electronics

MDPI

Article

Reinforcement Learning Based Passengers Assistance System for Crowded Public Transportation in Fog Enabled Smart City

Gone Neelakantam [1], Djeane Debora Onthoni [1] and Prasan Kumar Sahoo [1,2,*]

[1] Department of Computer Science and Information Engineering, Chang Gung University, Guishan 33302, Taiwan; D0421004@cgu.edu.tw (G.N.); D0421008@cgu.edu.tw (D.D.O.)
[2] Division of Colon and Rectal Surgery, Chang Gung Memorial Hospital, Linkou 33305, Taiwan
* Correspondence: pksahoo@mail.cgu.edu.tw; Tel.: +886-3-211-8800 (ext. 3804)

Received: 6 August 2020; Accepted: 10 September 2020; Published: 13 September 2020

Abstract: Crowding in city public transportation systems is a primary issue that causes delay in the mobility of passengers. Moreover, scheduled and unscheduled events in a city lead to excess crowding situations at the metro or bus stations. The Internet of Things (IoT) devices could be used for data collection, which are related to crowding situations in a smart city. The fog computing data centers located in different zones of a smart city can process and analyze the collected data to assist the passengers how to commute smoothly with minimum waiting time in the crowded situation. In this paper, Q-learning based passengers assistance system is designed to assist the commuters in finding less crowded bus and metro stations to avoid long queues of waiting. The traffic congestion and crowded situation data are processed in the fog computing data centers. From our experimental results, it is found that our proposed method can achieve higher reward values, which can be used to minimize the passengers' waiting time with minimum computational delay as compared to the cloud computing platform.

Keywords: reinforcement learning; Q-learning; fog computing; smart city; crowd management

1. Introduction

The world's population is moving towards urbanization. According to the United Nations Organization, the inhabitants in cities have increased to 68% of the world's population in 2018 [1]. Transportation system plays a vital role in this modern cites to address the primary issues of user mobility in and around city. Each local body organizes and maintains multiple modes of transportation in a city, which is considered as the public transportation. The transportation systems such as monorail, metro, and buses are the major components for passengers communication. The numbers of vehicles in public transport are increasing drastically to assist the growing number of passengers, where traffic congestion [2], traffic accidents, air pollution [3], energy consumption and overcrowding are inevitable. For instance, according to Taipei city Department of Transportation, Taiwan, the increase in density of pedestrian population [4] near the public transportation stations leads to mass crowding alongside increase in the number of vehicles leads to traffic congestions and road accidents [5]. The traffic congestion indicates that large vehicular density drastically increases the heat, forming urban hot zones in cities [6,7]. Therefore, the governments take steps to make hassle free transportation by embedding smart, sophisticated and reliable systems, which can be used in assisting passengers and encouraging private vehicle users to utilize the well organized public transportation system.

The smart city utilizes digital technology to build sustainable infrastructure for convenient and comfortable lifestyle of the people [8] with intelligent transport system. The smart transportation

consists of smart public transport vehicles and smart mobile devices, where the Internet of Things (IoT) devices like camera sensor, motion sensor and card reading Radio Frequency Identification (RFID) sensors are embedded in smart environment to monitor flow of the passengers. The GPS is embedded in buses to get real-time location information by the passengers and smart card reader assists in keeping track of the number of available seats. In smart transportation systems, numerous varieties of data are generated from smart IoT and mobile devices. The generated IoT data are conventionally processed in cloud platform. However, the issue of processing and computation latency is inevitable due to high transmission and computational delay. Therefore, a fog computing environment is used for data processing and to assist passengers, where the fog node data centers could be deployed in distributed computing environment closer to the end user's location. These fog nodes enable IoT data computation with low-latency as compared to the conventional centralized cloud computing [9] platform. The fog computing nodes are hosted with intelligent algorithms to analyze the user's request based on the input IoT data from various locations of a smart city.

Artificial Intelligence (AI) is an advanced technology that enables us to extract meaningful information from the generated data by various IoT devices. The various Machine Learning methods such as Supervised, Unsupervised and Reinforcement Learning enable the system to learn from the generated data. In Supervised Learning algorithms, the labelled data are fed to the system for learning and predicting purposes, whereas Unsupervised Learning learns from the inherent structure of the input data without any labelling. In both learning algorithms, historical data are essential to build the models and predict the outputs. However, in real-time learning environment like smart transportation, model should learn directly from the environment to predict the next step of action. For example, Reinforcement Learning (RL) performs consistently to assist the optimal decisions for smart transportation in real-time environment. The RL algorithm is built on dynamic programming that rewards and penalize the agent based on the real-time actions, where agent learns how to behave in the environment to obtain the maximum reward. The RL based Q-learning algorithm will be a better approach to assist the passengers in smart transportation system, which is used by most of the research works in minimizing the waiting time.

Motivation and Goals

Day-by-day crowding in smart cities is unpredictable due to scheduled and unscheduled events such as concerts hosted by popular stars, unpredictable protests, terrorist attacks, and so on. In order to reach the destination smoothly in a smart city, the passengers normally use applications like Google Maps and social media to find the information about less crowded routes, and transportation systems. The cloud based Google Maps can give information regarding traffic congestion and suggests better route to the users. However, for the passengers commuting through public transportation ignore the context in cities. The passengers can know the bus or metro arrival schedule in smart city transportation system, but do not know the information regarding the density of the crowd and number of passengers waiting at a station. Recently released Google Maps [10] has new features for predicting crowded bus and train stations that currently cover only 200 cities. These features have been developed using crowd-sourcing data, which are sent by users voluntarily, where users have asked about the seat availability. However, in reality the crowded public transportation can be affected by internal and external factors. The internal factor includes the inflow and outflow of passengers from the connected train and neighboring bus stations. External factor includes the scheduled and unscheduled events around the stations such as concerts, unpredictable protests, and so on. The external factors are unpredictable conditions and the historical data are also not enough for predicting the crowded situation. Therefore, in this paper, a Reinforcement Learning based passengers assistance system is designed that can be used by the commuters under the situation of crowded public transportation in the fog enabled smart city by utilizing the data from the IoT devices in real time to minimize the waiting time of the passengers. Based on these motivations, the goals of this work are summarized as follows.

- To assist the passengers in finding less crowded and most suitable transportation points at a particular instant of time.
- To minimize the waiting and traveling time of the passengers by maximizing the reward in the proposed RL model.
- To minimize the service time and data transmission latency in the fog computing environment.

The paper is organized as follows. Related works are presented in Section 2, the system model of the paper is given in Section 3. The Reinforcement Learning model is described in Section 4. Performance evaluation is made in Section 5 followed by the concluding remarks in Section 6.

2. Related Works

In this section, we described several related works, which are related to different technologies used in solving public transportation issues in smart city. Those technologies are Wireless Sensor Networks (WSNs), Internet of Things (IoT), Fog Computing, and Artificial Intelligence (AI). The growth in urbanization needs an IoT based public traffic management system to predict traffic congestion, passenger mobility and crowd monitoring in public transportation system. In Wireless Sensor Networks, large numbers of sensors generate huge volume and varieties of data that can be utilized to develop suitable applications in smart cities. The authors have analyzed recent scientific developments that exploit open source electronic boards to create IoT and smart city applications [11]. Moreover, huge volumes of data are generated from the network devices will directly affect the quality of service of the users utilizing the network services. The authors have proposed a load balancing mechanism based on Software Defined Wireless Sensor Network (SDWSN) to address the requirements of adaptability and high flexibility of the services [12]. Furthermore, the authors in [13] proposed cloud assisted deep learning based processing to utilize the mobile audio data generated in smart city and classify the noisy area. However, the cloud based processing does not consider the context of noise from real-time environment such as public events in the smart cities. Moreover, the events organized in and around the cities usher bulk crowds to a single destination, where an emergency situation leads to a stampede as in 1990 in Mecca and German love parade disasters in 2010. The authors [14] have utilized the mobile application data and Amazon Web Services (AWS) for context awareness and guide people to nearest exit points. However, the IoT data analysis in conventional cloud environment causes a delay in computation.

Internet of Things (IoT) enabled public transportation plays a vital role in reducing the traffic congestions by moving large number of commuters to various destination in and around the smart city. However, there are significant issues that need to be addressed such as overcrowding at the commuting stations, heterogeneous IoT data analysis in the real-time smart transportation system for passenger assistance. Since, accidents on the roads are unpredictable events due to lack of adequate surveillance and risk perception, the authors [15] have designed a prediction model to predict the crash injury severity. The designed model is based on the Support Vector Machine (SVM) algorithm for decision making during accidents. Moreover, in autonomous smart cars the emergency situation needs a priority based decision making. Similar issue is addressed in [16] with flexible parameter setting system, considering constrains from different IoT systems on a vehicle and built an inference based earliest deadline scheduling mechanism i.e., Fair Emergency First (FEF) algorithm. On the other hand, the authors in [17] have collected CCTV image data from the Singapore Public Transportation Company and have analyzed the data using a CNN based model to classify the crowd density in individual train cabins and to identify the vacant cabins and avoid overcrowding in a single cabin near to the escalators in the boarding trains. However, the computation and communication infrastructure cost for supporting the deployed monitoring and display system leads to higher service costs. Thus leads to a one of the prevailing issue in deciding suitable computational environments to a vastly generating IoT devices data. Furthermore, the authors in [18] adapted a fog based queuing model approach to optimize data transmission, delay, energy and cost for mobile devices, through which computational

resources are allocated at near edge fog computing environment based on the number of requests and fog node resource availability.

The prevailing fog computing is embedded into the smart transportation system to enhance the computation of the monitoring infrastructure near to the edge devices alongside the centralized cloud environment. For instance, the authors in [19] have proposed a fog based cost-effective crime detection assistance application, where the video data are analyzed on the fog computing platform to assist the security application. Fog enabled public vehicles play a vital role in smart city for near edge computing and real-time data analysis. In [20], authors proposed a distributed public vehicle scheduling system that integrates fog nodes and vehicular sensing, where authors have analyzed Taiwan, Ximen taxi positioning and operation trajectories record data sets. The analysis of user data nearer to the end user has primary advantage. Therefore, we consider fog computing for the transportation system to analyze heterogeneous IoT data. Furthermore in [21] the privacy preserving issue in customized bus sharing service has been addressed by adopting fog computing where the fog servers perform ride clustering, interacting with fog server near to the end user, on receiving pick-up and drop-off positions from user device. Equally important, in [22] authors have designed an application to find the optimal charging station in a real-time transportation. The route predicted in context aware fog computing assist in finding the suitable charging station with optimal distance and route, based on amount of energy left in the electric vehicles. Moreover, fog computing based IoT data collection, classification and real-time decision making in healthcare is a signification approach to assist multiple healthcare entities with real time alerts [23]. The authors in [24] have collected input IoT sensor data from the users and have provided data risk assessment by adopting the reinforcement learning method. Furthermore, authors in [25,26] have analyzed the challenges in ultra-dense deployment of the heterogeneous fog networks in advancement of moving towards the AI based machine-oriented communication such that wireless data acquisition, knowledge discovery, planning, operation, and management could be optimized in the fog computing nodes.

Average passenger flow increases day-by-day in Shanghai city subway, Shanghai rail transit, and the regular city dwellers change the platform to board next train that causes overcrowding during the peak hours. Authors in [27] adopted K-means clustering to optimize operation of subway service and improved the quality of management by analyzing the crowd monitoring data. Transportation capacity of Urban Rail Transit (URT) is not adequate enough to meet the travel demands where the density of the passengers waiting at the platform can exceed the critical density of the platform. At URT line in Shentong metro management center, Shanghai, passengers are left stranded at the metro station as the time table become infeasible. The authors in [28] proposed coordinated optimization scheme for URT line, which combines both the coordinated passengers inflow control with train rescheduling strategies. Similarly, authors in [29] proposed to optimize the inflow volume during a certain period of time at each station with the aim of minimizing the safety risks imposed on passengers at the metro stations. Both [28,29] designed the proposed model in Reinforcement Learning (RL). Moreover, decision making being a prominent part in various types of transportation systems for optimizing as well as to avoid undesirable events. Thus, the intelligent decision making becomes an important part of future decision on air confrontation, where authors in [30] adapted heuristic Q-Network method to enhance the efficiency of RL based algorithm. Furthermore, in automated vehicles prevailing issue is in decision making in lane changing. The authors in [31] defined a policy based on Deep Reinforcement Learning (DRL) to find a policy assisting the vehicle's lane changing decision making system.

3. System Model

Let, the geographical area of a smart city be divided into certain number of zones i.e., a grid of size $M \times N$, where M and N are the number of rows and columns of the grid, respectively. Each zone is represented as Z_i, $\forall i \in \{1, 2, 3, ...|M \times N|\}$. Let, f_i be a fog node data center located in the zone Z_i, where, $i \in \{1, 2, 3, ..., |M \times N|\}$. Let us consider two different types of transportation system such as a and b represented as metro train and bus, respectively. The associated nearest metro station is

denoted as α_a for metros and bus station is denoted as α_b. It is assumed that sensor gateways and Road Side Units (RSUs) are deployed in each zone to communicate data in the smart city. All parameters described above are depicted in the proposed system model as shown in Figure 1 and are listed in the Table 1.

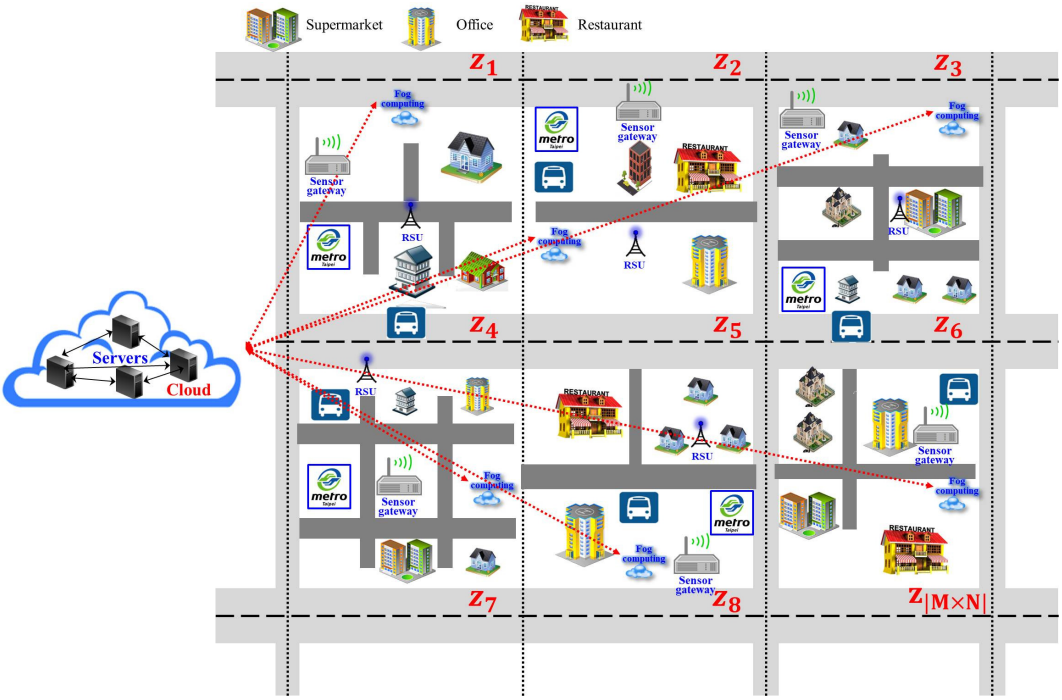

Figure 1. Proposed system model.

Table 1. Parameters name and description.

Parameters	Description
Z_i	Each zone
a, b	Metro train a, bus b
α_a	Metro station α_a
α_b	Bus station α_b
f_i	Fog node
$t, \forall t \in T$	Time stamp t, total duration T
p	Passenger
p_{id}^{loc}	Passenger's current location ID
p_{id}^{des}	Passenger's destination ID

3.1. Data Collection Phase

Various types of IoT devices are embedded in a smart city in synergy with fog integrated transportation systems. These IoT devices in each zone can be categorized as Passenger and Transportation IoT devices. The smart city passengers p can access information with help of the smart mobile phone along with smart payment ID card provided by various vendors, which are used in public transportation systems and are categorized as Passenger IoT devices. Two types of data such as passenger's current location p_{id}^{loc} and requested destination p_{id}^{des} can be obtained from the passenger's IoT devices. On the other hand, transportation IoT devices consist of bus and metro passenger's payment system that generate several data such as the total inflow and outflow of passengers, and number of passengers those are waiting in the line to reach at their destinations. Smart CCTV sensors generate the organized or emergency events monitoring data around the zones and passenger's crowd density at the station. All data are transmitted to the fog node data centers,

as shown in Figure 2 through the sensor gateway and RSU deployed in the smart city. It is to be noted that all data are collected based on the time stamp t.

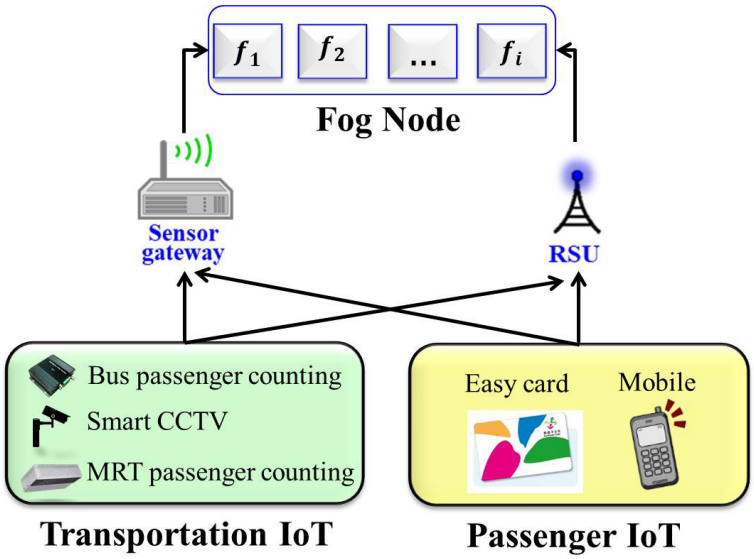

Figure 2. Data collection architecture.

3.2. Data Analysis Phase

Considering any passenger p at certain point of location p_{id}^{loc} in a smart city requests to know about the crowding status to travel to the destination p_{id}^{des} at time t, the request is sent to the fog node through the communication network. Based on passenger p's request, p_{id}^{des} and location p_{id}^{loc} are recorded and used to organize the collected data. Furthermore, the collected data in each zone are analyzed at the fog node f_{ij} located in the nearest zone. The collected data are processed based on the time stamp t, passenger's location p_{id}^{loc}, nearby station α and destination p_{id}^{des}, as shown in Figure 3. The data analysis is carried out using Reinforcement Learning (RL).

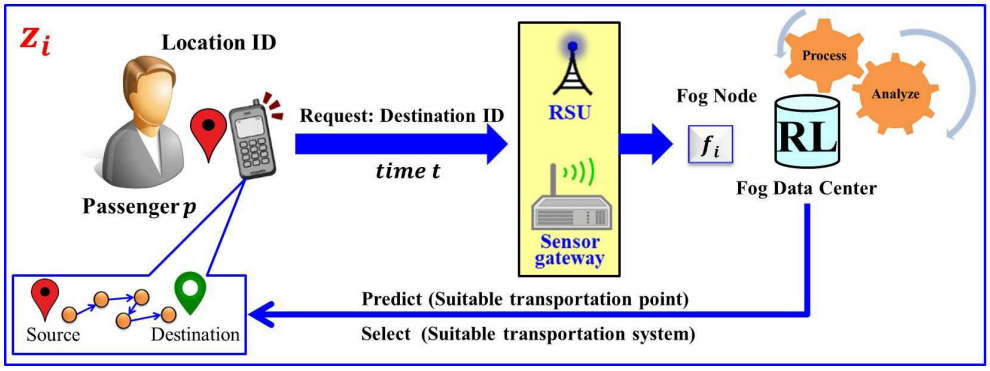

Figure 3. Analysis of data in fog node data centers.

3.3. Objective Function

The objective of our proposed work is to minimize the total waiting time of passengers from the source to the destination station. To achieve the minimum traveling time of the passengers, we consider that passengers try to avoid crowded stations in order to avoid delay and minimize the initial number of stations visited during the travel. As we know, social events, musical concerts, popular sports activities etc., in a city may generate a huge crowd, which make the stations crowded and the traffic is congested as people want to go back to their homes immediately after the event is over. Such an unavoidable situation makes the passengers delay in reaching their destinations. The Equation (1)

represents the objective function to minimize the delay $MinD$, where $W^{p}_{c,\alpha}$ is the waiting time of a passenger p and c is the passenger density representing the volume of the passengers $l = \{1, 2, 3, ...c\}$ at visited station α such that $h = \{1, 2, 3, ...\alpha\}$.

$$MinD = \sum_{c=1}^{l} \sum_{\alpha=1}^{h} W^{p}_{c,\alpha} \tag{1}$$

To optimize better available transportation methods for the passengers, two feasible constrains are considered to build our model. First, passenger p's waiting time at a station and the fog node context awareness to determine the passenger density. All used parameters are described as constraints in Table 2.

Table 2. Parameter's name and description.

Parameters	Description
p^{α}_{arriv}	Passenger arrival time at station
γ^{α}_{arriv}	Passenger arrival time at a station from the bus or metro train
W^{α}_{p}	Passenger waiting time at station
e_{α}	Events occur at station
c^{α}_{l}	Passenger density at station
P^{tot}	Total number of passengers
$G^{P^{tot}}_{enter}$	Total number of passengers that entered the platform
$Tr^{P^{tot}}_{transit}$	Total number of passengers transited
$Ex^{P^{tot}}_{exit}$	Total number of passengers exited
h	Total number of stations visited by passenger
$p^{tot}(\gamma_{\alpha} \rightarrow p^{des})$	Total number of passengers transited at station α from arrived bus or train

3.3.1. Passenger Waiting Time

The passenger's waiting time is the primary constraint that needs to be considered, where the passenger needs to wait until the arrival of any bus or train. The buses and metro trains follow different time intervals to run all over the city that connect multiple stations. The transportation commodity arrival time is denoted as γ^{α}_{arrv}. However, the passenger needs to reach any station to commute to the respective destination. The passenger arrival time at any transportation station is represented as p^{α}_{arrv}. Thus, the passenger p's waiting time at a particular metro or bus station α is determined from constraints as given in Equation (2).

$$W^{\alpha}_{p} = \gamma^{\alpha}_{arrv} - p^{\alpha}_{arrv} \tag{2}$$

3.3.2. Context Awareness at Stations

As shown in Figure 2, it is assumed in system model that the stations are equipped with IoT devices to collect transportation related data. These IoT devices can record and store the contexts of the environment in a station. For instance, CCTV keeps track of the events and passenger's density, and smart card reader records the inflow and outflow of passengers in the station. Two context awarenesses are considered in our proposed method as events and passenger density.

1. **Events**
   An event around the station environment is detected by IoT devices installed in and around the station. Events can be of organized programs like concerts by the popular stars, while unorganized hazardous events like bomb blasts lead to blocking the stations with overcrowded passengers. Thus,

IoT devices detect the events and store the event information e_α of a station α in binary values as illustrated in Equation (3).

$$e_\alpha = \begin{cases} 1 & e_\alpha = 1, \\ 0 & otherwise \end{cases} \tag{3}$$

where $e_\alpha = 1$ if any unpredictable crowding situation occurs, otherwise $e_\alpha = 0$.

2. **Passenger Density**

The passenger density is determined by the inflow and outflow of passengers at a particular station. The passenger crowding is occurred in the stations having more than one transit platforms where passenger transits from one metro train to another by changing the platform within station. Alongside, the passengers aligning from other transportation change to the nearest stations, and the passengers exit from the station. The following Equation (4) describes constraints for large passenger density l or crowding at the station α.

$$c_l^\alpha = (inflow) - (outflow) \tag{4}$$

The inflow of passengers at a station leads to huge crowds. It is the summation of total number of passengers arriving from different station and departing from the current station as given in Equation (5). Passengers who are waiting in a station may board or wait for the next bus or train as represented in Equation (7).

$$\boldsymbol{inflow} = \gamma_{arriv}^{p^{tot}} + G_{enter}^{p^{tot}} \tag{5}$$

$$\gamma_{arrive}^{ptot} = \sum_{\gamma=1}^{k} p^{tot} \tag{6}$$

$$G_{enter}^{ptot} = \sum_{t \in t} checkin \tag{7}$$

$$checkin = \begin{cases} 1 & passenger\ p\ check\text{-}in\ the\ payment \\ 0 & otherwise \end{cases} \tag{8}$$

Similarly, the passengers may exit the station or transfer to another vehicle within the same station to continue the journey towards passenger's destination as given in Equation (9). Passengers either exit the station or transfer the platform is represented in Equation (12).

$$\boldsymbol{outflow} = Tr_{transit}^{p^{tot}} + Ex_{exit}^{p^{tot}} \tag{9}$$

$$Tr_{transit}^{ptot} = \sum_{p^{des} \in Tr} p^{tot}(\gamma_{\alpha \to p^{des}}) \tag{10}$$

$$Ex_{exit}^{ptot} = checkout \tag{11}$$

$$checkout = \begin{cases} 1 & passenger\ p\ checkout\ the\ payment \\ 0 & otherwise \end{cases} \tag{12}$$

4. Reinforcement Learning Based Passenger's Assistance System

It is to be noted that any passenger who acts as an agent needs to choose better a transportation method such as bus or metro to reach the destination by avoiding possible traffic congestion related difficulties, which is the primary goal of this paper. Therefore, we modelled the transition of the passengers using a Markov Decision Process (MDP) to achieve the objectives described in the previous sections. The transition probability of the passengers is represented as 5-tuples t, S, A, T, R, where the parameters are listed in Table 3.

Table 3. Descriptions of Markov Decision Process (MDP) parameters.

Parameters	Description
t	Time step
S	Set of environment states s
A	Set of actions a
T	Set of transition probabilities
R	Set of reward, $R : SXA \rightarrow R$

4.1. State

Initially, any passenger starts from any state s at any given time t in the environment. As shown in Equation (13), the state or station α represents the metro station α_a and bus station α_b. The passenger may transfer within the same station or transition to different stations. Furthermore, the transition from state α at time t to the next state $\alpha + 1$ at time $t + 1$ can be performed by the passenger to reach the desired destination.

$$S_t = \{s_t^{\alpha_a}, s_t^{\alpha_b}\} \tag{13}$$

4.2. Action

When the agent sends a request from the free zone, information about all available nearby stations can be extracted from the fog node. Two possible actions can be taken by an agent to choose the suitable transportation point. Similarly, from the starting point, the agent again needs to select the next one of the two possible suitable transportation systems to move from the initial state s_α to the next state $s_{\alpha+1}$. The actions could be taking metro or taking bus as in $A = \{takingbus, takingmetro\}$. The states and actions are pictorially represented in Figure 4.

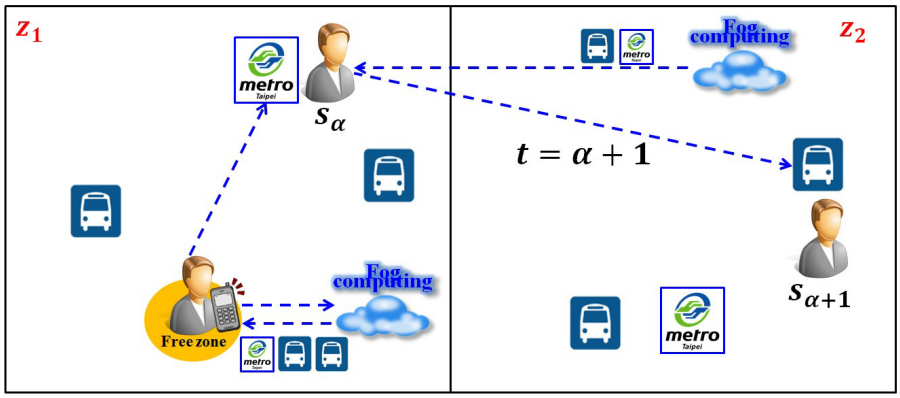

Figure 4. States and actions.

4.3. Reward

The goal of the agent is to minimize the travel and waiting time from the initial state to the destination state. By doing so, few constraints need to be considered such as total time step T, total visited stations h and total number of metros or buses associated with that particular station α. Reward function $R(s, a)$ is represented in Equation (14).

$$R(s, a) = \sum_{c=1}^{l} \sum_{\alpha=1}^{h} W_{c,\alpha}^{p} \tag{14}$$

4.4. Reinforcement Learning Algorithm

As an agent does not have any idea about the environment, model free Reinforcement Learning (RL) is chosen here. As one of the model free algorithms and based on off-policy Q-learning [32]

can benefit the agent's experience by exploiting and exploring the environments. Two algorithms are modeled using Q-learning, out of which an agent can learn and act optimally by choosing the suitable initial station state s_α from the free zone area in a particular zone based on the first algorithm. Upon deciding the suitable station based on Algorithm 1, an agent can smartly choose the next suitable transportation system state based on Algorithm 2 until agent reaches the destination station. The formal description of the first and second algorithms are given in Algorithms 1 and 2, respectively.

Algorithm 1 Prediction of suitable transportation point

For passenger $p = 1$
Input: Passenger request p_{id}^{des}
Parameter: Event e_i, passenger density c_l^α
Process:
1:Agent send a request: p_{id}^{des}
2:Fog node receive $(p_{id}^{des}, p_{id}^{loc})$
3:Check station $\alpha = p_{id}^{des}$ nearby p_{id}^{loc}
4: **IF TRUE**
5: Check station $\alpha = e_\alpha, c_l^\alpha$
6: **FOR** each station $\alpha = e_\alpha, c_l^\alpha$
7: **DO**
8: $X_\alpha^t = e_\alpha.c_l^\alpha$
9: **IF TRUE** $(X_\alpha^t == 0)$
10: Predict and send suitable transportation point (x_α^t)
11: Agent select the transportation point (x_α^t)
12: **Else**
13: Fog node search for other possible α
14:**END**

Algorithm 2 Selection of most suitable transportation system

Input: Total visited stations h, Total number of metros or buses I, Passenger waiting time $W_{c,\alpha}^p$
Parameter: Alpha α, γ, Policy π
Process:
1:For episode 1 to M do
2:Initialize state $s = x_{\alpha_a}^t$ or $x_{\alpha_b}^t$ /based on Algorithm 1
3:Repeat for each step of episode
4:Choose action a
5: Calculate π (using Boltzman-Distributed Exploration [33])
6: Take an action a based on π
7:Observe $\acute{s}$
8:Calculate Reward r (Equation (14))
9:Calculate $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{\acute{a}} Q(\acute{s}, \acute{a} - Q(s,a))]$
10:Update $Q(s,a)$
11:$s \leftarrow \acute{s}$
12:Until $s = p_{id}^{des}$
13:**END**

4.5. Service Time Latency Minimization

In this paper, we analyze that the fog computing nodes can have minimum latency in providing the computational services to the passengers of the smart city transportation system. In the fog computing paradigm [34], fog nodes process the data with low latency to meet the edge node's demands. The total

latency in locally distributed fog computing can be estimated based on n-dimensional Euclidean model, and the latency prediction formula as in [35].

$$D_{(f_1,f_2)} = \sqrt{\sum_{i=1}^{n}(f_{1_i} - f_{2_i})^2} \tag{15}$$

where, D is predicted delay, f_{1_i} and f_{2_i} are the coordinates of two fog nodes f_1 and f_2. The estimated latency E_f and actual measure distance, i.e., round trip time ν are used to estimate the absolute relative error R, which can give the difference between the estimated and total delay as represented in the Equation (16).

$$R = \frac{|E_f - \nu|}{min(E_f, \nu)} \tag{16}$$

For instance, if two applications' data with high and low computational resource requirements need to be processed in the fog network, the probability of offloading the high resource application data such as video is offloaded to the cloud computing environment, whereas the low computational application data are processed at the fog node data center. However, it is assumed that the fog nodes process the context aware related data in each zone. On the other hand, cloud being the core computing environment processes the image and video data from the camera sensors to determine the passenger's density c and events in the city e. The processed information is further updated at the fog node data centers based on the location information of the events. Based on the received task size R of the passengers, fog node data centers compute the estimated waiting time and provide the computing resources, where threshold θ amount of resources could be allocated. If the received requested task size is less than that of the threshold resources and average waiting delay, the request is accepted at the fog node. If the request is accepted at the fog node f_i, the waiting queue is updated with the request. Else, the request R_{fwd} is forwarded to another neighboring fog node f_i in the fog computing network among the available fog computing data centers, which is based on the maximum offloading limit of the fog layer. If it crosses the limit, data are forwarded to the cloud computing environment to minimize the delay in computation and provide the quality of service to the passengers.

$$L = \begin{cases} R < \theta \quad \textit{fog node accept and update the queue} \\ R_{fwd} \leq \theta \quad \textit{fog node forward the request to best neighbor node} \\ R > \theta \quad \textit{fog node forward the request to cloud} \end{cases} \tag{17}$$

5. Performance Evaluation

In this section, our proposed algorithms are evaluated by simulating in the fog computing and reinforcement learning based analytic environments. Fog computing based performance evaluation is executed using PureEdgeSim simulator [36]. For fog and cloud computing platforms, we compared and evaluated the computing resource consumption, task computing delay and energy consumption. From our simulation results, it is observed that distributed fog computing platform could provide the computing services consistently as compared to the conventional cloud computing platform. Furthermore, the Reinforcement Learning (RL) approach in finding better transportation for the smart city passengers was evaluated using Python and OpenCV. By doing so, the predicted transportation method can assist the passengers to reach at their destination faster by achieving the maximum reward and concurrently minimizing the waiting time.

5.1. Simulation Environment

Similar to our system model, we considered the simulation environment that consisted of the cloud data center, distributed fog node data centers, and end-user devices that included laptops, mobile phones, IoT actuators and IoT sensors. Accordingly, the cloud computing environment was assumed to have large amount of physical resources as compared to the fog node data centers.

Furthermore, the physical properties of the IoT devices and computing entities varied from one another. In detail, our simulation environment consisted of a cloud and fog nodes setup with two hosts, 16 VMs each, and devices with one and zero VMs as illustrated in Table 4. The IoT applications with different data size and computing requests were executed in two different computing platforms, where comparison and evaluation were performed with various input parameters.

Table 4. Fog simulation environment: specifications.

Parameters	Cloud DC	Fog DC	Laptop	Mobile Phone	IoT Actuator	IoT Sensor
Layer	0	1	2	2	2	2
CPU	2×10^6 MIPS	16×10^4 MIPS	11×10^3	25×10^3	16×10^2	4×10^2
Memory	150 GB	40 GB	8 GB	4 GB	2 GB	2 GB
Storage	10 TB	2 TB	300 GB	30 GB	20 GB	0
VMs	16	16	1	1	1	0
Hosts	2	2	-	-	-	-

The simulation environment was developed by considering a real-world urban transportation system in Taipei City that comprises Taipei Mass Rapid Transit (MRT) and Taipei Bus. We took an example of the main city transportation system. In MRT system, we included four lines such as GreenLine (GL), BlueLine (BL), RedLine (RL), and BrownLine (BRL), where each station was connected to each other with the corresponding lines. Besides, GL, BL, and RL are interconnected with BRL. Moreover, due to numerous bus stations nearby each MRT station, for simplicity, let $\{North, West, East, South\}$ be the four neighboring bus stations nearby any single MRT station. Different from the MRT connectivity, bus stations were randomly connected to each other to the respective MRT lines. The detail information about the considered simulation environment is depicted in Figure 5.

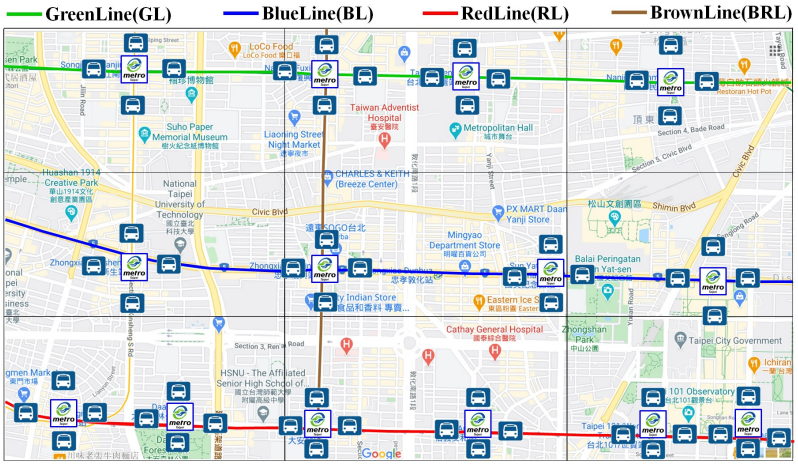

Figure 5. Simulation environment.

Based on our simulation environment, we set 12 nodes for the MRT, each of which had four bus stations as the neighboring nodes. Accordingly, we used $12(12 \times 4)$ nodes in our simulation along with the inter-connectivity. We assigned the initial inflow and outflow of passengers to any interconnected node either as a bus or MRT station based on the collected data [37,38]. Starting point and destination point of an agent was assumed to be fixed from any given point, i.e., either from a bus or MRT station. The implementation of the Reinforcement Learning based simulation was run on the GPU GeForce GTX 1070 Ti system, 418.56 NVIDIA-SMI, 32 GB memory, and Ubuntu 18.04.2 platform. The implementation was executed using Python 3.6.9 including python libraries Numpy 1.19.1, Matplotlib 3.3.0, and Networkx 2.4.

5.2. Simulation Results: Fog Computing

As per our assumptions in the system model, the fog enabled smart city consisted of various types of smart devices. Those smart devices were normally the IoT devices such as mobile phone, laptop, smart card reader, smart sensor and so on, which had limited computing resources. Though those IoT devices had limited computing resources, they generated a vast number of requests, which need to be processed and therefore need to be offloaded to the external computing environment like cloud and fog data centers. Conventionally, IoT device tasks were offloaded to the fog computing platform, even though the cloud computing platform's physical resource capabilities were significantly larger than that of the fog computing nodes. Furthermore, the fog nodes having limited computational capabilities were deployed to execute the requests from various IoT devices in the smart city. Upon receiving the requests of IoT users, those requests were executed in the neighboring fog nodes or needed to be forwarded to the cloud platform based on the availability of the computing resources as given in Equation (17). For instance, in our simulation we considered IoT devices of task requests with different data size ranging from $10^3 \sim 10 \times 10^3$ (Mega Bytes), which were randomly offloaded to the fog and cloud computing platform. We found that task execution latency in terms of execution delay was comparatively higher in cloud than that of the fog data center for average IoT requests. Figure 6 indicates that with increase in the average number of IoT device requests, data size impacted the average tasks execution latency at the fog and cloud data centers.

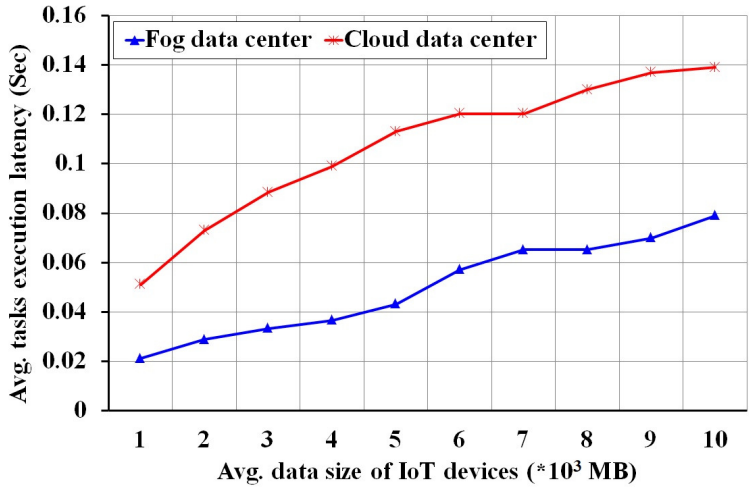

Figure 6. Comparison of Internet of Things (IoT) task execution latency in the fog and cloud platforms.

Furthermore, the fog node data centers that were widely deployed nearer to the edge devices provided computing services with high rate of resource utilization. For instance, passengers with smart phone device or IoT devices requested to find less crowded commuting stations, where the context aware fog nodes processed the data and provided computing resources with minimal delay as compared to the cloud platform. As shown in Figure 7, a number of user devices ranging from $10^3 \sim 10 \times 10^3$, with four different types of configurations, offloaded requests randomly to the computing environment. It is observed that IoT requests were mainly executed in the fog data centers as compared to the cloud data centers, which was inferred from the maximum CPU utilization of the fog data centers.

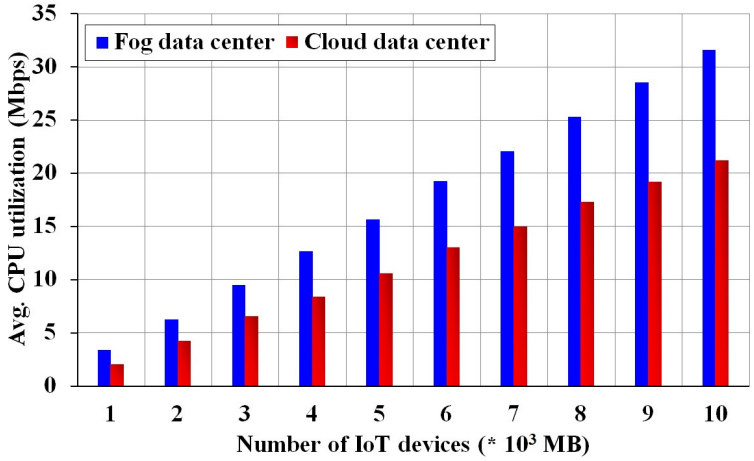

Figure 7. Comparison of CPU utilization in the fog and cloud platforms.

The IoT devices in the smart city generated task requests that needed to be executed in the fog or cloud computing platforms. The average computation time can be defined as the task waiting time and task execution time as shown in Figure 8. In a conventional cloud data center, the request processing queues were comparatively longer than that of the fog computing data centers. The cloud computing platform computes the global tasks of long queues that led to longer waiting time, as shown in Figure 8a. On the contrary, the distributively deployed fog computing data centers executed the local tasks with shorter queues of tasks that required processing the requests with minimum waiting time. Hence, as shown in Figure 8b, the waiting time of the tasks to be executed in the cloud needed a longer time as compared to the waiting time in the fog data centers.

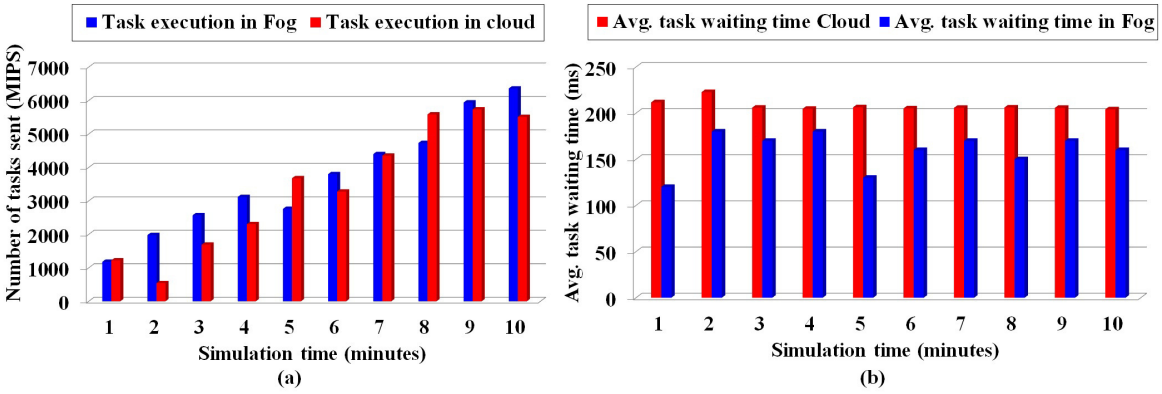

Figure 8. Computation time: (**a**) IoT tasks waiting time at fog and cloud data centers. (**b**) Avg. IoT tasks execution time at fog and cloud data centers.

In order to evaluate the energy consumption complexity, the generated tasks ranging from $10^3 \sim 10 \times 10^3$ MB are computed in the cloud and fog computing environments as depicted in Figure 9. It is observed that energy consumption in fog computing data centers was comparatively higher due to high execution rate of processing the data as compared to the cloud platform. It is due to the execution of large number of local tasks in the fog computing platform. Besides, due to extensive utilization of the computational resources, the energy consumption was also comparatively high in fog computing platform.

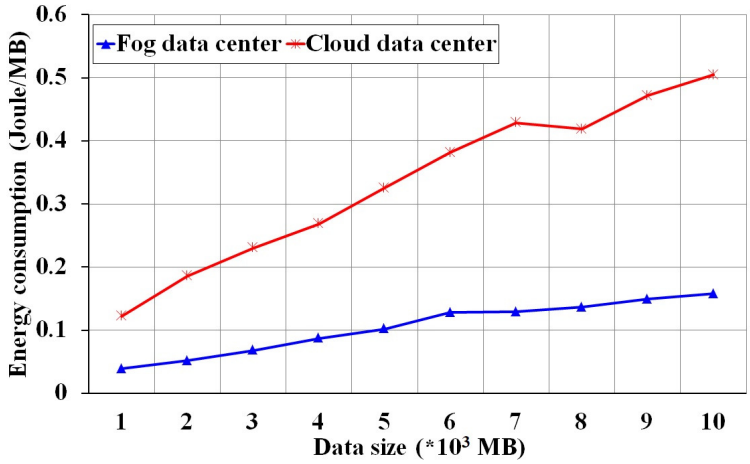

Figure 9. Energy consumption in fog and cloud platform.

5.3. Simulation Results: Reinforcement Learning

Based on Algorithms 1 and 2, we adopted Q-learning to conduct our simulation. The experiment was carried out in various episodes, and by considering the destinations nearer to and farther from the current location of the passengers. First, we optimized the value of the hyper-parameters of our adopted Q-learning using random search technique and by considering ten configurations. Table 5 shows the details of the hyper-parameter tuning configurations with the total number of episodes being 1000, which was composed of three hyper-parameters such as number of steps, learning rate α, discount factor γ, and $\epsilon-$ Greedy with range of $(0,1)$.

Table 5. Hyperparameter tuning configuration.

Configuration Number	Learning Rate (α)	Discount Factor(γ)	$\epsilon-$Greedy
1	0.6	0.9	0.09
2	0.7	0.7	0.07
3	0.8	0.8	0.08
4	0.9	0.3	0.07
5	0.7	0.5	0.05
6	0.8	1	0.04
7	0.2	0.6	0.03
8	0.3	0.7	0.01
9	0.4	0.1	0.02
10	1	0.2	0.1

Based on the configurations given in Table 5, we analyzed the maximum Q-Value and Cumulative Reward taking two scenarios, i.e., when the agent could have nearest and farthest destination stations either as bus or MRT from the current location of the passengers. The maximum Q-Value and cumulative reward were calculated when the training for any given episode was completed. Figures 10 and 11 show the results for nearest and farthest destination, respectively in terms of maximum Q-value (a) and cumulative reward (b). We observed that the highest maximum Q-value and total reward can be found in configuration number 1, 2, and 3 in both scenarios. These configurations have similarity, where all combinations of hyper-parameters value are >0.5. In contrast, low values can be found in configuration, which has average value <0.5 for any of the hyper-parameters.

By considering the optimal configuration of hyper-parameter number = 1 for a complex scenario such as farthest destination, we analyzed the performance of our adopted Q-learning algorithm in-depth by comparing with the well-known RL algorithm such as State-Action-Reward-State-Action (SARSA), SARSA Lambda, and Expected SARSA. First, we analyzed the maximum Q-value with different number of episodes. In episode 5000, it shows that SARSA and our adopted Q-learning had

similar performance. As the episode number increases, the maximum Q-value in adopted Q-learning increased as compared to SARSA. Similarly, SARSA lambda and expected SARSA had the lowest maximum Q-value as compared to the adopted Q-learning. It implies that the agent learned better by getting higher reward when the action was chosen to reach at the farthest destination. The comparison of the maximum Q-value is shown in Figure 12.

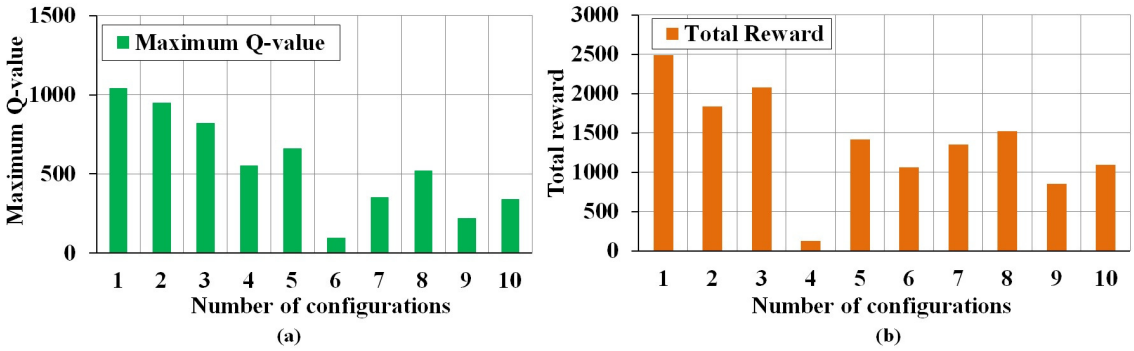

Figure 10. (**a**) Maximum Q-value and total reward comparison for nearest destination. (**b**) Total reward comparison for nearest destination

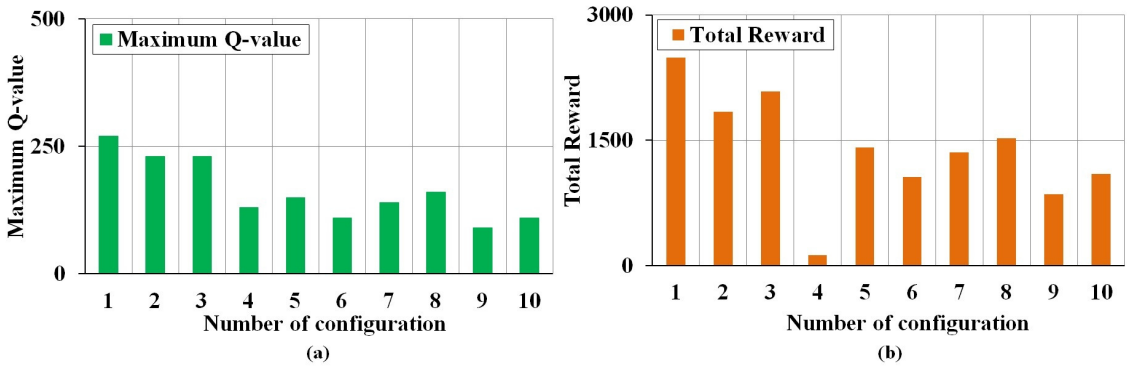

Figure 11. (**a**) Maximum Q-value and total reward comparison for farthest destination. (**b**) Total reward comparison for farthest destination.

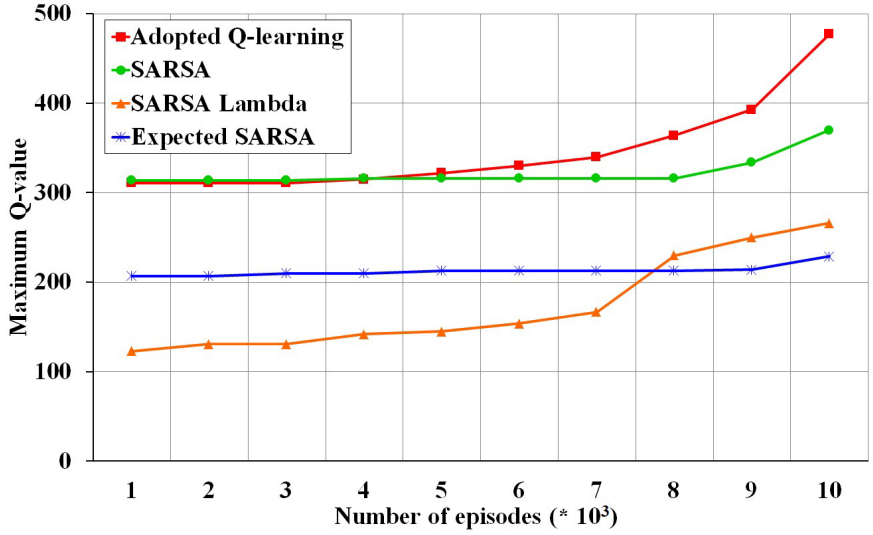

Figure 12. Maximum Q-value comparison.

It is observed that the maximum Q-value had an impact on the total reward performance. Based on the maximum Q-learning comparison, we compared the average rewards achieved by an agent.

As shown in Figure 13, the agent could successfully arrive at its destination by executing our adopted Q-learning, which was farthest from the starting point with highest average reward as compared to the other algorithms. Furthermore, we analyzed the execution time of our adopted Q-learning. As shown in Figure 14, the execution time was higher than the Expected SARSA as Q-learning took maximum value of the state–action pair, whereas in Expected SARSA, the expected value was used to define the likelihood of the actions to be taken.

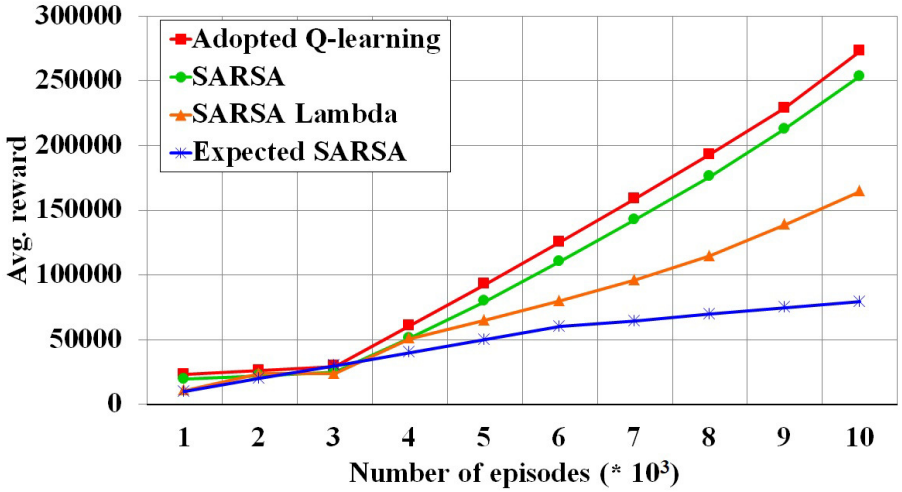

Figure 13. Average reward comparison.

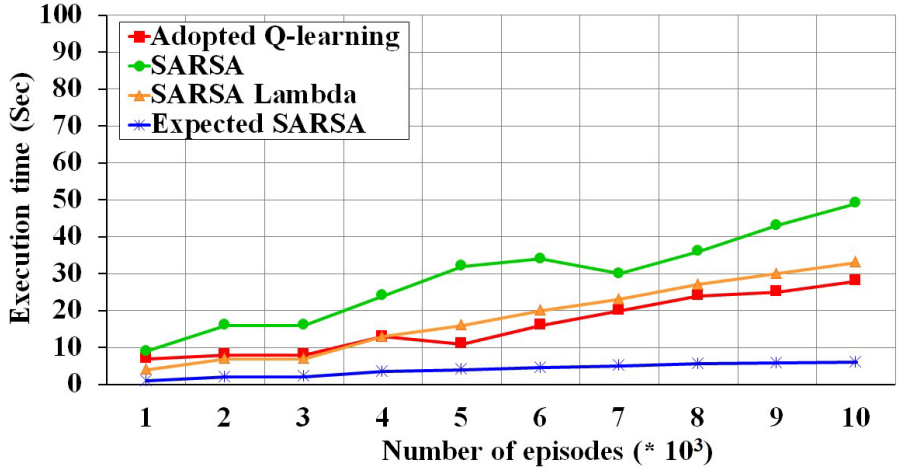

Figure 14. Execution time comparison.

6. Conclusions

In this paper, a passenger assistance system for crowded public transportation in fog-enabled smart city is designed by applying reinforcement learning, where the IoT data analysis is carried out in the fog computing environment. The data generated by the IoT devices of the passengers are collected and analyzed by fog nodes located in different zones of a smart city. In order to achieve our primary objectives of minimizing the passenger's waiting time, the highest reward is obtained by applying the Q-learning based reinforcement learning. Performance evaluation is made by considering the fog, cloud computing, and reinforcement learning environment. It is observed that the fog computing environment performs better over the cloud computing in terms of service time, computation time and energy consumption. Moreover, we also find that our adopted Q-Learning based reinforcement learning model achieves the highest maximum Q-Value and cumulative reward as compared to SARSA, SARSA lambda and expected SARSA, which indicates that the agent can concurrently reach the destination faster by reducing the waiting time. Finally, our proposed method can be beneficial

for reducing the travel time of the passengers under the crowded traffic conditions of the public transportation in a smart city.

Author Contributions: G.N., D.D.O. and P.K.S. conceived the idea and designed the algorithms, experiments, analyzed the data, and wrote the paper. G.N. performed the experiments on fog and cloud computing. D.D.O. performed the reinforcement learning related experiments. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partly supported by Ministry of Science and Technology (MOST), Taiwan under the grant number 109-2221-E-182-014.

Acknowledgments: We are thankful to Healthy Aging Research Center, Chang Gung University for supporting the GPU platform for our analysis from the Featured Areas Research Center Program within the Framework of the Higher Education Sprout Project (EMRPD1I0491) by the Ministry of Education (MOE), Taiwan.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. World Urbanization Prospects, The 2018 Revision. Available online: https://population.un.org/wup/Publications/Files/WUP2018-KeyFacts.pdf (accessed on 18 May 2019).
2. Wang, T.; Hussain, A.; Bhutta, M.N.M.; Cao, Y. Enabling bidirectional traffic mobility for ITS simulation in smart city environments. *Future Gener. Comput. Syst.* **2019**, *92*, 342–356. [CrossRef]
3. Jung, C.R.; Chung, W.T.; Chen, W.T.; Lee, R.Y.; Hwang, B.F. Long-term exposure to traffic-related air pollution and systemic lupus erythematosus in Taiwan: A cohort study. *Sci. Total Environ.* **2019**, *668*, 342–349. [CrossRef] [PubMed]
4. Shi, X.; Shao, X.; Guo, Z.; Wu, G.; Zhang, H.; Shibasaki, R. Pedestrian trajectory prediction in extremely crowded scenarios. *Sensors* **2019**, *19*, 1223. [CrossRef] [PubMed]
5. Department of Tranportation, Taipei City Governament. Available online: https://english.dot.gov.taipei/News.aspx (accessed on 28 April 2019).
6. Xiao, Z.; Dai, X.; Jiang, H.; Wang, D.; Chen, H.; Yang, L.; Zeng, F. Vehicular Task Offloading via Heat-Aware MEC Cooperation Using Game-Theoretic Method. *IEEE Internet Things J.* **2019**, *7*, 2038–2052. [CrossRef]
7. Wang, D.; Fan, J.; Xiao, Z.; Jiang, H.; Chen, H.; Zeng, F.; Li, K. Stop-and-wait: Discover aggregation effect based on private car trajectory data. *IEEE Trans. Intell. Transp. Syst.* **2018**, *20*, 3623–3633. [CrossRef]
8. Lin, C.; Zhao, G.; Yu, C.; Wu, Y.J. Smart City Development and Residents' Well-Being. *Sustainability* **2019**, *3*, 676. [CrossRef]
9. Singh, S.P.; Nayyar, A.; Kumar, R.; Sharma, A. Fog computing: From architecture to edge computing and big data processing. *J. Supercomput.* **2019**, *4*, 2070–2105. [CrossRef]
10. Transit Crowdedness Trends From Around the World, According to Google Maps. Available online: https://www.blog.google/products/maps/transit-crowdedness-trends-around/ (accessed on 23 August 2020).
11. Costa, D.G.; Duran-Faundez, C. Open-Source Electronics Platforms as Enabling Technologies for Smart Cities: Recent Developments and Perspectives. *Electronics* **2018**, *7*, 404. [CrossRef]
12. Cui, X.; Huang, X.; Ma, Y.; Meng, Q. A load balancing routing mechanism based on SDWSN in smart city. *Electronics* **2019**, *8*, 273. [CrossRef]
13. Hossain, M.S.; Muhammad, G. Environment classification for urban big data using deep learning. *IEEE Commun. Mag.* **2018**, *56*, 44–50. [CrossRef]
14. Franke, T.; Lukowicz, P.; Blanke, U. Smart crowds in smart cities: Real life, city scale deployments of a smartphone based participatory crowd management platform. *J. Internet Serv. Appl.* **2015**, *6*, 27. [CrossRef]
15. Liao, Y.; Zhang, J.; Wang, S.; Li, S.; Han, J. Study on Crash Injury Severity Prediction of Autonomous Vehicles for Different Emergency Decisions Based on Support Vector Machine Model. *Electronics* **2018**, *7*, 3813. [CrossRef]
16. Malik, S.; Ahmad, S.; Kim, B.W.; Park, D.H.; Kim, D. Hybrid Inference Based Scheduling Mechanism for Efficient Real Time Task and Resource Management in Smart Cars for Safe Driving. *Electronics* **2019**, *7*, 344. [CrossRef]
17. Goh, C.G.; Lim, W.H.; Chua, J.; Atmosukarto, I. Image Analytics for Train Crowd Estimation. In Proceedings of the 2018 Digital Image Computing: Techniques and Applications (DICTA), Canberra, Australia, 10–13 December 2018.

18. Liu, L.; Chang, Z.; Guo, X.; Mao, S.; Ristaniemi, T. Multiobjective optimization for computation offloading in fog computing. *IEEE Internet Things J.* **2018**, *5*, 283–294. [CrossRef]
19. Neto, A.J.; Zhao, Z.; Rodrigues, J.J.; Camboim, H.B.; Braun, T. Fog-based crime-assistance in smart iot transportation system. *IEEE Access* **2018**, *6*, 11101–11111. [CrossRef]
20. Lai, Y.; Yang, F.; Zhang, L.; Lin, Z. Distributed public vehicle system based on fog nodes and vehicular sensing. *IEEE Access* **2018**, *6*, 22011–22024. [CrossRef]
21. He, Y.; Ni, J.; Niu, B.; Li, F.; Shen, X.S. Privacy-preserving ride clustering for customized-bus sharing: A fog-assisted approach. In Proceedings of the 16th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), Shanghai, China, 7–11 May 2018; Volume 6, pp. 1–8.
22. Amini, M.H.; Karabasoglu, O. Optimal operation of interdependent power systems and electrified transportation networks. In Proceedings of the 16th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), Shanghai, China, 7–11 May 2018; Volume 11, p. 196.
23. Verma, P.; Sood, S.K. Fog Assisted-IoT Enabled Patient Health Monitoring in Smart Homes. *IEEE Internet Things J.* **2018**, *5*, 1789–1796. [CrossRef]
24. Huang, Y.; Guan, X.; Chen, H.; Liang, Y.; Yuan, S.; Ohtsuki, T. Risk Assessment of Private Information Inference for Motion Sensor Embedded IoT Devices. *IEEE Trans. Emerg. Top. Comput. Intell.* **2020**, *4*, 265–275. [CrossRef]
25. Bogale, T.E.; Wang, X.; Le, L.B. Machine intelligence techniques for next-generation context-aware wireless networks. *arXiv* **2018**, arXiv:1801.04223.
26. Zhang, Z.; Chen, H.; Hua, M.; Li, C.; Huang, Y.; Yang, L. Double Coded Caching in Ultra Dense Networks: Caching and Multicast Scheduling via Deep Reinforcement Learning. *IEEE Trans. Commun.* **2020**, *68*, 1071–1086. [CrossRef]
27. Chen, Y.; Yao, M.; Cai, Z. Research on the Classification of Urban Rail Transit Stations-Taking Shanghai Metro as an Example. In Proceedings of the 15th International Conference on Service Systems and Service Management (ICSSSM), Hangzhou, China, 21–22 July 2018; pp. 1–6.
28. Jiang, Z.; Gu, J.; Fan, W.; Liu, W.; Zhu, B. Q-learning approach to coordinated optimization of passenger inflow control with train skip-stopping on a urban rail transit line. *Comput. Ind. Eng.* **2019**, *127*, 1131–1142. [CrossRef]
29. Jiang, Z.; Fan, W.; Liu, W.; Zhu, B.; Gu, J. Reinforcement learning approach for coordinated passenger inflow control of urban rail transit in peak hours. *Transp. Res. Part C Emerg. Technol.* **2018**, *88*, 1–16. [CrossRef]
30. Zhang, X.; Liu, G.; Yang, C.; Wu, J. Research on Air Confrontation Maneuver Decision-Making Method Based on Reinforcement Learning. *Electronics* **2018**, *7*, 279. [CrossRef]
31. An, H.; Jung, J.I. Decision-Making System for Lane Change Using Deep Reinforcement Learning in Connected and Automated Driving. *Electronics* **2019**, *8*, 543. [CrossRef]
32. Christopher, W.; Peter, D. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292.
33. McFarlane, R. *A Survey of Exploration Strategies in Reinforcement Learning*; McGill University: Montreal, QC, Canada, 2018.
34. da Silva, R.A.C.; da Fonseca, N.L.S. On the Location of Fog Nodes in Fog-Cloud Infrastructures. *Sensors* **2019**, *19*, 2445. [CrossRef]
35. Li, J.; Zhang, T.; Jin, J.; Yang, Y.; Yuan, D.; Gao, L. Latency estimation for fog-based internet of things. In Proceedings of the 2017 27th International Telecommunication Networks and Applications Conference (ITNAC), Melbourne, VIC, Australia, 22–24 November 2017; Volume 1–6.
36. PureEdgeSim. Available online: https://github.com/CharafeddineMechalikh/PureEdgeSim (accessed on 28 August 2020).
37. Statistical Yearbook of Transportation, Taipei City. Available online: https://english.dot.gov.taipei/ (accessed on 24 August 2020).
38. Ridership Counts. Available online: https://english.metro.taipei/ (accessed on 28 August 2020).