Edge-Assisted Distributed DNN Collaborative Computing Approach for Mobile Web Augmented Reality in 5G Networks

Ren, Pei; Qiao, Xiuquan; Huang, Yakun; Liu, Ling; Dustdar, Schahram; Chen, Junliang

Published in:
IEEE Network

DOI:
10.1109/MNET.011.1900305

Edge-Assisted Distributed DNN Collaborative Computing Approach for Mobile Web Augmented Reality in 5G Networks

Pei Ren, Xiuquan Qiao, Yakun Huang, Ling Liu, Schahram Dustdar, and Junliang Chen

Abstract

Web-based DNNs provide accurate object recognition to the mobile Web AR, which is newly emerging as a lightweight mobile AR solution. Web-based DNNs are attracting a great deal of attention. However, balancing the UX against the computing cost for DNN-based object recognition on the Web is difficult for both self-contained and cloud-based offloading approaches, as it is a latency-sensitive service but also has high requirements in terms of computing and networking abilities. Fortunately, the emerging 5G networks promise not only bandwidth and latency improvement but also the pervasive deployment of edge servers which are closer to the users. In this article, we propose the first edge-based collaborative object recognition solution for mobile Web AR in the 5G era. First, we explore the fine-grained and adaptive DNN partitioning for the collaboration between the cloud, the edge, and the mobile Web browser. Second, we propose a differentiated DNN computation scheduling approach specially designed for the edge platform. On one hand, performing part of DNN computations on mobile Web without decreasing the UX (i.e., keep response latency below a specific threshold) will effectively reduce the computing cost of the cloud system; on the other hand, performing the remaining DNN computations on the cloud (including remote and edge cloud) will also improve the inference latency and thus UX when compared to the self-contained solution. Obviously, our collaborative solution will balance the interests of both users and service providers. Experiments have been conducted in an actually deployed 5G trial network, and the results show the superiority of our proposed collaborative solution.

Introduction

Mobile Web AR [1] is a lightweight implementation to provide an immersive experience on a mobile device. It enables cross-platform service provisioning that is challenging for the mainstream App-based implementations. Now it is seen as one of the most promising solutions for mobile AR. Recently, it has attracted a great deal of interest for its widespread application, such as HoloLeo Studios AR.js, Google WebARonARKit and WebARonARCore, Mozilla WebXR Viewer, and Tencent TBS AR.

Various components collaborate in the development of mobile Web AR. Object recognition is one of the most important, as it provides the key for subscribers to enter the mixed-reality world. Accurate and real-time recognition is therefore particularly required. Fortunately, deep learning techniques have emerged, with a surprising feature extraction capability. However, the dominant Web-based DNN implementations still face severe challenges for their practical application:

- Self-contained methods (e.g., TensorFlow.js and Keras.js) perform recognition inference on the Web, but all suffer from an unacceptable response delay due to the limited computing capability of the mobile Web browser (especially the built-in browsers in mobile Apps). Although there are already some lightweight approaches for inference acceleration, satisfactory recognition accuracy cannot be achieved at the same time.
- Cloud-based offloading methods leverage the resources of a remote central site for accurate object recognition. But these methods also have their inherent flaws:
  - The UX is degraded by wireless network fluctuations during continual long-distance data transmission.
  - There are also serious increases in the computing cost in the cases of high concurrent.

Offloading the DNN computations to the network edge seems a promising method [2], especially in the upcoming 5G networks [3] due to the pervasive deployment of edge servers, which will provide supplementary computing and storage resources for subscribers close to the access points.

Why Collaboration is Needed

Placing the DNN computations on edge servers is obviously able to provide accurate and real-time recognition services. Nevertheless, the edge-based collaborative method is still recommended for the following reasons.

Push from Service Providers: AR is a specific type of computation-intensive and data-intensive application. When faced with the ever-increasing computing cost, service providers are sorely in need of new approaches for cost saving. The idle computing resources on the end user's device have again gained attention.

Pull from End Users: With the continuous improvement of mobile devices, end users prefer to execute computations locally, not only for a

Digital Object Identifier:
10.1109/MNET.011.1900305

Pei Ren, Xiuquan Qiao, Yakun Huang, and Junliang Chen are with Beijing University of Posts and Telecommunications; Ling Liu is with Georgia Institute of Technology; Schahram Dustdar is with the Technische Universität Wien.

better experience (because it reduces the impact of data transmission on the applications) but also to protect their privacy.

By combining the advantages of all parties (namely, the mobile Web browser, the network edge, and the cloud server), an edge-computing based collaborative approach will undoubtedly be favored by users and service providers.

Fine-Grained and Adaptive DNN Partitioning

However, current collaborative methods [4, 5] are designed for collaboration between the user device and the cloud based on a single partitioning point. Apparently, the collaboration between three distributed platforms will be more complex. Although there have been some preliminary attempts at edge-based distributed DNN [6], these methods currently cannot deal with complex and varying situations due to their coarse-grained partitioning and lack of dynamic adaptability.

To this end, we propose the first fine-grained and adaptive collaboration mechanism for DNN-based object recognition on the mobile Web in 5G networks. Specifically, we analyze the characteristics of each layer in DNN, then obtain the inference latency and energy consumption prediction models. By considering the capability of the computing platforms and the mobile network performance, the proposed mechanism therefore enables adaptive DNN computation partitioning. Consequently, the DNN will be divided into several parts dynamically, and the computation-intensive parts are more likely be offloaded to the cloud so as to accelerate the process of inference, while others will be assigned to the end users, which achieves not only a cost saving for the service provider but also privacy protection for the users.

Differentiated DNN Computation Scheduling

Features learned at early layers in the DNN can provide credible recognition for simple samples, therefore an early exit mechanism will further accelerate the recognition inference [7]. When the server receives multiple DNN computing requests, a priority-based scheduling policy will be recommended. Especially since the network edge is a supplementary computing platform, and the resources are more precious, an appropriate scheduling approach can effectively improve the utilization of the resources, and thus system efficiency.

However, most of the current priority-based approaches are only concerned with one aspect of the problem, which lacks the consideration of the balance between time cost and benefits. An efficiency-based scheduling approach is therefore demanded, which considers the per-layer feature extraction capability and inference latency simultaneously, then assigns higher priority to DNN computing requests with higher efficiency. To the best of our knowledge, the proposed differentiated computation scheduling approach is also the first priority-based approach for the edge platform at DNN layer granularity.

Edge-Assisted Distributed DNN Collaboration

Edge-computing based collaboration promises an efficient computing paradigm with both opportunities and challenges for Web-based mobile applications. In this article, we present a complete edge-computing based collaborative object recognition solution for mobile Web AR to address the following two questions:

- How can one apply distributed DNN to an edge-based collaborative scenario in an adaptive manner?
- How can one optimize the computational efficiency of the edge platform based on the characteristics of DNN?

The exploration of this new computing paradigm aims to encourage new ideas and insights for in-depth study in the upcoming 5G era. Furthermore, this paradigm is proposed not only for mobile Web AR but also for a broad mobile application area that may benefit from this edge-computing based collaborative solution.

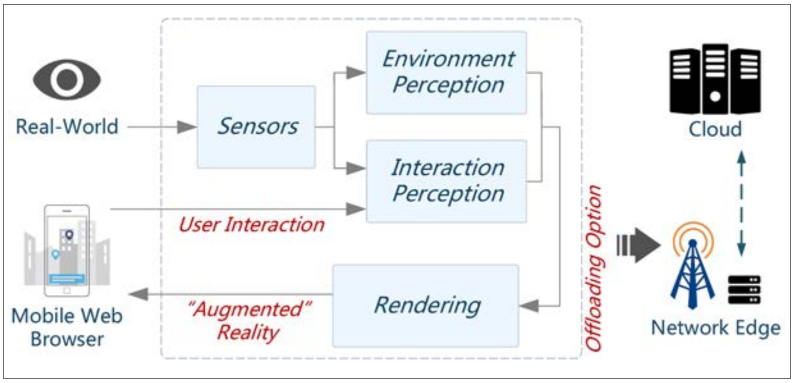


FIGURE 1. Typical mobile Web AR process.

State of the Art in Mobile Web AR

Although the first AR browser was introduced as early as 2001 [8], the Web-based mobile AR implementation, which promises a large scale service delivery, did not again receive attention until 2017, when AR.js emerged due to improvements in computing and networking.

Overall Architecture

A typical processing mechanism for mobile Web AR is illustrated in Fig. 1. The sensors, such as camera and gyroscope, are used to continually collect user ambient information for real-world perception and interaction analysis. Then the "augmented" contents will be rendered on the screen and presented to the mobile users. Unique to this Web-based mobile AR is that with the help of a computation offloading approach, it easily breaks through the limit of inefficient computing on the mobile Web browser, and thus facilitates its large-scale application as a lightweight mobile AR solution especially on the upcoming 5G networks.

DNN-Based Object Recognition on the Mobile Web

There are already many companies, such as Google and Microsoft, which have released their solutions for DNN-based recognition on the mobile Web. However, these approaches face the following challenges in practical application.

DNN Model Size: Unlike App-based solutions, every time the user experiences a mobile Web AR application, the DNN model needs to be re-downloaded. Although the data transmission in 5G networks will no longer be a performance bottleneck, the loading of DNN models for mobile Web browsers is still difficult. The compression of the DNN model is the major emerging approach for

Layer type	Inference latency models			Energy consumption models		
	α_l	β_l	γ_l	α_e	β_e	γ_e
Conv	6.240e–5	1.074e–4	–1.938e+0	9.240e–7	1.874e–6	3.810e–2
ReLU	1.534e–5	–	4.844e–1	1.435e–6	–	2.881e–1
Pooling	1.136e–5	1.313e–6	–1.695e+0	1.410e–6	1.312e–7	3.572e–1
Norm	5.182e–5	–	6.497e–1	5.187e–6	–	5.991e–1
FC	9.163e–5	3.990e–4	1.172e+0	9.213e–6	4.012e–5	1.125e+0

* We use *l* and *e* as subscripts to identify α, β, and γ in DNN inference latency and energy consumption prediction models, respectively.

TABLE 1. Parameters for the DNN Performance Prediction Models.

inference acceleration, but the flaws are also obvious: serious reduction of recognition accuracy.

Inference Latency: In addition to the inherent limitations of the mobile Web browser, the processing mechanism of JavaScript is another reason for inefficient computing. Current Web-based DNN inference solutions are basically unable to meet the real-time requirements of AR applications while providing satisfactory recognition accuracy. Although there are now emerging some advanced Web technologies, such as WebAssembly and Web Workers, time is still needed for their wide support.

The upcoming 5G networks provide another promising approach for Web-based mobile AR service to offload DNN computations to the edge or remote cloud where more powerful computing resources are available. Without the limitation of network transmission, this will be a practical approach for large-scale application of mobile Web AR by performing object recognition in a collaborative manner.

Elastic DNN Inference Collaboration

In this section, we first study the characteristics of the DNN layers, then introduce the core components of the collaboration mechanism for DNN-based object recognition in the distributed scenario.

DNN Layer Characteristics Analysis

The first requirement for achieving a fine-grained DNN computation partitioning is an in-depth study of the per-layer characteristics. Here, we discuss the proposed DNN inference latency and energy consumption prediction models, which are designed at the neural layer granularity.

Obviously, the computing capability has a close bearing on DNN inference performance. Therefore, our aim is to explore a conversion approach, which can directly obtain the inference performance for any DNN layers according to the capability of the mobile device.

To overcome the computational heterogeneity of mobile devices and servers, we take the cloud server as a standardized computing platform and use that as criteria for converting the computing capability of all mobile devices. Specifically, we investigated more than 20 mobile devices and suggested the following conversion relationship: standard mobile capability = 0.38 × mobile computing capability + 0.065.

Moreover, besides the AlexNet and VGGNet, we also study the compression DNN networks (e.g., MobileNet and ShuffleNet). Note that we focus on the per-layer input and/or output rather than the DNN layer structure when constructing the conversion relationship, which therefore can fit all types of neural network architectures.

Specifically, the DNN performance prediction models are obtained by collecting the inference latency and energy consumption of different DNN layers with randomly sampled 1000 images from the dataset (Table 1). We compared three regression methods and observed that linear regression method has a lower prediction error for the randomly sampled 300 images than logistic and polynomial methods. The obtained prediction models are detailed below:

- For convolution, pooling, and fully-connected layers, the regression models of both DNN inference latency and energy consumption can be expressed as: predicted value = capability scale × (α × input + β × output + γ).
- ReLU and normalization layers have fewer configurable parameters compared to the aforementioned three DNN layers. We only use capability scale and input feature size as the regression model variables, that is, predicted value = capability scale × (α × input + γ).

Note that the capability scale is the ratio of the standard mobile capability to standardized computing platform capability, and only the mobile energy consumption is considered; we use Battery Historian to inspect the battery status of devices.

Consequently, the proposed prediction models will provide the fundamental basis for the fine-grained and adaptive DNN computation partitioning. Moreover, they can also be applied to other DNN architectures for evaluation as a generalized solution without the execution in advance.

Adaptive DNN Computation Partitioning Pipeline

To facilitate collaborative object recognition in the distributed scenario, a DNN computation partitioning approach is required. Further, a context-aware mechanism enables adaptive partitioning between heterogeneous computing platforms, something which is completely necessary for achieving elastic collaboration. In more detail, a dynamic approach can effectively reduce the computing cost, since some of the DNN computations will be completed on the user device, while ensuring the quality of service by assigning computation-intensive parts to the edge or remote cloud, which accelerates the DNN inference process. It then provides a win-win service provisioning solution for Web-based mobile AR applications in 5G and beyond.

Based on these observations, we present an adaptive collaboration pipeline as shown in Fig. 2. Specifically, the computing and networking performance (including the bandwidth and the end-to-end latency) are collected to the cloud periodically by the performance monitoring part. Another important component is the decision-making system, which considers the collected context information and the DNN layer characteristics for the adaptive partitioning. As a result, the given DNN computations will be dynamically partitioned into several parts, then delivered to the distributed computing platforms for processing. Note that to carry out recognition inference on the mobile Web browser, it first needs to convert DNN models into the WebAssembly format. This

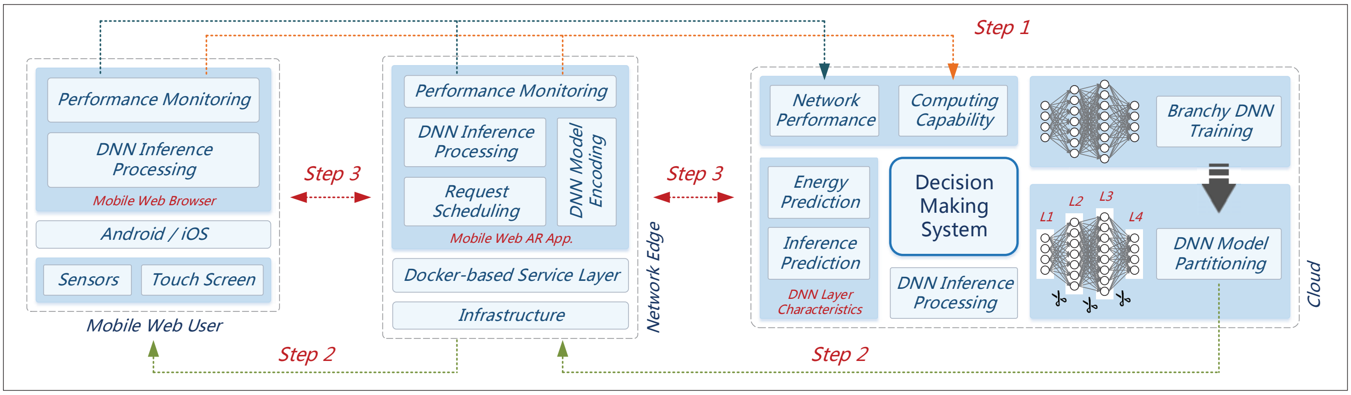


FIGURE 2. DNN computation partitioning pipeline in the collaborative scenario. The computing and networking performance are periodically collected in Step 1. And the pre-trained branchy DNN model will be partitioned into several parts based on the partitioning decision, then delivered to the network edge and mobile Web users in Step 2. Finally, the recognition process will be completed in a collaborative manner between the distributed computing platforms.

operation is performed on the edge server, which avoids the redundant storage of DNN models in the cloud and also alleviates the consumption of network resources caused by downloading the model from the cloud to the mobile Web users.

A well-executed collaborative process is more likely to offload computation-intensive parts in DNN to the cloud, so as to accelerate the inference process, while assigning others to be completed on the mobile Web browser, which will alleviate the computational burden on the service provider. But in the case of network fluctuations, on-device processing may achieve a better UX as it avoids any unacceptable transmission latency caused by offloading the DNN computations.

Elastic Partitioning Algorithm

For optimal performance of the DNN inference (either in terms of latency or energy consumption), current collaborative solutions leverage the enumeration approach to find the best computation partitioning point. However, each DNN layer is isolated. These approaches will apparently be inefficient and inflexible in an edge-based collaborative computing scenario because of the explosive growth in the decision space for a fine-grained partitioning. For example, there are only eight neural layers in AlexNet but there will be 6561 partitioning decisions when distributing these DNN computations among the cloud, network edge, and mobile Web browser.

Obviously, it will result in an unacceptable cost for orchestrating the DNN computations by using straightforward approaches. The newly emerging Reinforcement Learning (RL) method shows the effectiveness and potential for decision making and therefore provide us a new perspective for DNN computation partitioning. We therefore propose a deep reinforcement learning-based distributed DNN collaboration algorithm. The three core components in our approach are discussed:

- State consists of the DNN inference latency and mobile energy consumption at each time frame.
- Action refers to the specific partitioning decision at layer granularity for the DNN computations.
- Reward is defined as $-(T + \eta \cdot E)$, we denote by T the response latency and by E the mobile energy consumption.

However, with large state and action spaces, the value-based RL methods, such as Deep Q-Network, are all failed in our system. For this reason, we propose a Deep Deterministic Policy Gradient [9] based DNN computation partitioning approach, a policy-based RL method which outputs the probability of the actions directly according to the observed state information. The design details are as follows:

- The Actor and Critic networks are all designed with one hidden layer (i.e., fully-connected layer), which consists of 50 and 30 neurons, respectively.
- We adopt the Sigmoid activation function in the Actor network then use the piecewise function to convert successive values into a discrete action.
- The weighting factor η in the reward function is designed to adjust the latency-energy trade-off. Specifically, service providers can set η by adjusting the upper bound of latency and energy according to different AR application requirements. In detail, the upper bounds of latency and energy are T^u and E^u, if the "contribution" rate of latency and energy cost in reward is set to φ:ω, then η can be expressed as $\varphi \cdot T^u/\omega \cdot E^u$.

In practical applications, both training and execution of our proposed DNN partitioning approach are completed in the cloud. The agent generates actions based on the observed state then receives the reward at each time-step; meanwhile, the Actor and Critic networks will also be updated in the training phase. Since the network training is an iterative process with many steps in each episode, the computational complexity of these updates is therefore $O(mn)$.

Here the training process was set to 200 episodes, with each corresponding to 200 timesteps. Significantly, different application scenarios (including different DNN architectures and datasets) will obviously result in different convergence. It is therefore necessary to re-train the RL-based DNN computation partitioning agent for each application scenario. But all the training processes are completed in the cloud, where there are sufficient computing resources. By adopting an online learning mechanism, all the performance of decisions will feedback to the cloud, which will help to improve its effectiveness and practicabili-

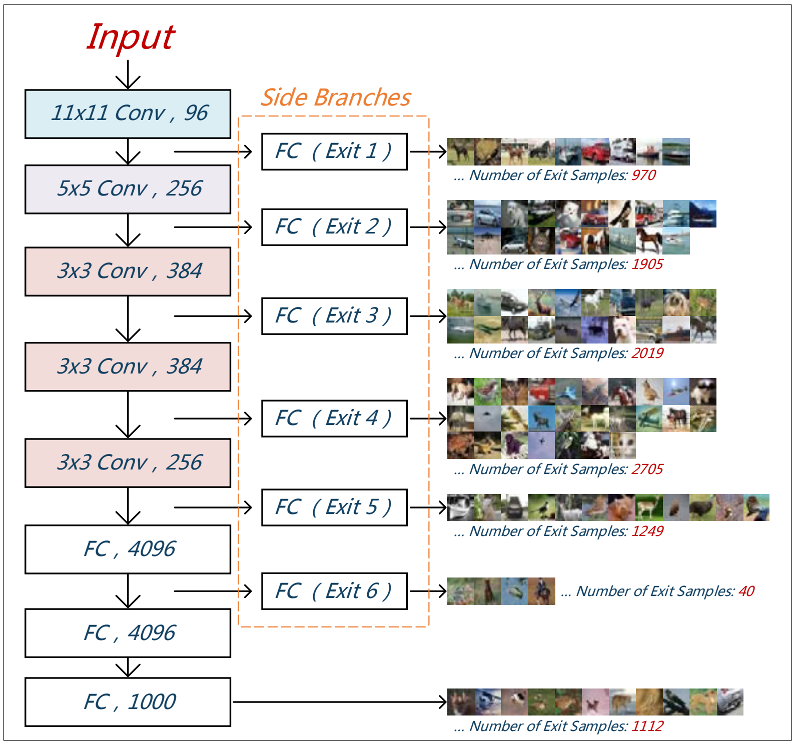

Input
11x11 Conv , 96
5x5 Conv , 256
3x3 Conv , 384
3x3 Conv , 384
3x3 Conv , 256
FC , 4096
FC , 4096
FC , 1000
Side Branches
FC (Exit 1)
... Number of Exit Samples: 970
FC (Exit 2)
... Number of Exit Samples: 1905
FC (Exit 3)
... Number of Exit Samples: 2019
FC (Exit 4)
... Number of Exit Samples: 2705
FC (Exit 5)
... Number of Exit Samples: 1249
FC (Exit 6)
... Number of Exit Samples: 40
... Number of Exit Samples: 1112

FIGURE 3. An example of the early exit mechanism for branch-based AlexNet on CIFAR-10. Early exit threshold is set to 1e-7. When the recognition accuracy exceeds the threshold, then the result will return to the users directly. Apparently, the number of samples exit from side branches is different because of the different feature extraction capability of DNN layers.

ty. Moreover, a variety of acceleration approaches [10] are also available to achieve more efficient training. Considering the simple architecture of the Actor network, it will take only a short time to generate an action in the execution phase.

Differentially Computation Scheduling

In this section, we discuss a priority-based computation scheduling policy for the sake of improving the performance of mobile Web AR application from the edge perspective.

Why Differentiation Is Needed

Many efforts have demonstrated that deeper neural networks are often able to handle more complex inputs, but the features learned at early layers are already sufficient for recognition of simple samples [7]. The adoption of an early exit mechanism (Fig. 3) can easily accelerate the inference process of the DNN since the subsequent computations no longer need to be executed.

On the other hand, resources are more precious on the network edge, which acts only as a supplementary computing platform. To maximize the utilization of edge resources, an in-depth study of the DNN computation scheduling mechanism is therefore also needed.

DNN Computation Scheduling Policy

When there are n DNN computations waiting to be processed, and each with different feature extraction efficiency $Cap_i = R_i/T_i$, $i \in [1, n]$. Here we denote by $R_i = E_i/D$ the recognition accuracy (the number of samples that can be accurately recognized by the current DNN block is E_i, and the total number of samples is D), and by T_i and W_i the DNN inference and waiting time. Our objective is to find a scheduling result to minimize the weighted processing time of the DNN computations on the edge: $\min\Sigma_{i=1}^{n}(T_i + W_i)/Cap_i$. Remarkably, the waiting time cannot exceed a certain threshold, that is, $W_i \le \mu \cdot T_i$, thereby preventing starvation.

In detail, our proposed scheduling policy consists of the following three steps:

- First, when two or more consecutive DNN layers from the same user are assigned to be completed on the network edge, they will be regarded as an integrated DNN block. The feature extraction efficiency of this block is the average of all included DNN layers.
- Then, the priority of the DNN layer or block is assigned according to their feature extraction efficiency. The edge server maintains a dynamic priority queue; DNN computations will be executed in order of priority.
- Finally, for the incoming two or more DNN layers or blocks, if they have the same feature extraction efficiency, the first arrived DNN computing request will also be served first by the edge server.

Note that because of the layer-wise structure and data dependency between successive layers, for disconnected layers that are assigned to the edge node, the subsequent layers will be activated only when the previous DNN computation has finished, then the intermediate result will be transferred to the edge server for further processing.

Obviously, by assigning a higher priority to DNN layers that have more powerful feature extraction efficiency, it can effectively reduce the overall waiting time of the system, and thus improve the resource utilization.

Performance Evaluation

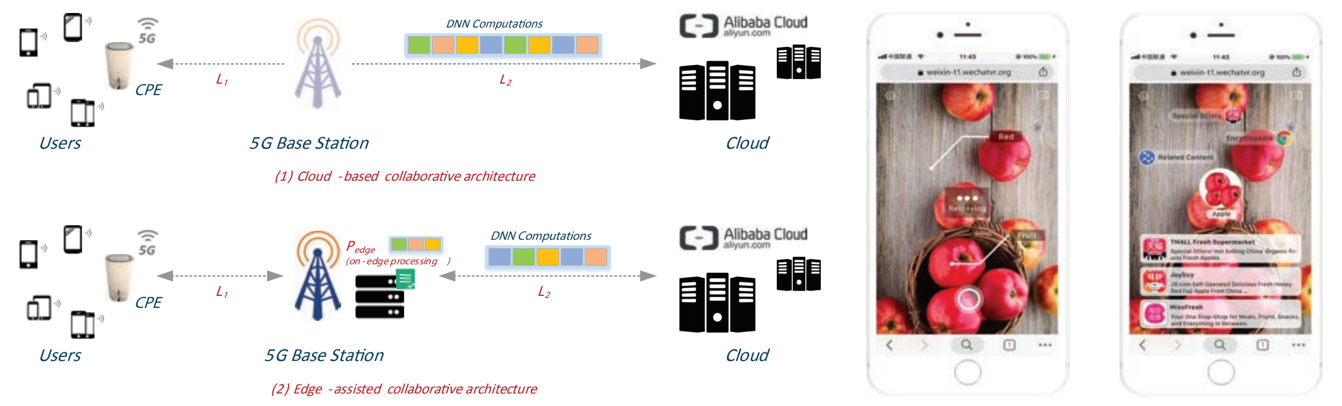
In this section, we present the experimental setting, and then detail the experimental results.

Experimental Setting

For demonstration purposes, we developed an AR-based instance retrieval and recommendation application as shown in Fig. 4b. By accessing the pre-defined URL, the relevant "augmented" information will be presented to users when they target a specific instance.

Mobile Network Performance: The experiments were conducted in an actually deployed 5G trial network supported by China Mobile Communications Group Beijing Co., Ltd. and Huawei Technologies Co., Ltd. The mobile device connects to the Internet via Customer Premise Equipment (CPE), and edge servers are deployed at the 5G base station to provide object recognition services. Specifically, the bandwidth (uplink/downlink) between the mobile device, network edge, and cloud is about 76.1/382.4 Mb/s and 73.3/542.3 Mb/s, and the end-to-end latency is about 8.76 ms and 27.04 ms.

Benchmark of Service Provisioning and Scheduling Approaches: For comprehensive comparison, we studied three kinds of service provisioning approaches. The self-contained approach performs all the DNN computations on the mobile Web browser. The cloud-based approach includes the remote and edge cloud-based ones. For the collaborative solution, we compare our proposed



(a) Collaborative architecture overview.

(b) Mobile Web AR application.

FIGURE 4. In edge-assisted DNN collaborative computing scenario, mobile user (Huawei Mate 10) connects to the cloud (AliCloud, Ubuntu 16.04, Intel Xeon E5-2683 v3) via Customer Premise Equipment (CPE), the edge server (Ubuntu 16.04, Intel Xeon E5-2600 v2) is deployed at the 5G base station in China Mobile Research Institute (CMRI) in Beijing. Compared with the cloud-based computing scenario, our proposed edge-assisted collaborative approach provides users with computing service at a close distance, which will significantly reduce the increase in response time due to the data transmission between the edge server and the cloud, that is, $P_{\text{edge}} \cdot L_2$, we denote by P_{edge} the DNN computations that are assigned to the edge server, and by L_2 the end-to-end latency between 5G based station and the cloud. Remarkably, different communication costs are caused by different DNN computation partitioning schemes. Moreover, we developed an AR-based instance retrieval and recommendation application for demonstration. Mobile users only need to access the pre-defined URL and target a specific object, such as apples, to experience the AR services.

approach with two existing collaborative approaches, Neurosurgeon and MAUI, the data-centric and control-centric approaches, respectively. But both approaches tend to offload DNN computations to the cloud due to the low communication cost in 5G networks, and therefore, they are all degraded into the cloud-based methods. To verify the effectiveness of our proposed scheduling policy, we also carry out a comparison with the following three basic scheduling approaches: First Come First Service (FCFS); Highest Value First (HVF); and Shortest Job First (SJF). The HVF and SJF approaches assign higher priority to DNN computations with the higher feature extraction capability and shorter processing time, which are the two important factors that affect system efficiency.

Benchmark of DNN Architecture: For the fine-grained computation partitioning and scheduling, the DNN architectures need to be re-designed with multiple side branches. We adopted AlexNet, VGGNet-16, ResNet-32, and MobileNet-v1 in the experiments for demonstration purpose, and then re-trained them on CIFAR-10 and CIFAR-100 datasets.

Experimental Results

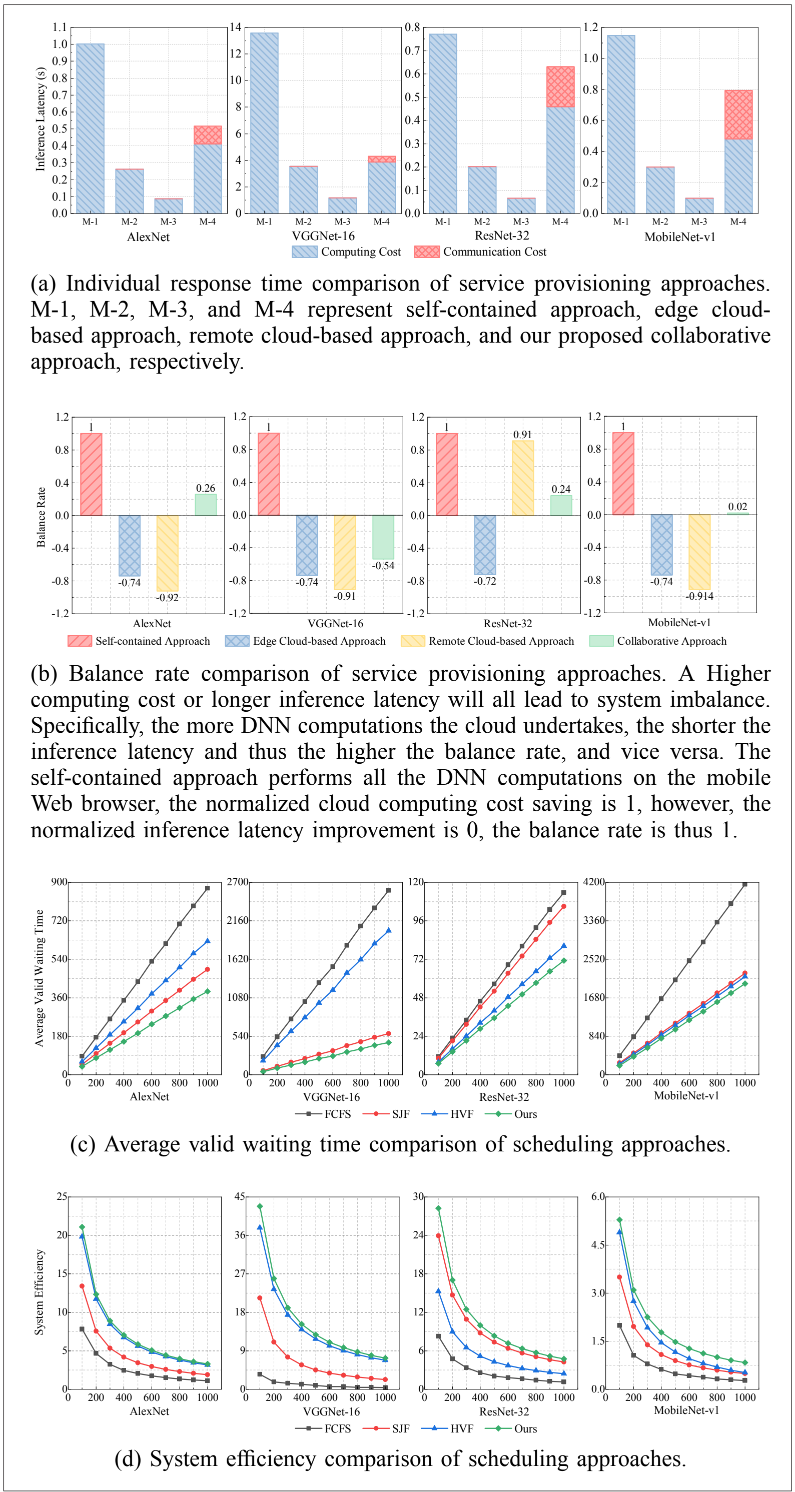
In this section, we present the comparison results to demonstrate the performance enhancement of our proposals against the benchmark approaches.

Performance Comparison of Service Provisioning Approaches: We proposed an edge-assisted collaborative approach to balance the interests of both users and service providers as mentioned earlier. The balance rate, which is defined as the difference between normalized cloud computing cost saving and normalized inference latency improvement, is therefore an important metric for the system performance evaluation. Moreover, mobile energy consumption is also considered in our experiment. The distribution of the DNN computations between the cloud server, the network edge, and mobile user were as follows according to the obtained partitioning results: 0.09/0.17/0.74 (AlexNet), 0.02/0.84/0.14 (VGGNet-16), 0.31/0.32/0.37 (ResNet-32), and 0.21/0.47/0.32 (MobileNet-v1).

As illustrated in Fig. 5, our proposed approach achieves a lower balance rate (absolute value) in all the three scenarios, which indicates that the interests of both users and service providers can be better satisfied. Specifically, our approach brings not only an improvement in the latency by about 39.75 percent on average compared with the self-contained approach, but also an almost 48.11 percent cloud computing cost savings compared with the remote cloud-based approaches. By adopting the collaborative mechanism, part of DNN computations will be offloaded to the cloud for processing. Although users need to pay the cost of communication additionally, the energy consumption by the mobile device can still be reduced significantly: it is about 70 percent on average compared with the self-contained approach in our experiment.

Performance Comparison of Scheduling Approaches: To demonstrate the improvement of edge system performance, we evaluate our scheduling approach from the following two aspects: average valid waiting time (i.e., (Σ waiting time × value)/layer number) and system efficiency (i.e., (Σ value/response time)/layer number). We denote by value the probability that the DNN layer can provide a credible recognition result for a given input. The response time includes waiting time and processing time.

The experimental results are illustrated in Fig. 5, which compares the two metrics for different lengths of request queue, from 100 to 1000. The DNN computation requests are generated randomly at layer granularity, and our proposed approach can reduce the average valid waiting time significantly by about 56.91 percent, 21.94 percent, and 33.82 percent compared with the FIFO, SJF, and HVF, respectively. Moreover, the system efficiency will also be improved: about 5.28×, 0.81×, and 0.35× value increase per unit time on average. Different DNN partitioning results will lead to different performance improvements. The shorter the aver-



(a) Individual response time comparison of service provisioning approaches. M-1, M-2, M-3, and M-4 represent self-contained approach, edge cloud-based approach, remote cloud-based approach, and our proposed collaborative approach, respectively.

(b) Balance rate comparison of service provisioning approaches. A Higher computing cost or longer inference latency will all lead to system imbalance. Specifically, the more DNN computations the cloud undertakes, the shorter the inference latency and thus the higher the balance rate, and vice versa. The self-contained approach performs all the DNN computations on the mobile Web browser, the normalized cloud computing cost saving is 1, however, the normalized inference latency improvement is 0, the balance rate is thus 1.

(c) Average valid waiting time comparison of scheduling approaches.

(d) System efficiency comparison of scheduling approaches.

FIGURE 5. The performance comparison of DNN-based recognition service provisioning and scheduling approaches: a) Individual response time comparison of service provisioning approaches. M-1, M-2, M-3, and M-4 represent self-contained approach, edge cloudbased approach, remote cloud-based approach, and our proposed collaborative approach, respectively; b) Balance rate comparison of service provisioning approaches. A Higher computing cost or longer inference latency will all lead to system imbalance. Specifically, the more DNN computations the cloud undertakes, the shorter the inference latency and thus the higher the balance rate, and vice versa. The self-contained approach performs all the DNN computations on the mobile Web browser, the normalized cloud computing cost saving is 1, however, the normalized inference latency improvement is 0, the balance rate is thus 1; c) Average valid waiting time comparison of scheduling approaches; d) System efficiency comparison of scheduling approaches.

age waiting time, the better the UX, also the more efficient the system will be.

Discussion

By arranging for a collaboration between heterogeneous computing resources, both the DNN inference latency and mobile energy consumption can be improved for object recognition in mobile Web AR applications. More generally, this collaborative approach can also benefit other mobile Web applications, as it provides a win-win solution for both users and service providers in 5G networks.

However, all these efforts currently are preliminary attempts at collaborative service provisioning for DNN-based object recognition in mobile Web AR with the help of the "edge," and much work remains to be done:

•Branchy DNN was proposed for fast inference via early exits [11]. In the training phase, the goal is to minimize the weighted sum of the loss of each exit branch, and meanwhile, the proper exit threshold is determined iteratively, which is a time-consuming process. Note that the cost for training such branchy DNN depends on the complexity of the structure and the number of added branches. Currently, it is trained on the cloud in advance, and the architecture of side branches is also designed manually. But more efficient DNN architecture design and training need further attention.

On one hand, by using the distributed deep learning [12] techniques, it can easily accelerate the training process with the help of more powerful computing resources; and on the other, more efficient branchy DNNs can be obtained benefitting from the emerging neural architecture search [13] techniques.

•In general, edge servers are deployed in a hierarchical structure, and the deployment of services on the edge platform depends on centralized management. While considering the user mobility scenarios, if the corresponding edge server is unable to complete the requests, then they will be forwarded directly to the upper edge platforms for execution. Meanwhile, the cloud can also dynamically adjust the deployment of services at the network edges according to the user mobility [14]. And in case the edge server is overloaded, by collaborating with the nearby servers, it can effectively balance the computational pressure of the edge system. Specifically, by having the multiple edges collaborate, the quality of service and the system efficiency could be further improved, and many efforts have been devoted to this field [15]. Moreover, many edge systems have also been proposed by industry, such as AWS Greengrass and Huawei KubeEdge, which provide good support for collaborative service provision and mobility management.

Conclusion

Web-based mobile AR as a lightweight and cross-platform solution brings opportunities for the large-scale application of AR as well as challenges. Compared to the self-contained and cloud-based or edge-based offloading approaches to recognize objects in AR applications, a collaborative solution is more desirable in 5G networks as it can balance the interests of both users and services providers. Specifically, assigning computation-intensive parts to the cloud accelerates the DNN inference, while placing other parts on the user device reduces the cloud computing cost. In this article, an edge-assisted distributed DNN collaborative computing approach has been discussed to improve the mobile Web AR applications in 5G networks. The evaluation results show that our proposals can balance the interests of both users and service providers. Finally, we presented a discussion of future research topics.

Acknowledgment

This work was supported in part by the National Key R&D Program of China under Grant 2018YFE0205503; in part by the National Natural Science Foundation of China (NSFC) under Grant 61671081; in part by the Funds for International Cooperation and Exchange of NSFC under Grant 61720106007; in part by the 111 Project under Grant B18008; in part by the Beijing Natural Science Foundation under Grant 4172042; in part by the Fundamental Research Funds for the Central Universities under Grant 2018XKJC01; in part by the BUPT Excellent Ph.D. Students Foundation under Grant CX2019213; and in part by the China Scholarship Council (CSC) under Grant 201906470033. The authors would like to express their sincere thanks to the anonymous reviewers for their helpful comments and valuable suggestions on this article.

References

[1] X. Qiao *et al.*, "Web AR: A Promising Future for Mobile Augmented Reality–State of the Art, Challenges, and Insights," *Proc. IEEE*, vol. 107, no. 4, Apr. 2019, pp. 651–66.

[2] X. Qiao *et al.*, "A New Era for Web AR with Mobile Edge Computing," *IEEE Internet Computing*, vol. 22, no. 4, July/Aug. 2018, pp. 46–55.

[3] X. Qiao *et al.*, "Mobile Web Augmented Reality in 5G and Beyond: Challenges, Opportunities, and Future Directions," *China Commun.*, vol. 16, no. 9, Sept. 2019, pp. 141–54.

[4] Y. Kang *et al.*, "Neurosurgeon: Collaborative Intelligence Between the Cloud and Mobile Edge," *ACM SIGARCH Computer Architecture News*, vol. 45, no. 1, Mar. 2017, pp. 615–29.

[5] E. Cuervo *et al.*, "MAUI: Making Smartphones Last Longer with Code Offload," *Proc. Int'l. Conf. Mobile Systems, Applications, and Services (MobiSys)*, ACM, June 2010, pp. 49–62.

[6] Y. Huang *et al.*, "A Lightweight Collaborative Recognition System with Binary Convolutional Neural Network for Mobile Web Augmented Reality," *Proc. 39th Int'l. Conf. Distributed Computing Systems (ICDCS)*, IEEE, July 2019, pp. 1497–1506.

[7] P. Panda, A. Sengupta, and K. Roy, "Conditional Deep Learning for Energy-Efficient and Enhanced Pattern Recognition," *Proc. 2016 Design, Automation & Test in Europe Conf. Exhibition (DATE)*, IEEE, Mar. 2016, pp. 475–80.

[8] R. Kooper and B. MacIntyre, "An Interface for a Continuously Available, General Purpose, Spatialized Information Space," *Proc. Int'l. Conf. Human-Computer Interaction (HCI International)*, 2001, pp. 5–10.

[9] T. P. Lillicrap *et al.*, "Continuous Control with Deep Reinforcement Learning," *Proc. Int'l. Conf. Learning Representations (ICLR)*, May 2016.

[10] A. Stooke and P. Abbeel, "Accelerated Methods for Deep Reinforcement Learning," arXiv preprint arXiv:1803.02811, 2018.

[11] S. Teerapittayanon, B. McDanel, and H. Kung, "BranchyNet: Fast Inference via Early Exiting from Deep Neural Networks," *Proc. Int'l. Conf. Pattern Recognition (ICPR)*, IEEE, Dec. 2016, pp. 2464–69.

[12] J. Dean *et al.*, "Large Scale Distributed Deep Networks," *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2012, pp. 1223–31.

[13] T. Elsken, J. H. Metzen, and F. Hutter, "Neural Architecture Search: A Survey," *J. Machine Learning Research (JMLR)*, vol. 20, no. 55, 2019, pp. 1–21.

[14] T. Ouyang, Z. Zhou, and X. Chen, "Follow Me at the Edge: Mobility-Aware Dynamic Service Placement for Mobile Edge Computing," *IEEE JSAC*, vol. 36, no. 10, Oct. 2018, pp. 2333–45.

[15] H. Zhang *et al.*, "Network Slicing Based 5G and Future Mobile Networks: Mobility, Resource Management, and Challenges," *IEEE Commun. Mag.*, vol. 55, no. 8, Aug. 2017, pp. 138–45.

Biographies

Pei Ren is currently working toward the Ph.D. degree at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China. His current research interests include the future Internet architecture, services computing, computer vision, machine learning, augmented reality, edge computing, and 5G networks.

Xiuquan Qiao is currently a full professor with the Beijing University of Posts and Telecommunications, Beijing, China, where he is also the Deputy Director of the Key Laboratory of Networking and Switching Technology, Network Service Foundation Research Center of State. He has authored or co-authored over 60 technical papers in international journals and at conferences, including the *IEEE Communications Magazine*, *Computer Networks*, *IEEE Internet Computing*, *IEEE Transactions on Automation Science and Engineering*, and *ACM SIGCOMM Computer Communication Review*. His current research interests include the future Internet, services computing, computer vision, augmented reality, virtual reality, and 5G networks. He was a recipient of the Beijing Nova Program in 2008 and the First Prize of the 13th Beijing Youth Outstanding Science and Technology Paper Award in 2016.

Yakun Huang is currently working toward the Ph.D. degree at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China. His current research interests include computer vision, distributed deep learning, machine learning, augmented reality, edge computing, and 5G networks.

Ling Liu [F] is currently a professor at the School of Computer Science, Georgia Institute of Technology, Atlanta, GA, USA. She directs the research programs at the Distributed Data Intensive Systems Lab, examining various aspects of large-scale big data systems and analytics, including performance, availability, security, privacy, and trust. Her current research is sponsored primarily by the National Science Foundation and IBM. She has published over 300 international journal and conference articles. She was a recipient of the IEEE Computer Society Technical Achievement Award in 2012 and the Best Paper Award from numerous top venues, including ICDCS, WWW, IEEE Cloud, IEEE ICWS, and ACM/IEEE CCGrid. She served as the general chair and the PC chair for numerous IEEE and ACM conferences in big data, distributed computing, cloud computing, data engineering, very large databases, and the World Wide Web fields. She served as the Editor-in-Chief for the *IEEE Transactions on Service Computing* from 2013 to 2016. She is the Editor-in-Chief of the *ACM Transactions on Internet Technology*.

Schahram Dustdar [F] was an Honorary Professor of Information Systems at the Department of Computing Science, University of Groningen, Groningen, The Netherlands, from 2004 to 2010. From 2016 to 2017, he was a visiting professor at the University of Sevilla, Sevilla, Spain. In 2017, he was a visiting professor at the University of California at Berkeley, Berkeley, CA, USA. He is currently a professor of computer science with the Distributed Systems Group, Technische Universität Wien, Vienna, Austria. He was an elected member of the Academy of Europe, where he is the Chairman of the Informatics Section. He was a recipient of the ACM Distinguished Scientist Award in 2009, the IBM Faculty Award in 2012, and the IEEE TCSVC Outstanding Leadership Award for outstanding leadership in services computing in 2018. He is the Co-Editor-in-Chief of the *ACM Transactions on Internet of Things* and the Editor-in-Chief of *Computing* (Springer). He is also an Associate Editor of the *IEEE Transactions on Services Computing*, the *IEEE Transactions on Cloud Computing*, the *ACM Transactions on the Web*, and the *ACM Transactions on Internet Technology*. He serves on the Editorial Board of *IEEE Internet Computing* and the *IEEE Computer Magazine*.

Junliang Chen received the B.S. degree in electrical engineering from Shanghai Jiao Tong University, Shanghai, China, in 1955, and the Ph.D. degree in electrical engineering from the Moscow Institute of Radio Engineering, Moscow, Russia, in 1961. He has been with the Beijing University of Posts and Telecommunications, Beijing, China, since 1955, where he is currently the Chairman and a professor with the Research Institute of Networking and Switching Technology. His current research interests include communication networks and next-generation service creation technology. He was elected as a member of the Chinese Academy of Sciences in 1991 and a member of the Chinese Academy of Engineering in 1994 for his contributions to fault diagnosis in stored program control exchange. He received the First, Second, and Third prizes of the National Scientific and Technological Progress Award in 1988, 2004, and 1999, respectively.