Article

Not peer-reviewed version

An AI Enhanced Strategy of Service Offloading for IoV in Mobile Edge Computing

Peng Hong Yu * , Zhang Xiao Song , Li Hong Wu , Xu Le Xi , Wang Xiao Dong

Posted Date: 22 May 2023

doi: 10.20944/preprints202305.1460.v1

Keywords: 6G; IoV; AI; edge computing; QoS; CNN; LSTM

Article

An AI Enhanced Strategy of Service Offloading for IoV in Mobile Edge Computing

Hongyu Peng [1,*], Xiaosong Zhang [1], Hongwu Li [2], Lexi Xu [2] and Xiaodong Wang [1]

[1] Department of Artificial Intelligence, TangShan University, TangShan, China; zhangxiaosong@tsc.edu.cn (X.Z.); wangxiaodong@tsc.edu.cn (X.W.)

[2] Research Institute, China Unicom, BeiJing, China; lihw153@chinaunicom.cn (H.L.); xulx29@chinaunicom.cn (L.X.)

* Correspondence: penghongyu@tsc.edu.cn; Tel.:+86-0315-2033957

Abstract: A full connected world is expected to gain in the 6th generation mobile network (6G). As a typical fully connected scenario, the internet of vehicle (IoV) enables intelligent vehicle operations via artificial intelligence (AI) and edge computing technologies. In the future of vehicular networks, wide variety of services need powerful computing resources and higher quality of service (QoS). Existing resources are insufficient to match these requirements. Aim to this problem, An intelligent service offloading framework is provided. Based on the framework, an Algorithm of Improved Gradient Descent (AIGD) is created to accelerate the speed of iteration. So, the convergence of convolutional neural network (CNN) based on AIGD is able to be accelerated too. Then, an Algorithm of convolutional long short-term memory (CN_LSTM) Based Traffic Prediction (ACLBTP) is designed to gain the predicted number of vehicles belonged to the edge node. At last, an Algorithm of Service Offloading Based on CN_LSTM (ASOBCL) is conducted to offload these services to the vehicles belonged to the edge node. In ASOBCL, sorting technique is adopted to speed up the offloading work. Simulation results demonstrate the fact that the prediction strategy designed in this paper has high accuracy. The low offloading time and maintaining stable load balance is gained via running ASOBCL. Low offloading time means short response time. And, the QoS is guaranteed. So, these strategies designed in this paper are effective and valuable.

Keywords: 6G; IoV; AI; edge computing; QoS; CNN; LSTM

1. Introduction

A full connected world is expected to gain in the 6th generation mobile network (6G). As a typical fully connected scenario, the internet of vehicle (IoV) is selected as the research object in this paper. According to GE Digital, new technology of IoT is estimated to unlock manufacturing savings and benefit 46 percent of the global economy [1]. The Internet of Vehicles, as an important subset of Internet of Things, develops rapidly in recent times [2]. The development of IoV delivers new insights into application of IoT [3]. At the same time, the progress of IoV substantially pushes the development of intelligent transportation [4]. With the increasing growth of sensors and perception devices, more and more valuable information is able to be obtained from the surrounding environment via vehicles [5]. In the meantime, the data processing capability of on-board equipment is constantly improving with the hardware upgrade [6]. Meanwhile, IoV services (e.g., auto navigation, traffic forecast and route planning etc.) are most delay sensitive. But, the conventional computing resources are not enough to meet these real time requirements of services [7]. Advanced intelligent vehicular applications (e.g., intelligent road environment perception, intelligent decision making and vehicle behavior controlling etc.) are envisioned [8]. These intelligent vehicular applications need powerful computing processing capability, low latency and stable load balance [9].

According to the problems above, exiting works mostly discuss how to compress and reduce neural networks on edge servers, ignoring the impact of service offloading [10]. Satveerrs S et al. provide a plan on service offloading to find the best power-delay trade-off, ignoring the problem of load balance [11]. H. Liu et al. propose parked vehicle edge computing for distributed task execution, ignoring the problem of delay [12,13]. To provide high-quality information services, Z. Su et al. suggest a strategy for caching content in parked vehicles in advance, ignoring the problem of large consumption of storage space [14]. W Sun et al. suggest edge computing is able to provide distributed computing service through small-scale data centers near the edge of the network [15].

With these observations, it is challenging to achieve an intelligent computing and service offloading architecture for high QoS. At the same time, a series of strategies are need to be designed for gaining low offloading time and maintaining stable load balance.

The key contributions of this paper are able to be summarized as follows.

(1) Develop an AI-based framework to deploy these strategies designed in this paper for IoV during service offloading process.
(2) Adopt AIGD(Algorithm of Improved Gradient Descent) to improve the speed of iteration. So, the convergence of CNN based on AIGD is able to be improved significantly.
(3) Design ACLBTP(Algorithm of CN_LSTM Based Traffic Prediction) to gain the predicted number of mobile nodes.
(4) Conduct ASOBCL(Algorithm of Service Offloading Based on CN_LSTM) to offload the services. Sorting technique is adopted in this algorithm. So, the work of offloading is more efficient.

The rest of this paper is organized as follows. Section 2 presents the framework of system. The model of system is proposed in Section3. These strategies of this paper are discussed in Section 4. The simulation analysis is given in Section5. Section 6 proposes the conclusion and future works of this paper.

2. The Framework of System

In this section, the notations summary of this paper and the framework of this system are designed and expounded. Firstly, the main terms of this paper are described in Table 1.

Table 1. Summary of Notations in Problem Formulation.

Notations	Descriptions
$D^{1/2}$	the non-singular matrix
R	the Raleigh quotient
v	the given direction
H	the function of Hessian
α	the gradient descent
γ	the eigenvalues of H
η	the eigenvectors of H
S_e	the set of edge nodes
S_s	the set of services
R_j	the resource utilization of j-th EN
R_{ave}	the average resource utilizations of ENs
l_b	the load balance
θ	the parameters of function
β	the learning rate
μ	the damping factor

It can be seen from Table 1 that the descriptions of notations are summarized. Then, the framework of system is developed as follows.

As shown as Figure1, there are three layers in this framework. These layers are data perception and computing layer, edge layer and cloud layer. The data perception and computing layer are composed by those mobile nodes selected via AI strategy deployed in edge nodes. These mobile nodes are able to exchange information with each other. The information includes traffic data and road conditions, etc. These mobile nodes upload perception data to the edge nodes in the edge layer.

The strategy deployed in edge node offloads computing task to the mobile nodes. At the same, the resources are allocated to these mobile nodes via the strategy.

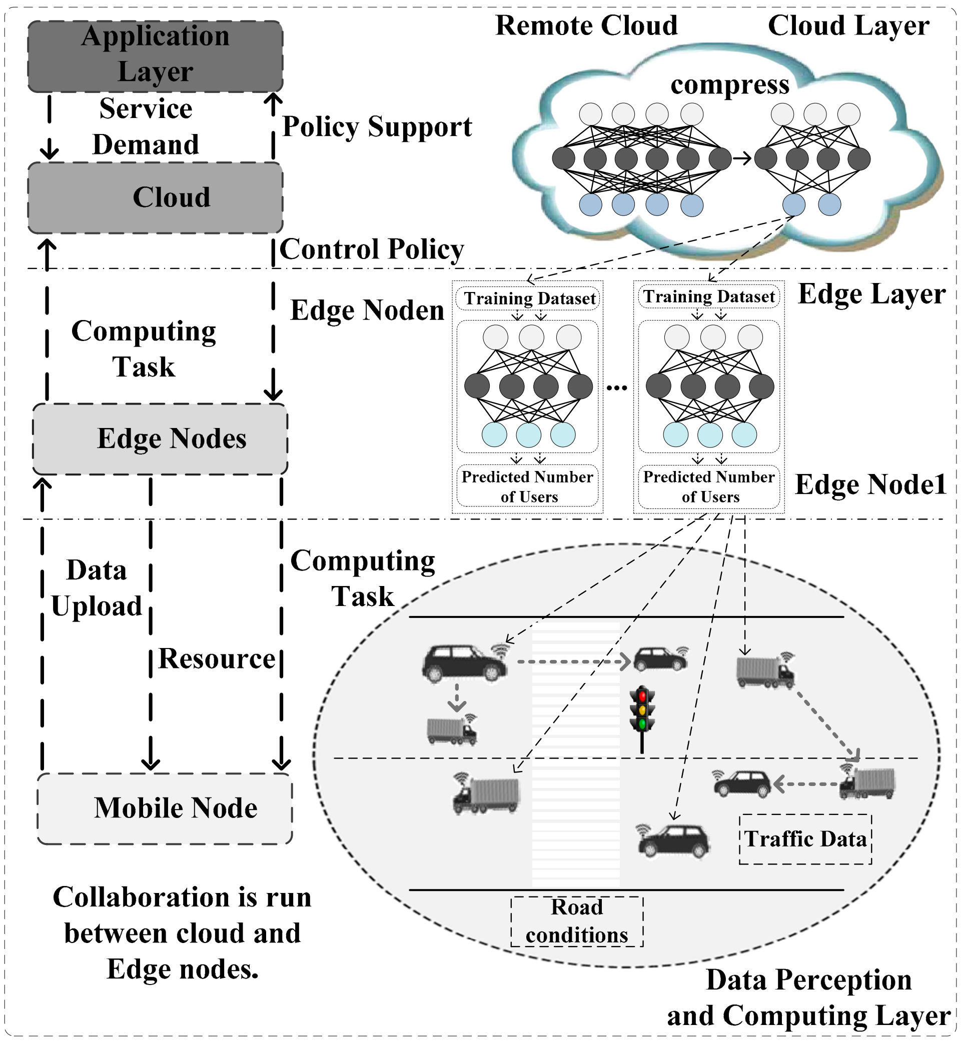

Figure 1. The Framework of System.

The edge layer is compose by edge nodes. The AI strategy designed in this paper is deployed in these edges. The AI strategy runs in the training set and the predicted number of mobile nodes is gained. These edge nodes assign hard computing task to the cloud. The cloud sends control policy back to these edge nodes. The service demands are provided to the applications in the application layer. At the same time, policy supports are given to these applications via cloud.

The collaboration is run between cloud and edge nodes. All the layers in the framework interact with each other. By analyzing the data set, edge knowledge bases are built. The knowledge is able to be used for the prediction of the traffic patterns. In this paper, It is used to predict the number of mobile edge nodes.

The models of system based on the framework above are designed in the next part.

3. The Model of System

Precondition is designed via introducing a linear change of variables.

$$\alpha = D^{\frac{1}{2}}\theta \quad \theta \in R \tag{1}$$

Where $D^{\frac{1}{2}}$ is a non-singular matrix. Then, a new function is able to be designed as

$$f(\theta)=f(D^{-\frac{1}{2}}\alpha) \tag{2}$$

The gradient of the function is expressed as follows.

$$f'(\alpha)=D^{-\frac{1}{2}}f'(\theta) \tag{3}$$

The H in this paper is defined as

$$H=f''(\theta) \tag{4}$$

The Hessian of the function is expressed as follows.

$$f''(\alpha)=(D^{-\frac{1}{2}})^T HD^{-\frac{1}{2}} \tag{5}$$

A gradient descent iteration for the transformed function is able to be gained from (3) and (5).

$$\alpha_t=\alpha_{t-1}-\lambda f'(\alpha) \tag{6}$$

According to the relationship of θ and 0, A gradient descent iteration for θ is able to be gotten as follows.

$$\theta_t=\theta_{t-1}-\lambda D^{-1}f'(\theta) \tag{7}$$

The Raleigh quotient R is defined as follows.

$$R(H,v)=\frac{v^T(Hv)}{v^Tv} \tag{8}$$

Where v is a given direction. R is used to measure the amount of curvature.
The Raleigh quotient R is able to be decomposed as follows.

$$R(H,v)=\sum_i^n \gamma_i\eta_i\eta_i^T v \tag{9}$$

Where, γ_i is the eigenvalues of H. η_i is the eigenvectors of H. So, Hv is able to be gained from (8) and (9).
The set of EN(Edge Node) is defined as follows.

$$S_e=(e_1,e_2,...e_n) \tag{10}$$

The set of services is defined as follow.

$$S_s=(s_1,s_2,...s_m) \tag{11}$$

A variable $k_{i,j}$ is given to represent whether the n-th service is executed by m-th EN

$$k_{i,j}=\begin{cases}1, \text{ i-th sercice is processed by the j-th EN}\\ 0, \text{ otherwise}\end{cases} \tag{12}$$

The resource utilization of j-th EN is calculated based on k n,m and it is given by

$$R_j=\sum_{i=1}^{m}k_{i,j} \tag{13}$$

Based on (13), the average resource utilizations of ENs R_{ave} is calculated as

$$R_{ave}=\frac{1}{n}\sum_{j=1}^{n}R_j \tag{14}$$

Based on (14), the load balance l_b is calculated by

$$l_b=\frac{1}{n}\sum_{j=1}^{n}(R_j\text{-}R_{ave})^2 \tag{15}$$

The problem of designing an effective offloading method is expressed as follows.

$$\min l_b \tag{16}$$

Then, these strategies based on the model above are designed as follows.

4. Strategy Design

In this section, there are three strategies provided in this part. These strategies are AIGD, ACLBTP and ASOBCL. Firstly, the strategy of AIGN is described as follows.

Table 2. Algorithm of Improved Gradient Descent.

```
Algorithm Realization
1: Initialization $\beta$, $\mu$
2: Initialization H to 0 matrix
3: Gain the min value of $f(\theta)$
4: foreach i in $(k,K)$
5:     get v randomly from N(0,1)
6:     $D=D+(Hv)^2$
7:     $\theta=\theta-\beta\frac{f'(\theta)}{\sqrt{D/k}+\mu}$
8: endfor
```

It can be seen from Table 2 that β is the learning rate. The μ is the damping factor. The contributions of these directions will take a large step in each direction. This will improve the speed of iteration. So the convergence speed of CNN is improved significantly.

Based on the strategy above, the ACLBTP is designed as follows.

It can be seen from Figure 2 that the CN_LSTM_Network Model of ACLBTP are composed by two convolution layers, a LSTM layer and a full connection layer. The input data is the traffic dataset. The output data from these convolution layers are put into the LSTM layer for extracting time features. The LSTM layer is composed of several LSTM blocks. The output from the full connection layer is the number of users in an area and in every time slot. The mean squared error(MSE) is adopted as the loss function of this paper [16]. MSE is used to minimize the number of network errors.

Base on the flowchart of strategy above, the algorithm's pseudo code is provided as follows.

Table 3. Algorithm of CN_LSTM Based Traffic Prediction.

Algorithm Realization

```
1: Initialization Mobile Edge Nodes Data Matrix: M, Layers: 3, Number of iteration: 2000, time
   slot:16, Number of data for each slot:32 Number of network errors: R
2: Establish Training Set: Input Data Set, Output result Set
3: Set Iteration Number n, Set Confidence Interval: MinValue
4: While(R< MinValue)
5:   {
6:     for i=1 to n
7:       {
8:         Extract Spatial Feature of Time Series Traffic Data via
             two Convolution Layers Based on AIGD.
9:         The Output Data from two Convolution Layers are
               put into LSTM Layer for extracting time feature.
10:        The Output Data from LSTM are put into a full
             connection layer and the output is the number of
              Users in each area in each time slot.
11:          i++
12:      }
13:     R=MSE(Output Data from LSTM, Input Data Set)
14:   }
15: Output Predicted Number of Users
16: End of Strategy
```

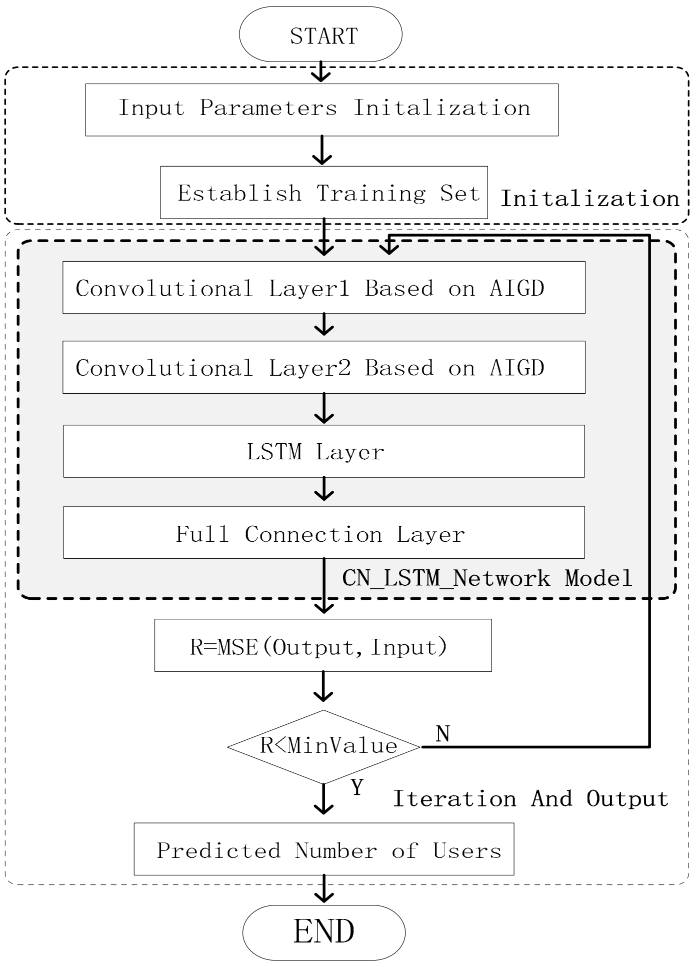

Figure 2. FlowChart for ACLBTP.

It can be seen from Table 3 that parameters are initialized and training set is established firstly. Then, the convolution iteration starts. The number of iteration is set to 2000 in this paper. In every iteration, the input data set is put into these convolution layers based on AIGD and the spatial feature of time series traffic data are extracted. The output data from two convolution layers are put into LSTM Layer for extracting time feature. The output data from LSTM are put into a full connection layer and the output is the number of users in each area in each time slot. The iteration is continued until the number of network errors is less than the confidence interval MinValue. At last, the predicted number of mobile nodes is gained.

Based on the strategy above, the strategy of ASOBCL is designed as follows.

It can be seen from Figure 3 that those mobile edge nodes gained via ACLBTP are regarded as the input data of the workflow. The n is defined as the number of these number of mobile edge nodes. The Ss is defined as the set of services. Firstly, these mobile edge nodes are sorted via (15) by load in descending order. The order of sort is to gain the mobile edge node whose load is min more quickly. One service is selected from Ss and offloaded to the mobile edge nodes whose load is min. The service which is offloaded is remove from the Ss. The work of offloading continues until the Ss is empty.

Base on the strategy flowchart above, the algorithm's pseudo code is provided as follows.

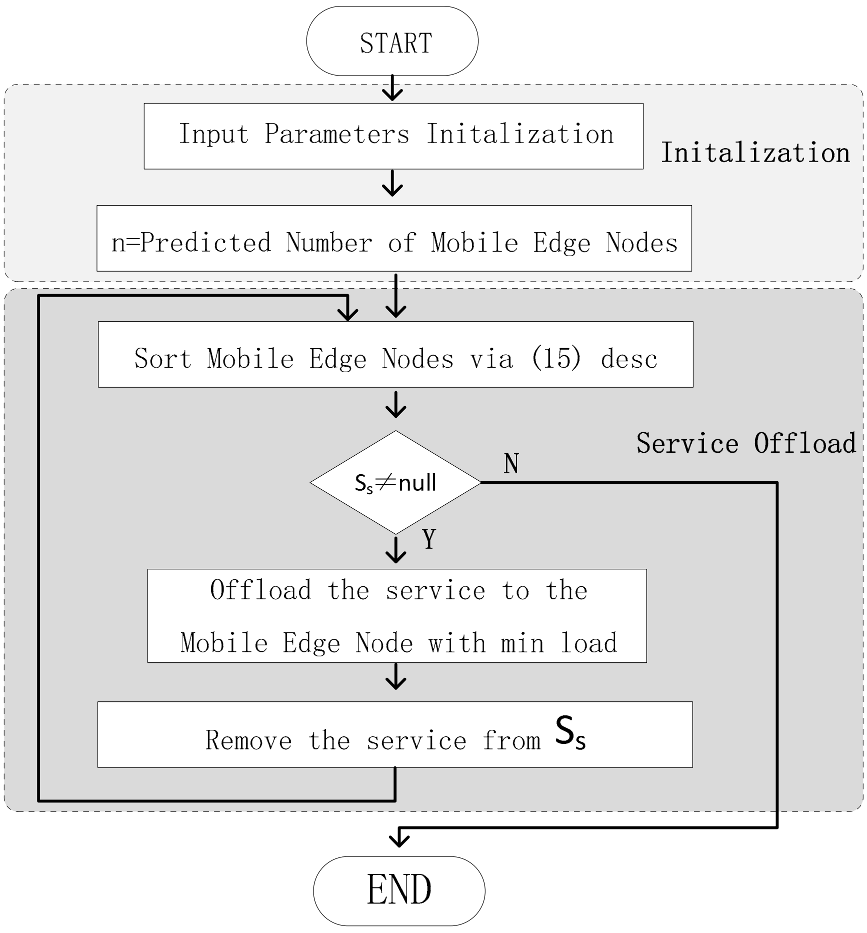
START
Input Parameters Initalization
Initalization
n=Predicted Number of Mobile Edge Nodes
Sort Mobile Edge Nodes via (15) desc
Service Offload
$S_s \neq$null
N
Y
Offload the service to the Mobile Edge Node with min load
Remove the service from S_s
END

Figure 3. FlowChart for ASOBCL.

Table 4. Algorithm of Service Offloading Based on CN_LSTM Traffic Prediction.

Algorithm Realization

```
1: Initialization Output Predicted Number of Mobile Edge
      Nodes: n, Service Set: $S_s$
2: Sort n Mobil Edge Nodes via load calculated with (15) desc.
3: While($S_s$ !=null)
4:   {
5:        Offload the service the Mobile Edge Node with min load.
6:        Remove the service from the $S_s$.
7:        Sort n Mobil Edge Nodes via load calculated with (15)
desc again
8:   }
9: End of Strategy
```

It can be seen from Table 4 that parameters are initialized. The n is initialized as the output predicted number of mobile edge nodes. The Ss is initialized as the service set. The load of every mobile edge nodes is calculated via (15). Firstly, these mobile edge nodes are sorted in descending order by the size of load. One service is select from Ss and offloaded to the mobile edge node with min load. Then, the service is removed from Ss. Those mobile edge nodes are sorted in descending by load again. These steps are repeated until the Ss is empty. The strategy comes to an end.

The simulation analysis based on these strategies are given in the next part.

5. Simulation Analysis

In this paper, the technique of lightweight machine learning is adopted on the edge nodes. Pre-trained dataset is used as the input to the new machine learning task. The adoption of pre-trained dataset is able to decrease the computing complexity significantly. The pre-trained AlexNet CNN is used in matlab. The ILSVRC 2012 is adopted in this paper. The strategy of ASOBCL is implemented in Tensorflow. According to [17], the channel is selected with parameter of 1. The noise power density is adopted as −120 dBm. The CabSpotting dataset [18] is taken as the dataset in this paper.

The performance of these strategy designed in this paper is evaluated from three aspects: the prediction accuracy, the load balance and the offloading time.

5.1. Prediction Accuracy

Figure 4 depicts the accuracy of prediction of LSTM over the varying training set size. As shown in Figure 4, the horizontal axis indicates the lapse of time. The unit is 60 seconds. The vertical axis represents the normalized number of mobile edge nodes. The blue line indicates the changing trends of real values. The red line indicates the changing trends of predicted values. When the time lapse is at around 100 minutes, the predicted value deviates significantly from the real value.

Figure 4 depicts the accuracy of prediction of ACLBTP over the varying training set size. As shown in Figure 5, the horizontal axis indicates the lapse of time, and the unit is 60 seconds. The vertical axis represents the normalized number of mobile nodes. The blue line indicates the changing trends of real values. The red line indicates the changing trends of predicted values.

Compare Figure 4 with Figure 5, we can see that the predicted values from ACLBTP designed in this paper are much closer to the real values than those from LSTM.

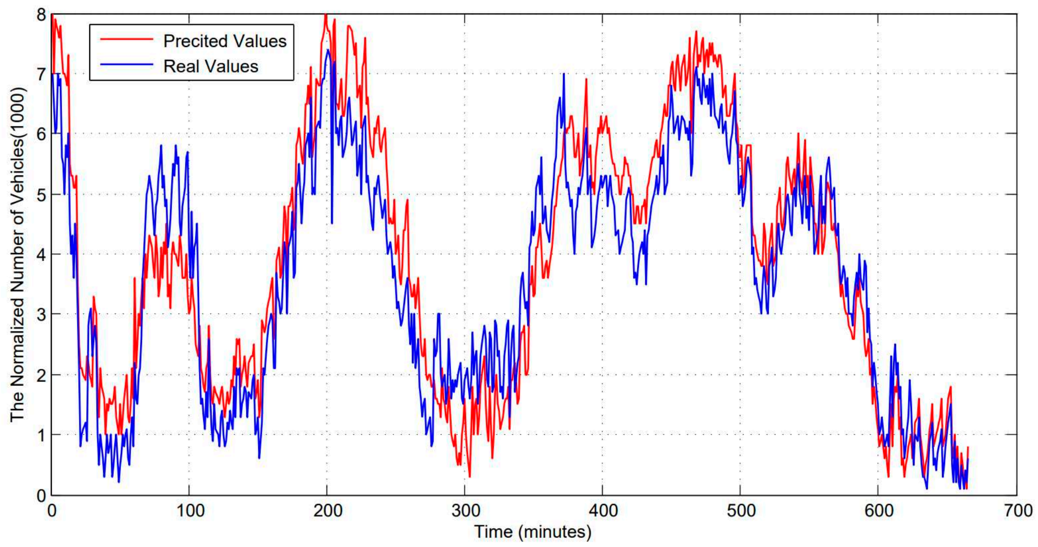

Figure 4. Comparisons of real values and predicted values from LSTM.

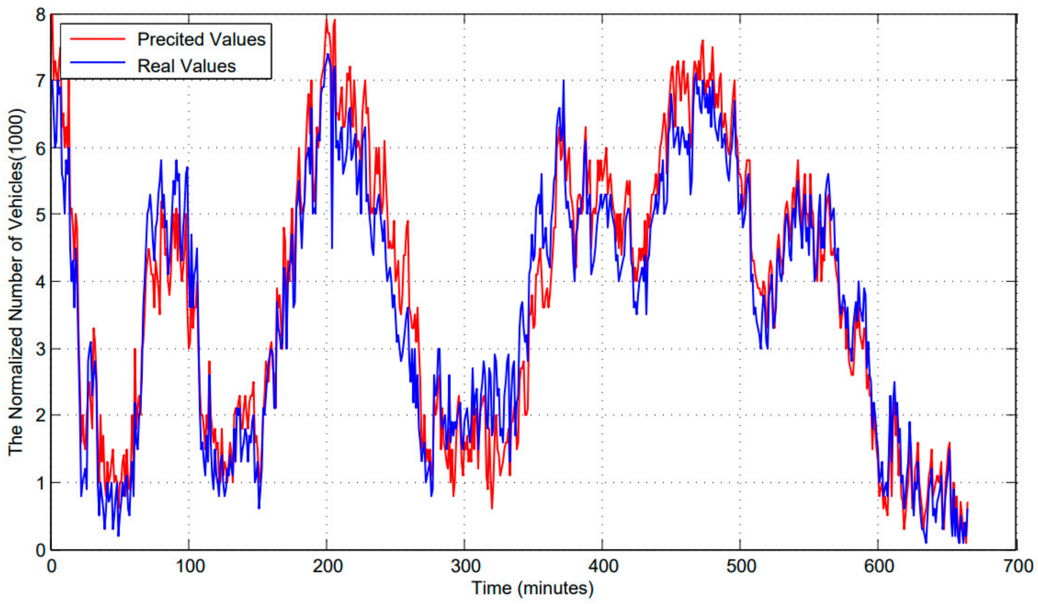

Figure 5. Comparisons of real values and predicted values from ACLBTP.

5.2. Load Balance

In this section, comparative algorithms and results analysis are proposed. Firstly, comparative algorithms are introduced as follows.

Comparative Algorithms

There are three comparative algorithms in this section, these algorithms are First Come First Service (FCFS), Next Come First Service (NCFSS) and Benchmark First Service (BCFS).

FCFS: When the service comes, the mobile edge node is selected randomly to be provided to the service in turn until the set of mobile edge nodes is empty.

NCFS: When the first service comes, the mobile edge node is selected randomly to be provided to the service. When the following service comes, the first selected mobile edge node is excluded. The mobile edge node is selected randomly in the remaining mobile edge nodes set to be provided to the service until the set of mobile edge nodes is empty.

BCFS: According to the mobile edge nodes' computing capability, these mobile edge nodes are sorted in descending order. The services are assigned to these mobile edge nodes in turn [19].

Results Analysis

In terms of load balance, simulation verification is carried out from the perspective of the number of services in this paper. And, simulation verification is introduced from the perspective of the load balance and the services' number.

Maintaining low load balance is an important goal in this paper. As shown in Figure 6, the horizontal axis indicates the number of services, and the unit is 1000. The vertical axis represents the load balance. As the number of services increases, the load balance for FCFS, BCSF, NCFS, and ASOBCL rise too. Among the four algorithms, the effect of ASOBCL is always the best, BCFS is second, NCFS is the worst, and FCFS's effect of storage reduction is better's than NCFS's and worsen than BCFS's. FCFS and NCFS occupy a handful of mobile edge nodes. This results high load balance in FCFS and NCFS. ASOBCL always maintains stable load balance rate in these mobile edge nodes because most edge mobile nodes are occupied reasonably.

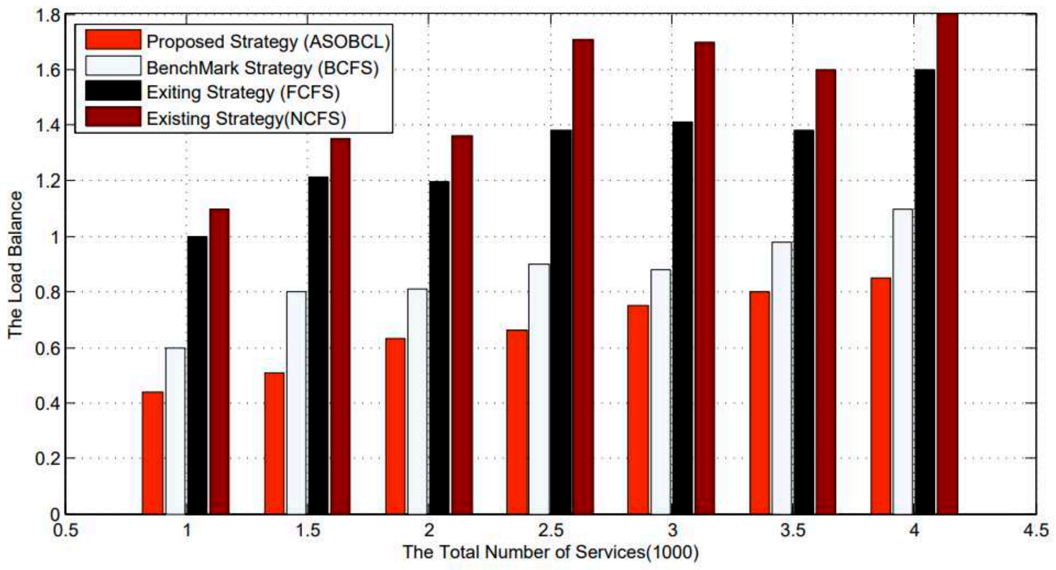

Figure 6. The number of services and load balance.

5.3. Offloading Time

In the part, comparative algorithms are adopted from the above section. In terms of offloading time, simulation verification is carried out from the perspective of the number of services in this paper. Simulation verification is introduced from the perspective of the offloading time and the services' number.

The offloading time is compared by FCFS, BCSF, NCFS, and ASOBCL. As shown in Figure 7, the horizontal axis indicates the number of services, and the unit is 1000. The vertical axis represents the offloading time, and the unit is 1000 milliseconds. As the number of services increases, the offloading time for FCFS, BCSF, NCFS, and ASOBCL rise too. Among the four algorithms, the effect of ASOBCL is always the best, BCFS is second, FCFS is the worst, and NCFS's effect is better's than FCFS's and worsen than BCFS's. The more the number of visits, the better the offloading time of ASOBCL. Compare with the other offloading methods, ASOBCL consumes low offloading time. Moreover, ASOBCL performs better than Benchmark due to the stable low offloading time.

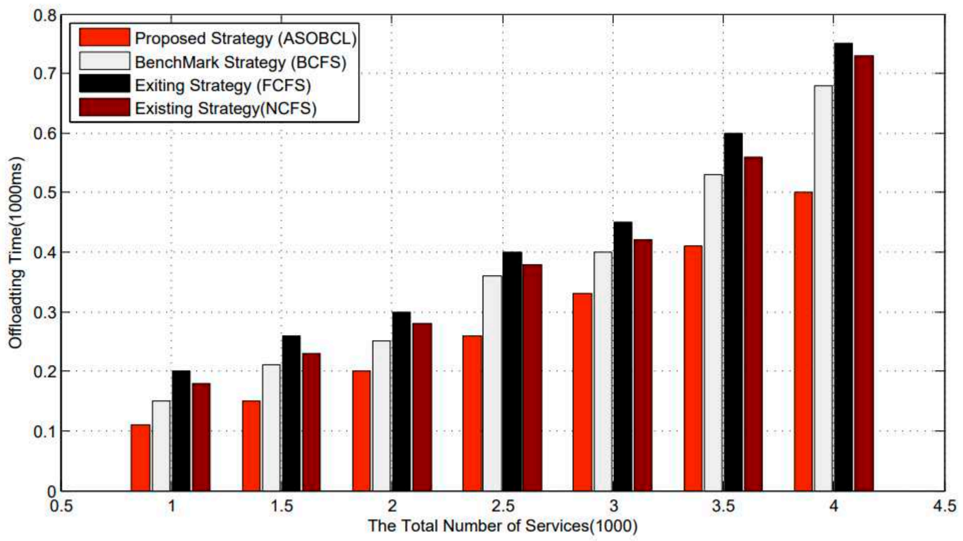

Figure 7. The number of services and Offloading Time.

6. Conclusion and Future Work

In this paper, an intelligent service offloading framework is provided firstly. Based on the framework, three strategies are proposed. The AIGD is created to accelerate the convergence of CNN. The ACLBTP is designed to gain the predicted number of vehicles belonged to the edge node. The ASOBCL is conducted to offload these services to the vehicles belonged to the edge node. Simulation results demonstrate that the prediction strategy designed in this paper has high accuracy. The low offloading time and maintaining stable load balance are gained via running ASOBCL. So, the effectiveness and efficiency of ASOBCL is verified by experiment evaluation. In our next work, we will devote to applying ASOBCL to real life, taking more real details of IoV environment into account.

Author Contributions: Conceptualization, H.P. and X.Z.; Formal analysis, H.P.; Funding acquisition, H.P.; Investigation, H.P. and X.W.; Methodology, H.P. and L.X.; Software, H.P.; Supervision, H.P.; Validation, H.P.and H L; Visualization, X.W.; Writing—original draft, H.P.; Writing—review & editing, L.X. All authors have read and agreed to the published version of the manuscript.

Funding: Southwest JiaoTong university cooperative intelligent water project.

Acknowledgment: This work was supported by Doctor Innovation fund project and Tangshan indoor positioning key laboratory construction project.

References

1. G. Gui et al., “6G: Opening New Horizons for Integration of Comfort, Security and Intelligence,” IEEE Wireless Commun.,vol. 27, no. 5, Oct. 2020, pp. 126–32.
2. GE Digital Report, “Everything You Need to Know about the Industrial Internet of Things,” 2017; https://www.ge.com/ digital/blog/everything-you-needknow-about-industrial-internet-things.
3. Liang T , Chen M , Yin Y , et al. Recurrent Neural Network Based Collaborative Filtering for QoS Prediction in IoV[J]. IEEE Transactions on Intelligent Transportation Systems, 2021, PP(99):1-11.
4. Xu X , Huang Q , Zhu H , et al. Secure Service Offloading for Internet of Vehicles in SDN-Enabled Mobile Edge Computing[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 22(6):3720-3730.
5. L. Zhou, W. Min, D. Lin, Q. Han, and R. Liu, “Detecting motion blurred vehicle logo in IoV using filter-DeblurGAN and VL-YOLO,” IEEE Trans. Veh. Technol., vol. 69, no. 4, pp. 3604–3614, Apr. 2020.
6. Wang Y , Tian Y , Hei X , et al. A Novel IoV Block-Streaming Service Awareness and Trusted Verification Scheme in 6G[J]. IEEE Transactions on Vehicular Technology, 2021, 70(6):5197-5210.
7. J. Qiu, L. Du, D. Zhang, S. Su, and Z. Tian, “Nei-TTE: Intelligent traffic time estimation based on fine-grained time derivation of road segments for smart city,” IEEE Trans. Ind. Informat., vol. 16, no. 4,pp. 2659–2666, Apr. 2020.
8. A. Mondal and S. Mitra, “Security issues in vehicular ad hoc networks for evolution towards Internet of vehicles,” in Connected Vehicles in the Internet of Things. Cham, Switzerland: Springer, 2020, pp. 253–307.

9. Mlika Z , Cherkaoui S . Network Slicing with MEC and Deep Reinforcement Learning for the Internet of Vehicles[J]. IEEE Network, 2021, PP(99):1-7.
10. Qi W, Li Q , Song Q , et al. Extensive Edge Intelligence for Future Vehicular Networks in 6G[J]. IEEE Wireless Communications, 2021, 10(99):1-8.
11. S. Wang et al., "When Edge Meets Learning: Adaptive Control for Resource constrained Distributed Machine Learning," Proc. IEEE INFOCOM, 2018.
12. G. Zhang et al., "Energy-Delay Tradeoff for Dynamic Offloading in Mobile-Edge Computing System with Energy Harvesting Devices," IEEE Trans. Industrial Informatics, vol. 14, no.10, 2018, pp. 4642–55.
13. H. Liu et al., "Parking-Area-Assisted Spider-Web Routing Protocol for Emergency Data in Urban VANET," IEEE Trans. Veh.Technol., vol. 69, no. 1, Jan. 2020, pp. 971–82.
14. X. Huang, P. Li, and R. Yu, "Social Welfare Maximization in Container-Based Task Scheduling for Parked Vehicle Edge Computing," IEEE Commun. Lett., vol. 23, no. 8, June 2019,pp. 47–51.
15. Z. Su et al., "A Game Theoretic Approach to Parked Vehicle Assisted Content Delivery in Vehicular Ad Hoc Networks," IEEE Trans. Veh. Technol., vol. 66, no. 7, Nov. 2016,pp. 64–74.
16. Sun W , Liu J , Yue Y . AI-Enhanced Offloading in Edge Computing: When Machine Learning Meets Industrial IoT[J]. IEEE Network, 2019, 33(5):68-74.
17. Y. N. Dauphin et al., "Equilibrated Adaptive Learning Rates for Non-Convex Optimization," Proc. NIPS, Dec. 2015, pp.1504–1512.
18. E. El Haber, T. M. Nguyen, and C. Assi, "Joint optimization of computational cost and devices energy for task offloading in multi-tier edge-clouds," IEEE Trans. Commun., vol. 67, no. 5, pp. 3407–3421,May 2019.
19. M. Piorkowski et al., "CRAWDAD Trace Set Epfl/Mobility/Cab (V. 2009-02-24)," http://crawdad.cs.dartmouth.edu/epfl/mobility/cab, 2009.
20. X. Xu et al., "An edge computing-enabled computation offloading method with privacy preservation for Internet of connected vehicles," Future Gener. Comput. Syst., vol. 96, pp. 89–100, Jul. 2019.