Bare Demo of IEEEtran.cls for IEEE Journals

Michael Shell, Member, IEEE, John Doe, Fellow, OSA, and Jane Doe, Life Fellow, IEEE

Abstract—With the emergence of advanced vehicular service, the challenge of meeting computation and communication needs of vehicles has become increasingly prominent. Computation offloading is a promising paradigm to .

Index Terms—Fog Computing, Computation offloading, Vehicular Network, Workload Allocation

I. Introduction

With the development of Internet of Things(IoT) and wireless technologies, vehicles become more intelligent and provide better services[1]. In this process, various complex-computing applications such as autonomous driving and image recognition are required to meet the demands of users. However, the computation limitation of vehicle terminal make it difficult to meet these demands and many advanced applications cannot be applied on vehicles.Without powerful computational support, various applications are still in the concept phase and cannot be applied in our daily life[2],[3].

Cloud computing can improve the computation performance by offloading tasks to the cloud which takes the advantage of abundant computational resource[4]. However, the long distance transmission of data to cloud has not only caused a heavy burden on communication links but also resulted in intolerable latency, which significantly degrades the QoS. To overcome the above problem, fog computing is a new paradigm extending the computation facilities to the user end [5],[6]. This paradigm is a promising method to solve the computation limitation of vehicle [7]. On the one hand, the fog device reduce the power consumption by relieving the workload of cloud. On the other hand, the geo-distributed fog devices can decrease the transmission delay. In fog computing, it is impossible to offload all tasks to the fog layer and some complex computation tasks are supposed to be offloaded to the remote cloud servers. Thus, it is critical to make offloading decisions to minimize the power consumption of the computational facilities and mobile vehicles with a delay constraint [8].

In this paper, we proposed a fog-cloud model in the vehicular network to minimize the pow consumption. Specially, the contributions of this paper are summarized as follows.

1) We modeled a mathematical framework to optimize the power consumption with delay constraint in the fog-cloud offloading model.
2) We decompose the whole system into two parts: the front-end and the back-end. We develop a combination-mode transmission in vehicular network and a deep learning model to respectively solve the optimal problem in the front-end and the back-end.
3) We conduct simulations to validate the fog-cloud model can significantly optimize the power consumption and our deep learning model is optimal.

The organization of the paper is as follows. In section II, we introduce the related work. The fog-cloud model and problem formulation is described in section III. In section IV, we give our solutions of the formulated problem. Numerical results and analysis are presented in section V. Eventually, in section VI we summarize the paper.

II. Related Work

With abundant computational resources,the cloud computing has attracted great attention on researches [9]. Ke Zhang et al. [10] proposed a promising network paradigm with predictive offloading on cloud to minimize the delay and power consumption in vehicular network, which seems to be a feasible solution in vehicular network. Whereas, the cloud computing has the following deficiencies. Initially,the cloud is far away from users. It is difficult to meet the demands of delay-sensitive applications. Morever,the power consumption of data center is queite expensive [11]. Eventually,cloud computing has a poor performance on supporting mobile vehicular appliacations. Similar to mobile cloud,cloudlet,and edge computing, Fog computing is a promised paradigm by extending cloud computing to the network edge [12]. In [13] ,a mobile cloud offloading model is proposed to meet multi-user computing offload requirements. The mobile cloud is a compensate for the deficiencies. Meng, Xianling et al proposed a Hybrid Computation Offloading with cloud and fog computing to minimize power consumption with delay constraint. These researches such as [14],[15] mainly consider the power consumption at the mobile side.

Different from the above researches, in this paper we aim to minimize the power consumption of the mobile users' side and the computational facilities. To our best knowledge, this is the first work to provide a framework of the overall power consumption in the offloading model.In the front-end(mobile users side), we design a combination-mode transmission to save energy. Moreover, we develop a deep learning method to optimize the workload in order to minimize the power consumption in the back-end (fog and cloud facilities). To our best knowledge, this is the first work to provide a detailed design about how to minimize

the power consumption by the deep learning method in offloading model.

III. SYSTEM MODEL AND PROBLEM FORMULATION

In this section,we give an overview of our system model. As shown in figure 1,the system architecture that we considered comprises : a set of Roadside Units(RSUs),a set of fog devices and a set of cloud servers.The RSUs receive requests from users and send them to a set N of fog devices,which act as users' interfaces. Fog nodes can process some requests and forward other requests to a set N of cloud servers through a Wide Area Network(WAN). Each cloud server hosts a lot of computing machines.Since the WAN covers a large geographical area from the fog to the cloud, the transmission delay can not be omitted(compared to the LAN) [16]. Moreover,the computation delay of fog node or cloud node also should be considered. In addition, the power consumption and delay of the front-end from vehicles to RSUs are also considered in our proposal system. Hence,we mainly consider consumption and delay in four components(i.e.cloud layer,fog layer, dispatch in WAN, transmission in VANET). The used notation is summarized in table I.

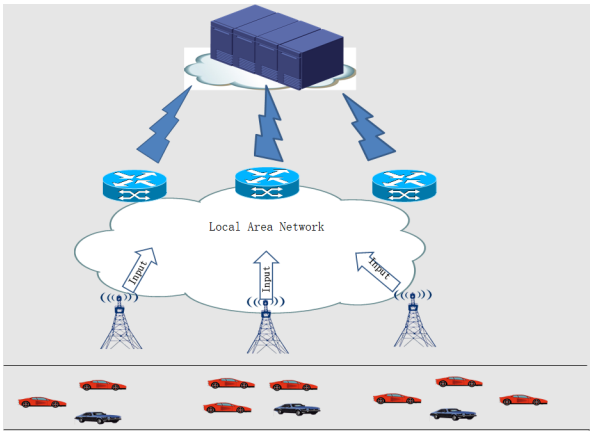


Fig. 1. Overall architecture of a fog-cloud computing system

A. System Model

We concern to minimize the consumption with an guaranteed delay in our system model. We are mainly concerned about the following parts.

1) Power Consumption of Fog Device: The computation power consumption of fog device i can be calculated by a function of the CPU frequency f_i.The function must be a monotonic increasing function.For simplicity but without loss of generality,the power consumption p_i^{fog} of device i is defined as followsA

$$p_i^{fog} \triangleq a_i f_i^2 + b_i f_i + c_i$$

,where f_i is the CPU frequency of fog device i and a_i, b_i, c_i are non-negative,pre-determined parameters.

Symbol	Definition
i	index of fog devices
j	index of cloud devices
M	size of the fog device set $\mathcal{M}$
N	size of the cloud server set $\mathcal{N}$
$r, R, \mathcal{R}$	index, number, set of RSUs
$u, U, \mathcal{U}$	index, number, set of vehicles
λ_{ij}	traffic rate dispatched from fog device i to cloud server j
y_j	workload assigned to cloud server j
f_i	CPU frequency of fog device i
L	total input from all RSUs
X	workload allocated for fog computing
Y	workload allocated for cloud server
P	power consumption
D	delay
v_i	service rate for fog device i
f_j	machine CPU frequency at cloud server j
b_{ur}	2×1 dimension matrix
n_j	machine number in cloud server j
$d_{i,j}$	communication delay from fog device i to cloud server j

TABLE I
Symbols

2) Communication Delay of Fog Device: We assume that it is a queueing system for the fog device to process requests. For the device i , the computation delay D_i^{fog} which includes waiting time and service time is

$$D_i^{fog} \triangleq \frac{1}{v_i - x_i}$$

,where x_i is the arrival rate and the v_i is the service rate.

3) Power Consumption of Cloud Server: As mentioned before, each cloud server hosts numerous computing machines. We simply assume that the CPU frequencies of these machines are equal in a cloud server. The number of machines in cloud server j is denoted by n_j. In this situation, the power consumption of cloud server j can be expressed as the product of the machine consumption value and the number of machines in server j. We approximate the power consumption value of each machine in server j by a function of CPU frequency f_j:$P^{mh} = A_j f_j^p + B_j$, where A_j and B_j are positive constants, and p is in the range of 2 to 3.

Hence, the power consumption P_j^{cloud} of the cloud server j can be calculated by multiplying the number of machines and the value of each machine power consumption.

$$P_j^{cloud} = n_j \left(A_j f_j^p + B_j\right)$$

4) Communication Delay of Cloud Server: The delay of cloud server can be obtained by $D_i^{cloud} = t_{queue} + t_{service}$. We model the system as a queueing network, and the cloud can be modeled as $M/M/n$ queue. Thus, the total computation delay of cloud server is given by

$$D_j^{cloud} \triangleq \sigma_j \left[\frac{C\left(n_j, y_j K_j / f_j\right)}{n_j f_j / K_j - y_j} + \frac{K_j}{f_j}\right]$$

,where the K_j denotes the average cycle of requests and the $C\left(n_j, y_j K_j / f_j\right)$ is Erlang's C formula [17].

5) Communication Delay of Transmission: Communication delay of dispatch: We use w_{ij} to record the bandwidth

from fog device i to the cloud server j. Let the binary variable λ_{ij} denote the traffic rate dispatched from the fog device i to the cloud server j. The communication delay of dispatch is

$$D_{ij}^{dis} \triangleq \lambda_{ij} \frac{data\ size}{w \cdot \log_2(1+\frac{S}{N})}.$$

Communication delay of vehicles and RSUs: For vehicle u and RSU r we get the time consumption through the V2I mode as $D_{ur}^{v2i} = t_{up} + hi_{ur} * t_{backhaul}$. In the V2V mode, it can be described as $D_{ur}^{v2v} = t_{up} + hv_{ur}^{j} * t_v$,where t_v,hv_{ur}^{j} denote a one-hop V2V delay and the V2V relay hops that are required in transmitting the input data to the j-hop-away RSU,respectively. The delay of VANET and RSUs can be similarly given as

$$D_{ur}^{trans} = b_{ur} \cdot \begin{bmatrix} D_{ur}^{v2v} \\ \\ D_{ur}^{v2i} \end{bmatrix}$$

,where b_{ur} is either $\begin{bmatrix} 0 & 1 \end{bmatrix}$ or $\begin{bmatrix} 1 & 0 \end{bmatrix}$ which depends on which mode to choose.

6) Power Consumption of transmission: The physical network infrastructures mainly include road side units (RSUs),Routers(fog layer) and vehicles. Let $\mathcal{R}$ and $\mathcal{U}$ be the sets of RSUs and vehicles ,respectively, where $\mathcal{R} = \{1, \ldots, R\}$,$\mathcal{U} = \{1, \ldots, U\}$.There are two transmission modes from vehicles to RSUs. The first is direct V2I mode and the other is V2V predictive- mode transmission. We can compute the power consumption of the task from vehicle u to RSU r in V2I mode as

$$p_{ur}^{v2i} = p_{upload} + hi_{ur} \cdot p_{backhaul},$$

where the $hi_{ru}, p_{backaul}$ donate the number of hops from the original RSU to the destination RSU and the cost for transmitting the output data across one road segment,respectively.

Furthermore, unlike the RSU access service that is provided by some infrastructures, the V2V communication is always self-organized by running vehicles and power costs are much less than V2I mode. The power consumption of the task in this mode from vehicle u to RSU r can be described as $p_{ur}^{v2v} = p_{upload} + hv_{ur}^{j} \cdot p_v$,where hv_{ur}^{j} denotes the V2V relay hops that are required in transmitting the input data to the j-hop-away RSU. Recall that j is the hops of the upload destination RSU away from the vehicle's current position. Thus,$j > 1$ means the vehicles adopt the predictive-mode transmission.In general, the consumption of transmission is

$$P_{ur}^{trans} = b_{ur} \cdot \begin{bmatrix} p_{ur}^{v2v} \\ \\ p_{ur}^{v2i} \end{bmatrix}$$

.

B. Constraints

Assuming that L denote the total requests input from all RSUs. These requests should be sent to a N set of fog devices. Thus, it satisfies

$$L \triangleq \sum_{i \in \mathcal{N}} l_i$$

Let X, Y respectively denote the workload for fog computing and cloud computing. Then we have

$$\begin{cases} X \triangleq \sum_{i \in \mathcal{N}} x_i \\ Y \triangleq \sum_{j \in \mathcal{M}} y_j. \end{cases}$$

The requests from RSUs can be devided into two parts, which are processed by fog devices or cloud servers,respectively. To be more specific, the corresponding relationship is as following.

Workload balance constraint for each cloud server

$$\sum_{i \in \mathcal{N}} \lambda_{ij} = y_j \qquad \forall j \in \mathcal{M} \tag{1}$$

Workload balance constraint for each fog device

$$l_i - x_i = \sum_{j \in \mathcal{M}} \lambda_{ij} \qquad \forall i \in \mathcal{M} \tag{2}$$

Obviously , from (6)(7) we can obtain

$$L = X + Y \tag{3}$$

4)WAN Bandwidth Constraint: For the transmission path from fog device i to the fog server j, let λ_{ij}^{max} denotes the max bandwidth capacity. Moreover, these transmission paths do not overlap with each other. The constraint of WAN communication bandwidth is as follows:

$$0 \leq \lambda_{ij} \leq \lambda_{ij}^{max} \qquad \forall i \in \mathcal{N}; \quad \forall j \in \mathcal{M} \tag{4}$$

In V2V transmission mode, the data is transmitted to the j-hop-away RSU by vehicles. In order to meet low delay, the hop number j should be no more than a threshold.

$$j \leqslant j_{max} \tag{5}$$

C. Problem Formulation

We propose our model toward trade-off on power consumption and delay in vehicular fog computing system. we want to minimize the power consumption ,meanwhile ensuring the low-delay The vehicular end experienced delay consist of the computation delay and transmission delay. The total delay of the proposal model is defined as

$$D^{sys} \triangleq \sum_{i \in \mathcal{N}} x_i \cdot D_i^{fog} + \sum_{j \in \mathcal{M}} y_j \cdot D_j^{cloud} + \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} D_{ij}^{dis} + \sum_{u \in \mathcal{U}} \sum_{r \in \mathcal{R}} D_{ur}^{trans}. \tag{6}$$

The power consumption consists of the fog layer and the cloud layer, which is shown as below.

$$P^{sys} \triangleq \sum_{i \in \mathcal{N}} x_i P_i^{fog} + \sum_{j \in \mathcal{M}} y_j P_j^{fog} + \sum_{u \in \mathcal{U}} \sum_{r \in \mathcal{R}} P_{ur}^{trans}.$$

We consider the problem of minimizing the power consumption ,as mentioned before, while guaranteeing the max tolerance delay $\overline{D}$ for vehicular end. Then we have the PP

$$\min_{x_i, y_j} p^{sys} \quad s.t. \begin{cases} D^{sys} \leq \overline{D} \\ (1) - (4). \end{cases} \tag{7}$$

The exact solution of the above problem consists in solving the above problem as a mixed integer non-linear programming (MINLP) problem, which is a NP-Hard problem. Thus, in case of large scale applications, the problem solving time is unacceptable.

IV. Solutions

We divide the whole system into two parts: the front end and the back end. The front end consists of vehicles and RSUs. The cost of front end is mainly consumption and delay of communication between vehicles and RSUs and we want to minimize the D_{ur}^{trans} and $P_u r^{trans}$. We adopt the predictive combination-mode scheme to solve the above problem. The back end,however, contains the fog nodes and cloud servers. We proposed a deep learning model to minimize the back-end cost as follows.

A. Greedy Algorithm

We propose a simple and well-understood greedy algorithm[18]. In the greedy algorithm, the requests are queued and we successively process each request. We choose the server,which has the minimum power consumption under the constraints conditions,for the request in each step. For each request, we place it to the current optimal position. And we get the total power consumption and delay of these requests. The pseudocode of the algorithm is given in the Algorithm 1. Although the result obtained by greed algorithm is not the global optimal solution, it will not be too bad. In the simulation section we compare our model with the cloud-only model using the greedy algorithm.

Algorithm 1 Greedy algorithm

Require:
The set of requests $Q = \{q_1, q_2, \ldots, q_m\}$;
The status list of servers $S = \{s_1, s_2, s_3, \ldots, s_n\}$;
The tolerable total delay D
Ensure:
The minimum of power consumption ;
The serial number of server nodes for requests

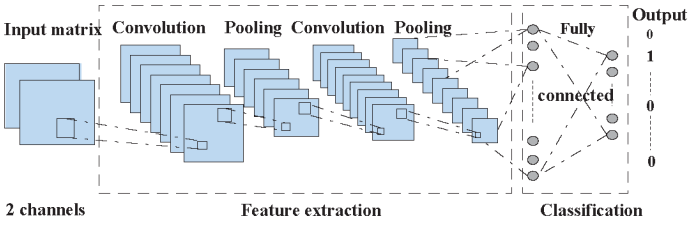
```
serial_list ← {}
sum ← 0
D̄ ← D/|Q|
for each q ∈ Q do
    minimum ← +∞
    for each s_i ∈ S do
        q → s_i
        if satisfy the constraint D̄ then
            p ←calculate power consumption of q
            if p < minimum then
                minimum ← p
                no ← i
            end if
        end if
    end for
    sum ← sum + minimum
    serial_list ⇐ no
    Update the status info of server i
end for
return sum, serial_list;
```

B. Deep Learning Model

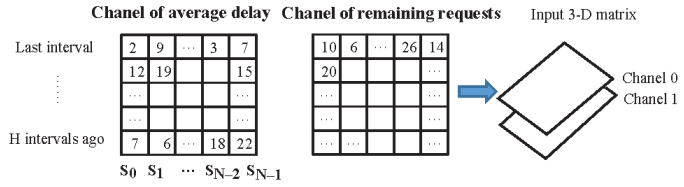
1) Input and Output Design: Our considered system model is depicted in Fig 1. In our model, we totally have n optional edge computing nodes and cloud servers to handle requests from the front-end. Therefore, each node holds a record of the number of requests and average power consumption in the last H periods. We adopt these records as the input of our deep learning model [19] and our considered deep CNN structure is shown in Fig 2. In order to train our CNN model, labeled data (i.e.many sets of (x, y)) are required to perform supervised training.

The deep CNN comprises two main components,respectively,the feature extraction and classification parts. In the feature extraction part,convolution layers are used to filter the low level features of the input data while the pooling layers are used to reduce the size of features and parameters ,and speed up calculation in the network. Features of the input data can be extracted from the convolution and pooling layers. Based on these extracted features, the fully connected layer compute the output results of the classification.

The deep learning structure is utilized to compute the candidate service provider. Therefore, we choose the server number as the output in our deep learning model.Thus, the output value is in the range of $[0, N-1]$.



(a) considered deep CNN structure



(b) input characterization

Fig. 2. Our input characterization for the deep CNN

2) Initialization Phase: As described in Input and Output Design, we need labeled data to train our CNN model. In the initialization phase, we need to obtain the labeled data. The purpose of the initialization phase is to get the labeled data which consist of the input vector and the corresponding output result [20]. It is best to train our CNN model with the global solution of formula 12. However, the time complexity of solving the optimal solution is $O(n^m)$, which is not tolerated. We choose a

compromised method to get the labeled data. We adopt a heuristic algorithm to obtain the near-optimal solution. As shown in algorithm 2, the simulated annealing(SA) algorithm mainly consists of two key step: generate a new solution with some functions and accept the new solution with a certain probability [21]. We take the solution of our greedy algorithm as the initial solution in SA algorithm.

The SA algorithm iterates L times at a certain temperature to search the global optimal solution. In the searching process, the SA algorithm poor solution with the probability of $\exp(-\frac{\triangle t}{T})$, where Δt is the evaluation difference between the new solution and the origin. With the SA algorithm, we can obtain the labeled data for our CNN model. The input of our CNN model is the records information of servers in last H intervals. Thus, we obtain the record information of servers and the corresponding offloading result.

Algorithm 2 Simulated Annealing Algorithm

Require:
The set of requests $Q = \{q_1, q_2, \ldots, q_m\}$;
The list of servers $S = \{s_1, s_2, s_3, \ldots, s_n\}$;
The tolerable total delay D
Ensure:
The minimum of power consumption ;
The serial number of server nodes for requests
Initial temperature T,temperature threshold $T^{'}$, iterations L
Initial solution
$s \leftarrow GreedyAlgorithm(Q, S, D)$
while $T > T'$ do
  repeat
    Displace some requests from high performance to low performance randomly
    if Delay<D then
      Get new solution s'
    end if
    calculate $\Delta t = C(s') - C(s)$
    if random [0,1] $< \mathrm{e}^{\frac{-\Delta \mathrm{t}}{\mathrm{T}}}$ then
      accept new solution s'
      record serial_list,$C(s')$
    end if
    if satisfy final condition then
      return $C(s')$, serial_list
    end if
    $K \leftarrow K + 1$
  until K==L
  $T = \alpha \cdot T$, where α is an attenuation factor
end while
return $C(s')$, serial_list

3) Training Phase: In the training phase, we use the data obtained in initialization phase to train the CNN model. The training phase consists of two step: initializing the parameters in our designed CNN and fine-tuning the parameters with BP(back-propagation) algorithm. It is necessary to initial the parameters,which is benefit to accelerate the convergence speed. The CNN parameters are initialized by normal Gauss distribution with a mean value of zero.

For feed-forward neural network, parameter optimization depends on the error back-propagation. The optimization phase can be done by the stochastic gradient descent(SGD) algorithm or the Adam algorithm. Since the Adam algorithm is adaptive, we choose adam as the optimization algorithm in our training phase as shown in Algorithm 3. In the training phase, we take the cross-entropy cost function as the loss function. Thus, the output is a scalar which is the neuron index of the maximum value in the output layer.

Algorithm 3 Training Algorithm

Require:
$(x, y) = \left\{(x^{(t)}, y^{(t)}) | t = 1, 2, \ldots, m\right\}$
Ensure:
θ
initial θ randomly
for $(x_i, y_i) \in (x, y)$ do
  predict = forward(x_i,θ)
  loss = loss_fun(predict,y_i)
  loss.backward()
  update θ
end for

4) Running Phase: In running phase all servers need to record the number of received requests and average power consumption over a period of time and send these records to edge computing nodes. In this way, each edge node can take these records as an input to calculate the offloading result. However, it is possible to obtain an inappropriate result which doesn't meet the constraints in the CNN running phase. In such situation, we take the greedy algorithm as an compensation.

V. PERFORMANCE EVALUATION

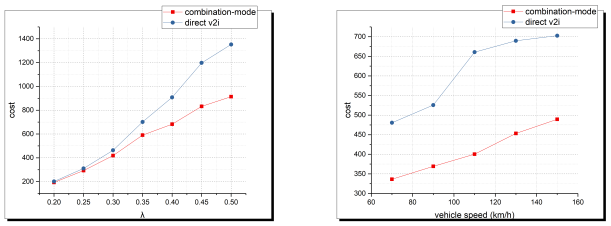
In order to validate the performance of our model, we conduct the simulations on the front end and back end. The related parameters in the simulation is shown as table 2.

Variable	Value
N	20
M	5
n_j	$20 \sim 25$
f_i	$4.5 \sim 5.5$
f_j	$2.5 \sim 3.5$
request packet size	$5 \sim 15MB$
request need cycles	$0.7 \sim 0.9$

TABLE II
parameters

As mentioned before, we used the predictive combination-mode model in the front-end. Before sending requests, broadcast packets are sent to ask the nearby vehicles or RSU for the back-end processing delay. With the delay result and vehicle speed, vehicles can calculate the arriving RSU when the requst returns. In the combined-model, the power consumption and delay of V2I,V2V

model are estimated to choose the optimal model. In the simulation, we assume that vehicles are travelling in a straight line and the simulation result is shown as figure 3



(a) house (b) hotel

Fig. 3.

Figure 3 illustrates the performance of cost with different vehicle speed and vehicle density. With higher vehicle speed, the cost of the combined-mode is much lower than the V2I mode. For the V2I mode, the cost grows significantly with the speed of 100 to 120. That is because with that speed most vehicles are at the edge of RSU coverage when the request returns. Therefore, when the speed exceeds a threshold, the vehicles will drive into another RSU. The cost of V2I mode will increase significantly. In addition, the optimization effect of combined-mode is obvious with a heavy traffic density. With a heavy traffic density, the delay and power consumption of each hop in V2V mode reduced a lot.

In order to validate the performance of fog-cloud model. We compare the performance of fog-cloud model with cloud-only and fog-only mode.

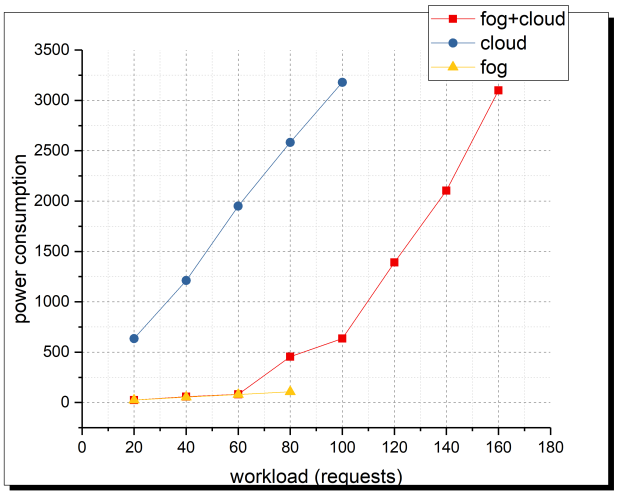


Fig. 4. Comparison of energy consumption

As shown in figure 4, the power consumption of fog-only is extremely low but no more than 80 requests can be handled at one time. The fog-cloud model is much better than the cloud-only model with the workload of 20 to 100. The power consumption of the fog-cloud model is almost the same as that of fog-only model with the workload of 20 to 80. In such situation, the workload of fog layer is not saturated and most requests are processed by the fog node. When the workload is more than 100, the fog layer reaches saturation,after which the power consumption growth trend in the fog-cloud model is similar to the cloud-only model.

Moreover, the maximum workload of fog-cloud model is much larger than the other two models. Figure 5 illustrates the performance of delay with different workload. The average delay of fog-cloud model is less than 1.5 seconds. Whereas, in the other two model the average delay is fast-growing.

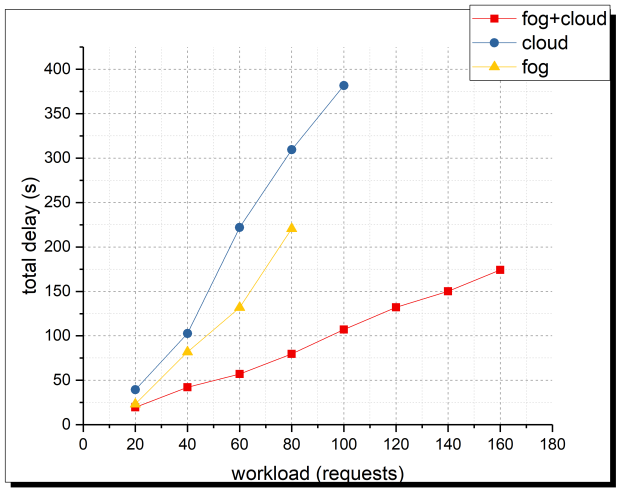


Fig. 5. Comparison of delay

As mentioned before, we take the last H interval records of servers as input in CNN model. The CNN model in the simulation is shown in figure 6. We take the records of last four Δt intervals as input, where Δt is equal to 2 seconds. In each interval Δt, the average number of request is 50. The training data is obtained by SA algorithm in 20 consecutive intervals.

Input layer		Convolutional layers			Full connection layers			Output layer	
Width	25	Layer	1	2	Layer	1	2	Node	25
		Width	1	3					
		Height	3	3					
Height	4	Channel	20	20	Node	2000	200	Active	NULL
		Stride	1	1					
		Padding width	0	1					
Channel	2	Padding height	1	1	Active	Relu	Relu	Initialize	Xavier
		Active	Relu	Relu					
		Initialize	Xavier	Xavier	Initialize	Xavier	Xavier		

Fig. 6. CNN model structure

After training, we obtain the offloading result of requests by CNN model. The power consumption of different algorithm is shown in figure 7. Our deep learning model can effectively reduce the computational complexity in the running phase and the performance is much better than the greedy algorithm

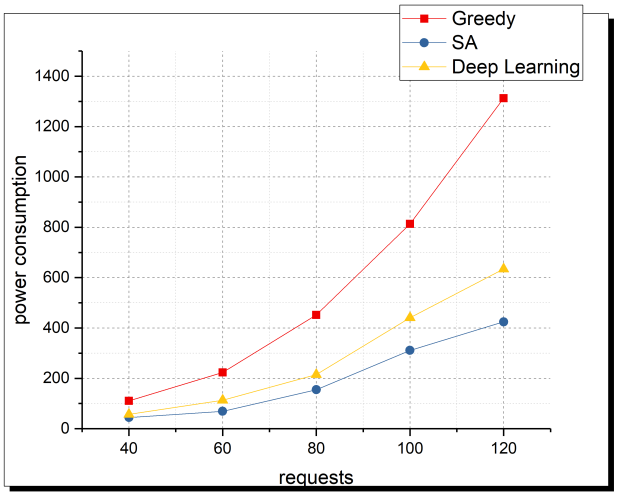


Fig. 7. Power Consumption of Deep Learning and other algorithm

VI. Conclusion

In this paper,we propose a fog-cloud model based on deep learning ,which is a feasible solution to minimize the power consumption and delay in the back-end. The offloading optimization problem is formulated in our proposed model.For the fog nodes, they can be modeled as M/M/1 in queueing theory. For the cloud servers, they can be modeled as M/M/n queue. In addition, we propose a predictive combination-mode model to minimize the cost in front-end. In the simulation, we can draw the conclusion that the fog-cloud model shows good performance compared to the cloud-only mode or the fog-only mode. Our deep learning model is an approximate approach to solved the formulated problem.

References

[1] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications," IEEE Internet of Things Journal, vol. 4, no. 5, pp. 1125–1142, 2017.

[2] M. Gerla, E.-K. Lee, G. Pau, and U. Lee, "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds," in Internet of Things (WF-IoT), 2014 IEEE World Forum on, pp. 241–246, IEEE, 2014.

[3] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen, "Vehicular fog computing: A viewpoint of vehicles as the infrastructures," IEEE Transactions on Vehicular Technology, vol. 65, no. 6, pp. 3860–3873, 2016.

[4] D. Mazza, D. Tarchi, and G. E. Corazza, "A unified urban mobile cloud computing offloading mechanism for smart cities," IEEE Communications Magazine, vol. 55, no. 3, pp. 30–37, 2017.

[5] S. Yi, C. Li, and Q. Li, "A survey of fog computing: concepts, applications and issues," in Proceedings of the 2015 Workshop on Mobile Big Data, pp. 37–42, ACM, 2015.

[6] T. Francis and M. Madhiajagan, "A comparison of cloud execution mechanisms: Fog, edge and clone cloud computing," Proceeding of the Electrical Engineering Computer Science and Informatics, vol. 4, pp. 446–450, 2017.

[7] Z. Zhou, P. Liu, Z. Chang, C. Xu, and Y. Zhang, "Energy-efficient workload offloading and power control in vehicular edge computing," in Wireless Communications and Networking Conference Workshops (WCNCW), 2018 IEEE, pp. 191–196, IEEE, 2018.

[8] A. Bozorgchenani, D. Tarchi, and G. E. Corazza, "An energy and delay-efficient partial offloading technique for fog computing architectures," in GLOBECOM 2017-2017 IEEE Global Communications Conference, pp. 1–6, IEEE, 2017.

[9] X. Chen, "Decentralized computation offloading game for mobile cloud computing," IEEE Transactions on Parallel and Distributed Systems, vol. 26, no. 4, pp. 974–983, 2015.

[10] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," IEEE Vehicular Technology Magazine, vol. 12, no. 2, pp. 36–44, 2017.

[11] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, and R. S. Tucker, "Fog computing may help to save energy in cloud computing," IEEE Journal on Selected Areas in Communications, vol. 34, no. 5, pp. 1728–1739, 2016.

[12] T. H. Luan, L. Gao, Z. Li, Y. Xiang, G. Wei, and L. Sun, "Fog computing: Focusing on mobile users at the edge," arXiv preprint arXiv:1502.01815, 2015.

[13] M.-H. Chen, M. Dong, and B. Liang, "Resource sharing of a computing access point for multi-user mobile cloud offloading with delay constraints," IEEE Transactions on Mobile Computing, 2018.

[14] X. Meng, W. Wang, and Z. Zhang, "Delay-constrained hybrid computation offloading with cloud and fog computing," IEEE Access, vol. 5, pp. 21355–21367, 2017.

[15] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," IEEE Journal on Selected Areas in Communications, vol. 34, no. 12, pp. 3590–3605, 2016.

[16] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," IEEE Internet of Things Journal, vol. 3, no. 6, pp. 1171–1181, 2016.

[17] N. Gautam, Analysis of queues: methods and applications. CRC Press, 2012.

[18] W. Chen, D. Wang, and K. Li, "Multi-user multi-task computation offloading in green mobile edge cloud computing," IEEE Transactions on Services Computing, 2018.

[19] F. Tang, B. Mao, Z. M. Fadlullah, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "On removing routing protocol from future wireless networks: A real-time deep learning approach for intelligent traffic control," IEEE Wireless Communications, vol. 25, no. 1, pp. 154–160, 2018.

[20] B. Mao, Z. M. Fadlullah, F. Tang, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "Routing or computing? the paradigm shift towards intelligent computer network packet transmission based on deep learning," IEEE Transactions on Computers, vol. 66, no. 11, pp. 1946–1960, 2017.

[21] L. Wei, Z. Zhang, D. Zhang, and S. C. Leung, "A simulated annealing algorithm for the capacitated vehicle routing problem with two-dimensional loading constraints," European Journal of Operational Research, vol. 265, no. 3, pp. 843–859, 2018.