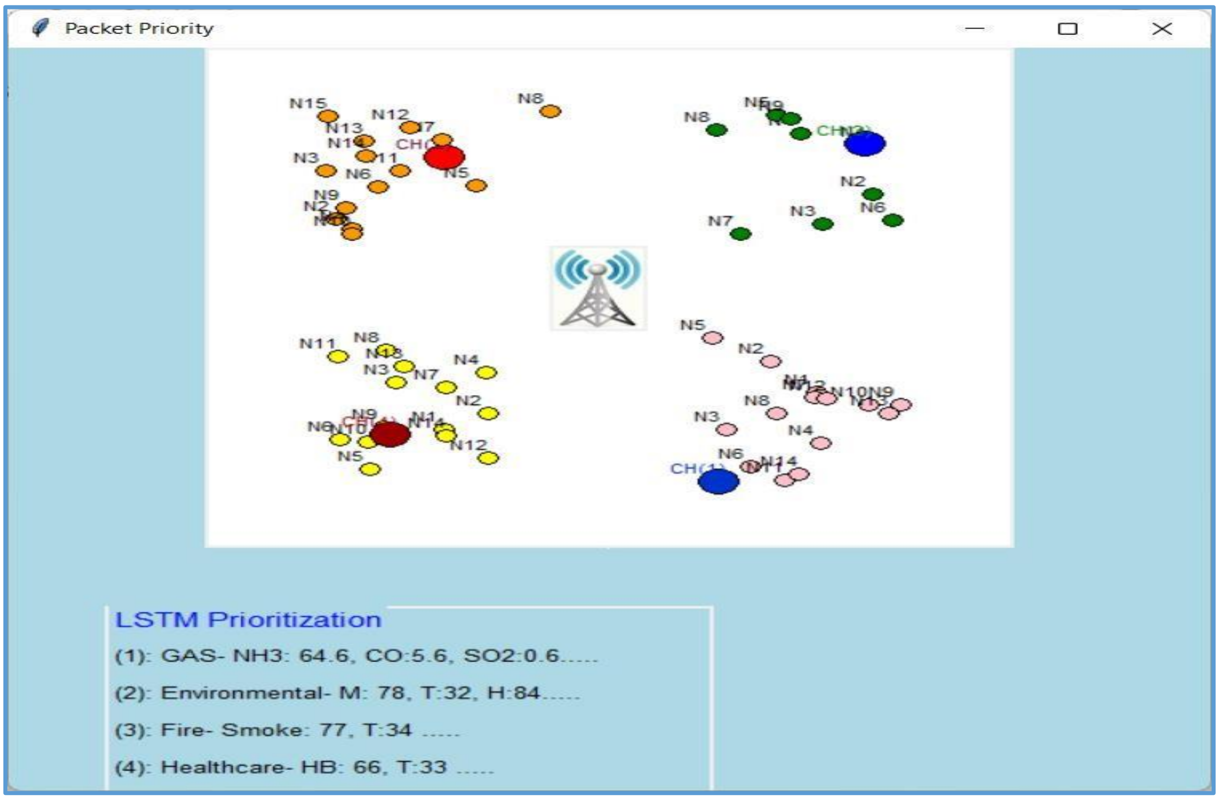


Figure9.5.Analyzing the Packets



LSTM Prioritization

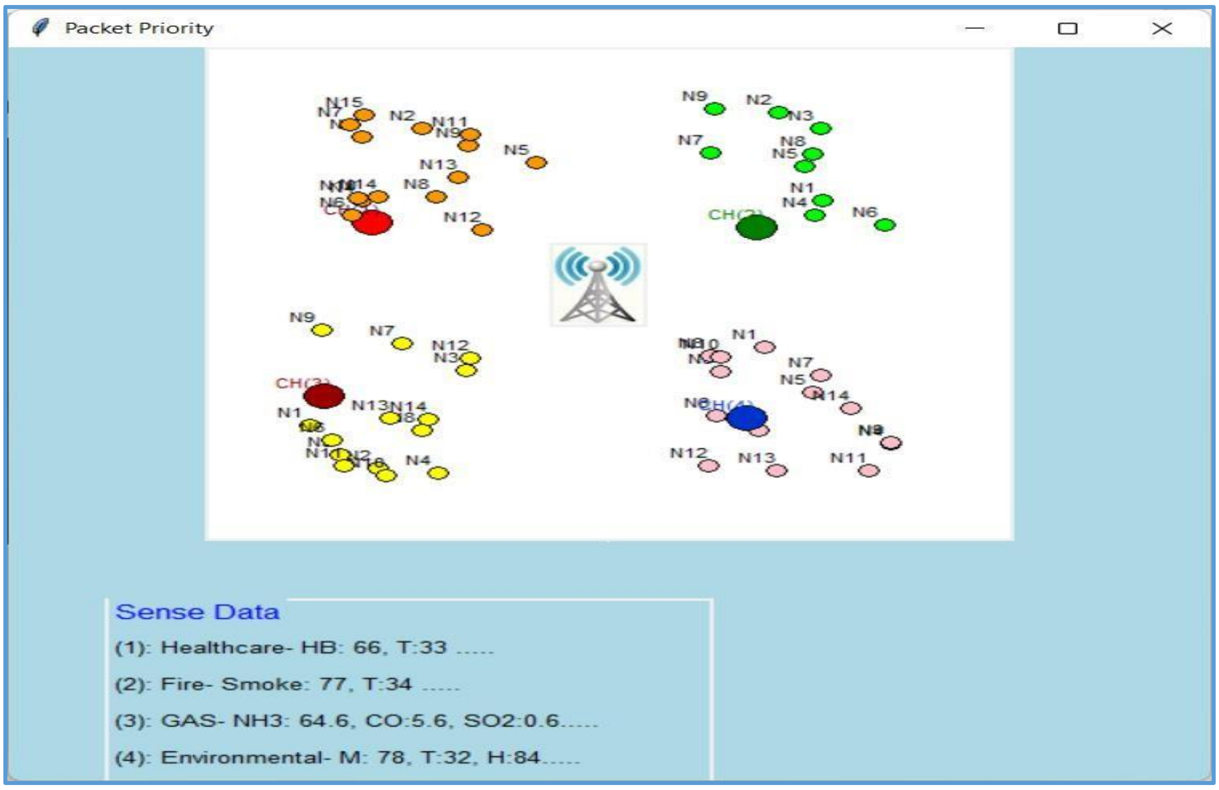
(1): GAS- NH3: 64.6, CO:5.6, SO2:0.6.....

(2): Environmental- M: 78, T:32, H:84.....

(3): Fire- Smoke: 77, T:34 .....

(4): Healthcare- HB: 66, T:33 .....

Figure9.6.Analysed Packets for LSTM



Sense Data

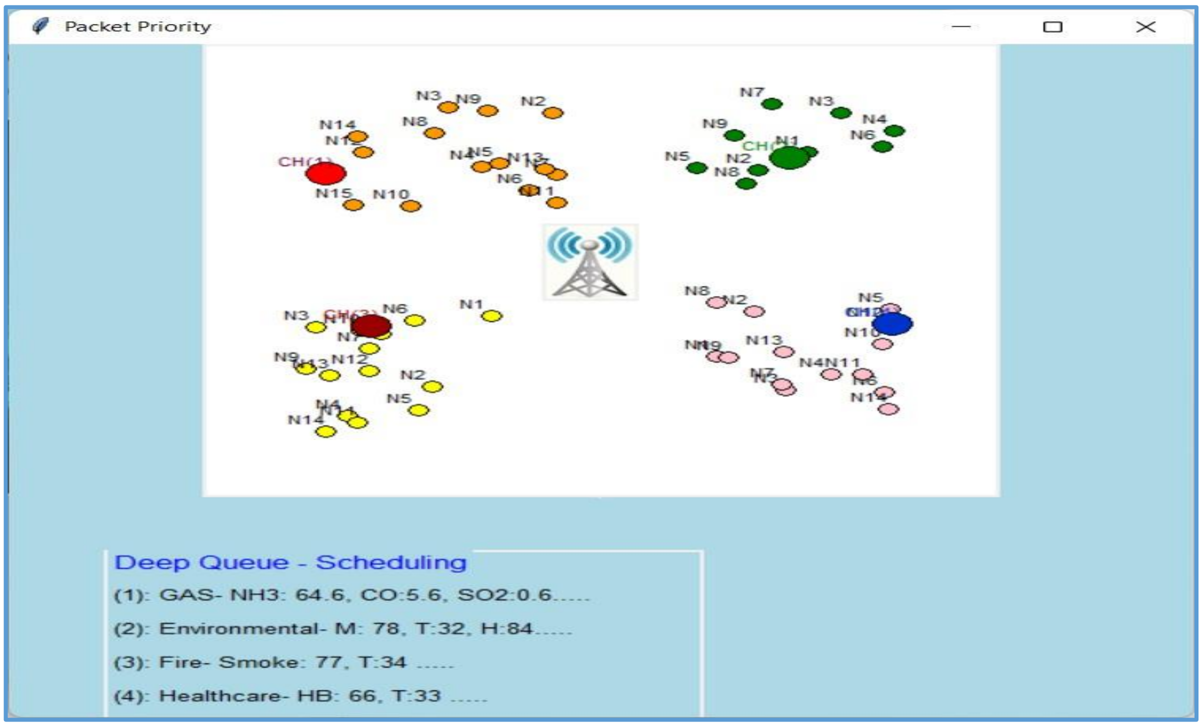
(1): Healthcare- HB: 66, T:33 .....

(2): Fire- Smoke: 77, T:34 .....

(3): GAS- NH3: 64.6, CO:5.6, SO2:0.6.....

(4): Environmental- M: 78, T:32, H:84.....

Figure9.7.Analysed Packets for Sense Data



Deep Queue - Scheduling

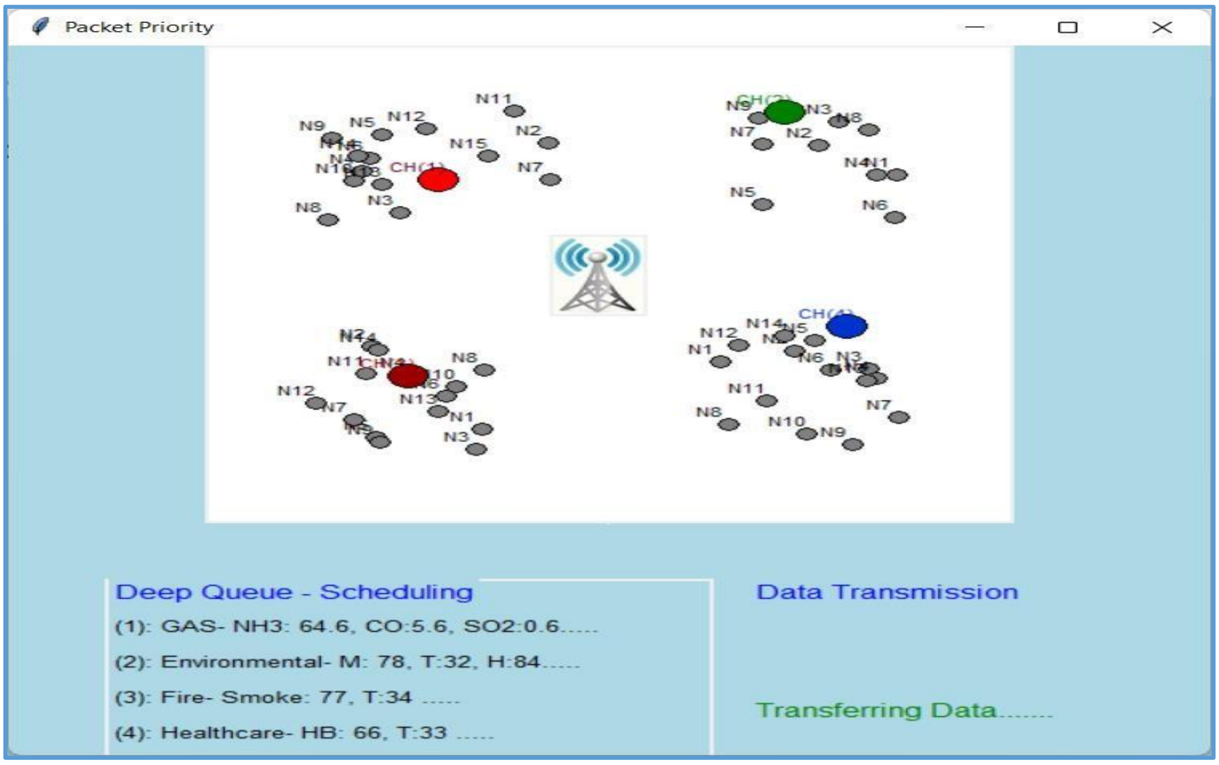
(1): GAS- NH3: 64.6, CO:5.6, SO2:0.6.....

(2): Environmental- M: 78, T:32, H:84.....

(3): Fire- Smoke: 77, T:34 .....

(4): Healthcare- HB: 66, T:33 .....

Figure9.8.Analysed Packets for Deep Queue



(1): GAS- NH3: 64.6, CO:5.6, SO2:0.6.....

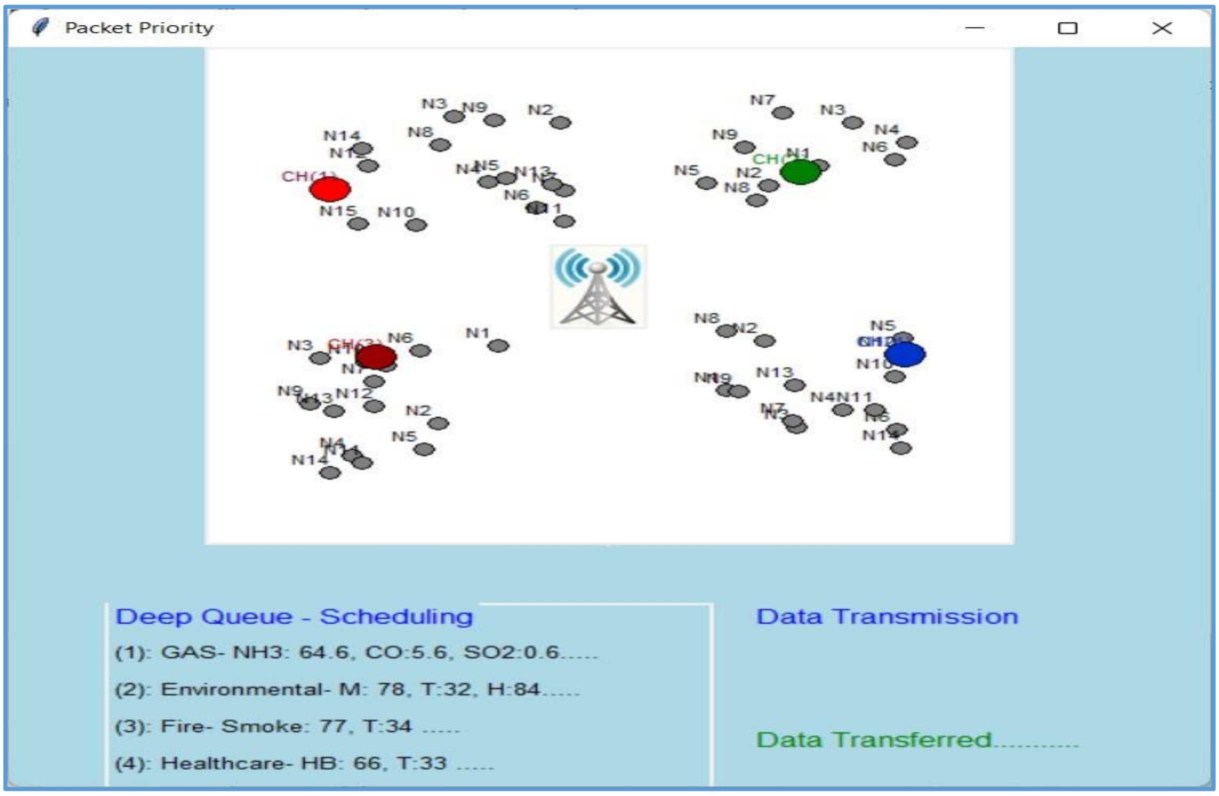
(2): Environmental- M: 78, T:32, H:84.....

(3): Fire- Smoke: 77, T:34 .....

(4): Healthcare- HB: 66, T:33 .....

Transferring Data.......

Figure9.9.Emergency and Non-Emergency Packets are Queuing



(1): GAS- NH3: 64.6, CO:5.6, SO2:0.6.....

(2): Environmental- M: 78, T:32, H:84.....

(3): Fire- Smoke: 77, T:34 .....

(4): Healthcare- HB: 66, T:33 .....

Data Transferred...........

Figure9.10.Emergency Packets are Detected

9.2. SOURCECODE

Import Packages

```
import PIL.Image, PIL.ImageTk
import random
import time
import threading
from PIL import Image from PIL import ImageTk
class PacketPriority:
def_init (self, master):
self.master = master
self.frame = Frame(self.master)
self.a1 = Label(self.master, text='Packet Priority Queuing Model an Emergency Packet
Scheduling',bg='lightblue', fg='black', font=("Helvetica", 16))
self.a1.pack() self.a1.place(x=50, y=30)
self.a1 = Label(self.master, text=' in IoT Network Using ReinforcementLearning',bg='lightblue',
fg='black', font=("Helvetica", 16))
self.a1.pack() self.a1.place(x=100, y=60)
mm = PIL.Image.open("dttr.jpg")
img2 = PIL.ImageTk.PhotoImage(mm)
panel2 = Label(self.master, image = img2)
panel2.image = img2 # keep a reference!
panel2.pack()
panel2.place(x=100,y=100)
self.b2 = Button(self.master, text=" << Packet Information >> ",command=self.simulate)
self.b2.pack() self.b2.place(x=260, y=450)
```

Sense Data

```
class Sense_data():
def_init (self, master):
self.master = master
self.frame = Frame(self.master)
self.a10 = Label(self.master, text='Packet Priority Queuing Model',bg='lightblue',fg='blue',
font=("Helvetica", 16))
```

```
self.a10.pack() self.a10.place(x=150, y=10)
self.a11 = Label(self.master, text='Monitor',bg='lightblue', fg='blue',font=("Helvetica", 12))
self.a11.pack() self.a11.place(x=130, y=90)
for cs in cc:
i=1
j=0
number=cs
self.n=number
f=cs
#print("f="+str(f) +"k="+str(k))while i <= number
if k==1:
#f=cs
xr = random.randint(60,180)yr = random.randint(50,150)
elif k==2:
#f=cs
xr = random.randint(250,350)yr = random.randint(50,150)
elif k==3:
#f=cs
xr = random.randint(50,150) yr = random.randint(230,350)
else:
#f=cs
xr = random.randint(250,350) yr = random.randint(230,350)
tt="N"+str(i)
a = xr
b = yr
r = 5
x0 = a – r
y0 = b - r
x1 = a + r
y1 = b + r
rh=10
xh0 = a - rh
yh0 = b - rh
xh1 = a + rh
```

```
yh1 = b + rh
bob="b"+str(i)
if k==1 and f==i:
self.cir = self.canvas.create_oval(xh0, yh0, xh1, yh1, fill="red", tags=bob)
 self.canvas.pack()
self.txt = self.canvas.create_text(xr-10, yr-10, text="CH", font=("purisa",8),
fill="#660033", tags=bob)
elif k==1:
self.cir = self.canvas.create_oval(x0, y0, x1, y1, fill="#FF9900", tags=bob)
 self.canvas.pack()
self.txt = self.canvas.create_text(xr-10, yr-10, text=tt, font=("purisa",8),
fill="black", tags=bob)
if k==2 and f==i:
#print("c")
self.cir = self.canvas.create_oval(xh0, yh0, xh1, yh1,
fill="#669900",tags=bob)
self.canvas.pack()
 self.txt = self.canvas.create_text(xr-10, yr-10, text="CH", font=("purisa",8),
fill="#669900", tags=bob)
elif k==2: #print("d")
self.cir = self.canvas.create_oval(x0, y0, x1, y1, fill="#00FF00", tags=bob)
self.canvas.pack()
self.txt = self.canvas.create_text(xr-10, yr-10, text=tt, font=("purisa",8),fill="black", tags=bob)
if k==3 and f==i:
elif k==3:
self.cir = self.canvas.create_oval(x0, y0, x1, y1, fill="yellow", tags=bob)
self.canvas.pack()
self.txt = self.canvas.create_text(xr-10, yr-10, text=tt, font=("purisa",8),fill="black", tags=bob)
if k==4 and f==i:
self.cir = self.canvas.create_oval(xh0, yh0, xh1, yh1, fill="#0033CC", tags=bob)
def movement(self):
k=1
while k <= self.n:
bb="b"+str(k)
```

```
ab = random.randint(-10, 10)cd = random.randint(-10, 10)
movenode=self.canvas.move(bb, ab, cd)k += 1
self.canvas.after(700, self.movement)
##Gas
gv4=""
gr1=randint(10,80)
gr2=randint(10,80)
gr3=randint(1,30)
ga1=randint(50,70)
ga2=randint(1,7)
gv1=str(ga1)+"."+str(gr3)
gv2=str(ga2)+"."+str(gr3)
gv3="0"+"."+str(gr3)
if ga2>4:
gas_st=1
gv4="Abnormal"
else:
gv4=" "
gas_st=0
gas_val="GAS: NH3: "+gv1+", CO:"+gv2+", SO2:"+gv3+" "+gv4
get_msg.append(gas_val)
gas_val2="GAS: NH3: "+gv1+", CO:"+gv2+", SO2:"+gv3
get_msg2.append(gas_val2)
#########fire
fr1=randint(20,200)
fr2=randint(30,34) fr3=""
if  fr1>100:
first=1 fr3="Abnormal"
else:
fr3=""
first=0
fir_val="Fire: Smoke: "+str(fr1)+", T:"+str(fr2)+" "+fr3
get_msg.append(fir_val)
```

```
fir_val2="Fire: Smoke: "+str(fr1)+", T:"+str(fr2)
get_msg2.append(fir_val2)
##health
hr1=randint(40,120)
hr2=randint(30,34)
hr3=" "
 if hr1<60 or hr1>80:
 hea st=1 hr3="Abnormal"
else:
hr3="" hea_st=0
 hea_val="Health: HB: "+str(hr1)+", T:"+str(hr2)+" "+hr3
 get_msg.append(hea_val)

hea_val2="Health: HB: "+str(hr1)+", T:"+str(hr2)
 get_msg2.append(hea_val2)
 sr1=randint(40,120)
 sr2=randint(30,34)
 sr3=randint(40,100)
 env_val="Environmental: M: "+str(sr1)+", T:"+str(sr2)+", H:"+str(sr3)
 get_msg.append(env_val)
 env_val2="Environmental: M: "+str(sr1)+", T:"+str(sr2)+", H:"+str(sr3)
 get_msg2.append(env_val2)
 ########self.a1 = Label(self.master, text=gas_val,bg='lightblue', fg='blue', font=("Helvetica",12))
#self.a1.pack()

#self.a1.place(x=50, y=50)
farr="|".join(get_msg)
farr2="|".join(get_msg2)
f1=open("data1.txt","r")
data1=f1.read()

f1.close() f1=open("data2.txt","r")
data2=f1.read() f1.close()
```

LSTM Prioritization

```
##LSTM
def load_data(stock, seq_len): amount_of_features = len(stock.columns)
```

```
data = stock.as_matrix()
#pd.DataFrame(stock)
sequence_length = seq_len + 1
result = []
for index in range(len(data) - sequence_length):
result.append(data[index: index + sequence_length])
result = np.array(result)
row = round(0.9 * result.shape[0])
train = result[:int(row), :]
x_train = train[:, :-1]
y_train = train[:, -1][:,-1]
x_test = result[int(row):, :-1]
y_test = result[int(row):, -1][:,-1]
x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1], amount_of_features))
x_test = np.reshape(x_test, (x_test.shape[0], x_test.shape[1], amount_of_features))
return [x_train, y_train, x_test, y_test]
def build_model(layers):
model = Sequential()
model.add(LSTM(
input_dim=layers[0],output_dim=layers[1], return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(layers[2],return_sequences=False))
model.add(Dropout(0.2))model.add(Dense(output_dim=layers[2]))
model.add(Activation("linear"))
start = time.time()
model.compile(loss="mse", optimizer="rmsprop",metrics=['accuracy'])
print("Compilation Time : ", time.time() - start)
return model
def build_model2(layers):d =0.2
model = Sequential()
model.add(LSTM(128, input_shape=(layers[1], layers[0]), return_sequences=True))
model.add(Dropout(d))
model.add(LSTM(64, input_shape=(layers[1], layers[0]), return_sequences=False))
model.add(Dropout(d))
model.add(Dense(16,init='uniform',activation='relu'))
model.add(Dense(1,init='uniform',activation='linear'))
model.compile(loss='mse',optimizer='adam',metrics=['accuracy'])
return model
def chPriority(self):
f3=open("det2.txt","r")
v3=f3.read()
f3.close()
v4=int(v3)+1
mss=" "
```

Deep Queue Scheduling

```
def DeepQLearning(num_of_nodes):
enviroment = "Packet" action_space="1"
action=0
alpha=1
reward=0
for customer in range(0, num_of_nodes):
# Reset the enviroment
state =  environment
#Initialize variables
reward = 0
def chPriority(self):
f3=open("det2.txt","r")
v3=f3.read()
f3.close()
v4=int(v3)+1
mss=" "
```

Deep Queue Scheduling

```
def DeepQLearning(num_of_nodes):
enviroment = "Packet" action_space="1"
action=0
alpha=1
reward=0
for customer in range(0, num_of_nodes):
```

```
# Reset the enviroment
state =  environment
#Initialize variables
reward = 0
terminated = False
j=1
n=num_of_nodes
while j<n:
#Take learned path or explore new actions based on the epsilonif
random.uniform(0, 1) < num_of_nodes:
i=0
k=0
while i<=num_of_nodes:
i+=3
k+=1
action = i
else:
action = np.argmax(q_table[state])
#Take action
gamma=1
#next_state, reward,
terminated, info = action
q_table=num_of_nodes/3
# Recalculate
q_value = k
max_value = q_table
#np.max(q_table[next_state])
new_q_value = (1 - alpha) *
int(q_value) + alpha *
(reward + gamma *
max_value)
 # Update Q-table
```

```
#q_table[state, action] =
new_q_value state
=new_q_value
j+=1
#if (queue + 1) % 100 == 0:
#clear_output(wait=True)
#print("Queue:
{}".format(queue + 1))
#enviroment.render()
def
QueuePredict(enviroment,
optimizer):
# Initialize attributes
_state_size = enviroment
_action_size = "1"
#enviroment.action_space.n
_optimizer = optimizer
expirience_
replay = int(enviroment/2)
# Initialize discount and
exploration rate
 gamma = 0.6
epsilon = 0.1
# Build networks
q_network = optimizer
target_network =
expirience_replay
def store(state, action,
reward, next_state,
terminated):
expirience_replay.append((st
ate, action, reward,
next_state, terminated))
def _build_compile_model():
 model = Sequential()
model.add(Embedding(_state
_size, 10, input_length=1))
model.add(Reshape((10,)))
model.add(Dense(50,
activation='relu')
model.add(Dense(50,
activation='relu'))
model.add(Dense(_action_siz
e, activation='linear'))
model.compile(loss='mse',
optimizer=self._optimizer)
return model
def alighn_target_model():
if np.random.rand() <=
epsilon:
return
enviroment.action_space.sam
ple()
return
np.argmax(q_values[0])
def retrain(batch_size):
minibatch =
random.sample(expirience_re
play, batch_size) for state,
action, reward, next_state,
terminated in minibatch:
target =
q_network.predict(state)
if terminated:
target[0][action] = reward
else:
```

```
t =
target_network.predict(next_
state)
target[0][action] = reward +
gamma * np.amax(t)
q_network.fit(state, target,
epochs=1, v
```

9.3. ALGORITHMS

Algorithm 1: LSTM Algorithm for Packet Priority classification

```
Data: Sequence of raw packets from network
1: dictionary = array () % Create the mapping Type ii to the index array
2. Translated Words = null; %variable to store the values of words to converted to integer
format;
3: while true do3: while true do
4: Parse the packet;
5: Extract 54-byte from each packet 6: if packet length < 54 then
7: Pad zeros;
8: end if
9: Feed forward to the second layer LSTM; 10: Dropout;
11: Feed forward to the third layer LSTM; 12: Dropout;
13: Apply RMS Prop optimizer;
14: Use soft max to output 0 or 1 to the model; 32: Use binary cross entropy as loss
function 33: for epoch = 1; epoch < 200; epoch++ do
15: Evaluate Loss, Validation Loss
16: Evaluate Accuracy and Validation
Accuracy 17: end for
```

Algorithm 2: DQN for Packet Scheduling

```
1. Input : PKa
2. Output: PA
3. Upon receiving emergency information packets
4. for n ← 1 to N do
5. if PKa [n][1] == i then
6. PNi ← PNi + 1
7. Put Hid into Pi
8. end if
9. end for
10. for i ← 1 to 3 do
11. if  PNi > 1 then
12. Sort the elements in Pi based on their deadlines;
13. end if
14. end for
15. Get the packet with the highest priority and the shortest deadline to update PA;
16. Broadcast the PA;
17. The node whose MAC address is equal to PA send
18. The whole packet;
```

CHAPTER 10

CONCLUSION

The IoT Network Emergency packet classification using LSTM and Packet Scheduling using Deep Q Learning is a complex system designed to classify and prioritize packets from IoT devices in emergency situations while maintaining the security and reliability of the network. The system uses machine learning techniques such as LSTM and Deep Q Learning to classify and prioritize emergency packets over regular packets, providing a fast and efficient response to emergency situations. In this project, by employing LSTM Model to classify the emergency packets and the classic deep Q network (DQN) architecture in intelligent scheduling, our PPQM can identify reasonable channel and time slot combinations with competitive performance. Extensive simulation experiments were implemented, demonstrating that our PPQM can obtain better network performance than the traditional scheduling schemes. In conclusion, the IoT Network Emergency packet classification using LSTM and Packet Scheduling using Deep Q Learning system has the potential to revolutionize emergency response systems in IoT networks. By accurately identifying emergency packets and prioritizing them, the system can help ensure that emergency responders receive critical information in a timely manner, potentially saving lives and minimizing damage. Overall, the IoT Network Emergency packet classification using LSTM and Packet Scheduling using Deep Q Learning system has the potential to significantly improve emergency response times in IoT networks, providing a fast and efficient response to critical situations.

FUTURE ENHANCEMENT

Future enhancements for the IoT Network Emergency packet classification using LSTM and Packet Scheduling using Deep Q Learning system:

1. **Multi-objective optimization**: The system could be enhanced to optimize not only for emergency packet classification and packet scheduling, but also for other objectives such as energy efficiency, network congestion, and latency.
2. **Dynamic adaptation to changing network conditions**: The system couldbe enhanced to dynamically adapt to changing network conditions, such as fluctuations in traffic volume, changes in network topology, and failures innetwork nodes.
3. **Integration with other IoT devices and networks**: The system could be enhanced to integrate with other IoT devices and networks, such as wearable devices and smart home systems, to provide a more comprehensive and seamless emergency response system.
4. **User interface improvements**: The system could be enhanced by improving the user interface, making it more intuitive and user-friendly foremergency responders and network administrators.

Hence, the future enhancements of the IoT Network Emergency packet classification using LSTM and Packet Scheduling using Deep Q Learning systemcould help to further improve the accuracy, efficiency, and effectiveness of the system, making it an even more valuable tool for emergency response in IoT networks.

REFERENCES

1. H. Xu, X. Liu, W. G. Hatcher, G. Xu, W. Liao, and W. Yu, “Priority-aware reinforcement learning-based integrated design of networking and control for industrial Internet of things,” IEEE Internet of Things Journal, vol. 8, no. 6, pp. 4668–4680, 2021.
2. J. Wang, Y. Zhang, Y. Liu, and N. Wu, “Multiagent and bargaining-game-based real- time scheduling for Internet of things-enabled flexible job shop,” IEEE Internet of Things Journal, vol. 6, no. 2, pp. 2518–2531, 2019.
3. L. Farhan, L. Alzubaidi, M. Abdulsalam, A. J. Abboud, M. Hammoudeh and R. Kharel, 2018 An efficient data packet scheduling scheme for Internet of Thingsnetworks,1st International Scientific Conference of Engineering Sciences - 3rd Scientific Conferenceof Engineering Science (ISCES), Diyala, 2018, pp. 1-6.
4. M. O. Ojo, S. Giordano, D. Adami, and M. Pagano, “Throughput maximizing and fair scheduling algorithms in industrial Internet of things networks,” IEEE Transactions on Industrial Informatics, vol. 15, no. 6, pp. 3400–3410, 2019.
5. T. Qiu, K. Zheng, M. Han, C. P. Chen, and M. Xu, “A data emergency-aware schedulingscheme for Internet of things in smart cities,” IEEE Transactions on Industrial Informatics, vol. 14, no. 5, pp. 2042–2051, 2018.
6. V. Kotsiou, G. Z. Papadopoulos, P. Chatzimisios, and F. Theoleyre, “LDSF: low- latency distributed scheduling function for industrial Internet of things,” IEEE Internetof Things Journal, vol. 7, no. 9, pp. 8688–8699, 2020.
7. X. Guo, H. Lin, Z. Li, and M. Peng, “Deep-reinforcement learning- based QoS-aware secure routing for SDN-IoT,” IEEE Internet of Things Journal, vol. 7, no. 7, pp. 6242–6251, 2020.
8. X. Lyu, W. Ni, H. Tian et al., “Optimal schedule of mobile edge computing for Internet of things using partial information,” IEEE Journal on Selected Areas in Communications, vol. 35, no. 11, pp. 2606–2615, 2017

9. X. Wang, Z. Ning, S. Guo, and L. Wang, “Imitation learning enabled task scheduling for online vehicular edge computing,” IEEE Transactions on Mobile Computing, 2020.
10. Z. Ning, P. Dong, X. Wang et al., “Partial computation offloading and adaptive task scheduling for 5G-enabled vehicular networks,” IEEE Transactions on Mobile Computing, p. 1, 2020..

BOOK REFERENCES

1. "Deep Learning with Python" by Francois Chollet.
2. "Designing Distributed Systems: Patterns and Paradigms for Scalable, Reliable Services" by Brendan Burns.
3. "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems" by Aurélien Géron.
4. "IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things" by David Hanes, Gonzalo Salgueiro, and Patrick Grossetete.
5. "Machine Learning: The Art and Science of Algorithms that Make Sense of Data" by Peter Flach.
6. "Natural Language Processing with Python" by Steven Bird, Ewan Klein, and Edward Loper.
7. "Network Warrior: Everything You Need to Know That Wasn't on the CCNA Exam" by Gary A. Donahue.
8. "Packet Guide to Core Network Protocols" by Bruce Hartpence.
9. "Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow" by Sebastian Raschka and Vahid Mirjalili.

10. "Reinforcement Learning: An Introduction" by Richard S. Sutton and Andrew G. Barto.

These books provide a comprehensive understanding of the different aspects of IoT network, packet classification, LSTM, and Deep Q Learning and can be helpful in designing and implementing the IoT Network Emergency packet classification using LSTM and Packet Scheduling using Deep Q Learning system.

WEB REFERENCES

1. A Comprehensive Review of Deep Learning for Image Captioning in IoT" by Mehar Shafiq, Hassan Tahir, et al. **(https://www.mdpi.com/2079-9292/8/8/879)**
2. "An Overview of Deep Learning for IoT Big Data: Opportunities, Challenges, and Future Directions" by Muhammad Imran, Ayesha Kashif, et al. **(https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6174013/)**
3. "Packet Classification Using Machine Learning Techniques: A Survey" by FaisalAlmalkiandAimanErbad**.(https://www.hindawi.com/journals/scn/2020/8838749/).**
4. "Packet Scheduling for IoT Networks: A Comprehensive Survey" by Hailong Sun, Keqiu Li, et al. (https://ieeexplore.ieee.org/document/8490751)
5. "Reinforcement Learning for Packet Scheduling in Communication Networks: A Comprehensive Survey" by Yuxuan Jiang, Shengli Zhang, et al. **(https://ieeexplore.ieee.org/document/9192686)**

These web references provide a wealth of information related to IoT network, packet classification, LSTM, and Deep Q Learning that can be helpful in designing and implementing the IoT Network Emergency packet classification using LSTM and Packet Scheduling using Deep Q Learning system.