Journal of Computer and Knowledge Engineering, Vol. 6, No. 1, 2023. (47-58)
DOI: 10.22067/cke.2023.78257.1066

An Intelligent Control Method for Urban Traffic using Fog Processing in the IoT Environment based on Cloud Data Processing of Big Data*

Research Article

Alireza Soleimany[1] Yousef Farhang[2] Amin Babazadeh Sangar[3]

Abstract: Due to such disadvantages of current traffic light control methods as waste of time, waste of fuel and resources, increased air pollution, providing an intelligent traffic light control system that leads to the shortest waiting time for vehicles and pedestrians becomes so significant. Given the high priority of this issue, this paper presents an intelligent urban traffic system method based on IoT data and fog processing. Fog processing is a platform that is at the edge of the network and provides powerful services and applications for users. Compared to cloud computing, cloud computing is closer to users and therefore collects information faster and disseminates it over a network of sensors. It also helps cloud computing to perform tasks such as preprocessing and data collection. Cloud computing is a new type of distributed processing structure used for the Internet of Things. This paper proposed a method called GW-KNN. According to this method, we first collect data through the Internet of Things. Then, the preprocessing operation and extraction of effective fields in the cloud processing section are performed using the k-nearest neighbor improved machine learning algorithm. Traffic on each road is predicted in the next time slot and this information is sent for use in the fog processing layer to make traffic control decisions. The concept of Euclidean distance network with Gaussian weight was used to predict the future traffic situation and KNN model was included in the algorithm output to increase the forecasting accuracy and finally solve the problem of traffic light control. This idea was implemented and simulated using MATLAB. To get the results, the implementation was done on a computer with an i7-10750 processor and 16 GB of main memory and 1 TB of external memory. The results of the evaluations show that the proposed method has a much better performance than the previous two methods in terms of absolute mean error percentage, absolute mean error percentage of traffic forecast, and average waiting time of each vehicle.

Keywords: Intelligent Urban Traffic Control, IoT, K-Nearest Neighbor Machine Learning Algorithm, KNN Model

1. Introduction

The world's population is growing rapidly, and as you know, a large part of this population lives in cities. The United Nations Population Fund estimates that approximately fifty-two billion people, or 64% of the world's population, will live in cities by 2025 [1]. Managing modern cities using the currentg methods have made challenges leading to the need of achieving smart cities. Today, road transport is widely used for moving passengers, products, and services. Many people use their own personal vehicles for transportation, and the number is increasing. The growth of these issues has increased the problems related to urban traffic. Traffic jams cause many problems, including passenger delays, waste of time, waste of energy and resources and fuel, air pollution, increase of pollutants, even increase of road accidents and loss of life and property. These issues have caused that urban traffic control attract a lot of attention, and researchers in both industry and academia are looking for effective solutions to improve existing traffic control systems. There are two solutions to the problem [2]. The first solution is to develop transportation infrastructure, which has a high cost and related challenges. An important point about this solution is that it is temporary and short-term, because it loses its effectiveness over time as the number of vehicles increases. The second solution is to use the existing infrastructure with the highest quality and in the best way, and to improve their efficiency and the systems that are used for manage them like the traffic light control system. The second solution is smarter, is able to achieve high efficiency, and can maintain its efficiency in the long run. The best way to properly control traffic congestion is using intelligent traffic light systems. In this paper, we use the second method to provide an intelligent urban traffic system method based on IoT big data. Urban traffic light control systems often use fixed scheduling for lights and only a small number of systems consider traffic conditions to a small extent [3]. Systems such as SCOOT [4] and RHODES [3] have been used and are able to adjust the color of traffic lights based on traffic situation, but there are still many challenges to improve the performance of these systems and introduce new systems with high efficiency. Traffic light control systems that use fixed schedules are not real-time and dynamic, and have already been calculated and the duration corresponding to each color of the light is set. These systems use traffic data that has already been collected offline, based on which the data for traffic lights is considered to be of the same or different duration. The point is that these systems may become useless due to events such as accidents, sports matches, bad weather, etc., and cause traffic jams at intersections. The presence of a traffic police at the crossroads can be effective, but in this case, manpower and time are wasted. It is clear that if we have a human operator

* Manuscript received: 2022 August 15, Revised, 2022 November 15, Accepted, 2023 February 12.
1. PhD student in computer, Department of Computer Engineering, Islamic Azad University, Urmia Branch, Urmia, Iran.
2. Corresponding author. Assistant Professor, Department of Computer Engineering, Islamic Azad University, Khoy Branch, Khoy, Iran. **Email**: Yfarhang@yahoo.com.
3. Assistant Professor, Department of Computer Engineering, Islamic Azad University, Urmia Branch, Urmia Iran..

who can measure traffic conditions and based on his experience and background about the intersection, as well as the current traffic conditions he observes, he can immediately determine the duration of each traffic light, the problem of congestion traffic is largely eliminated. These observations led us to have systems that take the traffic situation as input and learn in a way that, like a human force, can reasonably control traffic by interacting with the environment. The traffic management system is the best that allows vehicles and pedestrians experience the minimum delay and waiting time to cross the intersection [4].

One of the essential steps in creating an intelligent traffic light system is data collection. The tools used to collect data must be able to work properly in the worst weather conditions, 24 hours a day, 7 days a week. Among the devices used to collect data are wireless sensor network (WSN), IoT, video surveillance, satellite system, metrological sensors, traffic control centers, mobile infrastructure, and so on. In recent years, IoT has been used for smart traffic systems, which have made it possible to expand and develop smart cities. Numerous industry and academic studies have been conducted to improve IoT-based traffic control [5, 6, 7]. The authors in [1] used IoT to collect traffic data and used the concept of big data to process and analyze the collected dat. In this article, due to the large amount of data collected by the IoT to predict traffic conditions and future conditions, which includes such data as the data on vehicles lie the number of buses, taxis and personal vehicles, the data on non-motorized vehicles like bicycles, the data on roads such as the number of lanes, speed limits, distance between intersections, and other data such as weather conditions, road accidents, road repairs, sporting and political events, etc. Classical methods of data processing are not a good option and therefore the concept of big data is used.

The main idea of this paper is providing an intelligent traffic light system that can determine traffic lights based on IoT big data like an expert manpower that regulates the duration of each traffic light.

The main contributions of this article are:

1. In this paper, data is first collected through the Internet of Things. Then the preprocessing operation and extraction of effective fields in the cloud processing section are performed using the k-nearest neighbor improved machine learning algorithm;
2. In this paper, the method of using fog computing and cloud computing is different from other studies done in this field. Thus, in the proposed method, the improved k-nearest neighbor algorithm is used in the cloud processing center, which leads to the pre-processing of the number and speed of the collected vehicles, so that the database update operation can be performed better. Then, the equivalent distances between the road intersections, the calculated intersections, and the temporal-spatial correlation between them are obtained, which form the time vectors representing the traffic situation of each section. After that, the equivalent distances between the previous and current data are calculated based on Euclidean distance with Gaussian weighting to select KNN. The idea of applying fog computing and cloud computing in other studies was that they were only used to store the collected data and also to achieve lower latency than fog computing;
3. The traffic on each road is predicted in different time slices and based on this collected data, which is done in the fog processing layer, a decision is made to control the traffic lights;
4. To predict the future state of traffic, the concept of Euclidean distance network with Gaussian weight is used, and KNN model is included in the output of the algorithm to increase the accuracy of the prediction.

The rest of the article is organized as follows: Section 2 deals with the basic concepts including IoT and fog processing. Section 3 reviews the works that have been done to provide intelligent traffic control systems. Section 4 presents the proposed algorithm. Section 5 evaluates the performance of the proposed algorithm and compares it with the other two approaches. Finally, Section 6 concludes the study. Table 1 shows the list of abbreviations used in this article.

Table 1. List of abbreviations

Symbols	Abbreviation
KNN	K-nearest neighbors
IoT	Internet of Things
WSN	Wireless Sensor Networks
SLA	Service-level agreement
TLC	Traffic light control
MARDDPG	Multi-agent recurrent deterministic policy gradient
DRL	Deep reinforcement learning
SAE	Neural network deep stacked autoencoders
MADRL	Multi-Agent DRL
MDP	Markov Decision Process
ATSC	Adaptive Traffic Signal Control
DQN	Deep Q Network
MAP	Mean absolute percentage error
RMSE	Mean squared error
SVM	Support vector machine

2. Basic concepts

In this section, we will take a look at some of the basic concepts we need in this article, such as IoT and fog processing.

2.1. IoT

IoT is a popular and growing paradigm whose primary idea is to connect the data of various electronic devices over the Internet so that they can exchange information with each other and create new applications and services. Using IoT, people and objects can exchange information with each other anytime, anywhere. In IoT, sensors and devices that collect data for different applications detect any changes that occur in the environment and send it to a device. There are different types of sensors such as thermal, electrical, mechanical, active sensors. IoT has had many positive impacts on human life and has facilitated and improved the quality of human

life. Applications of IoT include health and medical systems, transportation, smart city, smart home, smart traffic control, education and so on. Figure 1 shows various applications of IoT [8].

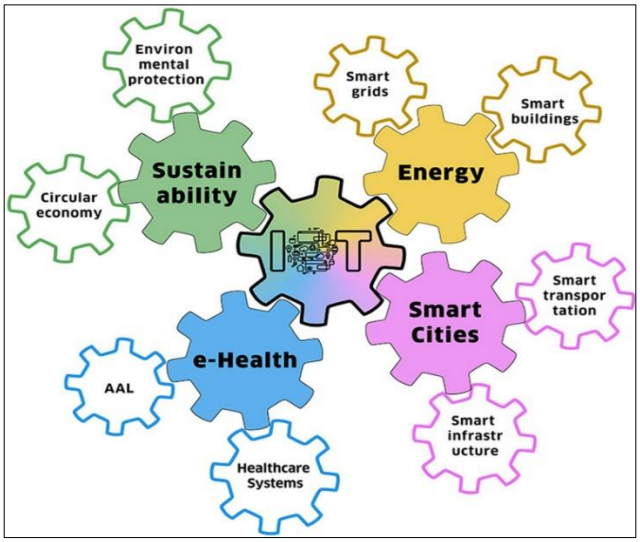


Figure 1. Various applications of IoT [9]

Consider the IoT application of refrigerated sensors as a simple example. When these sensors detect that the volume of water inside the water tank is less than a certain amount, they try to inform the person inside the house that the water bottle is empty. For this purpose, these sensors send a search message to all the sensors inside the house. The IoT can be used to track disease, which has grown in popularity in recent years, especially due to Covid-19 pandemic. By following the patient and knowing who the patient has dealt with, the disease can be controlled and its spread prevented. In more complex and important IoT applications, the data collected by IoT devices can be used to predict natural disasters such as earthquakes, tsunamis, and so on.

IoT has always attracted a lot of attention in industry and research and has always been one of the most attractive and popular topics. The IoT market is projected to be in the $ 2.7 to $ 6.2 trillion range by the end of 2025 [9]. Of this amount, 41% is for medicine and health, 33% for industry, 7% for energy and the rest of the market for other IoT applications. It is noteworthy that about IoT is the limited resources of IoT devices. This resource constraint stems from low battery power, low computing power, and network protocols. Experts disagree on a unique model for the IoT architecture. The most commonly used architectural model is a 5-layer model, including download layer, network layer, middleware layer, application layer, and business layer. As mentioned, among the the appication of the data collected by IoT devices is the intelligent traffic light system, which is the focus of this study.

2.2. Fog processing

Although cloud computing has been an efficient way to process and store data, challenges such as increased demand for real-time or latency-sensitive applications and applications with limited network bandwidth still cannot be solved using cloud computing. In recent years, on the other hand, the advent of the IoT has created a ubiquitous connection between all ubiquitous devices, resulting in the unprecedented production of huge and heterogeneous volumes of data, known as explosions, as a model for supporting time-sensitive demands. A new computing called fog computing has been introduced. Cloud computing extends cloud services to the edge of the network, bringing computing, communications, and storage closer to edge devices and end users, with the goal of increasing mobility, network bandwidth, security, privacy, and reducing latency. Figure 2 shows the fog processing structure.

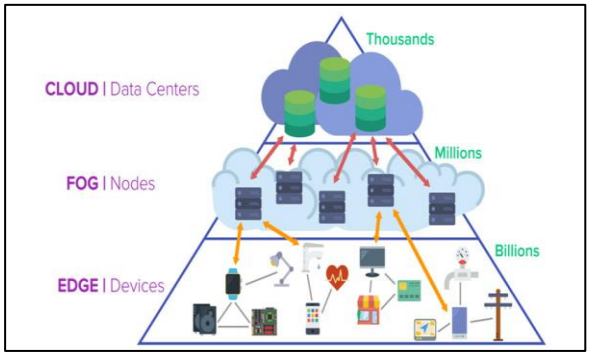


Figure 2. Distributed architecture fog processing along with cloud processing [10]

Cloud computing has three key aspects: infrastructure as a service, platform as a service, and software as a service. It has such inherent advantages as resilience and scalability, but still face many challenges. SLA service level agreements may also limit processing in locations where the cloud provider does not have data centers. Fog Computing is a new paradigm for addressing such issues, enabling the provision of out-of-cloud resources and services at the edge of the network, near end devices, or in locations specified by SLAs. Fog Computing is not a replacement for cloud computing, but a powerful complement that enables edge processing while still allowing cloud interaction. Fog computing, as an emerging computational model that has not yet been implemented, improves cloud and edge performance by extending the cloud to the edge of the network. In this training, the basic concepts of fog are introduced, and then its potential applications are pointed out, and in the end, the basic training is concluded by presenting the challenges ahead in fog computing.

3. Revew of related works

Given the importance of intelligent traffic light systems, much work has been done in this field in industry and academia. Early works used fuzzy logic and linear programming, in which a small amount of data was considered and could not be used on a large scale. Deep learning was then used to solve problems related to the dynamic and adaptive control of traffic lights [11].

Sharif et al. [1] tried to provide a dynamic traffic light management control system that works efficiently and inexpensively. They used IoT, image processing, Infrared sensors and Raspberry-Pi in their proposed system. Their proposed system works in such a way that the more crowded the side, the green on that side stays green longer.

Kuppusamy et al. [4] proposed a traffic light control system based on IoT and genetic algorithm. They used IoT devices to collect data and genetic algorithms to optimize vehicle density. Their evaluations were performed using the Arduino uno kit and the results of the evaluations showed that their proposed system is able to reduce the waiting time, latency, and congestion compared to the normal traffic system.

Wu et al. [12] introduced a traffic light control (TLC) algorithm at several intersections in real time. Their algorithm is a multi-agent recurrent deterministic policy gradient (MARDDPG) algorithm. Each traffic light controller does not operate alone at each intersection and is able to anticipate the policies that will be adopted by traffic light controllers at other intersections in order to make an appropriate traffic control decision, and thus achieving a good result. Their algorithm takes into account both the vehicle and the pedestrians who want to cross the intersection. They also took into account the waiting time of passers-by and also gave different priorities to vehicles and buses. Buses were given a higher priority due to the large number of passengers. Their simulations are performed on a traffic simulation software called SUMO. The simulation results showed the efficiency of their proposed algorithm in reducing congestion of vehicles and pedestrians [12].

Kai et al. [13] used deep reinforcement learning (DRL) to intelligently control the duration of each traffic light. Their proposed algorithm takes the current state of traffic as input. They determined reward as the total vehicle delay between actions. Their evaluations are performed on the SUMO v 0.22 simulator. The simulation results showed that their proposed algorithm is able to reduce the overall latency, queue length and travel time by 82%, 66% and 20%, respectively, compared to STSCA. One of the limitations of their work is ignoring how fair their traffic signal controller is.

Wei et al. [14] proposed several algorithms for traffic light scheduling problems using DRL. They used deep stacked autoencoders (SAE) neural networks to implement the Q-function. To evaluate their proposed approach, they compared criteria such as traffic latency and queue length, and compared their proposed approach with the conventional DRL algorithm. The results of their simulations show that in the tested scenarios, their proposed approach is able to reduce the delay by 14% and also the length of the queue generated by their proposed approach is shorter than the method. Their approach is able to design traffic light timing, give better results.

Their algorithm takes state as input. State is defined as the state of the environment, which indicates the state of traffic and the phase of the traffic light. Their algorithm takes as a reward the number of vehicles and their waiting time. Their evaluations were performed on the SUMO simulator and they used reward, queue length, delay, etc. as evaluation parameters. They compared their proposed algorithm with other tasks such as fixed time control and several other approaches to synthetic data and real-world datasets. The simulation results showed that their proposed algorithm performs better than the three compared methods in terms of all evaluation parameters. However, they had limitations in their approach, for example, they considered two-phase traffic lights. Moreover, the intersection they considered was very simple and different from the real world.

Wang et al. [15] introduced a DRL algorithm to control traffic lights. Their algorithm is able to obtain the features needed to adjust the traffic light and learn the proper procedure. They used SUMO tool for debugging and tested their proposed algorithm based on such criteria as algorithm stability, latency, and vehicle waiting time. The results of the evaluations showed that their proposed algorithm is able to reduce the delay by 47% compared to the longest queue algorithm and by 86% compared to the fixed time control algorithm. In the reward function of their algorithm, items such as queue length, vehicle latency, and number of vehicles are considered. Their algorithm takes the number of vehicles as input. Their tests were performed on VISIM and ChainerRL. They compared their proposed algorithm with two controllers. The results of the evaluations showed that the pre-timed controller cannot perform well in the face of a sudden increase in traffic, although the fully-actuated controller performs better and their proposed algorithm performs the best.

Zhu et al. [16] proposed an algorithm using Multi-Agent DRL (MADRL) to solve the traffic light control problem. They used 3DQN method in their algorithm. Their algorithm takes the state as input, which includes the current traffic mode and the current signal phase. They defined the reward function as minimizing the overall delay of a vehicle. They performed their evaluations on Aimsun Next 8.2.3 simulator on a real-world scenario in Florida. They compared their algorithm to the real-world benchmark. The copared results showed that their algorithm works better in terms of travel time and latency.

Zhou et al. [17] used traffic flow prediction techniques and optimized the duration of traffic lights and combined them to propose an adaptive traffic light control system. Their ultimate goal was to reduce and minimize the number of vehicles waiting at the intersection and increase the operational capacity of the intersection. Neural network was used to predict traffic flows. The results of evaluations and comparisons on real-world data sets showed that their proposed system is able to achieve good performance and its efficiency is better than the compared methods.

In [18], a dynamic ITS traffic light control system based on a distributed fog architecture was proposed. In this architecture, using a wireless sensor network, the local gateway collects real-time local traffic data at each intersection. It only considers adjacent lane data from the nearest fog node, which requires very little data exchange, and therefore also provides intersection coordination and traffic flow management. It also receives adjacent traffic data from distributed fog nodes. The proposed efficient traffic light dynamic control architecture calculated the optimal green light sequence and duration for multiple intersections using local and global traffic data. The advantages of this architecture are reducing waiting time and fuel consumption, and increasing power.

In [19], a distributed multi-agent ITL was proposed to solve the real-time control problem of traffic flow data. The proposed ITL is based on fog computing and Q learning algorithm. Moreover, the proposed ITL used hash collision and Diffie-Hellman-based method to maintain security. The CDH puzzle-based ITL security control scheme is less efficient when the vehicle density increases, while the hash-collision puzzle-based scheme is very friendly to fog

equipment. As a result, the proposed ITL control method based on fog computing and Q learning algorithm can effectively reduce traffic congestion. So, the proposed method has high security. The results of the simulation showed the superiority of the proposed ITL in terms of delay time and vehicle durability.
Table 2 summerizes the characteristics of all related works.

4. Proposed method

The proposed approach consists of the following phases: In the first phase, data is collected using IoT, and then pre-processing and data transmission steps are performed by Fog. In the second phase, problem is defined to explain a solution for it. In the third phase, the advantages of using fog processing are presented to determine the advantages of using fog computing. In the fourth phase, improved k-nearest neighbor algorithm is used to find the distance between vehicles based on the shortest vehicles. In the next section, Euclidean distance with Gaussian weight are given to predict the nearest vehicle.

4.1. Phase 1: Fog processing

Fog processing is a platform that is at the edge of the network and provides powerful services and applications for users. Compared to cloud computing, fog processing is closer to users and therefore faster data collection and dissemination in the sensor network are done. It also helps cloud computing to perform tasks such as preprocessing and data collection. Although fog processing and cloud processing are complementary, fog processing is a new type of distributed processing structure used for the IoT. Its flagship features include intelligent detection, high performance, low latency, real-time data transfer, online data analysis, and interaction with the cloud environment. Figure 3 shows the structure of fog processing.

Table 2. The recent works done on traffic light control

Research	Approach	Metrics	Simulation tool	Evaluation & Results
[1]	DRL & MDP	Waiting time	SUMO	Good results compared to fixed time, DQN, and ATSC
[4]	Genetic algorithm & IoT	Staying time & delay	Arduino Uno kit	Better results compared to the normal traffic system
[12]	DRL	Total delay, queue length, travel time	SUMO	Total delay, queue length and travel time reduction by 82%, 66% and 20%, respectively, compared to STSCA
[13]	DRL	Vehicle delay & waiting time	SUMO	Delay reduction by 47% compared to the longest queue first algorithm and by 86% compared to the fixed time control algorithm
[14]	DRL	Delay & queue length	VISIM & ChainerRL Library	Better performance compared to pre-timed controller and fully-actuated controller
[15]]	DRL	Delay, queue length, vehicle speed, number of stops	SUMO	Better than fixed time approach in terms of all metrics, better than actuated control approaches in all metrics except the number of stops
[16]	DRL	Delay & queue length	SUMO	Better than other three compared methods in terms of all metrics. Their limitations include two-phase traffic lights and very simple intersection that was different from the real world
[17]	MADRL & 3DQN	Delay & travel time	Aimsun Next 8.2.3	Better performance compared to the real-world benchmark
[18]	Fog-based distributed architecture	Average waiting time, Minimizing the fuel consumption and maximizing the throughput	SUMO	The simulation results show that the proposed EDTLCM minimizes the waiting time, reduces fuel consumption, and improves system throughput compared to the other dynamic strategies.
[19]	Computational Diffie–Helleman (CDH) and Hash Collision	Regulation efficiency, Average throughput, and Time to solve CDH puzzles (ms)	VISSIM-Excel VBA-MATLAB simulation platform	The results of this study may provide a theoretical basis for reducing traffic congestion and improving the efficiency and safety performance for data dispatch of intelligent TF.

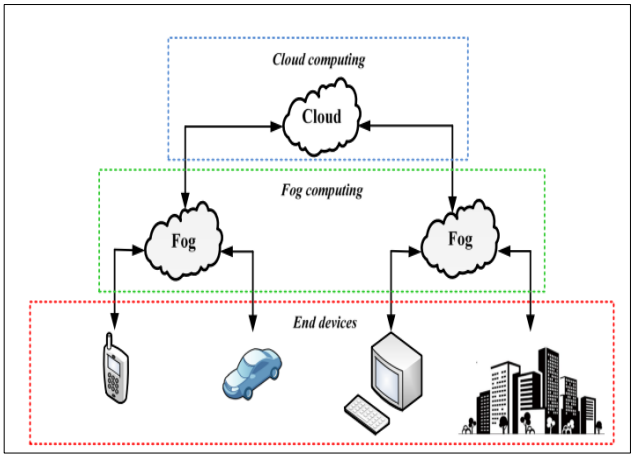


Figure 3. Distributed architecture fog processing along with cloud processing

Fog processing is based on mobile communications, wireless communication, and the Internet. As can be seen in Figure 3, fog processing is related to cloud processing, and a TCP/IP connection is required for this connection. Bbecause fog processing layer interacts with terminal equipment, it needs to support IoT protocols and heterogeneous equipment. Fog processing is the interface between the IoT and cloud computing servers.

4.2. Phase 2: Problem defination

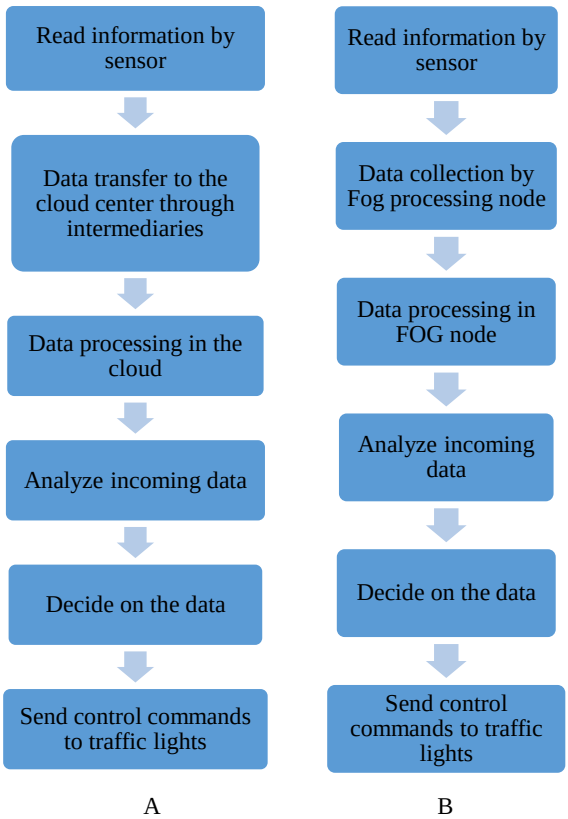
With the rapid rise of the Internet of Things, interconnected devices have expanded. Therefore, the need to reduce service delays has increased. The intelligent traffic control system consists of several components that perform real-time data collection and processing, traffic monitoring, and traffic light control. In this study, we provide a method that collects traffic data on the roads with IoT equipment and makes appropriate decisions in a timely manner to control traffic lights to reduce congestion and improve urban traffic. This method tries to respond faster and improve performance by combining fog processing and cloud processing. For monitoring and preventing traffic jams, the data stream generated by sensors in the city is used. In this method, sensors located on city roads collect the data about the number and speed of vehicles passing through each street and send it to the collecting nodes. The data collection nodes send the data to the fog processing node. The dta collected over a period of time is processed by the fog processing node and decisions are made based on the data about the amount of traffic and congestion, and based on these decisions, traffic lights in the area are controlled and scheduled. These decisions are made and enforced by the traffic controller. The dta received in the fog processing node for cloud processing services is also sent to the central node via the Internet. In the cloud processing center, the collected dta is stored in the form of bulk data and then used by machine learning algorithms to generate decision conditions and criteria. These criteria and conditions are used by fog processing nodes to make decisions about traffic light control.

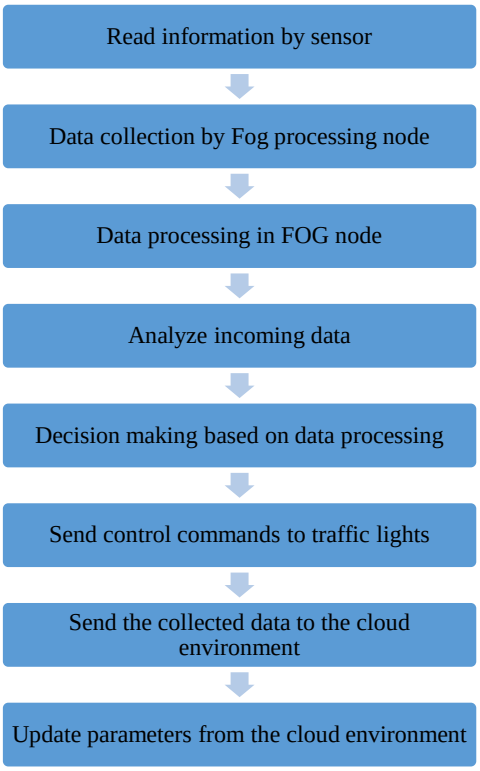
4.3. Phase 3: Presenting the advantage of using fog processing

If cloud computing is used without fog processing, traffic data collected by road surface nodes should be sent through interfaces to the cloud processing center, where large data received from all nodes is are processed and decisions made for traffic control nodes are sent to traffic lights.

If distributed fog processing is used without the use of cloud computing, the traffic data collected at the fog processing nodes are processed and decisions are made to control the traffic lights based on the parameters specified by the operator. This approach has less latency than the cloud processing method due to the locality and proximity of the processor node to the executable environment. However, due to the limited storage and processing capacity of the nodes, fog processing nodes cannot use the data history or data of many nodes to make decisions based on different conditions. However, if fog processing nodes are used as the interface between the IoT for data collection and traffic control and the cloud processing layer, instantaneous decisions based on data from the same geographic area are made in real time and faster respond to traffic changes are done. On the other hand, the computing load is reduced from the cloud processing center and the computing is done in a distributed way. Figure 4 shows the flowcharts related to cloud computing, fog processing, and the proposed method.

According to Figure 5, in the traffic light control section at each intersection, a decision is made based on the amount of traffic on each street (road) connected to that intersection. Predicted traffic dta for each street in the cloud processing center is calculated based on current and previous data collected from all roads and streets, and this dta is received and updated at the fog processing node over a regular period of time. Fog processing, based on this data and traffic data received from IoT sensors, schedules the duration of each traffic light in that area.





C

Figure 4. Flowchart related to (a) cloud processing (b) fog processing (c) the proposed method

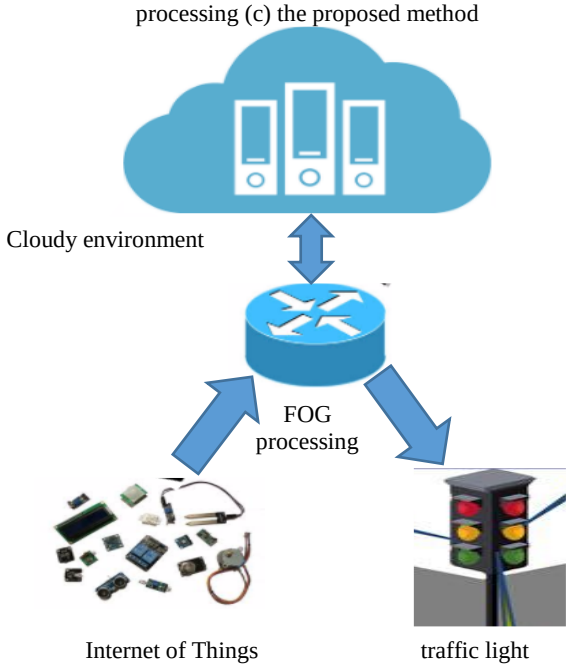


Figure 5. Method architecture provided

4.4. Phase 4: Cloud processing center

4.4.1. k-nearest neighbor algorithm improved

In the cloud computing environment, there is a collection of massive data collected from the street and urban road environments by IoT sensors. This data that shows the history of urban traffic in the form of time series is used to control traffic. The improved k-nearest neighbor algorithm is used to process this data.

The improved k-nearest neighbor algorithm consists of several steps here. First, the number and speed of the collected vehicles are preprocessed and the database is created. Then, the equivalent distances between road sections, the calculated intersection and the temporal-spatial correlation between them are obtained that the time vectors represent the traffic situation of each section. Then, the equivalent distances between the previous and current data are calculated based on the Euclidean distance with the Gaussian weight for the KNN selection.

4.4.2. Calculation of time-space situation matrix and equivalent distance

The traffic situation of different parts of the road in the road network changes based on the increase or decrease of vehicle movement. For example, a congestion or accident on a street causes one or more nearby streets to become congested in a short period of time. This study focuses on the temporal-spatial correlation of roads and streets with the collection of massive data via the IoT located on roads and streets.

Temporal-spatial correlation is done by analyzing the structure of roads, the characteristics, and the data received from them. A city or part of a city consists of several streets and roads that carry a stream of vehicles. Hence, the hierarchy of the connection of different roads is considered, as in Figure 6. The first degree (g = 1) indicates the part whose traffic should be predicted. The second degree (g = 2) is the streets that are directly connected to it. The third degree (g = 3) is the streets that are directly connected to the second-degree streets and the rest of the degrees continue in the same way.

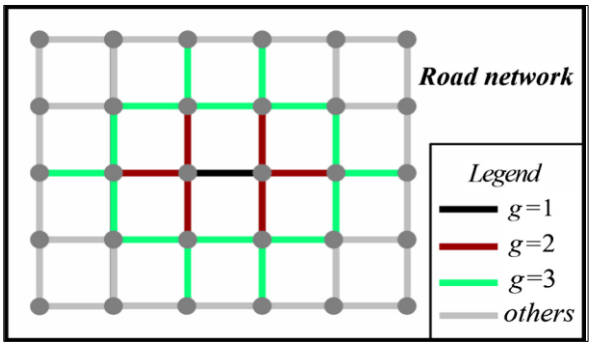


Figure 6. Relationship between adjacent roads (red and green) and the road selected for traffic forecasting (black)

The concept of distance is the physical distance of roads. Correlation coefficient is a concept that is calculated based on the time series of roads and indicates the characteristics of traffic flow. Degree of connection is an artificial concept that shows the spatial correlation of roads. In this study, the distance between roads is considered equal. Therefore, to obtain the dist distance, a combination of data and physical properties is used according to Equation 1:

$$dist = \begin{cases} (h.g)^{1-r} & g \neq 1 \\ 1 & g = 1 \end{cases} \quad (1)$$

where h is the physical distance between the roads, g represents the degree of connection of that road, and r is the correlation coefficient between the time series of the two roads, which is obtained from the collected data. If X and Y show two roads and the time series of two are respectively {x1, x2,…, xN } and {y1, y2,…, yN }, then the value of r

is obtained using Equation 2:

$$r = \frac{Cov(X,Y)}{\sqrt{\Delta X}.\sqrt{\Delta Y}} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2 \sum_{i=1}^{N}(y_i - \bar{y})^2}} \quad (2)$$

The value of dist measures the temporal-spatial correlation between the selected road and other roads. The minimum value of this variable is one and increases with increasing parameters *h*, *g* or *1-r*.

The general procedure for calculating it is as follows: First, the maximum value for *g* is selected to include a sufficient number of adjacent roads. Then, the correlation coefficient between the desired road and other roads is calculated. The value of distance is then calculated according to Equation 1, and finally the threshold value for equal distances is obtained to select the related roads.

The k-nearest neighbor algorithm uses a one-dimensional vector, but here uses a two-dimensional matrix V (m, n) for the temporal-spatial state of traffic. *m* is the length of the time series and *n* is the number of related roads. The elements of the status matrix represent the average speed at a given time. The general shape of this matrix is shown below:

$$V(m,n) = \begin{bmatrix} v_{1,1} & v_{1,2} & \cdots & v_{1,n} \\ v_{2,1} & v_{2,2} & \cdots & v_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ v_{m,1} & v_{m,2} & \cdots & v_{m,n} \end{bmatrix} \quad (3)$$

Each column of this matrix represents the traffic situation in m stages (m-1 of the previous stage) and each row represents *n* roads.

4.4.3. Euclidean distance with Gaussian weight

KNN models typically use Euclidean distances to calculate distances. However, this method is not suitable for predicting traffic alone. We use the Euclidean distance with the Gaussian weight to determine the similarity of two temporal-spatial states whose weight is determined by the bias of the Gaussian function.

The temporal weight matrix W_t and the spatial weight matrix W_s are defined and their elements are w_t and w_s.

$$W_t = \begin{pmatrix} w_{t,1} & & & \\ & w_{t,2} & & \\ & & \ddots & \\ & & & w_{t,m} \end{pmatrix}$$ که

$$w_{t,i} = \frac{1}{4\pi a_1^2} \exp\left(-\frac{|t_i - t_m|^2}{4a_1^2}\right) \quad (4)$$

Here t_m represents the current time and t_i represents the *m-i* time slope before t_m.

$$W_s = \begin{pmatrix} w_{s,1} & & & \\ & w_{s,2} & & \\ & & \ddots & \\ & & & w_{s,n} \end{pmatrix}$$ که

$$w_{s,i} = \frac{1}{4\pi a_2^2} \exp\left(-\frac{\left|dist_j\right|^2}{4a_2^2}\right), j \in [1,n] \quad (5)$$

$Dist_j$ is the distance between the *j* road and the road whose traffic will be forecast. Parameters a_1 and a_2 are also derived from real data for optimization.

If V and V_p (m, n) represent the current state and the *p* state of the previous state, respectively, then the degree of similarity between the two states is obtained by calculating the Euclidean distance with the Gaussian weight according to Equation 6:

$$SD_p = \left\|W_t V W_s - W_t V_p W_s\right\|_2 = \left\|W_t(V - V_p)W_s\right\|_2 \quad (6)$$

In the last step, KNN model is used to select the next road condition forecast. Here, by using the weighted average, more accuracy in predicting road traffic based on the previous condition and adjacent roads is obtained. In this study, λ_q according to Equation 7 is used to normalize the weights.

$$\lambda_q = \frac{1}{4\pi a_3^2} \exp\left(-\frac{|SD|^2}{4a_3^2}\right) \quad (7)$$

Here a_3 is the regulator parameter and is derived from real data. The final prediction for time (t_m + l) is calculated according to Equation 8:

$$F_{t_{m+l}} = \frac{\sum_{s=1}^{k} v_{q,t_{m+l}}.\lambda_q}{\sum_{s=1}^{k}\lambda_q}, q \in [1,k] \quad (8)$$

where $F_{t_{m+l}}$ is the result of the traffic prediction and $v_{q,t_{m+l}}$ is the predicted velocity *q* of the nearest neighbor at several time intervals before the current time.

5. Simulation and Results

This idea has been implemented and simulated using MATLAB. To get the results, the implementation was done on a computer with an i7-10750 processor and 16 GB of main memory and 1 TB of external memory. The simulation was performed with the parameters of Table 3.

Table 3. Simulation parameters

Value	Parameter name
30	Number of streets and roads
600 pcs	Number of vehicles
Between 30 and 60 km / h	Speed of vehicles
120 min	Simulation time
5 min	time slice

5.1. Evaluation criteria of cloud processing department

In the cloud processing section, using the improved k-nearest neighbor machine learning algorithm, it is tried to predict the traffic of each road in the next time slice and send this data for use in the fogging layer to make a decision in controlling the traffic lights.

The efficiency of the model used to predict traffic is calculated by two criteria: The percentage of absolute mean error (MAPE) and the mean square error (RMSE). MAPE, which is an important criterion, indicates the relative error of the model, while RMSE indicates significant errors in the specific time cut that exacerbate the problem [20, 21].

$$MAPE = \frac{1}{M}\sum_{i=1}^{M}\frac{\left|F_{t_m+1,i} - v_{t_m+1,i}\right|}{v_{t_m+1,i}} \quad (9)$$

$$RMSE = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\left|F_{t_m+1,i} - v_{t_m+1,i}\right|^2} \quad (10)$$

where M is the number of roads predicted (here 30), $F_{t_m+1,i}$ is the amount of traffic predicted for the road, and $v_{t_m+1,i}$ is the actual amount of traffic for that road after a time slice from t_m.

Tables 4, 5 and 6 show the compared data of RMSE, MAPE, and total average error vs. number of time cuts used to predict, and number of vehicles. These tables show the superiority of the proposed method over the previous three methods.

Figure 7 (A, B, and C) shows the percentage of absolute mean error (MAPE), Mean Square Error (RMSE), and Total Average Error for the algorithms KNN, SVM and GW-KNN for different modes of the number of sections used for prediction. As shown in the diagram, the proposed method (GW-KNN method) has less error in predicting road traffic than other methods. The reason is the use of Gaussian weights, which makes the accuracy of traffic forecasting closer to reality. Examination of this chart shows that this method improves by 32% in MAPE, 23% in RMSE, and 26% in Total Average Error. Another reason for the superiority of the proposed method over the other three methods is the use of Euclidean distance with Gaussian weight to determine the similarity of two time-space situations whose weight is determined based on the bias of the Gaussian function. KNN and SVM models usually use Euclidean distance to calculate distances, which is not suitable for traffic prediction alone. The next reason for the superiority of the proposed method over the other three methods is the prediction of traffic in different time spaces, which sends this predicted data to be used in fog processing layer to make decisions in controlling traffic lights. Therefore, using this algorithm to predict urban road traffic is very accurate.

Table 4. Number of time cuts used to predict vs MAPE

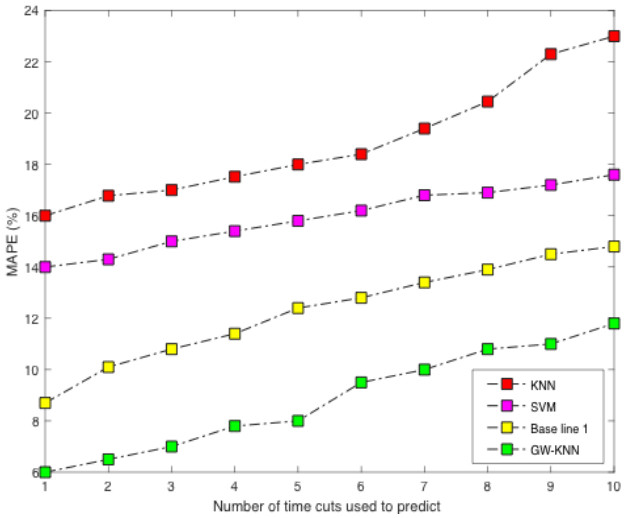
Number of time cuts used to predict	KNN	SVM	Base line 1	GW-KNN
1	16	14	8.7	6
2	16.78	14.3	10.1	6.5
3	17	15	10.8	7
4	17.52	15.4	11.4	7.8
5	18	15.8	12.4	8
6	18.4	16.2	12.8	9.5
7	19.4	16.8	13.4	10
8	20.45	16.9	13.9	10.8
9	22.3	17.2	14.5	11
10	23	17.6	14.8	11.8

Table 5. Number of time cuts used to predict vs RMSE

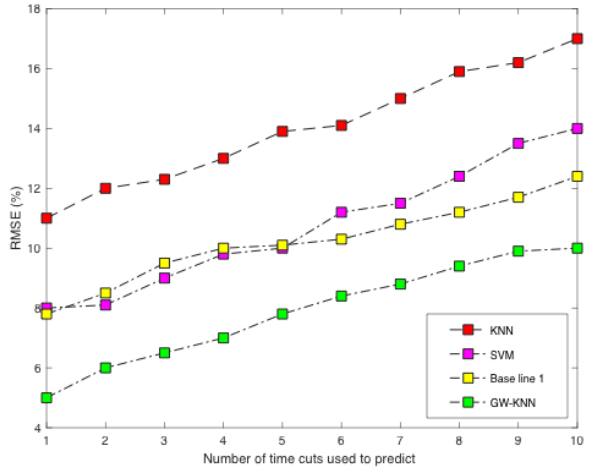
Number of time cuts used to predict	KNN	SVM	Base line 1	GW-KNN
1	11	8	7.8	5
2	12	8.1	8.5	6
3	12.3	9	9.5	6.5
4	13	9.8	10	7
5	13.9	10	10.1	7.8
6	14.1	11.2	10.3	8.4
7	15	11.5	10.8	8.8
8	15.9	12.4	11.2	9.4
9	16.2	13.5	11.7	9.9
10	17	14	12.4	10

Table 6. Number of vehicles vs total average error

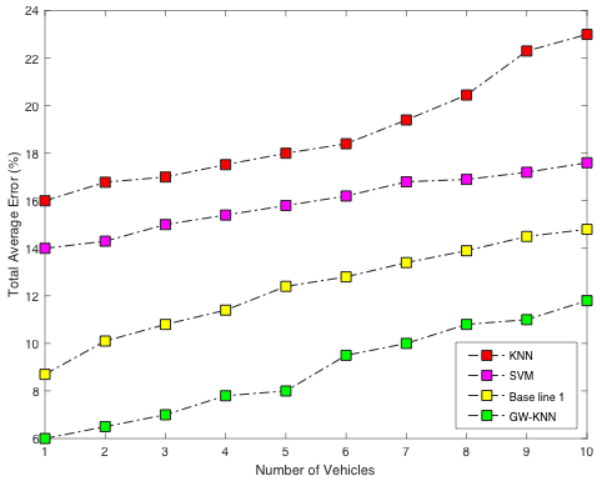
Number of Vehicles	KNN	SVM	Base line 1	GW-KNN
1	21	18.1	12.5	8
2	21.78	18.3	13.5	10.4
3	22.5	19.2	13.9	11.2
4	23.4	20.3	14.2	12
5	24.6	20.9	14.5	12.4
6	25.8.	21.8	14.9	12.7
7	26.1	22.9	15.4	12.9
8	28.4	23.8	15.8	13.4
9	28.9	24.5	16.5	13.5
10	30.2	25.8	16.7	13.9



A. Number of time cuts used to predict vs MAPE



B. Number of time cuts used to predict vs RMSE



C. Number of vehicles vs total average error

Figure 7. MAPE, RMSE, and total average error

5.2. Evaluation criteria of the presented method

5.2.1. Average vehicle waiting time

An important criterion used in intelligent traffic control systems is the average waiting time for vehicles at intersections and traffic lights. For this section, for each vehicle, we calculate the waiting time at each intersection in seconds and calculate the average total waiting time for all vehicles in the simulated time.

Tables 7 and 8 present the compared data of number of time cuts used to predict vs RMSE and number of vehicles vs total average error. These tables show the superiority of the proposed method over the previous methods.

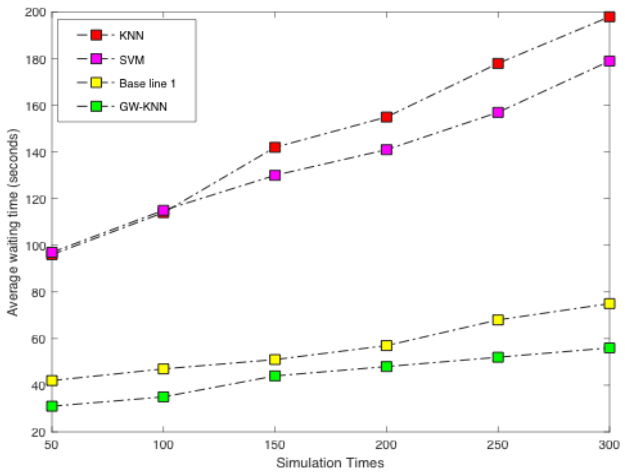
Table 7. Number of time cuts used to predict vs RMSE

Number of Vehicles	KNN	SVM	Base line 1	GW-KNN
100	195	200	98	68
200	214	220	115	73
300	248	235	148	81
400	268	248	158	92
500	289	261	169	105
600	310	286	172	110

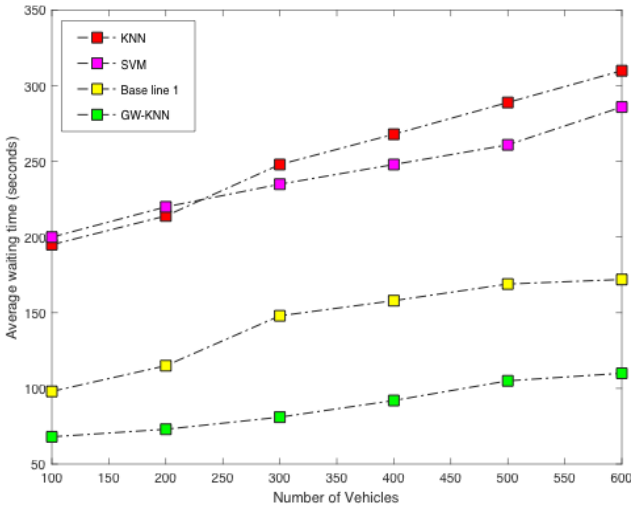
Table 8. Number of vehicles vs total average error

Simulation Times	KNN	SVM	Base line 1	GW-KNN
50	96	97	42	31
100	114	115	47	35
150	142	130	51	44
200	155	141	57	48
250	178	157	68	52
300	198	179	75	56

As it is clear from the graphs in Figure 8 (A and B), the average waiting time of the proposed method in two cases is compared with the three methods KNN, SVM, and Base line 1. Both KNN and SVM methods were performed in the mode without fog processing of the traffic control system (traffic lights) by the cloud section. As shown in Figure 8, the proposed method obtains less waiting time for vehicles than KNN and SVM. The reason for that is the use of the traffic prediction algorithm based on the data collected by the Internet of Things and then processing in the cloud environment in the form of a large set of related data. For both KNN and SVM methods, since there is no traffic forecast, decisions are made based on the traffic of the same roads connected to the intersection, and therefore the waiting time for vehicles at busy intersections increases. Moreover, in the Base line 1 method, the delay in receiving and sending data and the absence of a local data filtering section (located in the fog processing node) lead to delayed decision making and reduced data convergence. Therefore, traffic management is somewhat weaker than the presented method. In this section, the results of using the method presented in the simulation showed that this method reduces the average waiting time of vehicles. Therefore, the proposed method has 42%, 32%, and 25% superiority over all three KNN, SVM, and Base line 1 methods in the average waiting time criterion.



A. Number of time cuts used to predict vs RMSE



B. Number of Vehicles vs Total Average Error

Figure 8. Average vehicle expectation in simulation

6. Conclusion

To overcome the problems that occurred due to inefficient traffic light control systems, this paper proposed a method for intelligent traffic light control systems using Fog processing. Our proposed system used the concepts of Internet of Things, Big Data, Fog Processing. To evaluate the efficiency of the proposed method, such criteria as forecast error including mean square error (RMSE) and mean absolute error percentage (MAPE) as well as the average waiting time of vehicles were used. The results of extensive simulations showed high accuracy of our prediction model up to 96% and efficiency of our proposed algorithm compared to previous models. Our proposed algorithm is able to obtain the shortest waiting time among the compared methods.

7. References

[1] Sharif, A., et al. "Internet of things—smart traffic management system for smart cities using big data analytics", in 2017 14th international computer conference on wavelet active media technology and information processing (ICCWAMTIP). 2017.

[2] Genders, W. and S. Razavi, Using a deep reinforcement learning agent for traffic signal control. arXiv preprint arXiv:1611.01142, 2016.

[3] Saifuzzaman, M., N.N. Moon, and F.N. Nur. IoT based street lighting and traffic management system. in 2017 IEEE region 10 humanitarian technology conference (R10-HTC). 2017.

[4] Kuppusamy, P., et al., Design of smart traffic signal system using internet of things and genetic algorithm, in Advances in Big Data and Cloud Computing. 2018, Springer. p. 395-403.

[5] Seliem, M., K. Elgazzar, and K. Khalil, Towards privacy preserving iot environments: a survey. Wireless Communications and Mobile Computing, 2018. 2018.

[6] Mohammadi, M., et al., Deep learning for IoT big data and streaming analytics: A survey. IEEE Communications Surveys & Tutorials, 2018. 20(4): p. 2923-2960.

[7] Xia, J., et al., Improving random forest with ensemble of features and semisupervised feature extraction. IEEE Geoscience and Remote Sensing Letters, 2015. 12(7): p. 1471-1475.

[8] Ma, M., et al. Design and analyze the structure based on deep belief network for gesture recognition. in 2018 Tenth international conference on advanced computational intelligence (ICACI). 2018.

[9] Smolensky, P., Information processing in dynamical systems: Foundations of harmony theory. 1986, Colorado Univ at Boulder Dept of Computer Science.

[10] Taneja, M. and A. Davy. Resource aware placement of IoT application modules in Fog-Cloud Computing Paradigm. in 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM). 2017.

[11] Gao, J., et al., Adaptive traffic signal control: Deep reinforcement learning algorithm with experience replay and target network. arXiv preprint arXiv:1705.02755, 2017.

[12] Wu, T., et al., Multi-agent deep reinforcement learning for urban traffic light control in vehicular networks. IEEE Transactions on Vehicular Technology, 2020. 69(8): p. 8243-8256.

[13] Arulkumaran, K., et al., Deep reinforcement learning: A brief survey. IEEE Signal Processing Magazine, 2017. 34(6): p. 26-38.

[14] Wei, H., et al. Intellilight: A reinforcement learning approach for intelligent traffic light control. in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2018.

[15] Wang, B., et al., Deep Reinforcement Learning for Traffic Light Timing Optimization. Processes, 2022. 10(11): p. 2458.

[16] Zhu, R., et al., Multi-Agent Broad Reinforcement Learning for Intelligent Traffic Light Control. Information Sciences, 2022.

[17] Zhou, M., X. Qu, and X. Li, A recurrent neural network based microscopic car following model to predict traffic oscillation. Transportation research part C: emerging technologies, 2017. 84: p. 245-264.

[18] Hossan, S. and N. Nower, Fog-based dynamic traffic light control system for improving public transport. Public Transport, 2020. 12(2): p. 431-454.

[19] Qin, H. and H. Zhang, Intelligent traffic light under fog computing platform in data control of real-time traffic flow. The Journal of Supercomputing, 2021. 77(5): p. 4461-4483.

[20] Sepasgozar, S.S. and S. Pierre, Fed-NTP: A Federated Learning Algorithm for Network Traffic Prediction in VANET. IEEE Access, 2022.

[21] Gayathri Devi, K., et al. An IoT Based Automatic Vehicle Accident Detection and Rescue System. in International Conference on Artificial Intelligence for Smart Community. 2022.