FedCo: A Federated Learning Controller for Content Management in Multi-party Edge Systems

Venkatraman Balasubramanian[1], Moayad Aloqaily[2], and Martin Reisslein[1]

[1]Arizona State University, AZ, USA, {vbalasubramanian; reisslein}@asu.edu
[2]Qatar University, Doha, Qatar, maloqaily@ieee.org

***Abstract*—Managing cache content at the edge is one of the many use cases of 5G-and-beyond networks. However, increasing the density of Edge Data Centers (EDCs) to service requests is a crucial problem. To overcome this problem, recent research has advanced mobile device architecture paradigms and the content caching in a Mobile Device Cloud (MDC). These two service locations (EDCs and MDC) are registered with the Mobile Network Operator (MNO), enabling the MNO to control the content placement for profit maximization. As the user demands for content items are directly related to the QoS perceived by the user, it is important to understand the future popularity of the content items and to place them appropriately. Additionally, privacy issues have increased over time because of sensitive user information being divulged at the MDC. To preserve privacy, a branch of machine learning called Federated Learning (FL) can train machine learning models leaving the data in the end user devices. The paper contributions are as follows: (1) We introduce an FL algorithm called *FedCo*, that trains a deep-neural network (DNN) to predict the user demand of a specific content, so as to manage the content files placement at EDC and MDC sites. (2) We then conduct a theoretical evaluation of user demand behavior via prospect theory to justify revenue maximization for an MNO. (3) We show numerically via a multimedia content delivery use-case how the proposed model compares favorably with two state-of-the-art designs.**

***Index Terms*—Federated Learning (FL), Mobile Device Cloud (MDC), Mobile Edge Computing (MEC), Edge Management.**

I. Introduction

In order to de-congest the backhaul network, the management of content caching has been closely investigated in the last decade. Further, reducing the delay for the last-mile users has been a major research focus. To this end, placing popular contents closer to the user so that frequently requested files can be retrieved faster, has been a key endeavour for many engineering applications that have been developed post-2015. There are two prevalent models studied in the edge caching paradigm, namely the costly edge data center (EDC) model [1] and the low cost 5g-D2D model [2]. Both of these models deployments have their own pros and cons [2], [3], [4]. In general, there is broad agreement that MEC deployments are costly, especially when they serve demand of multimedia services in upcoming 5G networks [5].

Recent research has begun to involve the mobile equipment (MEs) nodes in the multimedia delivery loop via Device-to-Device (D2D) communications. In this paradigm, MEs lend their own storage resources [6] [7] to serve each other via D2D links, thereby flexibly enhancing the edge cloud capability and scalability. In such models, all mobile devices already have a 5G subscription and MAC-IDs which are known centrally by a base station (owned by an MNO). The client requesting a service forms a D2D resource composition using the idle resources of close-by devices. Note that MEs increasingly feature acceleration modules for network-intensive functions [8], [9]. Such a resource-rich environment of MEs replacing the expensive MEC is called Mobile Device Cloud (MDC) [10]. However, despite the MDC promises, privacy issues are still prevalent. Further, users uploading contents to a central data-center inherit the limitations of the existing cloud infrastructure, such as single point of failure, latency, redundancy, and security risks. Motivated by the need to comprehensively address these issues, we propose a model that carries the benefits of edge entities with an underlying Federated Learning (FL) technique.

Recent efforts such as in [11], study the case of distributed machine learning algorithm wherein parameters are distributed across multiple edge nodes. In this model, raw data transmission to a centralized location is not considered important rather, the gradient descent results from each local node are updated and averaged at a central place. The study [11] shows that using such an environment results in close to optimal results in terms of training time with a given resource budget. Extrapolating this distributed FL to an MDC environment, we show in our model that the users at the lowest layer may request services from a 5G-D2D MDC (i.e registered with the MNO) or an EDC collocated with the 5G-NR (New Radio) base station. Having multiple content placement locations, not only forms a convenient business model, but also empowers the customers to select judiciously the QoS level that they would like to pay for. As most of the QoS related metrics depend on where the popular content items are placed, a close examination of the content placement is needed. Further, as both of these service locations (i.e., the EDC and the MDC) are registered with the MNO, the question of where the content with a high demand should be placed influences the MNO's profit margin.

A controller which is situated at the edge, e.g., within a backhaul architecture [12], [13], makes the key decisions of content placement. As shown in Figure 1, when each user processes the FL from the controller for training the model with the local data, the main goal of the end user is to upload only the necessary model parameters' updates. After updating the parameters, the recommendation for the content is registered in the controller. This recommendation is then used to place the content in an appropriate resource

Supported in part by the U.S. National Science Foundation grant number 1716121

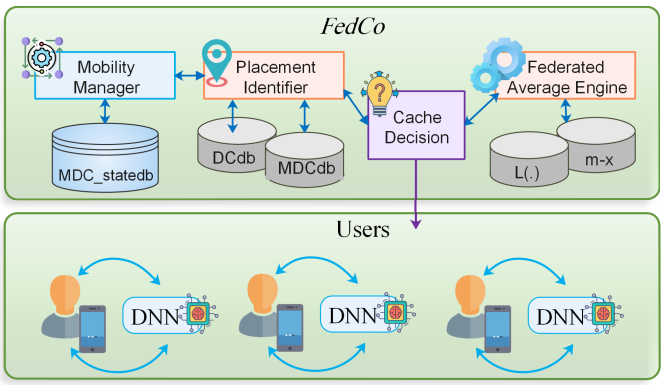

Fig. 1: When each user downloads the FL framework from the controller for training the model with the local data marked as red arrow, the main goal of the end user is to upload only the necessary parameter updates.

block (i.e. a device cloud or an EDC). The edge controller is responsible for aggregating the information from the users and the choice of the most popular content. This translates into understanding the QoS levels of the user, thereby resulting in the maximization of the MNO's revenues. Additionally, by keeping the locally trained data with the end-user, security risks are minimized. In line with these two objectives, the contributions of this paper are:

- We design a hierarchical architecture for multimedia content delivery and caching management where the MNO places the 5G-D2D compositions in different strategic areas along with other edge computing entities, allowing users to select their offloading points.
- We formulate a business model for the MNO to maximize its revenues with the key constraint of satisfying the user's QoS. The management is supported by a FL algorithm that predicts the users' demands for specific content in order to provide the exact location for its placement.
- Extensive simulations were conducted to show the effectiveness of the proposed model compared to two state of the art models, namely Random Caching [14] and Edge-Boost [2].

The rest of the paper is organized as follows. Section II delineates the novelty of the proposed solution with a brief discussion of the most recent related work. Section III presents the system design and the associated problem formulation. Section IV presents the performance evaluation and Section V provides concluding remarks.

II. Related Work

A. Software Defined Network (SDN) Paradigm

It is challenging to address the difficulties in a network whose mobility characters are not known. In [15] Wang et al. capture the ubiquity of operations at the edge that encompasses state of the art models. Authors discuss the key notions behind the merger of software defined networks, artificial intelligence and Deep Learning implementation at the edge. An SDN usage in wireless network is proposed in [16]. In this architecture, a cluster head is chosen that acts as a controller. Similar to these models, our control plane logic utilizes the OpenFlow protocol to feed control information via the south bound interface to the data plane. Figure 2 shows how the collocation of the BS-controller assists in reliably placing content.

B. Edge Caching Models and Federated Learning

Extensive studies have recently been conducted for replacing the edge caching capacity by device storage resources via D2D links [10] [17]. For instance, the authors in [18] have utilized mobile vehicles as smart caching agents to offload the caching tasks from the BS using a vehicular edge structure. However, the random vehicular movements had not been considered. Neglecting the vehicular movements significantly impairs the caching demand estimation, which in turn negatively affects the caching performance. Similarly, studies [19] [20] proposes a distributed caching framework based on the D2D assisted caching paradigm. The main difference between the two is that in [20] a delay-aware caching algorithm over D2D links is proposed that locates the best carrier. The key idea is to minimize the transmission delay and improve the throughput. These solutions regulate the caching capacity but ignore the demand variations.

In [21], an overlay structure is employed to effectively search for content providers in D2D networks without having a reliable content popularity estimate. This would in turn result in a sub-optimal revenue generation for the operator. In [22], Deng et al. discuss a roadmap for how edge computing and the interdisciplinary fields of AI and Machine Learning together brought about change in various communications and computations aspects at the edge. The main insights relate to how devices resource constraints can be overcome training machine learning models at the edge. However, this work ignores some key aspects of revenue generation that is directed towards the MNO who is deploying such a service.

C. Related Federated Learning (FL) Research

FL is a branch of machine learning that enables decentralization of the training model by letting the end users be part of the training and prediction loop. An over-the-air computation model that exploit the super-position property of wireless channels to aggregate data has been proposed in [23]. This is one of the preliminary models that provides a close observation of applying FL in wireless networks. The study [24] targets a unique FL problem in *challenged networks*, where an LSTM model is in place to take the inputs. The output obtained here is the routes that are feasible in disaster management scenarios or challenged scenarios as the authors define it. However, this model cannot be reused in cases where we need to search for computation sites to determine satisfactory QoS levels for users. The study [25] proposed a FL based proactive content caching (FPCC), which is a hierarchical architecture where users upload only the requisite updates to the edge server and keep all the remaining sensitive data on their devices. However, due to the complex nature of the model, a scalable deployment might not be possible. In contrast, our model is

purely based on the probability of outcomes that is judged via a predictive user-behaviour model.

In [26], Chen et al. study FL training models in wireless networks. In this work, the authors study the various network parameters that come into play while the local FL models are transmitted to the Base Station (BS) that aggregates them and maintains the global FL model. They formulate an optimization problem capturing wireless network parameters and users choices. We take inspiration from this work for the MDC scenario where there are network factors that come into play while uploading the local FL models to the central controller situated at the base station. The key difference from that work is that our interest is towards producing a seamless content placement/retrieval environment for cloud users which not only benefits the MNO but also provides a better Quality of Experience (QoE) to the user. To that end, while training, we ensure that our algorithm produces optimal results while minimizing losses.

Zhao et al. in [27], elaborate on the use of Federated Radio Access Networks where the combination of edge computing and AI is explained. Fundamentally, authors study the use of loss functions at the time of training and the commensurate reduction in prediction accuracy over a period of time with learning enabled Radio Access Networks. Similar to this Vu et al. [28] proposes a FL supported MIMO framework that optimizes the local accuracy, transmit power, processing frequency and data-rate. The key idea behind this paper is to study the complex non-convex behavior of the training time, computation and transmission of the computed values. The two FL models cited above have similarity in training time and accuracy predictions but differ in the overall complexity. On the other hand, although we evaluate a similar scenario with a central controller collocated with the base station, our interest is driven by multiple hierarchical controllers. The training time reduction caused by the presence of these controllers is much better compared to the other models.

Park et al. in [29] show theoretically and via simulations the different blocks of an ML based network edge, discussing how training and inference occurs when devices share their local training models with the base station in wireless networks. Similar to the above two models, a distributed, low-latency and reliable ML model is proposed that incorporates different neural network architectural splits. While our work focuses on training and accuracy, at the same time we ensure acceptable placements of the content for the other devices to retrieve from. This not only reduces the latency but also reduces cost for the users, due to the decentralized nature of the system.

In [30] Chang et al. propose an adaptive content delivery framework that is built on a Q learning technique for video streaming at the edge. The authors prove how quality of experience and fairness improves with such a design. Further, this work considers HTTP Adaptive Streaming (HAS) video chunks for its use cases, which enables the various adaptation schemes (such as Buffer-based adaptation, Rate- Based Adaptation and DASH based adaption) implemented in the paper.

In [31] Li et al. propose a deep neural network framework called Edgent that adaptively partitions a job among different computation entities. Authors make use of a predictive strategy that enables low latency communication with the edge that works seamlessly in static and dynamic scenarios. Specifically, they propose an online point detection algorithm that accommodates changing bandwidth conditions in the last mile networks. However, the execution plan proposed for the MNO is costly as the edge intelligence is distributed across devices that is outside the control of the operator. Further, job partitioning might not always result in proper resource retrieval as there are many false positives which occur in the very beginning of the training period.

TABLE I: Summary of key notations.

Notat.	Meaning
$\mathcal{R}$	Set of resource points (locations)
j	Content item (e.g., a video segm.) to be placed
o_j	Amount of space in storage units consumed by a content item j
$\mathbf{V}$	Vector of content popularities
u	User
d_u	Deadline specified by a user u
$T_{u,r,j}$	Total time spent for serving a request
$\mathcal{P}(e_u)$	Price fct., i.e., price paid by the user depending on the QoS
$L(\cdot)$	Weighting function
$m(\cdot)$	Probability of attaining the highest QoS that is achievable for r resource units
$\chi_{(\cdot)}$	Prospect pairs

III. FedCo Model

A. Design

The proposed model is depicted in Figure 2. At the control plane, we have four important modules namely, the *Mobility Manager*, the *Placement Identifier*, the *Cache Decision* making module, and the *Federated Averaging* module. The mobility manager plays the role of maintaining seamless sessions between the devices forming the MDCs. The state of a device is stored in the $Mdc\_statedb$. Every time a device moves out of a location, a state variable associated with the device is updated. The placement identifier keeps a log of where a particular content is placed, i.e., MDC or the EDC. It makes use of the on-going procedural calls to the MDCdb and the DCdb to understand the popularity of a content and manages the placement. The Cache decision is made based on the inputs received from the *Federated Averaging* module, that holds and computes the two important quantities namely the $L(\cdot)$ and m_s in our FL framework. In the data-plane, the users compute the results via a Recurrent/Deep Neural Network (DNN) (popularly called RNN) algorithm that will be uploaded to the control plane once computation is complete. We do not discuss the DNN algorithm (i.e. LSTM) implementation in depth due to space limitations.

B. System Model

We consider a typical 5G scenario where the end-users specify their demands to the controller. The content is stored in a set of virtual resources in the EDC or it can be split among the 5G-D2D MDC composition [2]. The concern is that the users request content delivery and the MNO is responsible for placing the content and providing the requisite service to the

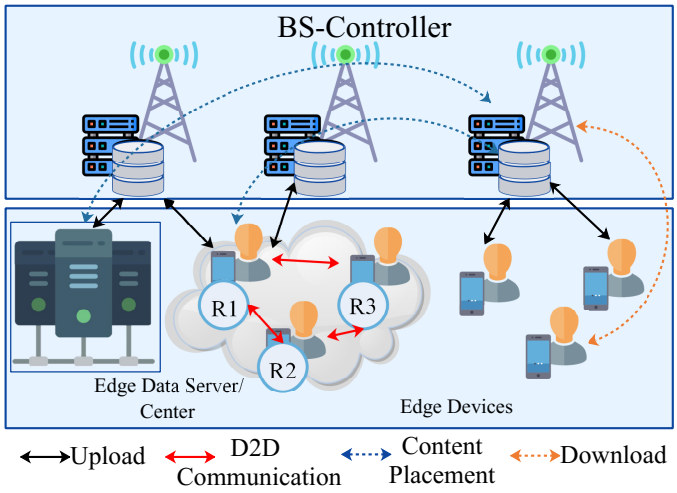

Fig. 2: *FedCo* Federated Learning Controller Framework: The edge computing units (including the idle device resources (R_1, R_2, R_3) and data-center servers) are the service locations for content placement which the users request.

user. More specifically, the user u requests a content file j which could be placed either in an edge server or across an MDC composition. The MNO, which is housed along side the central controller, is responsible for placing the content across these resource points. The request from the end user u is for one such content file. We define three popularity classes $c \in \mathcal{C}$ for these content files:

1) High demand, High popularity, $(c = 3)$
2) Average demand, Average popularity, $(c = 2)$
3) Low demand, Low Popularity, $(c = 1)$.

We assume that the MNO has some (external to our model) prediction that gives the popularity level c_j for each content file j, represented by a vector of content popularities $\mathbf{V}$.

We form a resource pool with the set of available resource points. We refer to the set of resource points (locations) as $\mathcal{R}$ and each resource point $r \in \mathcal{R}$ corresponds to a virtual resource which has some content items (files) in storage (see Table I for a summary of notations). Resource point $r \in \mathcal{R}$ has a storage capacity s_r, $s_r \leq s_{\max}$ in terms of number of storage units. Placing a content item (object) j in any of these resource points requires (consumes) o_j storage units.

C. MNO Revenue Optimization Model

Now, we are considering this problem from an MNO's perspective: content placement and the resource allocation has associated costs, such as speed of computation and storage on a specific resource unit. To satisfy a customer's request, the MNO has to satisfy the end users QoS requirements. As each of these requests has a strict deadline to be met, the MNO has to complete the computation prior to a deadline. Specifically, each user u is mapped to a deadline d_u. Once a content file j is retrieved, the user u may use it for a time $\tau_{u,r}$ during which o_j units are dedicated for this transaction on the resource pool. Denoting by $q_{u,r,j}$ the queuing time of the user u at the resource location r, the total time spent serving user u is $T_{u,r,j} = \tau_{u,r} + q_{u,r,j}$ In scenarios where requests for two contents of the same class occur, then the choice is dependent on the resource point r which can retrieve the content with the lowest queuing time. A key element here is that once the content delivery is complete, we map the task completion time to the request's deadline. The QoS satisfaction makes the customer pay more to the MNO for a future service.

In order to map this price division we define four main QoS classes for the user with κ_1 being the best QoS to κ_4 being the worst QoS. For each user u, the experienced QoS level e_u is defined based on the difference between the given deadline requirement d_u and the total service time $T_{u,r,j}$.

If $d_u - T \geq 2d_u$ then $e_u \in \kappa_1$. If $\frac{2d_u}{3} \leq d_u - T \leq 2d_u$, then $e_u \in \kappa_2$. If $\frac{d_u}{2} \leq d_u - T < \frac{2d_u}{3}$, then $e_u \in \kappa_3$. Finally, if $d_u - T \leq d_u/2$, then $e_u \in \kappa_4$.

These different QoS levels are mapped to different price levels v_1 to v_4 as per the service satisfaction. More specifically, the price function $P(e_u)$ represents the price paid by the user based on the service satisfaction level. We define this price function as

$$P(e_u) = \begin{cases} v_1, & \text{if } e_u \in \kappa_1 \\ \vdots & \\ v_4, & \text{if } e_u \in \kappa_4. \end{cases} \tag{1}$$

As the value of e_u shows the exact user demands that need to be satisfied and its price, the FL prediction of e_u is our goal.

The model represents the multi-factorial dependence on the type of content placed, the deadlines met, and the QoS that the end user perceives. Let, $n_{u,r,c} = \begin{cases} 1, & \text{if } u \text{ retrieves content of pop. class } c \text{ from res. } r \\ 0, & \text{otherwise.} \end{cases}$.
A reasonable operator revenue model is:

$$H = \sum_{u \in \mathcal{U}} \sum_{r \in \mathcal{R}} \sum_{c \in \mathcal{C}} n_{u,r,c} \left(d'_u \mathcal{P}(e_u) - \chi_{r,c} \right) \tag{2}$$

where the binary variable $d'_u \in \{0, 1\}$ is set to 1 when the deadline time associated with a user u is met, i.e., d'_u is set to 1 when the deadline d_u associated with user u is satisfied. The cost $\chi_{r,j}$ represents the MNO cost price in deploying and provisioning a content caching resource unit at resource point r for popularity class c.

The accurate prediction of the user demand plays an important role in the MNO revenue maximization. It is important to note that the mobile end users who register for these services have a highly variable behaviour. It is therefore hard to predict a user's level of desired QoS.

In order to maximize the profit earned by the MNO, the goal is to place the content appropriately for maximizing the user's QoS which is dependent on producing the profit. Since higher levels of QoS can be achieved if the content placed is retrieved and processed faster within the completion times, to achieve the goal we need to accurately predict the demands of the users for the content. To realize this prediction, we make

use of the FL framework. Therefore, the revenue maximization can be expressed as:

$$\max_{n_{u,r,c} \ \forall \ u,r,c} H \tag{3}$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{A}_r} o_j \leq s_r, \ \forall \ r \in \mathcal{R} \tag{4}$$

$$T_u \leq d_{c_u} \ \forall u \in \mathcal{U}; \ c_u \in \mathcal{C}. \tag{5}$$

The set $\mathcal{A}_r$ represents the contents j that are already allocated in resource location r. The first constraint ensures that all content items j that are on location r fit with their sizes o_j into the available space s_r. Thus, the first constraint enforces compliance with the resource capacities of the various nodes of the different resource pool types (EDC and 5G-D2D MDC), i.e., the content is placed based on the knowledge of the resource point capacities. The second constraint ensures that for a user u who requests a content file j belonging to popularity class c_u, the time $T_{u,r,j}$ spent to serve a request from user u should be less than or equal to the deadline d_{c_u} specified by the user for the class c.

D. FL Model

As the problem is hard, mainly due to the presence of the last constraint, we use the FL framework to provide a heuristic solution. The FL technique performs machine learning based training on the end users and the output is then aggregated in a central location. The local controller is the first aggregation location. Each user u that requests a content file j belonging to popularity class c maintains historical data in the vector $\mathbf{B}_{u,c}$. This vector stores the information of the content that was recently requested. Each user is passed this vector in order to predict the next content it will request. The end user is now responsible for solving the ML model having a loss function that needs to be minimized. As noted in [23], if we have a function $f_l(x)$, where l is a data sample, the output of $f_l(x)$ represents the error in data while training the model. If the users in $\mathcal{U}_c$ have requested a content belonging to class c, the loss function can be re-written from [32] as

$$F_{u,c}(z) = \frac{1}{|\mathbf{B}_{u,c}|} \sum_{u \in \mathbf{B}_{\mathcal{U},c}} f_u(z). \tag{6}$$

The objective reduces to finding z^*. It is given as the $\arg\min F(z)$ which we solve with gradient descent. The presence of a hierarchical FL model enables training at the users locally with the aim of aggregation at the edge controller. The key here is the communication between the user's training model and the aggregation at the controller within a specific number of iterations. We design the following Algorithm 1 using the steps in gradient decent to update the quantity $z_{i,c}(\alpha)$, where α is the number of iterations. We have,

$$z_{u,c}(\alpha) = \hat{z}_{u,c}(\alpha - 1) - \beta \nabla F_{u,c}(\hat{z}_{u,c}(\alpha - 1)), \tag{7}$$

where β is the learning rate and $\hat{z}_{u,c}(\alpha - 1)$ represent the global aggregation. Once this information is passed to the edge controller we have the weighted average calculated as

$$z(\alpha) = \frac{\sum_{u \in \mathcal{U}} |\mathbf{B}_{u,c}| z_{u,c}}{\sum_{u \in \mathcal{U}} |\mathbf{B}_{u,c}|}. \tag{8}$$

Algorithm 1: *FedCo*

Input: Users $\mathcal{U}_c$ requesting a service of class c
Output: Weighted Average $z(\alpha)$

1 Calculate the $z_{u,c}(\alpha)$;
$z_{u,c}(\alpha) = \hat{z}_{u,c}(\alpha - 1) - \beta \nabla F_{u,c}(\hat{z}_{u,c}(\alpha - 1))$
2 Calculate the weighted average $z(\alpha)$;
$z(\alpha) = \frac{\sum_{u \in \mathcal{U}} |\mathbf{B}_{u,c}| z_{u,c}}{\sum_{u \in \mathcal{U}} |\mathbf{B}_{u,c}|}$
3 Placement Strategy:
Input: Content Popularity Vector $\mathbf{V}$, QoS levels e_u
Output: Content Placement Location
4 Calculate : $L(v_{\kappa_1,c,j,r})$
5 Update: $m(x_{u,c,r,j})$
6 Repeat Calculation of $L(\cdot)$ until all content items have been placed.

The main benefit of FL is inheriting the advantages in securing the privacy of the end user. To that end, we have the user updating the quantity z_u. Further, using gradient enables optimal resource consumption, hence would mitigate negative impacts on the battery usage

E. Decision Making

An important step above is the decision making process. To this end, a real-life decision making process via the theoretical design called Prospect Theory has been defined in [33]. In strongly uncertain scenarios, where the customer demands for a particular content may fluctuate, prospect theory helps to model the customer behavior in a reasonable manner. According to this theory, we have a set of possible prospects which are the outcomes and the probabilities related to these outcomes. In association with our model, these outcomes are the request completion times $T_{u,r,j}$ and each resource point r implies the presence of a prospect. If a user u requests a content item j of popularity class c in resource point $r \in \mathcal{R}$, the number of prospects is computed as the total available resource points. Hence, the prospects can be written as a combination of the following pairs:

$$\mathcal{X}_{u,c,1} = (m(x_{u,c,1,j}), L(v_{\kappa_1,c,j,1})) \tag{9}$$

$$\mathcal{X}_{u,c,2} = (m(x_{u,c,2,j}), L(v_{\kappa_1,c,j,2})) \tag{10}$$

$$\vdots$$

$$\mathcal{X}_{u,c,\mathcal{R}} = (m(x_{u,c,|\mathcal{R}|,j}), L(v_{\kappa_1,c,j,|\mathcal{R}|})). \tag{11}$$

Note that we use κ_1 to establish the probability of having the highest QoS for a specific area for a set of requests; specifically, κ_1 stands for the highest QoS level, while κ_4 stands for the worst QoS.

The quantity $m(x_{u,c,r,j}) = \frac{1}{\delta_{u,c,r,j} v_{\kappa_1,c,j,r}}$ is the probability of attaining the highest QoS on resource computation point r for a content request c. It is easy to observe the intuition behind the ratio of the number of users allocated to a resource requesting a content c experiencing the highest QoS and the total users in accessing the same resource queued for the

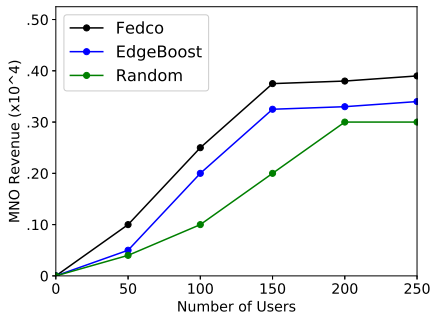

Fig. 3: MNO Revenue Accrued

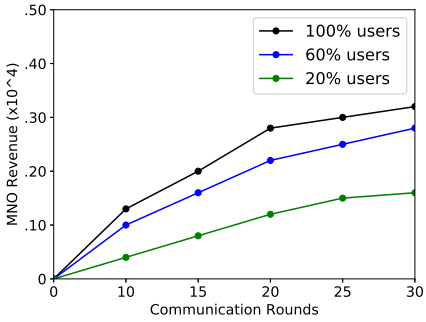

Fig. 4: FedCo MNO Revenue for different percentages of FedCo participating users

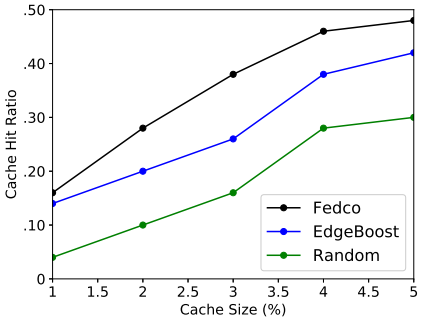

Fig. 5: Cache Hit Ratio

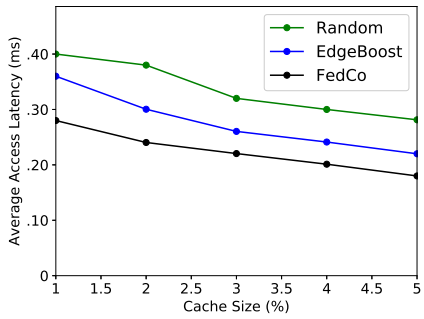

Fig. 6: Average Access Latency

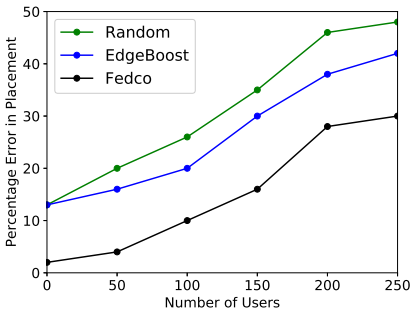

Fig. 7: Percentage of Error in Placement

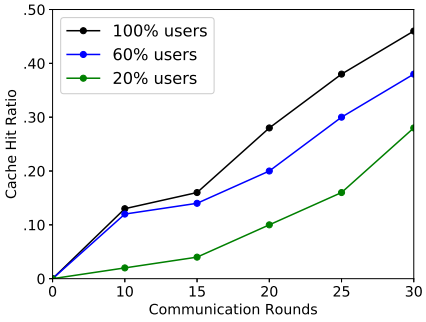

Fig. 8: FedCo Cache Hit Ratio

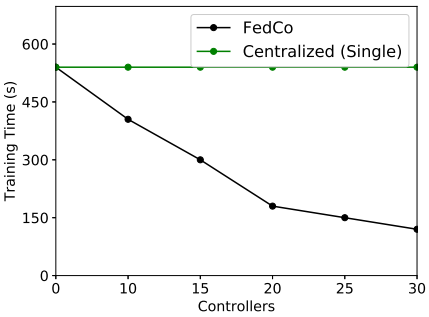

Fig. 9: Training Time

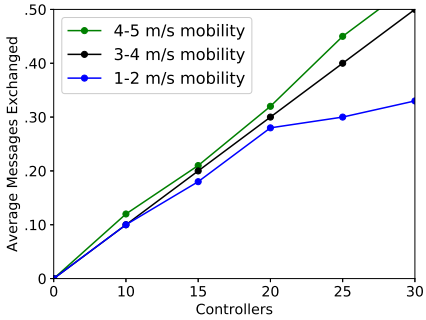

Fig. 10: FedCo controller message interactions for different mobility speeds

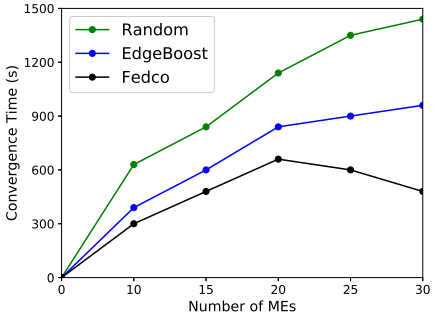

Fig. 11: As more users register with the FedCo controllers, more samples are trained which reduces convergence time

content c. The $L(\cdot)$ function is the weighting function defined as in [33]:

$$L\left(v_{\kappa_1,c,j,r}\right) = \frac{v^{\gamma}_{\kappa_1,c,j,r}}{v^{\gamma}_{\kappa_1,c,j,r} + (1 - v^{\gamma}_{\kappa_1,c,j,r})^{1/\gamma}}. \quad (12)$$

Note that prospect theory is based on evaluating a prospect. For our problem setting, the outcomes of evaluation are represented by the completion times of a request, and the service areas (MDC, EDC) are the prospects.

The definition of the value function can be written as $m(x_{u,c,r,j}) = \begin{cases} x^{\omega}_{u,c,r,j}, & \text{if } x_{u,c,r,j} \geq 0 \\ -\zeta(-x^{\theta}_{u,c,r,j}), & \text{if } x_{u,c,r,j} < 0, \end{cases}$ with $0 \leq \omega, \theta \leq 1$ and $\zeta \geq 1$. Thus, the final objective can be written as $\arg\max_r (m(x_{u,c,r,j}), L(v_{\kappa_1,c,j,r}))\ \forall u \in \mathcal{U}$. The next step after the FL framework is executed to define the content popularity $\mathbf{V}$ vector. Each content c that belongs to the vector $\mathbf{V}$ requiring o_j amount of resources where it is executed. Each of the requests arriving at this resource point accepts the content such that it minimizes the cost of the operator. Further, the perceived QoS levels play a key role in mapping the user to the content.

As observed before, each user u that requests a content is allocated to a resource point r. If there are multiple requests arriving at the resource point, we evaluate the timelines/deadlines of completion. Only the highest QoS levels that are guaranteed for a particular u are accepted and others are

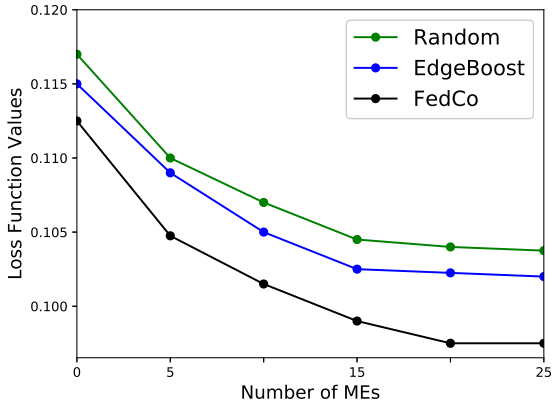

Fig. 12: As the number of samples increases, FedCo training loss reduces and remains unchanged

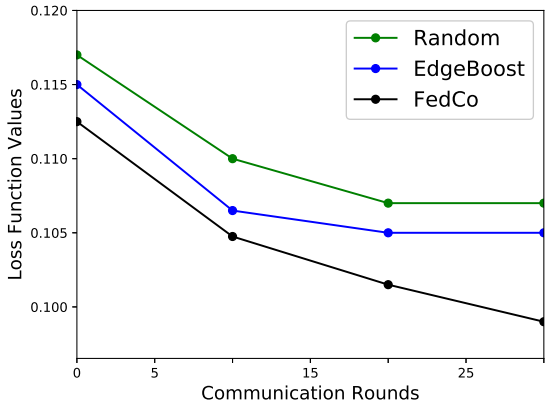

Fig. 13: FedCo training loss has a significant impact with increase in communication rounds

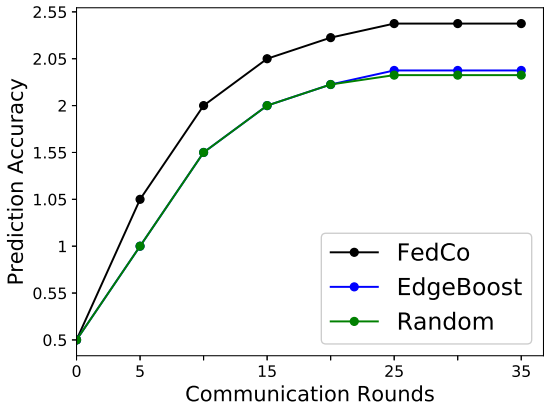

Fig. 14: FedCo accuracy outperforms traditional approaches

rejected. If there are more resources available on the resource r, then we allocate the content on that resource. The choice of the placement that guarantees the highest QoS level is chosen. However, if the QoS levels are the same, we iterate over the value functions $L(\cdot)$ for breaking the ties. After each allocation, the queues are updated for each resource point whether the choice is an EDC or 5G-D2D MDC. The user does not know the exact location of the content as only a set of resource points $\mathcal{R}$ is visible to the user.

IV. Performance Evaluation

This section first presents the performance evaluation set-up of our experiments and the data-sets generated for carrying out the simulations. We then compare our model with two state-of-the-art models, namely Random Caching by Xu et al. [14] and Edge Boost by Balasubramanian et al. [2].

A. Set-up

We use the Python simulator Mininet-Wifi for our simulations. A custom POX controller is used for control plane decision making of the OpenFlow switches (learning and forwarding). We use a pre-processed LSTM model with 4 layers and 128 neurons. We will provide the accuracy analysis for this choice Section IV-D. As multimedia caching is an important use-case of 5G, we generate video samples of varying sizes following the DASH standards [34]. We set the base-station user communication range to 500 m. We simulate a total of 250 users, each user has a 5G-D2D communication module for MDC communication with the physical parameters communication range of 150 m and bandwidth of 30 Mbps. Each video segment is divided such that the chunks are 2 s long with a bit-rate of 4000 kbps, resulting in a chunk size of 1 MB. The arrival rate of each video request follows a Poisson distribution over [2, 20]. When the requested content is determined, the corresponding segments will be consequently accessed by the ME. The simulation time is set to 1000 s and 95% confidence intervals are evaluated.

B. Evaluation

In this section, we compare the proposed FedCo model with *Random Caching* [14] and *Edge-Boost* [2]. Initially, we consider a static (non-mobile) scenario in this subsection to get a basic understanding of the performance, and then bring in mobility in Subsection IV-C. In the random caching model, the placement strategy is based on random content selection. In the Edge-Boost model, content placements are deployed with MDCs as the primary resource. In the FedCo model, however, the choice of placement is based on a pool of resources with high computation resource points of the EDC as well as the MDC. In Figure 3, the overall revenue is higher while using the FedCo module mainly because of the prediction technique applied on the content. More specifically, Figure 4 shows the accrued revenue by FedCo with varying communication rounds. Due to the learning model of FedCo, the requests that are learnt over a period of time enable the MNO to properly place the contents in appropriate locations, thereby producing adequate QoS for the end users. It is very clear that accurate predictions have led to accruing more profit for the MNO, and when the FedCo learning is removed, intuitively profits go down. In Figure 5 we can observe the Cache Hit Ratio (CHR), which is defined as the total number of requests

generated to the total number of requests satisfied from caches. The caching space of each resource unit ranges from 1%–5%, arbitrarily containing MDC compositions and EDC resource points. We can observe that the CHR is higher for FedCo than for the benchmarks. Further, a larger caching size increases the CHR. However, Edge-Boost and Random Caching lack the prediction of the cached content; therefore, their accuracy is lower by a margin of over 10% compared to FedCo.

The Average Access Latency (AAL) performance is also better for FedCo as shown in Figure 6. AAL is defined as the time interval between the request generated by the user to the first packet received for the requested content. AAL is relatively low for FedCo due to the exact content retrieval which means the extra time required to search the content is saved. Further, the FedCo module empowers the user by allowing exact mapping via the prospect theory value function. The outcome of this AAL evaluation also suggests a better QoS for the users as low access latency means better QoS.

In Figure 7, content placement failure is observed. This is the dropping probability of the requests which cannot be completed within the deadline due to placement errors. FedCo has the least percentage error in placement; although, Edge-Boost was expected to perform better in this case because of the stricter assignments with the MDC. The users who are already using the MDC for content retrieval can do so without any service disruption. This outage probability is minimal for the FL controller due to the current knowledge of the state of the devices. Thus, the controller consistently places content between the two main resource blocks from the resource pool $\mathcal{R}$. Figures 4 and 8 show the need for a large number of communication rounds for having a close to optimal CHR. The CHR becomes around 0.5 after 30 rounds for 100% of users participating while its over 0.3 and 0.4 for 60% and 20% participating users. In Figure 4, the revenue relationship with the communication rounds is clear. This variability indicates that higher information exchanged during the learning process shows that the model has a higher effectiveness. The appropriate information learnt is actually establishing the fact that the content has been placed suitably which directly translates into MNO profit. Thus, we see over 40% gain in all cases within the prescribed resource limits of $\mathcal{R}$.

In Figure 9, we calculate the time required to train our RNN/D-RNN model. We use an LSTM and consider a case where we have just one FedCo controller that takes over 70% more time compared to using 30 controllers. The main reason for this reduction with multiple FedCo controllers is that the traffic load is spread across more FedCo agents which perform training on smaller parts of the data.

C. MDC Management with Mobility

We consider a case of mobility in Figure 10 for moving speeds between [1, 5] m/s. We observe from Figure 10 that using multiple controllers results in a constant exchange of messages between the controllers. We observe that for lower mobility scenarios there are fewer messages exchanged, mainly because the devices which are registered with one controller have already been passed to the nearby controller even before the movement occurs. Further, we see that the FedCo convergence time changes with the increase in the number of MEs. In Figure 11, we observe that as the number of MEs increases, more data is used to train the LSTM model. This means that more training data samples are available and the time taken by FedCo to converge decreases. On the contrary, in state-of-the-art models, such as EdgeBoost and Random Caching, we do not observe this reduction, mainly because the control is distributed among the data-plane devices, which in turn results in a longer time to convergence.

D. Loss Function Evaluation

Figures 12 and 13 show how the loss function values, specifically, the values of the general linear regression loss function used for prediction of FL algorithms [26] values vary with the number of MEs and progressing iterations. We note that unlike FedCo, the EdgeBoost and Random Caching benchmarks, do not have prediction capability. In order to enable the loss function comparison, we generalized the loss function for EdgeBoost and Random Caching as follows. The loss function normally determines the classification errors. However, EdgeBoost and Random Caching models do not conduct classifications, they only select devices based purely on a search mechanism and provide a decision based on this search. This search for devices and placing the content implicitly follows the result of the search, which is essentially a "general" way to place content. Thus, there is going to be a loss associated with the search (depending on the input data and output data); similar to FedCo, where the loss function assesses the correctness of the prediction.

Firstly, in Figure 12 we see that as the number of users registering with the FedCo controller increases, the loss function values show a commensurate reduction, with FedCo achieving lower loss function values than the EdgeBoost and Random Caching models. This is mainly attributed to the fact that as the number of generated data samples increases, LSTM can use more data for training itself. This leads to lower training loss. As the number of MEs reaches 15, the training loss reduces faster, which shows that the initial increase in the MEs registering causes most of the reduction in the training loss, with FedCo achieving lower loss function values than the state-of-the-art EdgeBoost and Random Caching models. Similarly, in Figure 13, we see that as the iterations continue increasing, the loss function reduces by a substantial margin. This is because, as time progresses, there are enough data samples to approximate the loss function gradient. Another reason for this reduction is that there are more MEs contributing to the global FedCo controller. This enables a more extensive training, resulting in lower training loss. Overall, FedCo performs better, i.e., achieves lower loss function values, mainly because Edgeboost and Random Caching primarily search for devices to optimize the QoS, irrespective of the cost/revenue maximization objective. This comparison shows the balance between meeting user QoS and MNO revenue maximization that FedCo strives to achieve.

Finally, Figure 14 shows the prediction accuracy of all three considered controllers. The prediction accuracy is defined as

the ratio of the number of correct predictions to the total number of predictions made. We observe that FedCo achieves a better prediction compared to the state-of-the-art models. This shows how a future device is chosen for placing a content in the MDC scenario. We observe a noticeable increase in prediction for FedCo mainly because of its federated averaging algorithm that creates an aggregation of the local MEs' models which is later useful for prediction.

V. Conclusion

A Federated Learning (FL) framework named *FedCo* to predict the user demands of a particular edge content has been proposed. FedCo provides suitable content management across different placement sites. In doing so, we show how revenue maximization can be achieved for a Mobile Network Operator (MNO) that enables 5G services, such as 5G-D2D Mobile Device Cloud (MDC) and 5G-Edge Data Center (EDC) access for content caching. The paper also provides theoretical evaluation of user demand behavior modelling via prospect theory to justify how revenue maximization can be achieved by closely investigating user behavior. Finally, we show via performance comparisons how FedCo outperforms two state-of-the-art designs while considering multimedia content delivery use-case. Over 40% profit margin is accrued while the FedCo framework is deployed for 5G mobile edge content caching scenarios.

Acknowledgement

We are very grateful for interactions with Prof. Anna Scaglione of Arizona State University.

References

[1] B. Brik, P. A. Frangoudis, and A. Ksentini, "Service-oriented MEC applications placement in a federated edge cloud architecture," in *Proc. IEEE ICC*, 2020, pp. 1–6.

[2] V. Balasubramanian, M. Wang, M. Reisslein, and C. Xu, "Edge-boost: Enhancing multimedia delivery with mobile edge caching in 5G-D2D networks," in *Proc. IEEE ICME*, 2019, pp. 1684–1689.

[3] C. Long, Y. Cao, T. Jiang, and Q. Zhang, "Edge computing framework for cooperative video processing in multimedia IoT systems," *IEEE Transactions on Multimedia*, vol. 20, no. 5, pp. 1126–1139, 2018.

[4] V. Balasubramanian and A. Karmouch, "An infrastructure as a service for mobile ad-hoc cloud," in *Proc. IEEE Comp. Commun. Workshop and Conf. (CCWC)*, Jan 2017, pp. 1–7.

[5] "5G Developments," "https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails".

[6] K. Habak, E. W. Zegura, M. Ammar, and K. A. Harras, "Workload management for dynamic mobile device clusters in edge femtoclouds," in *Proc. ACM/IEEE Symp. Edge Comp.*, 2017, pp. 6:1–6:14.

[7] X. Li, X. Wang, K. Li, and V. C. Leung, "CaaS: Caching as a service for 5G networks," *IEEE Access*, vol. 5, pp. 5982–5993, 2017.

[8] L. Linguaglossa, S. Lange, S. Pontarelli, G. Rétvári, D. Rossi, T. Zinner, R. Bifulco, M. Jarschel, and G. Bianchi, "Survey of performance acceleration techniques for network function virtualization," *Proceedings of the IEEE*, vol. 107, no. 4, pp. 746–764, 2019.

[9] P. Shantharama, A. S. Thyagaturu, and M. Reisslein, "Hardware-accelerated platforms and infrastructures for network functions: A survey of enabling technologies and research studies," *IEEE Access*, vol. 8, pp. 132 021–132 085, 2020.

[10] A. J. Ferrer, J. M. Marquès, and J. Jorba, "Towards the decentralised cloud: Survey on approaches and challenges for mobile, ad hoc, and edge computing," *ACM Comp. Sur.*, vol. 51, no. 6, pp. 1–36, 2019.

[11] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[12] D. King, A. Farrel, E. N. King, R. Casellas, L. Velasco, R. Nejabati, and A. Lord, "The dichotomy of distributed and centralized control: METRO-HAUL, when control planes collide for 5G networks," *Optical Switching and Networking*, vol. 33, pp. 49–55, 2019.

[13] P. Shantharama, A. S. Thyagaturu, N. Karakoc, L. Ferrari, M. Reisslein, and A. Scaglione, "LayBack: SDN management of multi-access edge computing (MEC) for network access services and radio resource sharing," *IEEE Access*, vol. 6, pp. 57 545–57 561, 2018.

[14] C. Xu, M. Wang, X. Chen, L. Zhong, and L. A. Grieco, "Optimal information centric caching in 5G device-to-device communications," *IEEE Trans. Mob. Comp.*, vol. 17, no. 9, pp. 2114–2126, Sep. 2018.

[15] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Commun. Surv. & Tut.*, vol. 22, no. 2, pp. 869–904, 2020.

[16] A. Detti, C. Pisa, S. Salsano, and N. Blefari-Melazzi, "Wireless mesh software defined networks (wmSDN)," in *Proc. IEEE WiMob*, 2013, pp. 89–95.

[17] M. Mehrabi, D. You, V. Latzko, H. Salah, M. Reisslein, and F. H. Fitzek, "Device-enhanced MEC: Multi-access edge computing (MEC) aided by end device computation and caching: A survey," *IEEE Access*, vol. 7, pp. 166 079–166 108, 2019.

[18] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Cooperative content caching in 5G networks with mobile edge computing," *IEEE Wireless Commun.*, vol. 25, no. 3, pp. 80–87, 2018.

[19] D. Wu, L. Zhou, Y. Cai, and Y. Qian, "Collaborative caching and matching for D2D content sharing," *IEEE Wirel. Commun.*, vol. 25, no. 3, pp. 43–49, 2018.

[20] Y. Li, M. C. Gursoy, and S. Velipasalar, "A delay-aware caching algorithm for wireless D2D caching networks," *arXiv:1704.01984*, 2017.

[21] G. S. Park, W. Kim, S. H. Jeong, and H. Song, "Smart base station-assisted partial-flow device-to-device offloading system for video streaming services," *IEEE Transactions on Mobile Computing*, vol. 16, no. 9, pp. 2639–2655, 2017.

[22] S. Deng, H. Zhao, W. Fang, J. Yin, S. Dustdar, and A. Y. Zomaya, "Edge intelligence: The confluence of edge computing and artificial intelligence," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7457–7469, 2020.

[23] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 2022–2035, 2020.

[24] A. Sacco, F. Esposito, and G. Marchetto, "A federated learning approach to routing in challenged sdn-enabled edge networks," in *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, 2020, pp. 150–154.

[25] Z. Yu, J. Hu, G. Min, H. Lu, Z. Zhao, H. Wang, and N. Georgalas, "Federated learning based proactive content caching in edge computing," in *Proc. IEEE GLOBECOMx*, 2018, pp. 1–6.

[26] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 269–283, 2021.

[27] Z. Zhao, C. Feng, H. H. Yang, and X. Luo, "Federated-learning-enabled intelligent fog radio access networks: Fundamental theory, key techniques, and future trends," *IEEE Wireless Communications*, vol. 27, no. 2, pp. 22–28, 2020.

[28] T. T. Vu, D. T. Ngo, N. H. Tran, H. Q. Ngo, M. N. Dao, and R. H. Middleton, "Cell-free massive MIMO for wireless federated learning," *IEEE Trans. Wirel. Commun.*, vol. 19, no. 10, pp. 6377–6392, 2020.

[29] J. Park, S. Samarakoon, M. Bennis, and M. Debbah, "Wireless network intelligence at the edge," *Proceedings of the IEEE*, vol. 107, no. 11, pp. 2204–2239, 2019.

[30] Z. Chang, X. Zhou, Z. Wang, H. Li, and X. Zhang, "Edge-assisted adaptive video streaming with deep learning in mobile edge networks," in *Proc. IEEE Wireless Comm. Netw. Conf. (WCNC)*, 2019, pp. 1–6.

[31] E. Li, L. Zeng, Z. Zhou, and X. Chen, "Edge AI: On-demand accelerating deep neural network inference via edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 447–457, 2020.

[32] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "When edge meets learning: Adaptive control for resource-constrained distributed machine learning," in *Proc. IEEE INFOCOM*, 2018, pp. 63–71.

[33] A. Sanjab, W. Saad, and T. Başar, "Prospect theory for enhanced cyber-physical security of drone delivery systems: A network interdiction game," in *Proc. IEEE ICC*, 2017, pp. 1–6.

[34] Y. Li, A. Markopoulou, J. Apostolopoulos, and N. Bambos, "Content-aware playout and packet scheduling for video streaming over wireless links," *IEEE Trans. on Multimedia*, vol. 10, no. 5, pp. 885–895, 2008.