Asynchronous Federated Learning for Vehicular Edge Caching of Consumer Content

Xiaolong Xu, Guanming Bao
Nanjing University of Information Science and Technology, China

Muhammad Bilal
Lancaster University, United Kingdom

***Abstract*—Edge caching is one of the key technologies, which enhances consumer content delivery and reduces service latency by caching consumer content and services on edge nodes. Learning-based caching algorithms have been proposed in the literature to achieve high caching efficiency. However, in the complicated task of edge caching for consumer content in vehicular networks, it is challenging to achieve a high cache hit rate and low consumer content delivery delay. This paper proposes AFRL, an Asynchronous Federated Learning with Deep Reinforcement Learning edge consumer content caching algorithm for vehicular networks. AFRL utilizes federated learning to collaboratively train a shared DRL agent among Roadside Units(RSUs), and an efficient asynchronous federated learning algorithm is also introduced to accelerate convergence and improve cache hit rates in dynamic environments. Simulation results demonstrate the superior performance of AFRL compared to traditional and state-of-the-art caching algorithms, showcasing its potential in handling varying traffic densities, achieving higher cache hit rates, and low consumer content delivery delay.**

CLOUD COMPUTING provides users with services by integrating multiple computing entities into a single powerful computing system through the network, thereby reducing the processing burden on user terminals. However, traditional cloud computing struggles to support low-latency services and faces problems such as privacy leakage [1]. To address these challenges, distributed cloud systems enable the deployment of storage and computation resources closer to the edge. A distributed cloud at the edge greatly reduces communication complexity and consumer content delivery latency and enhances privacy by employing localized data storage [2]. Furthermore, deploying servers at the network's edge reduces communication and computation load from the core network.

Digital Object Identifier 10.1109/MCE.2022.Doi Number
Date of publication 00 xxxx 0000; date of current version 00 xxxx 0000

Real-time information transfer is essential for vehicular networks to provide intelligent and effective modern transportation systems for applications such as path planning, autonomous driving, and real-time traffic monitoring [3]. However, with the growing popularity and use of various vehicular mobile applications, low-latency communication between vehicles and road infrastructure has become crucial to enable real-time information flow. To support real-time information flow, computation offloading has become increasingly important, but traditional cloud computing models cannot meet the requirements of highly dynamic vehicular networks. Thus, distributed cloud systems are introduced to provide more efficient services in vehicular networks. By deploying more Distributed Cloud Nodes(DCNs) adjacent to the road to provide nearby edge caching and computation services for vehicles, distributed cloud systems can support rapidly growing low-latency onboard mobile applications. Learning-based approaches have been proposed to achieve efficient edge caching using RSUs scattered across the roadside. Qiao et al. [4] used a DDPG framework on RSUs to build a dual time-scale Markov decision model, reduce consumer content delivery latency and system exhaustion, and continuously train the agent to make better caching decisions.

As a distributed learning framework, federated learning enables a group of users to collaboratively train a shared model. In this framework, training is performed locally by users, and aggregation is carried out at a central server, with only the model parameters communicated, ensuring user data privacy [5]. Utilizing federated learning to train models in DCNs holds great promise for improving edge caching, consumer content delivery delay. However, traditional synchronous federated learning approaches are not efficient for highly dynamic vehicular networks, as they require vehicle clients to wait for all local updates before performing global aggregation for the next round of local training. Asynchronous federated learning frameworks are therefore advantageous, and combining them with deep reinforcement learning offers even more benefits.

Deep reinforcement learning(DRL) is an effective method for training an optimal decision network by having an intelligent agent continuously interact with the environment, thereby achieving efficient decision-making in numerous interactive application scenarios. In this paper, we propose an edge caching algorithm, named AFRL, that combines Asynchronous Federated learning and deep Reinforcement Learning. AFRL has shown high caching performance in simulation experiments, outperforming traditional caching algorithms, the synchronous benchmark based on FedAvg, and the compared state-of-the-art edge caching algorithm in terms of cache hit rate and consumer content delivery delay.

ARCHITECTURE FOR DISTRIBUTED CLOUD IN VEHICULAR NETWORKS

Various distributed cloud models have been proposed to address the challenges of conventional cloud computing. Mohan et al. [6] proposed a distributed cloud model for the Internet of Things (IoT) named Edge-Fog cloud. They assign computation tasks to multiple data center networks for execution, following the idea of spreading computation across different data center networks. Generally, distributed cloud systems are not universal due to different application scenarios.

Figure. 1 depicts the application scenario of distributed cloud in vehicular networks, where vehicles maintain connections with RSUs while driving across RSUs' coverage areas. An RSU with a limited storage space is deployed with wireless communication devices, and is connected to the data center in through cable. Adjacent RSUs communicate through wireless networks. The coverage of each RSU does not overlap. It is popular to DRL agents on RSUs. Federated learning can be performed between the RSUs federation and the Cloud Data Center(CDC). Through collaboration among vehicles, RSUs, and CDC, many vehicle-oriented applications can be supported. In this section, we focus on the proposed distributed cloud architecture for vehicular networks from the perspective of edge computing, where RSUs function as DCNs.

In this architecture, A vehicle client establishes a connection with a DCN server and requests consumer content. In contrast to traditional cloud computing, the request is initially received by the local DCN, which connects with the requesting vehicle client and provides the desired consumer content if it is available on the local server. However, because DCNs have limited storage capacity, we have devised a collaborative caching technique to enhance cache hit rate and reduce consumer content delivery delay. Specifically, a request by a vehicle client is handled in one of three different ways: (i) The local DCN caches the previously requested consumer content, and when a newly connected vehicle client requests consumer content, the DCN delivers the cached consumer content directly, as shown in Figure. 2 with vehicle client 1 requesting consumer content B. (ii) When the connected DCN does not have the consumer content in the local cache, but the adjacent DCNs do have the requested consumer content in the cache, as in the case of vehicle client 2 requesting consumer content C in Figure. 2, an adjacent DCN sends the requested consumer content to the local DCN, which then forwards it to the requesting vehicle client. (iii) When the local DCN and the adjacent DCNs do not have the requested consumer content in their local cache, the local DCN requests to the remote CDC and obtains the requested consumer content, and then the local DCN forwards it to the requesting vehicle client, as illustrated by vehicle client 3 requesting consumer content K in Figure. 2.

PROPOSED ALGORITHM FOR VEHICULAR EDGE CACHING

Based on the distributed cloud architecture for vehicular networks, we have designed the AFRL vehicular edge caching algorithm. AFRL combines asynchronous federated learning and deep reinforcement learning.

DRL Model

We partition time into brief five-minute segments. Each vehicle's request is represented as a triplet, consisting of the request time, the corresponding RSU number connected to the vehicle client, and the requested content's number. We deploy an agent on each RSU. Each time segment corresponds to a step in DRL, and 144 time segments(12 hours) corresponds to an episode. At the start of a time segment, the agent observes the current state, performs an action, and upon completion of the time segment, the agent receives a reward. A new state is then observed, and the sequence of state, action, reward, and new state is stored in the agent's experience pool. The elements of DRL is introduced as follows:

State: The number of all contents that a vehicle client may request is recorded as C_n. We record the number of times each consumer content is requested in each time segment and set an observation length L. At any time segment, the observation or state of a DRL agent is a two-dimensional vector composed of the number of times each content has been requested in the past L time segments, and the shape of the state is $L \times C_n$. The state indicates the short-term request frequency of consumer content, which is the most significant basis for edge caching.

Action: At the beginning of each time segment, the agent performs an action based on the current observation. The action is represented as a one-dimensional vector of length C_n. Each element of the vector is a number ranging from 0 to 1, representing the preferences of caching the corresponding content. The agent then ranks the content based on these preferences and selectively caches the highest-preferences content until the maximum capacity of the RSU storage is reached.

Reward: At the end of each time segment, the agent accumulates the total reward based on the action

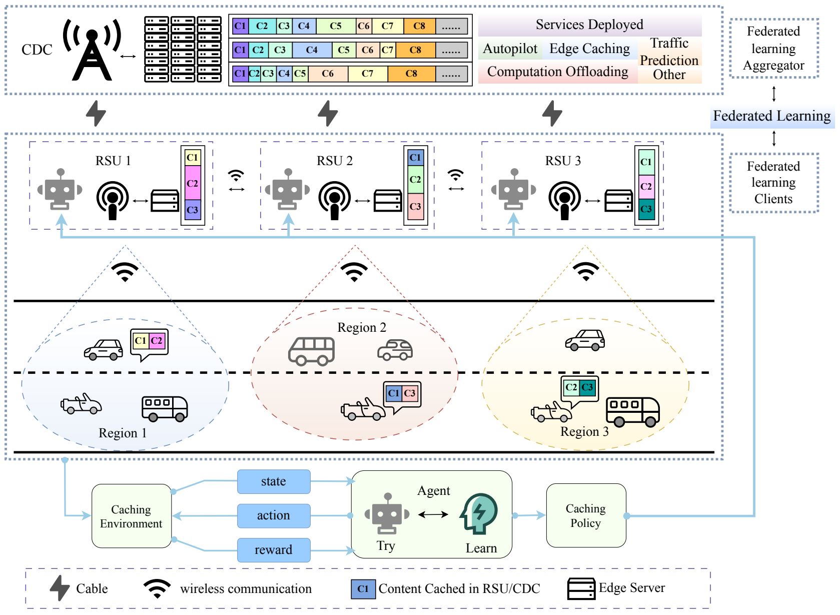

Figure 1. Application scenarios of distributed cloud for vehicular networks.

performed at the beginning of that time segment. The reward function is designed considering the objectives of the task. Let's denote the cache hit rate as hr, the cache utilization rate as hu, and the content delivery delay as sd. We introduce three weights: $w1, w2, w3$, where $w1>0, w2>0, w3<0$. The reward is then calculated as $w1 \times hr + w2 \times hu + w3 \times sd$. This reward function encourages the agent to perform actions that increase the cache hit rate, increase the cache utilization rate, and decrease the content delivery delay.

Asynchronous Federated Learning

Clients: The clients of federated learning are RSUs. We regard all RSUs in a region as an RSU federation. The motivation of each RSU is the same, that is, to enable the agent to obtain the maximum reward in the local environment. Therefore, it is reasonable for RSUs to cooperate in training a global model.

Asynchronous model aggregation and distribution: To the edge caching of consumer content in vehicular networks, policy of the agents deployed on the RSUs updates after processing a certain amount of requests. Different RSUs are in different environments and receive requests at different frequencies, resulting in different frequencies of local updates, and traditional federated learning performs model aggregation synchronously. Therefore, using synchronous federated

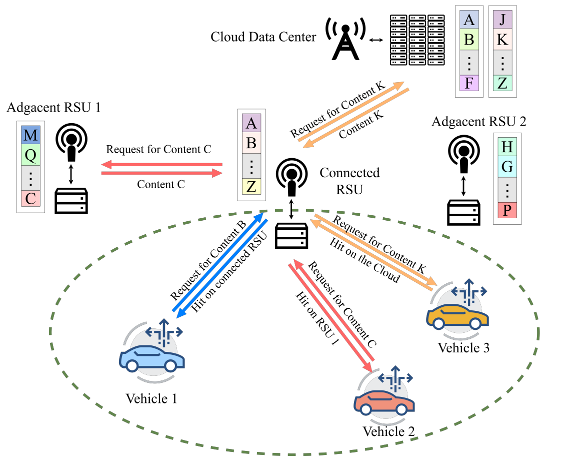

Figure 2. Framework of edge consumer content caching in vehicular networks.

learning in the edge caching of consumer content in vehicular networks will cause a waiting interval. Part of RSUs will wait for other RSUs to upload their local models, and every time model aggregation is performed, there is a high possibility that some RSUs will be waiting, which reduces the learning efficiency of the agents. Therefore, we propose to use an asynchronous federated learning algorithm for the edge caching of vehicular consumer content. Specifically, we adopt asynchronous model update and distribution strategies. The specific aggregation method is shown in the next subsection.

AFRL for Vehicular Edge Caching

In this subsection, we provide a detailed description of the proposed AFRL algorithm. The DRL agent deployed on the RSU is trained using the DDPG algorithm. Each agent consists of four networks: main actor network, target actor network, main critic network, and target critic network. The following are the steps of the AFRL algorithm.

- (i) Initialization: CDC is designated as the aggregation server. Four networks, namely global main actor network, global target actor network, global main critic network, and global target critic network, are initialized. These four networks are distributed to each agent. The set of agents is denoted as $\{agent_i|i = 1, 2, \cdots, num_1\}$, where num_1 represents the number of agents.
- (ii) Local Concurrent Execution by each $agent_i$:
  - 1. $agent_i$ receives a state, performs an action, receives a reward, and obtains a new state. The tuple (state, action, reward, new state) is stored in the local experience pool.
  - 2. Whenever $agent_i$ processes num_2 requests, it randomly samples a batch of experiences from the local experience pool and updates its two main networks. The two target networks are periodically updated by copying from the main networks.
  - 3. Asynchronous aggregation is performed every num_3 updates.
- (iii) Asynchronous Update Strategy: The global model in federated learning is denoted as w_g, and the local model parameters of $agent_i$ are denoted as w_i. The number of requests received by $agent_i$ is represented as $request_i$. The contribution of $agent_i$ to the model $contribution_i$ is defined as $request_i / \sum_{k=1}^{num_1} request_k$. Based on the contribution of $agent_i$, the global model w_g is updated as $contribution_i \times w_i + (1 - contribution_i) \times w_g$.

EXPERIMENT SETTINGS AND PERFORMANCE EVALUATION

We generated a dataset simulating vehicles on a highway with speeds ranging from 80 km/h to 120 km/h. The coverage radius of RSUs is 5 km. Five RSUs are continuously deployed to handle vehicle requests. The number of contents requested by vehicles is 1000, with each content's size ranging from 0 to 10 MB. Each vehicle has a preference for each content and requests content from RSUs based on its preference with a certain level of activity. The generated dataset contains 8669971 requests in a year. The storage capacity of each RSU is 1024 MB. In the DDPG algorithm, the discount factor used for soft updates is 0.99, and the experience replay buffer has a capacity of 50000. Both the actor and critic networks have a hidden layer with 128 neurons. To demonstrate the advantages of the AFRL algorithm, we compare it with the following baselines:

- RC: Random Cache, where the RSUs update their cached content randomly.
- PFC: Popular First Cache, where the RSUs cache contents based on the popularity ranking among the observed window of the request of each content.
- RL: each agent learns independently and locally without collaboration using DDPG.
- FRL: Federated Reinforcement Learning with RSUs performing synchronous aggregation. The aggregation function used is the classical synchronous

federated averaging algorithm[7].

- CODR: state-of-the-art edge caching algorithm[8], where the RSUs make caching decisions based on predicted popularity using the simplex algorithm.

In Figure. 3, the AFRL algorithm demonstrates its advantages. When RSUs are trained locally and do not cooperate with each other, the limited amount of local experience results in the agents acquiring less experience during the learning process. In the complex environment of vehicular edge caching of consumer content, this makes it difficult for a single agent to make satisfactory caching decisions. In Figure. 3, the reward curve of the RL algorithm quickly declines after a brief increase because the individual agent lacks sufficient data and the learning process is incomplete. In the later stages of training, the agent's exploratory behavior decreases, and performing caching decisions using suboptimal strategies becomes challenging to achieve high rewards, leading to a gradual decrease in rewards. In contrast, the AFRL algorithm allows RSUs to cooperate by enabling global model updates and distribution. This enables RSUs to share their experiences and knowledge, and achieve more thorough learning, resulting in higher rewards and better caching decisions. However, traditional federated learning is not perfectly suited for vehicular edge caching of consumer content. In the case of global updates by the agents, there are situations where part of agents have to wait. AFRL overcomes this limitation by allowing the agents to perform global model updates immediately after completing a batch of local training and obtaining the updated global model. This enables the agents to learn more thoroughly and achieve higher rewards. In Figure. 3, both training runs of AFRL converge slightly earlier than FRL and achieve higher rewards.

Figure. 4 illustrates the cache hit rates under different traffic densities. Traffic density refers to the number of vehicle clients within the coverage area of all RSUs. The cache hit rate of the learning-based caching algorithms represent the average cache hit rate of its converged episodes including AFRL, FRL, RL, and CODR. At traffic densities of 30 and 60, AFRL exhibits higher cache hit rates compared to all other caching algorithms. Particularly, at a higher traffic density of 60, AFRL achieves a 6.02% improvement in cache hit rate over RC, a 5.2% improvement over PFC, a 2.22% improvement over RL, a 0.5% improvement over FRL, and a 3.2% improvement over CODR. Among these, all the caching algorithms utilizing DRL agents outperform the popularity-based caching algorithm CODR in terms of hit rate. CODR caches content based on predicted content popularity and RSUs' demands, while each RL-based approaches assigns a time window of past content requests to the agent. The difference lies in the fact that the RL-based approaches directly observe the environment instead of relying on human-made popularity and demands for caching decisions. This allows the RL-based approaches to directly respond to the environment, resulting in higher cache hit rates.

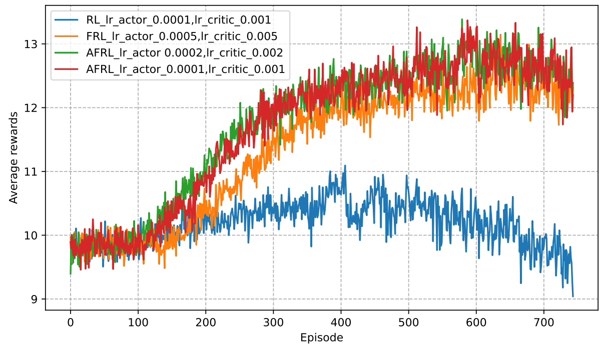

Figure 3. Comparison on average rewards under different episodes.

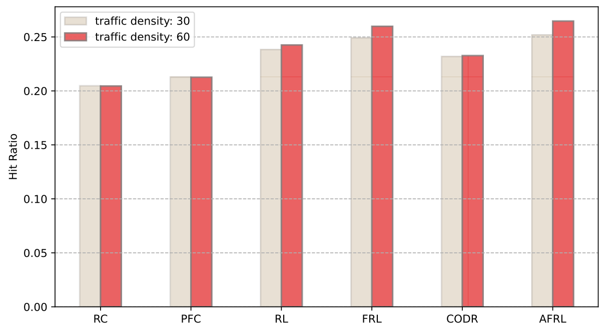

Figure 4. Comparison on cache hit ratio under different traffic densities.

In Figure. 5, we varied the storage space per RSU and compared the consumer content delivery delay brought by different caching algorithms. Starting from 1GB, we incrementally increased and decreased the storage of RSUs by 128MB. In these different storage settings, AFRL consistently achieved the lowest delay. Consistent with the previous discussion, delay of RL and FRL were lower than that of CODR, indicating the effectiveness of AFRL's use of agents for consumer content caching. Furthermore, we observed that as the

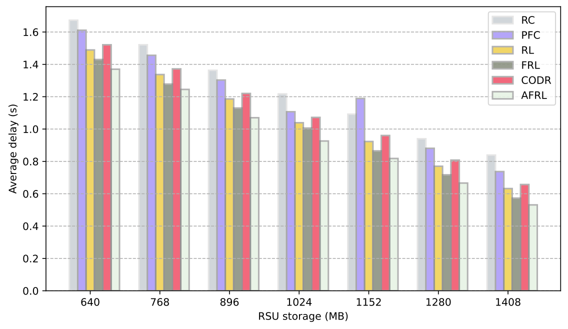

Figure 5. Comparison on average service delay under different RSU storage.

RSU storage increases, the delay differences between AFRL and CODR, RL, and FRL gradually decreased. Under a smaller storage space of 640MB, AFRL had delays 0.151s, 0.119s, and 0.06s shorter than CODR, RL, and FRL, respectively. Under a larger storage space of 1408MB, AFRL had delays 0.126s, 0.101s, and 0.042s shorter than CODR, RL, and FRL, respectively. This indicates that AFRL can provide shorter delays for consumer content delivery in verhicular networks, particularly when the RSU storage space is smaller.

CONCLUSION

In this paper, we propose a collaborative edge caching algorithm that combines asynchronous federated learning with deep reinforcement learning named AFRL. The AFRL algorithm achieves significant improvements in consumer content delivery delay, cache hit rate. Notably, we employ a federated learning framework that enables RSUs to collaboratively train a shared DRL model. Additionally, we utilize an efficient asynchronous federated learning algorithm to expedite the convergence of models. Through simulation experiments, we demonstrate that the proposed AFRL approach achieves faster convergence compared to the synchronous approach. Furthermore, it outperforms classical and state-of-the-art caching algorithms in terms of cache hit rate and consumer content delivery delay, and it is robust under different RSU storage and different traffic densities.

REFERENCES

1. A. Chakraborti, R. Curtmola, J. Katz, J. Nieh, A.-R. Sadeghi, R. Sion, Y. Zhang, and others, ”Cloud Computing Security: Foundations and Research Directions,” Foundations and Trends® in Privacy and Security, vol. 3, no. 2, pp. 103-213, 2022
2. Y. Coady, O. Hohlfeld, J. Kempf, R. McGeer, and S. Schmid, ”Distributed cloud computing: Applications, status quo, and challenges,” ACM SIGCOMM Computer Communication Review, vol. 45, no. 2, pp. 38-43, 2015
3. W. Duan, J. Gu, M. Wen, G. Zhang, Y. Ji, and S. Mumtaz, ”Emerging technologies for 5G-IoV networks: applications, trends and opportunities,” IEEE Network, vol. 34, no. 5, pp. 283-289, 2020
4. G. Qiao, S. Leng, S. Maharjan, Y. Zhang, and N. Ansari, ”Deep reinforcement learning for cooperative content caching in vehicular edge computing and networks,” IEEE Internet of Things Journal, vol. 7, no. 1, pp. 247-257, 2019
5. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Aguera y Arcas, ”Communication-efficient learning of deep networks from decentralized data,” in Artificial intelligence and statistics, pp. 1273-1282, 2017
6. N. Mohan and J. Kangasharju, ”Edge-Fog cloud: A distributed cloud for Internet of Things computations,” in 2016 Cloudification of the Internet of Things (CIoT), pp. 1-6, 2016
7. McMahan B, Moore E, Ramage D, et al. ”Communication-efficient learning of deep networks from decentralized data,” Artificial intelligence and statistics (PMLR), pp. 1273-1282, 2017
8. Zhou X, Bilal M, Dou R, et al. ”Edge Computation Offloading with Content Caching in 6G-Enabled IoV,” IEEE Transactions on Intelligent Transportation Systems, pp. 1-15, 2023

Xiaolong Xu (Senior Member, IEEE) is with the School of Software, Nanjing University of Information Science and Technology. He has published more than 100 peer review papers in international journals and conferences.

Guanming Bao is with the School of Computer Science, Nanjing University of Information Science and Technology. His research interest includes big data and deep learning.

Muhammad Bilal (Senior Member, IEEE) is Senior lecturer (Associate Professor) with the School of Computing and Communications, Lancaster University, United Kingdom. He authored more than 100 articles published in renowned journals and registered many US and Korean patents in the area of Networking, Cyber Security, IoT, and Cloud/Fog Computing.