Privacy-Preserving Serverless Edge Learning With Decentralized Small-Scale Mobile Data

Shih-Chun Lin, Chia-Hung Lin, and Myungjin Lee

Abstract

In next-generation (i.e., 6G) networking systems, the data-driven approach will play an essential role, being an efficient tool for networking system management and bringing popular user applications. With those unprecedented and novel usages, existing frameworks fail to consider the complex nature of the next-generation networking system and consequently fail to be applied to future communication systems directly. Moreover, existing frameworks also fail to support popular privacy-preserving learning strategies efficiently by presenting special designs to respond to the resource-demanding nature of the aforementioned strategies. To fill this gap, this paper extends conventional serverless platforms with serverless edge learning architectures, providing a mature and efficient distributed training framework by fully exploiting limited wireless communication and edge computation resources in the considered networking system with the following three features. Firstly, this framework dynamically orchestrates resources among heterogeneous physical units to efficiently fulfill privacy-preserving learning objectives. The design jointly considers learning task requests and underlying infrastructure heterogeneity, including last-mile transmissions, computation abilities of edge and cloud computing centers, and loading status of infrastructure. Secondly, the proposed framework can easily work with data-driven approaches to improve network management efficiency, realizing AI for network promise of next-generation networking systems to provide efficient network automation. Lastly, to significantly reduce distributed training overheads, small-scale data training is proposed by integrating with a general, simple data classifier. This low-load enhancement can seamlessly work with various distributed deep models in the proposed framework to improve communications and computation efficiencies during the training phase. Based on the above innovations, open challenges, and future research directions encourage the research community to develop efficient privacy-preserving learning techniques.

Introduction

Data-driven approaches will play essential roles in next-generation networking systems, being efficient tools for networking system management and bringing popular user applications for end users. To not only hold the unique opportunities but also tackle the unprecedented challenges brought by data-driven approaches, next-generation networking systems are supposed to achieve artificial intelligence (AI) for network and network for AI simultaneously [1]. On the one hand, a network for AI promises to provide data-driven approaches friendly networking structure for increasingly popular learning-based client applications. On the other hand, AI for networks aims to employ data-driven strategies to update networking systems from network softwarization to network intelligence for more strict quality of service requirements [1], [2], [3]. Given that over 90% of data is generated and stored in end devices, distributed learning is now an irreversible trend. Furthermore, the 2018 Facebook Cambridge Analytica data scandal, in which at least 87 million users' information was disclosed, led to governments worldwide amending related laws to protect user-collected data, which has advocated for the development of privacy-preserving distributed learning research. Consequently, the development of advanced networking systems is essential in this direction.

Considering a typical scenario where a group of clients conducts privacy-preserving learning tasks wirelessly, low latency and reliable communication are crucial to facilitate the learning process and achieve the network for AI's promise. Introducing the concept of edge intelligence to handle the learning tasks using in-network edge resources could significantly reduce end-to-end latency. However, the current radio access network (RAN) [4] has limited reconfigurability and coordination among network nodes to enable edge intelligence to respond locally to learning tasks, which affects the latency improvement. In line with this, the recently proposed and advocated open-radio access network (O-RAN) architecture [5], [6], promoted by the authoritative organization in the telecommunication field, the 3rd Generation Partnership Project (3GPP), aims to provide disaggregated, virtualized, and software-based components. This will enable network nodes to be connected via open and standardized interfaces, and interoperable across different vendors, thereby achieving the aforementioned edge intelligence. However, with the emergence of O-RAN architecture, two challenges must be addressed to truly bring edge intelligence to clients. Firstly, how can we efficiently build a next-generation networking framework for service providers to

Digital Object Identifier: 10.1109/MNET.135.2200611
Date of Current Version: 10 May 2024
Date of Publication: 29 May 2023

Shih-Chun Lin (corresponding author) and Chia-Hung Lin are with the Electrical and Computer Engineering Department, North Carolina State University, Raleigh, NC 27606 USA; Myungjin Lee is with Cisco Systems, Inc., San Jose, CA 95134 USA.

control and optimize O-RAN components? Secondly, after establishing the framework, how can we incorporate popular learning-based network management tools to achieve the AI for network promise? To address these questions, we focus on important and popular privacy-preserving learning tasks as applications and develop and implement a solution for next-generation networking systems.

In this paper, we extend serverless platforms [7] (e.g., Amazon Web Services (AWS) Lambda and Google Cloud Functions) with *serverless edge learning architectures* from the networking perspective, providing an efficient framework for distributed training by fully exploiting limited communications and edge computation resources in the considered networking system. Notably, to better serve distributed training scenarios, the proposed framework can dynamically allocate resources and intelligently assign communications and computation tasks to cell sites, edge computation centers, and cloud computing centers. Based on O-RAN architecture, the intelligent allocation is according to the heterogeneous and dynamic nature of wireless conditions, computation abilities, and loading status of infrastructure. Especially, our framework is actually suitable to tackle the resource-demanding issues of current privacy-preserving learning schemes [8] to improve training efficiency. Moreover, the proposed framework can easily work with data-driven approaches to improve network management efficiency, realizing AI for the network promise of next-generation networking systems. To the best of our knowledge, this work is the first to develop a networking framework from the networking perspective for efficient distributed training and efficient network automation deployment. Furthermore, to further reduce distributed training overheads, in this work, we also offer a learning enhancement that distributed training can be operated on small-scale data to increase communications and computation efficiency dramatically and simultaneously. While conventional communications-efficient or computation-efficient algorithms work on communications or computation efficiency separately, our design brings huge potential to distributed training development with small data. We summarize the contributions of this paper below:

- We propose a novel serverless edge learning platform designed for next-generation networking systems, which aims to realize the promises of 6G networking systems based on the latest specifications. This platform can fully leverage in-network resources to support resource-demanding learning applications through its provided controlling capability. Furthermore, it enables the quick development, deployment, and execution of data-driven approaches to improve network efficiency.
- To illustrate the capabilities of our proposed platform, we implemented reinforcement learning (RL)-based network management using online training/inferring signaling procedures for popular federated learning applications. This was done within our proposed framework, utilizing the controlling capability to perform dynamic resource allocation and improve the end-user satisfaction rate.
- We also investigated the use of a few-shot learning (FSL)-enabled distributed learning strategy within our proposed framework. Our simulation results show that FSL can effectively balance the required network resources and performance achieved, enabling efficient and effective federated learning for end-users.

The rest of this paper is organized as follows. Section II reviews the latest progress of related research topics. Section III introduces distributed privacy-preserving learning over serverless edge architectures. Section IV further provides a few-shot learning enhancement to realize small-scale data training. Section V lists open research directions, and Section VI concludes the paper.

State-of-the-Arts

This section reviews the latest research progress of privacy-preserving learning and distributed training strategies.

Privacy-Preserving Learning Approaches

For privacy-preserving deep learning, training data and model parameters are the two elements that need protection. We consider a group of clients conducting distributed learning with the support of wireless networking systems while malicious clients also operate within coverage, aiming to steal private data and model parameters by eavesdropping on the information-exchanging process wirelessly. In the black-box attacking mode, shadow training and reverse engineering can be utilized to obtain or recover sensitive model parameters and data if no additional protection mechanisms are built into the information-exchanging process. Thanks to the recent developments in the privacy-preserving research area, the required data and parameter protection can be realized by federated learning and secure multi-party computation, respectively, being the considered privacy-preserving mechanisms in this paper.

However, these privacy-preserving learning strategies require significant network resources for two reasons. First, the amount of model parameters in powerful neural networks can be up to several million, requiring substantial communication resources in each epoch of the training process. Second, computationally heavy key generation and management will be required in advanced federated learning scenarios, such as vertical federated learning and federated transfer learning. Therefore, current privacy-preserving learning requires significant resources supported by network systems. To provide services to numerous end-users with different privacy-preserving learning applications, next-generation networking systems must offer dynamic resource allocation capabilities for smooth training processes.

Distributed Deep Training Strategies

The goal of existing distributed training strategies is to maintain the achieved performance while reducing communications or computation overheads. To improve communications efficiency [9], coding or model compression schemes (e.g., model pruning, model quantization) can be applied to the model parameter uploading step and download step during training. However,

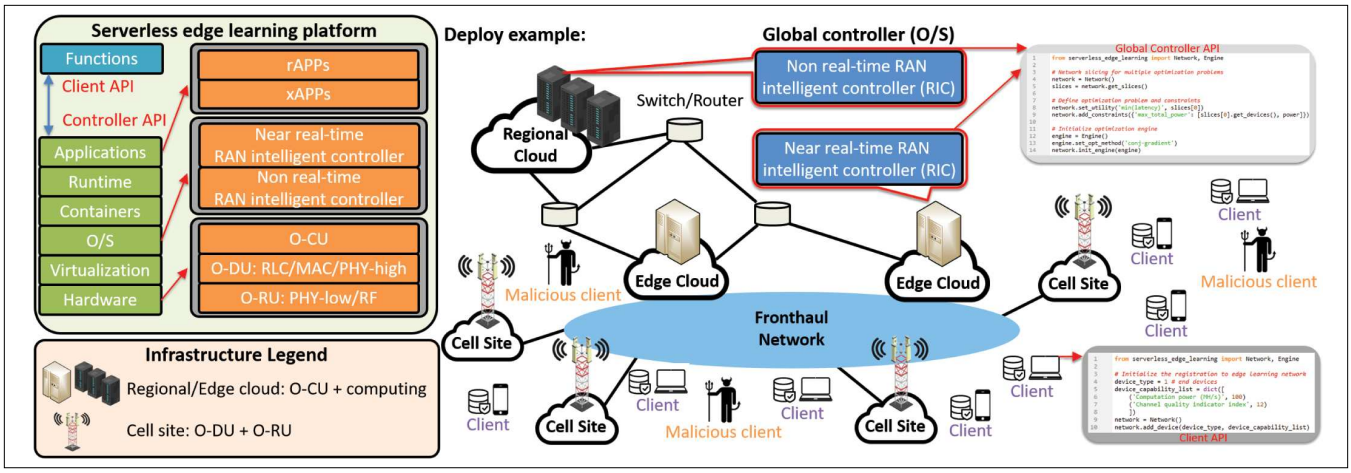

FIGURE 1. The considered scenario of serverless edge learning framework in a distributed environment. We consider clients with local training data distributed in the environment. To prevent data leakage to malicious clients, federated learning is adopted. In such cases, our main goal is to develop a networking framework to satisfy the resource-demanding learning tasks by intelligently allocating in-network resources according to dynamic resource and task status.

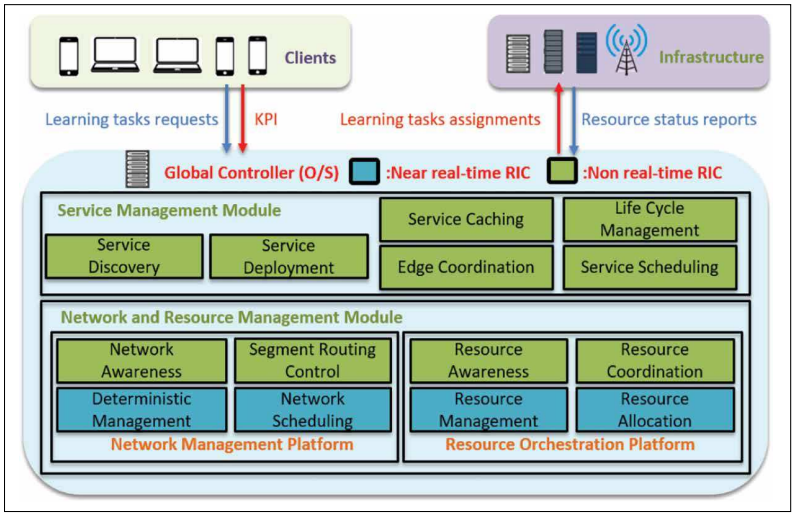

FIGURE 2. The function blocks of the serverless edge learning framework. With the provided resource status and learning task requests, the global controller is responsible for performing service management and network and resource management to facilitate distributed deep learning training by exploiting limited communications and computation resources efficiently.

these schemes work after local model training and cannot improve the computation efficiency, and they also create a computation burden by performing additional compression schemes. On the other hand, to improve computation efficiency [10], existing algorithms introduce the concept of importance sampling or similarity calculation (i.e., active sampling) to choose the samples with higher importance or lower similarity for the model training to mitigate computation overhead. However, these approaches also bring a new challenge to distributed training, as estimating the behavior of a neural network on a specific sample is very difficult. Computation-efficient algorithms need remarkable computations, which cannot be ignored in the distributed training scenarios, to perform accurate estimations, threatening their practicality. In short, the development of distributed deep training strategies is still in its infancy stage. It requires innovations as a comprehensive networking framework, which simultaneously considers the device heterogeneity and underlying edge computing capability.

Efficient Distributed Privacy-Preserving Learning Over Serverless Edge Learners

We introduce a serverless edge learning framework to facilitate resource-demanding distributed training in next-generation networking systems.

Next-Generation Networking Systems

We consider next-generation networking systems as shown in Fig. 1. That is, several clients with local data aim to conduct federated learning to finish distributed training without data leaking to malicious clients. To support those learning tasks, our goal is to create a networking platform, controlling and commanding in-network infrastructures (i.e. cell sites, edge clouds, and regional clouds) to provide efficient communication and computation services to those clients. To enable the required edge intelligence, we introduce O-RAN architecture to disaggregate traditional base station functionalities, realizing fine-granted designs to achieve our goal. Specifically, based on 3GPP specifications, next-generation infrastructures can be further divided into three categories: O-RAN central unit (O-CU), O-RAN distributed unit (O-DU), and O-RAN radio unit (O-RU), providing different communication and computation functionalities to clients in coverage. O-RU is responsible to perform lower physical layer and radio frequency (RF) front-end operations, O-DU is designed to conduct higher physical layer and lower data link layer operations in a real-time manner, and O-CU aims to provide higher data link layer and network functionalities in a non-real-time manner. We consider a general case, where each cell site is consisting of O-RU and O-CU. Moreover, each edge cloud and regional cloud is with O-CU and computing resources, connecting to cell sites via wired fronthaul. Note that our work is even suitable to work with popular

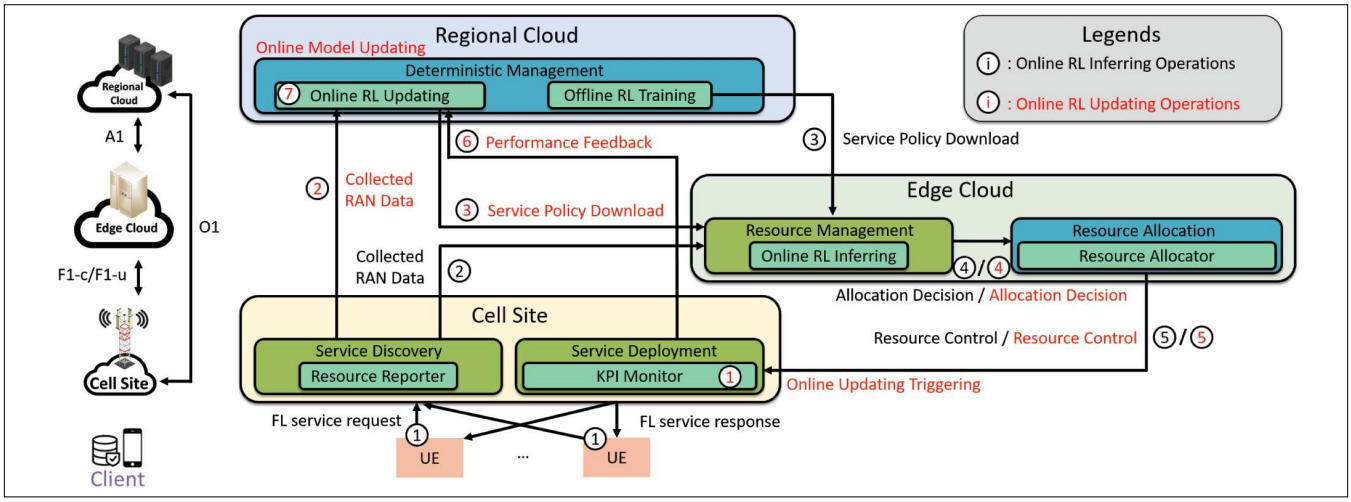

FIGURE 3. The signaling procedures to deploy RL solution in the proposed framework and execute online RL inferring and online RL updating operations.

6G heterogeneous networks (HetNets), such as Space-Air-Ground Integrated Network, to handle the underlying heterogeneity by performing different operations for different targets. Also note that how to dynamically allocate in-network resources in such systems to support resource-demanding federated learning applications is still an open question, being the main promise of the proposed framework.

Serverless Edge Learning Framework Overview

To achieve the promise, we designed and implemented a serverless edge learning platform, which is also shown in Fig. 1. Hence, all in-network infrastructure, including O-RUs, O-DUs, and O-CUs can be virtualized and be provided to the operation system (O/S) for further use. In the O/S layer, two controllers, near real-time RAN intelligent controller (RIC) and non real-time RIC, are created to execute different network management functionalities in different time-scales. Near real-time RIC is responsible for single-cell site level timely operations with stricter latency requirement (10–1000 ms) while non real-time RIC is focusing on multi-cell sites level operations (> 1 s) in the network. Moreover, to support future networking scenarios with various service requirements, different operations can be placed in the application layer as xApps and rApps, allowing near real-time RIC and non-real-time RIC to realize network automation by selecting appropriate operations to achieve different service requirements. With this design, advanced network management designs can be easily developed, deployed, and executed with next-generation infrastructure.

Fig. 2 provides a closer look at the O/S layer for different network management operations. In our implementations, we further provide two application programming interfaces (APIs) for end devices (i.e., in-network infrastructures and clients) and O/S, respectively. As a result, on the one hand, infrastructures can utilize the provided API to report resource status. On the other hand, clients can use the provided API to submit learning task requests with different quality of service requirements. The collected information will be provided to O/S for dynamic network operations. By doing so, *global control and distributed executing* are realized to fully utilize in-network resources to aid resource-demanding learning applications, being able to design the optimal network operations based on collected global information and assign those operations to specific infrastructure for executions. In our designs, two management modules and several containers are designed as function blocks for different network management operations since different network operations require different Key Performance Indicators (KPIs) and are expected to deliver different network instructions. Service level operations are belonging to non-real-time RIC, utilizing multi-cell site resources to fulfill service requirements. Similarly, network and resource awareness blocks are also belonging to non-realtime RIC to monitor all in-network infrastructures to catch the dynamic nature of networking systems, also letting routing control and resource coordination blocks design multi-cell site operations based on the provided information. Finally, network scheduling and resource management blocks are implemented in near real-time RIC to offer timely single-cell site operations.

A Case Study: RL-Based Network Management xApp

An important aspect of the proposed O/S layer design is its ability to easily support advanced AI solutions to facilitate AI for networks in next-generation networking systems. For example, popular RL-based network management algorithms can be easily supported by placing agents in the O/S layer, which can generate network operations as actions and commands for in-network infrastructure executions based on the given state (i.e., KPI, learning tasks requests, and resource status reports). In this subsection, we provide a case study that demonstrates the development of a network management xApp, which can aid in-network resource allocation utilizing RL-based solutions. We consider a practical case where an RL-based network management solution is pre-trained offline, and we introduce how to support online RL inferring and updating if required. The detailed signaling procedures are provided in Fig. 3. Our implementations include

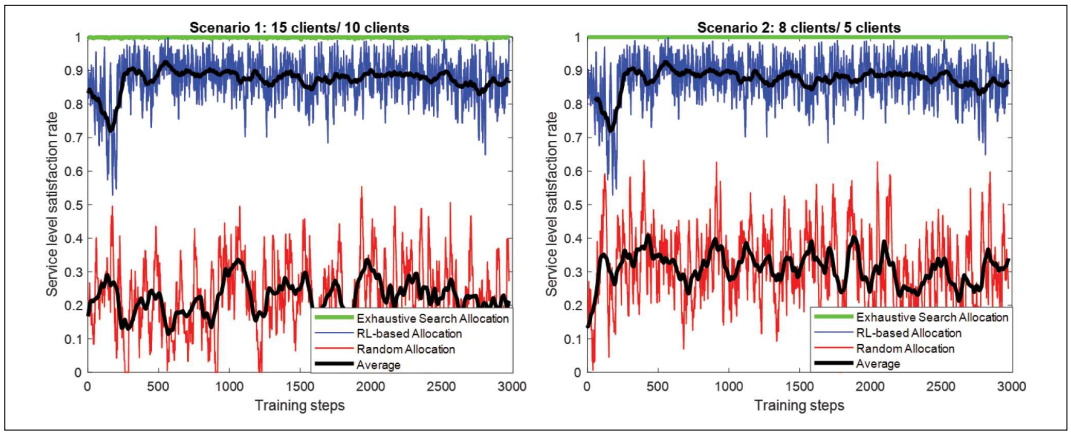

FIGURE 4. The achieved performance of the proposed federated distributed learning in two different scenarios.

building the necessary open interfaces to connect all in-network infrastructure according to 3GPP specifications. Specifically, the F1-c and F1-u interfaces connect the cell site and edge cloud for control plane and data plane data exchanges. The A1 interface is used to connect the edge cloud and regional cloud, and the O1 interface is utilized to bridge the cell site and regional cloud.

In the case of online RL inferring operations, a resource reporter is implemented at the cell site to collect service requests and relay the information to the resource management container in the edge cloud. The resource management container then retrieves the latest RL policy from the regional cloud and uses it to generate allocation decisions, which are passed to the resource allocation container in the edge cloud. Upon receiving the decisions, the resource allocator preserves computation resources and instructs the cell site to allocate communication resources to specific clients based on the decisions, completing the online RL inferring. For online RL updating operations, a KPI monitor records the satisfaction rate of clients in the cell site. If the rate drops below a predefined level for a specified time, the online RL updating is triggered to adjust the trainable parameters in the RL algorithms. To do so, the collected service requests are sent to the RL updating container in the regional cloud. Then, the adjusted policy is downloaded by the resource management container to conduct the same procedures as online RL inferring. The only difference is that after executing allocation decisions, the KPI monitor container provides feedback on the satisfaction rate of clients to the regional cloud. The procedures are iteratively performed until convergence, and the fixed policy is stored in the regional cloud for future downloading. Although existing works have discussed similar research topics to this paper, such as networking framework construction for network management and network orchestration, those papers often focus more on algorithm development or concept elaboration. In contrast, this work designs and implements the proposed framework using the latest O-RAN specification released by 3GPP to demonstrate its flexibility. Specifically, the detailed operation procedures provided in this paper enable online updating/inferring of RL-based network management xApps. These results facilitate the effortless deployment and maintenance of RL-based network management algorithms in future networking systems, being the main difference between our work and existing papers.

Network Management xApp Simulation Results

To test the above designs, we consider a single-cell site scenario with 500×500 (m^2) coverage, where several clients with mobility submit learning task requests with latency requirements to perform federated learning. Specifically, we consider that there are two groups of clients conducting federated model training with different applications. To satisfy the received requests, the cell site needs to allocate bandwidth, transmission power, and computation power to each client to upload/distribute the model parameters and encrypt the model parameters. We consider a cell site with a total bandwidth of 1.5 GHz, total computing power 50 Mega CPU cycles, and total transmission power 30 W as system configuration. To simplify the resource allocation process, we assume there are 4 discrete levels to each resource for the cell site to choose from. The goal of the considered optimization is to maximize the service-level satisfaction rate (i.e., satisfying latency requirements) obeying system constraints by dynamically adjusting each resource. To do so, in the network management block, we develop an RL-based algorithm for automated network management. Note that the considered optimization is an NP-hard problem and thus no closed-form solutions can be utilized, motivating us to consider the RL-based algorithm as an efficient searching solution. Specifically, the signal-to-noise ratio (SNR) of all clients in the previous time slot is provided as the state of the developed RL-based algorithm, and all feasible sets satisfying system constraints are considered as the action space. Moreover, we set the service-level satisfaction rate as the reward to perform direct optimization to the interested network management problem. We consider two scenarios with different numbers of clients attending the training federated learning to test our algorithm. We compare the service-level satisfaction rate results of RL-based, random search, and exhaustive search algorithms. In the exhaustive search algorithm, all possible actions are computed to obtain the optimal action while the random search algorithm picks an action randomly to serve each client. The results are presented in Fig. 4, where

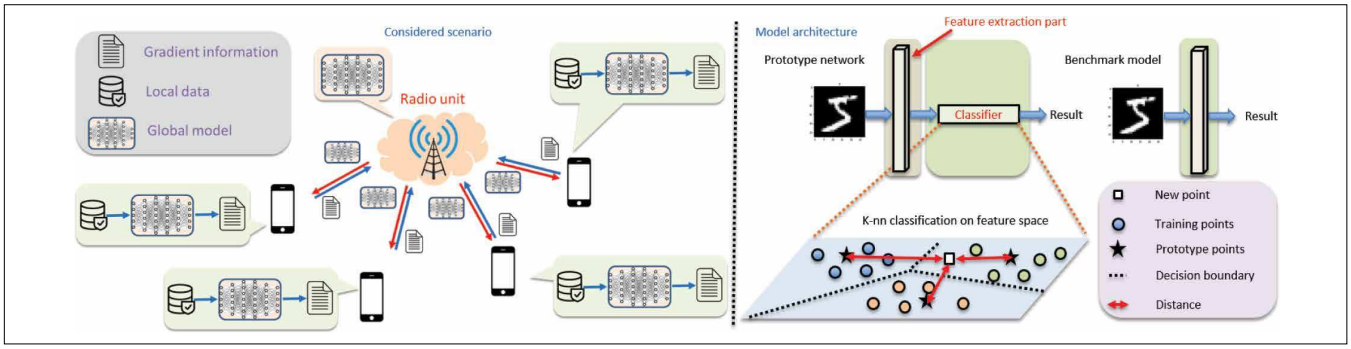

FIGURE 5. The model architectures of prototype, benchmark models, and the classification mechanism of the prototype network. In particular, we implement these models in a considered distributed scenario using a serverless edge learning framework to evaluate the training overheads of different models.

we consider two scenarios with different numbers of participants to show the generalization capability of RL-based algorithms. In each training step, the presented results are obtained via testing the achieved performance in a testing dataset, which is mutually exclusive to the training dataset to reflect the achieved performance. Due to the higher variance of the RL-based algorithm and random search algorithm, we use a solid line to present the average testing results of the previous 100 training steps. Two considered scenarios show a similar tendency, where the RL-based algorithm converges around 500 training steps. Moreover, the exhaustive search algorithm shows around 100 % satisfaction rate while the RL-based algorithm achieves 90 % and the random search algorithm provides a 30 % satisfaction rate. Note that although the exhaustive search algorithm provides the best performance, only the RL-based algorithm and random search algorithm meet near real-time requirements and the RL-based algorithm brings acceptable performance to aid network automation. Moreover, more advanced RL-based algorithms can be trained and deployed by the near real-time RIC to generate more powerful xApp for network automation. Also note that satisfaction rate is crucial when considering the training process of federated learning. Failure to satisfy the resource requirements will lead to significantly slower convergence since enough gradient information cannot be collected from clients in each training round.

Few-Shot Mobile Computing Enhancement

To further mitigate communications and computation overheads for next-generation networking systems, we provide a novel enhancement for efficient distributed training in Section II and study a use case to show that communications and computation overheads during distributed training can be improved simultaneously. Thus, this enhancement can bring a better trade-off between achieved performance and training overheads in serverless edge learning.

Prototype Network Implementation

Usually, clients are motivated to employ deep learning models with more trainable parameters to increase the capacity of deep learning models for better performance, adding communication and computation overheads during training at the same time. Alternatively, novel few-shot learning [11] matches the need for distributed deep training strategies naturally. Specifically, few-shot learning aims to classify new data, having seen only a few training samples. Unlike conventional deep learning, few-shot learning algorithms combine classical data-driven schemes (e.g., *k*-nearest neighbor algorithm) and deep learning models, limiting the number of trainable parameters to work on the over-parameterization issue of conventional deep learning models. As the number of trainable parameters in few-shot algorithms is reduced significantly, the number of training data can also be reduced as long as the overfitting does not occur, leading to improved communication and computation overheads simultaneously. Note that this is crucial when considering privacy-preserving distributed learning owing to the fact that the communication and computation demands of such applications are even higher than conventional learning applications, being the main motivation to consider few-shot learning enhancement in this work.

Based on this concept, we aim to build a specialized neural network containing a lower amount of trainable parameters to address the training challenges of distributed deep learning. As a result, the computation and communication efficiency can be improved simultaneously as the number of training samples and parameters are reduced. However, no such studies in the literature investigate the benefits of employing few-shot learning to aid distributed deep training. Therefore, we present a case study here to show the potential of this research direction and encourage researchers to contribute efforts to it. Toward this end, we develop our solution based on a prototype network [12], a classic algorithm of few-shot learning, and evaluate the benefits in distributed deep learning scenarios. Conventionally, in a deep learning classification model, pixel-level input sample will be transformed to a feature-level latent vector first (i.e., feature extraction part), then classification will be performed based on the feature-level latent vector to get the final result (i.e., classification part). The idea of the prototype network is to employ a deep learning model to perform feature extraction automatically, then use the concept of *k*-nearest neighbor to finish the classification task. As shown in Fig 5, in the prototype model, the feature extraction part is done by the neural network model, and the classification part is aided by the *k*-nearest

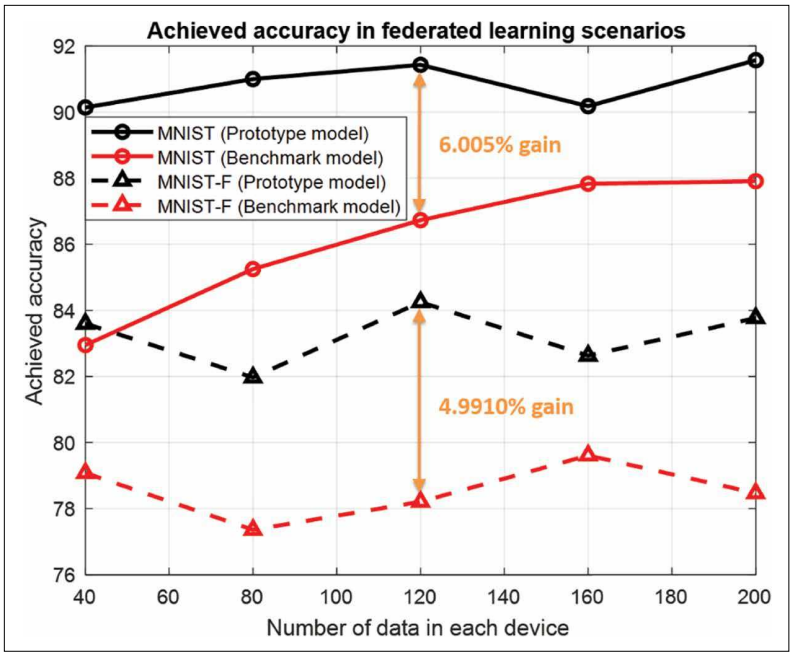

FIGURE 6. The achieved performance in federated learning scenarios when considering different learning tasks and model architectures. The x-axis indicates the number of training samples in each client, and the y-axis reflects the achieved performance. Moreover, black lines use prototype models, while red lines are employing benchmark models to conduct assigned learning tasks.

neighbor algorithm in the prototype model. As for the benchmark model, the classification and feature extraction parts are all performed by the neural network model. Note that the number of trainable parameters of the prototype network and benchmark model are the same owing to the fact that there are no trainable parameters in the *k*-nearest neighbor algorithm. Specifically, in the prototype model, given training samples belonging to each class, the deep learning-based feature extraction module will extract high-level features from the training samples to present corresponding feature vectors in feature space with a fixed dimension. For each specific class, the mean value of that class (i.e., the prototype point) in the feature space can be calculated by computing the arithmetic average of feature vectors belonging to the case. When a new data sample is obtained for classification, the deep learning-based feature extraction module will be employed again to project the data point to the feature space. Then the probability that the data point belongs to each class can be calculated by the distance between the feature vector of the new sample and the mean values of training samples belonging to each class. In order to minimize the cross-entropy loss function of the prototype network, the deep learning-based feature extraction module is forced to find the best projection way to present data in the feature space, consequently improving the final classification accuracy.

Performance Evaluation of Prototype Network

We implement the prototype network in a serverless edge federated learning framework to show the potential of few-shot learning algorithms in a distributed deep learning training environment. MNIST handwriting image classification task and MNIST-fashion clothing image classification task [13] are implemented in distributed deep learning scenarios to obtain the simulation results in this section. We also set the number of clients as four, and each client holds non-overlapping data and a local deep-learning model to perform image classification. Next, we introduce the communication and computation overhead we use in this paper to evaluate the total overhead to train a deep learning model. Considering a distributed training scenario with several clients, we assume that a deep learning model with N_w model parameters is trained, and each client is with N_d training samples. We further assume that the training process is consisting of N_e epochs. With the above information, we define the communication overhead to train a deep learning model as N_wN_e since those N_w model parameters need to be transmitted in each round of N_e epochs. Similarly, we define the computation overhead to train a deep learning model as N_dN_e since N_d training samples need to be processed in each round of N_e epochs. In our simulations, we set N_e = 100 and N_e = 200 for MNIST and MNIST-Fashion, respectively since MNIST-Fashion is a more challenging task than MNIST. When it comes to the model architecture, as for the prototype-based deep learning model, we construct a feature extractor with a single-layer neural network with 256 neurons and employ a *k*-nearest neighbor algorithm-based classifier to finish the classification task while the same neural network model is directly adopted by benchmark model without the aid of *k*-nearest neighbor algorithm. Finally, we adjust N_d and record the achieved performance in Fig. 6.

As shown in Fig. 6, one can notice that there is a constant gap between the achieved performance provided by the prototype model and benchmark model, no matter which task is considered. Specifically, with the aid of the *k*-nearest neighbor algorithm, the prototype model can provide about 6% and 5% performance gain in MNIST and MNIST-Fashion tasks, respectively. Note that the employment of the *k*-nearest neighbor algorithm will not introduce any computation or communication overheads since no trainable parameters are involved in the *k*-nearest neighbor algorithm.

Challenges and Research Directions

This section presents open challenges for the proposed serverless edge learning platform to enable efficient privacy-preserving distributed learning.

Intelligent Resource Allocation Algorithms

In this paper, we present the development of a next-generation serverless edge learning framework, providing controlling capability to aid resource-demanding learning applications. Moreover, we also demonstrate that AI for network algorithms can easily be developed, deployed, and executed via the provided framework. Despite the proposed reinforcement learning-based resource allocation mechanism, we believe advanced intelligent resource allocation algorithms can be further developed to utilize the provided controlling capability of the proposed platform fully. For example, proactive resource allocation algorithms [14] can be developed, utilizing spatial (i.e., network topology) and temporal (i.e., time-varying user location

distribution) correlation [15] to predict and preserve communication and computation resource for future application usages. Thus, the response time for serverless edge infrastructure can be further reduced.

In-Network Processing Algorithms

Another interesting research direction of the proposed framework is the development of advanced in-network processing algorithms. Specifically, with the provided controlling capability, advanced in-network processing algorithms can be executed to present multi-cell sites and multi-applications operations for enhanced efficiency in the proposed framework. For example, given that the initial weightings of neural networks will affect the neural network convergence significantly, a transfer learning-aided initial weighting design mechanism can be developed to accelerate the convergence of new applications in the network. To explain, recent literature suggests that essential features for learning development are mostly task-independent. While focused tasks are different in different research areas, basic learning features might be highly relevant. This similarity suggests that transfer learning algorithms can be used to perform *intelligent caching*, providing designed initial weightings to new applications in the network to facilitate their distributed training.

Data-Centric Algorithms

Finally, we also point to the potential of advanced edge learning algorithms for improved communication and computation efficiency in distributed learning scenarios. In light of this direction, some interesting works can still be extended. For example, current few-shot learning solutions are still limited in a classification setting, further works can be contributed to extending few-shot learning solutions to regression and structure learning settings for more usages. Moreover, due to the fact that the proposed few-shot learning distributed training strategy can still be trained with small data, the degree of freedom of training data selection is provided to select the most informative training samples to accelerate neural network convergence. Similar ideas can be found in recent literature regarding *data-centric deep learning*, focusing on the development of the aforementioned training data selection process. We expect the combination of data-centric deep learning and few-shot learning distributed training strategy to have huge potential to aid current in-network distributed learning.

Conclusion

In this paper, a serverless edge learning framework is proposed to fulfill two important promises of next-generation networking systems: network for AI and AI for networks. The framework achieves dynamic orchestration to utilize in-network infrastructure resources to aid resource-demanding learning applications. Moreover, data-driven approaches can be easily developed, deployed, and executed to achieve efficient network automation. Finally, we also demonstrate the potential of few-shot learning in distributed training scenarios by evaluating the reduced communication and computation overheads during the training phase. Open challenges and future research directions of serverless edge learning are summarized to encourage the development of efficient distributed training strategies.

Acknowledgment

This work was supported by Cisco Systems, Inc.

References

[1] K. B. Letaief et al., "The roadmap to 6G: AI empowered wireless networks," *IEEE Commun. Mag.*, vol. 57, no. 8, pp. 84–90, Aug. 2019.
[2] Y.-C. Lin et al., "Deep learning phase compression for MIMO CSI feedback by exploiting FDD channel reciprocity," *IEEE Wireless Commun. Lett.*, vol. 10, no. 10, pp. 2200–2204, Oct. 2021.
[3] C.-H. Lin, S.-C. Lin, and E. Blasch, "TULVCAN: Terahertz ultrabroadband learning vehicular channel-aware networking," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM) Workshop*, May 2021, pp. 1–6.
[4] C. L. Stergiou, K. E. Psannis, and B. B. Gupta, "InFeMo: Flexible big data management through a federated cloud system," *ACM Trans. Internet Technol.*, vol. 22, no. 2, pp. 1–22, 2021.
[5] I. Chih-Lin, S. Kuklinskí, and T. Chen, "A perspective of O-RAN integration with MEC, SON, and network slicing in the 5G era," *IEEE Netw.*, vol. 34, no. 6, pp. 3–4, Nov./Dec. 2020
[6] M. Polese et al., "Understanding O-RAN: Architecture, interfaces, algorithms, security, and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 2, pp. 1376–1411, 2nd Quart., 2023.
[7] R. Xie et al., "When serverless computing meets edge computing: Architecture, challenges, and open issues," *IEEE Wireless Commun.*, vol. 28, no. 5, pp. 126–133, Oct. 2021.
[8] B. Liu et al., "When machine learning meets privacy: A survey and outlook," *ACM Comput. Surv.*, vol. 54, no. 2, pp. 1–36, 2021.
[9] S. Shi et al., "A quantitative survey of communication optimizations in distributed deep learning," *IEEE Netw.*, vol. 35, no. 3, pp. 230–237, May/Jun. 2020.
[10] R. Han et al., "Accelerating deep learning systems via critical set identification and model compression," *IEEE Trans. Comput.*, vol. 69, no. 7, pp. 1059–1070, Jul. 2020.
[11] Y. Wang et al., "Generalizing from a few examples: A survey on few-shot learning," *ACM Comput. Surv.*, vol. 53, no. 3, pp. 1–34, 2020.
[12] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," 2017, *arXiv:1703.05175*.
[13] A. Baldominos, Y. Saez, and P. Isasi, "A survey of handwritten character recognition with MNIST and EMNIST," *Appl. Sci.*, vol. 9, no. 15, p. 3169, Aug. 2019.
[14] A. R. Mohammed, S. A. Mohammed, and S. Shirmohammadi, "Machine learning and deep learning based traffic classification and prediction in software defined networking," in *Proc. IEEE Int. Symp. Meas. Netw. (M&N)*, Jul. 2019, pp. 1–6.
[15] C.-H. Lin et al., "GCN-CNVPS: Novel method for cooperative neighboring vehicle positioning system based on graph convolution network," *IEEE Access*, vol. 9, pp. 153429–153441, 2021.

Biographies

Shih-Chun Lin (Member, IEEE) (slin23@ncsu.edu) received the Ph.D. degree from the Georgia Institute of Technology in 2017. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, North Carolina State University, leading the Intelligent Wireless Networking (iWN) Laboratory. His research interests include 6G networks, software-defined infrastructure, machine-learning techniques, mathematical optimization, and performance evaluation.

Chia-Hung Lin (clin25@ncsu.edu) received the B.S. degree in electrical engineering from Chang Gung University, Taoyuan, Taiwan, in 2016, the M.S. degree in communication engineering from National Sun Yat-sen University, Kaohsiung, Taiwan, in 2018, and the Ph.D. degree in electrical and computer engineering from North Carolina State University, Raleigh, NC, USA, in 2023. His research interests include 6G radio, intelligent networking, machine learning, and its application in wireless communications.

Myungjin Lee (myungjle@cisco.com) received the Ph.D. degree from Purdue University. He is a Senior Researcher at Cisco's Emerging Technologies and Incubation (ET&I) Group. He is broadly interested in systems and networking. He currently leads research on edge computing. Prior to joining Cisco, he worked at Salesforce as a Software Engineer. He was also an Assistant Professor at the University of Edinburgh, U.K., where he led research activities around data center networks, network telemetry and debugging, SDN, and so forth.