Privacy-Preserving Federated Deep Learning for Cooperative Hierarchical Caching in Fog Computing

Zhengxin Yu, Jia Hu, Geyong Min, Zi Wang, Wang Miao, Shancang Li

***Abstract*—Over the past few years, Fog Radio Access Networks (F-RANs) have become a promising paradigm to support the tremendously increasing demands of multimedia services, by pushing computation and storage functionalities towards the edge of networks, closer to users. In F-RANs, distributed edge caching among Fog Access Points (F-APs) can effectively reduce network traffic and service latency as it places popular contents at local caches of F-APs rather than the remote cloud. Due to the limited caching resources of F-APs and spatio-temporally fluctuant content demands from users, many cooperative caching schemes were designed to decide which contents are popular and how to cache them. However, these approaches often collect and analyse the data from Internet-of-Things (IoT) devices at a central server to predict the content popularity for caching, which raises serious privacy issues. To tackle this challenge, we propose a Federated Learning based Cooperative Hierarchical Caching scheme (FLCH), which keeps data locally and employs IoT devices to train a shared learning model for content popularity prediction. FLCH exploits horizontal cooperation between neighbour F-APs and vertical cooperation between the BaseBand Unit (BBU) pool and F-APs to cache contents with different degrees of popularity. Moreover, FLCH integrates a differential privacy mechanism to achieve a strict privacy guarantee. Experimental results demonstrate that FLCH outperforms five important baseline schemes in terms of the cache hit ratio, while preserving data privacy. Moreover, the results show the effectiveness of the proposed cooperative hierarchical caching mechanism for FLCH.**

***Index Terms*—Cooperative Caching, Fog Computing, Federated Learning, Internet-of-Things, Privacy**

I. Introduction

With the rapid development of Internet-of-Things (IoT), the number of IoT devices is expected to grow to 14.7 billion by 2023 [1]. The unprecedented amount of data generated from the emerging IoT devices places a heavy burden on traditional mobile networks. To deal with this challenge, Fog Radio Access Networks (F-RANs) has been regarded as a promising solution by pushing computation and storage functionality towards the edge of networks. F-RANs are able to reduce the service latency and alleviate the network traffic.

In the F-RANs, edge nodes, *e.g.*, Remote Radio Heads (RRHs) and Fog Access Points (F-APs), are capable of local signal processing, distributed caching and cooperative radio resource management [2]. The distributed caching among edge nodes is a key functionality of F-RANs. Placing popular contents as close as possible to users at the edge nodes can greatly reduce duplicate data transmission and service latency. As the cache capacity of edge nodes is limited compared to a huge number of contents [3], the effective utilization of the available cache capacity has a profound effect on the service performance.

Cooperative caching is an effective way to optimise the management of cache capacity for edge nodes to satisfy users' requests. However, it largely depends on the popularity of contents, which is highly dynamic and difficult to predict. Machine Learning (ML) has demonstrated its great potentials to make accurate predictions. Therefore, it is important to design a learning-based cooperative caching scheme that can learn dynamic content popularity trends to make smart caching decisions under the complex environment of F-RANs.

Despite many research efforts, there are still open challenges for learning-based cooperative caching. 1) *Privacy*: Popularity prediction models in the majority of the existing caching schemes are built on the gathered massive IoT data. However, sending these data to a central server for model training may result in severe privacy issues. 2) *Utilisation*: The contents may be stored repeatedly in the cache capacity of edge nodes, which lacks the global optimisation of cache resource utilisation. It is non-trivial to decide how and where to cache, given the limited cache sizes at the different edge nodes. 3) *Mobility*: Users are likely to request similar contents, but they frequently move from one edge node to another. This means that the cached contents at one edge node might become out of date after users move out, while another edge node has not cached the contents for the incoming users. Thus, the lack of consideration of user mobility may lead to low cache efficiency.

To address the above challenges, we propose a privacy-preserving Federated Learning based Cooperative Hierarchical edge caching scheme (FLCH). It enables to make the intelligent decision for caching contents at the edge while protecting data privacy. The FLCH scheme trains a shared global learning model at the F-AP with training data distributed over users' IoT devices. Only the parameters of the trained local models are uploaded to the central server, instead of the IoT data [4]. To further protect the privacy of user data, the FLCH scheme integrates a differential privacy mechanism. We consider a Fog Computing scenario that consists of three tiers. The bottom tier contains users equipped with IoT devices. The middle tier includes F-APs with small cache storages. The top tier has a BaseBand Unit (BBU) pool with large cache storage. This hierarchical caching architecture achieves better utilisation

Z. Yu, J. Hu, G. Min, Z. Wang, W. Miao are with the Department of Computer Science, College of Engineering Mathematics and Physical Sciences, University of Exeter, Exeter, EX4 4QF, U.K. Email: {zy246, J.Hu, G.Min, zw300, Wang.Miao}@exeter.ac.uk

S. Li is with the Department of Computer Science and Creative Technologies, University of the West of England, Bristol BS16 1QY U.K. Email: shancang.li@uwe.ac.uk

Corresponding authors: Jia Hu and Zi Wang

of available caches with mobile users, since the requested contents can be obtained from the local F-AP, neighbour F-APs and BBU.

The major contributions of this work are as follows:

1) A privacy-preserving federated learning based caching scheme is designed to make caching decision by analysing the content retrieval history and context information of users. In this caching scheme, the training data is left locally on IoT devices and a shared global model is learnt by aggregating locally-computed models. An integrated differential privacy mechanism is applied to achieve a strict privacy guarantee.

2) A hierarchical cooperative caching architecture is proposed to leverage the horizontal cooperation between the F-APs and vertical cooperation between the BBU pool and F-APs in order to enhance the overall caching performance and global cache resource utilisation.

3) We propose a one-class collaborative variational autoencoder (OCC-VAE) to predict content popularity. The variational autoencoder is utilised to extract the hidden representations of users and contents. Moreover, the one-class collaborative filtering is exploited to effectively process the input data for a better recommendation of popular contents.

The remainder of this paper is structured as follows: Section II introduces the related work. Section III describes the system architecture of the proposed FLCH caching scheme. The detailed implementation of the FLCH is presented in Section IV. Section V provides the performance evaluation of the FLCH. Finally, Section VI concludes this paper.

II. Related Work

Extensive efforts have been made by using ML techniques to solve problems in different aspects of IoT, such as communication [5], privacy [6], and security [7] [8]. For example, Sagduyu *et al.* [7] utilised adversarial machine learning to attack and defend IoT networks. Salimitari *et al.* [8] designed an AI-enabled blockchain platform with a two-step consensus protocol to improve the fault tolerance of hyperledger fabric. In this section, we summarise the existing works about cooperative caching and federated learning in IoT and smart environment paradigms, respectively.

A. Cooperative Caching

Yang *et al.* [9] proposed a cooperative caching strategy, where base stations within a local cloud cooperate with each other for caching contents. Li *et al.* [10] developed a cooperative caching system in wireless networks. This caching system is combined with Device-to-Device (D2D) transmission, which mobile devices collaboratively cache popular contents. All the above caching schemes assume that the content requests from users are followed by Zipf distribution. In reality, content popularity is dynamic and has temporal and spatial dependencies, which is hard to be accurately modelled. Therefore, ML has been widely used in the cooperative caching scheme to improve caching efficiency.

Hou *et al.* [11] investigated a cooperative proactive caching scheme in Mobile Edge Computing (MEC) to reduce transmission cost and improve user's quality of experience by exploiting transfer learning approach. Mehrizi *et al.* [12] presented a proactive cooperative caching scheme by utilising a probabilistic dynamical model to predict content popularity. Variational Bayes approach is proposed to tackle overfitting. Zhang *et al.* [13] designed a spatially cooperative caching strategy to improve caching efficiency and reduce the storage space taken by optimising the caching probabilities of cache nodes. Qiao *et al.* [14] developed a cooperative caching scheme that is based on deep reinforcement learning. A double time-scale markov decision process is exploited to model the caching problem to update the caching contents. Huang *et al.* [15] introduced a cluster-based cooperative caching strategy in vehicular named data networking with considering the mobility of vehicles. Mo *et al.* [16] presented a cooperative caching strategy by considering content popularity and making full use of the rules of node distribution. Wu *et al.* [17] proposed a social-aware cooperative caching scheme to improve content sharing performance and utilise cache resource of users. Gao *et al.* [18] exploited a cooperative coded caching scheme by using reinforcement learning and the maximum-distance separable coding. It also models the proposed caching scheme as a Markov decision process to realise the dynamic content popularity. Yang *et al.* [19] developed a recurrent neural network based cooperative caching scheme to predict user mobility and content popularity.

Most of the prior works on cooperative caching schemes are designed for a highly controlled environment, where training data needs to be uploaded to a central server for processing. Security and privacy concerns are key obstacles for the existing cooperative caching schemes. Thus, federated learning has been considered as a promising framework to train ML models without sharing data.

B. Federated Learning

Federated Learning (FL) is firstly proposed by Google [4], which presents a new approach to fitting ML techniques into the edge of networks. Mcmahan *et al.* [4] developed the primitive FL protocol. It performs synchronous optimization in federated settings. Many FL variants have been designed to improve FL from different aspects such as communication cost, model accuracy, and round efficiency [20]. Wang *et al.* [21] developed a control algorithm to decide the interval of global model aggregation adaptively. Nishio and Yonetani [20] designed a protocol to filter out slow users in MEC framework, which is based on the estimated work time of users, in order to reduce round length. Konevcny *et al.* [22] proposed the sketched updates and structured updates to reduce communication costs. Xie et al. [23] introduced an asynchronous federated learning (FedAsync), which adopts the non-blocking update of the global model and regularises local optimization. Similar to [23], Sprague *et al.* [24] exploited an asynchronous protocol to train a global model for a geo-spatial application, which allows users to join the training halfway. However, Chen *et al.* [25] proved that stochastic gradient descent in a synchronous manner outperforms asynchronous methods in terms of model accuracy.

The distinct advantage of FL is to protect users' privacy and reduce security risks, because only the parameters of the model

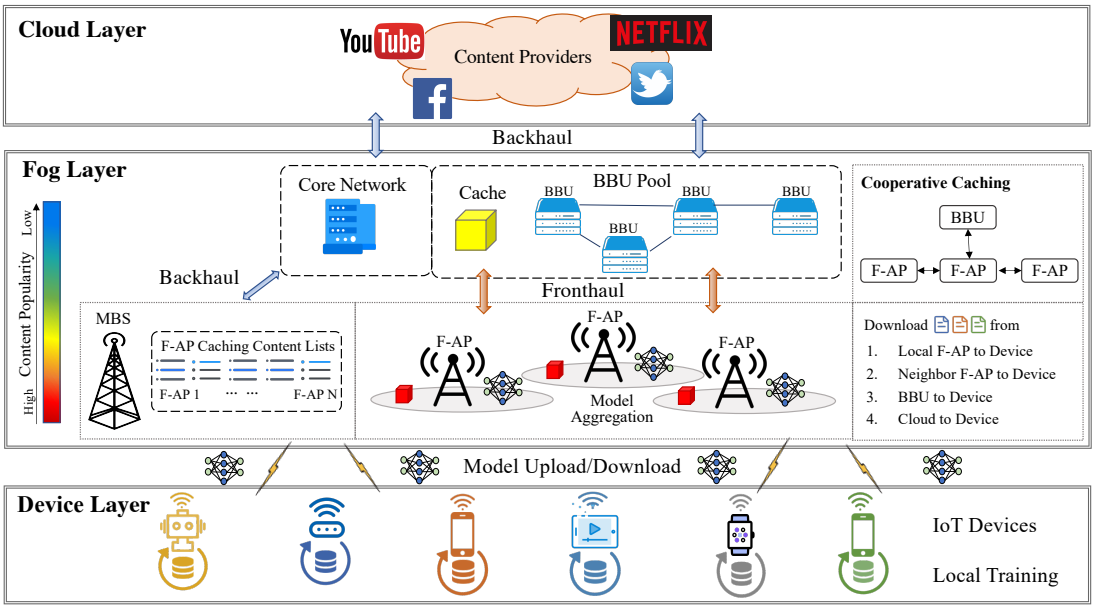


Fig. 1. System Architecture of the Federated Learning based Cooperative Hierarchical Caching for F-RANs.

are uploaded to the server, instead of users' data. However, some works demonstrate that analysing the model parameters from users still can divulge the users' private data and the model remains as attack surfaces. Therefore, we proposed a privacy-preserving federated deep learning for cooperative hierarchical caching scheme to improve caching efficiency and protect the privacy of user data.

III. Overall Design of Federated Learning Based Cooperative Hierarchical Caching in F-RANs

A. Fog Radio Access Networks

The F-RANs integrate fog computing with radio access networks (RANs), which evolved from cloud radio access networks (C-RANs) [26]. In C-RANs, a crowd of RRHs are deployed distributively within a particular region and are connected to a centralised BBU pool via high-bandwidth fronthaul links. A BBU pool greatly reduces power consumption and largely improves resource utilisation, while it also places a heavy burden on the fronthaul at the same time, resulting in the high service latency for users. F-RANs address the above issues of C-RANs through extending computing and caching functions to the edge of the network. For example, F-APs, which evolved from traditional RRH, are equipped with caches in F-RANs and are able to manage the caches flexibly.

B. The System Architecture of Federated Learning based Edge Caching

Fig. 1 illustrates the system architecture of our proposed federated learning based edge caching scheme in F-RANs. We consider a three-level Device-Fog-Cloud hierarchical structure. The Device level distributes users who are equipped with IoT devices. The Fog level consists of two tiers. The upper tier contains a BBU pool, while a Macro Base Station (MBS) and some F-APs are located in the lower tier. The MBS and F-APs all connect to the BBU pool. More specifically, the F-APs are connected to the BBU pool via high-speed fronthaul links. The MBS connects to the BBU pool with a reliable backhaul link. It is responsible for delivering the overall control signalling and providing seamless coverage for IoT devices. The Cloud level is the Internet where content providers are located remotely. In F-RANs, the BBU pool and F-APs are equipped with caches. We assume that the storage capacity for the BBU and F-APs are up to M and N contents, respectively, where $M \geqslant N$.

The aim of our work is to take advantage of distributed caching storages at the F-RANs. It requires to know content popularity distribution. However, content popularity is subject to change dynamically due to the mobility of users. Different users prefer different contents since the preference of users are various. The preference of users may link to many factors, such as gender, age and occupation. Even the location of connected users, the connected time of day and the type of connected equipment device also influence the preference of users and further affect the content popularity. Hence, the popularity of contents changes according to the fluctuating users and their context. In our scheme, the content popularity decides what and where to cache at edge nodes.

Federated learning (FL) is to train a shared high-quality model distributively without gathering users' data. In our proposed FL based caching scheme, each user associated with an F-AP performs local training by using its own data. The parameters of the training model from these users are then aggregated at the F-AP to jointly learn a model that is used to predict the local content popularity, as shown in Fig. 1. Based on the predicted content popularity, each user uploads a list of popular contents to the local F-AP. The F-AP aggregates the lists from each connected user and constructs an aggregated list of local popular contents that will be uploaded to the MBS. The N highly popular contents are selected to cache at F-APs,

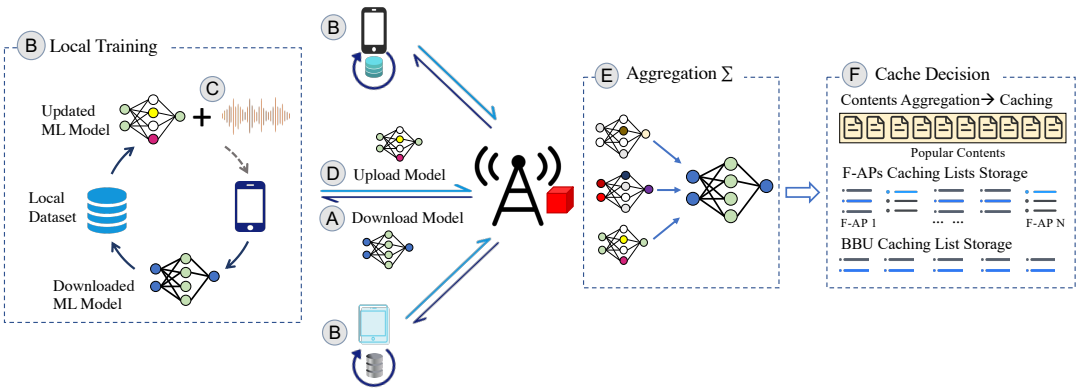


Fig. 2. Federated Learning Process for Edge Caching

to provide local caching services to their associated users. By contrast, the BBU pool caches the M less popular contents. Based on the knowledge of the content popularity from F-APs, the place of caching (at F-APs or BBU pool) can be decided, as described in the next subsection.

C. The Design of Cooperative Hierarchical Edge Caching

The cache hit ratio is used to evaluate the performance of caching schemes, which describes the percentage of content requests that can be served by cache. The proposed FLCH aims to maximise the cache hit ratio by leveraging the vertical cooperation (V-FLCH) between the BBU pool and F-APs, and the horizontal cooperation (H-FLCH) between the local F-AP and neighbour F-APs. Users can fetch the requested content not only from the cache of the local F-AP but also from the neighbour F-APs and the BBU pool. Specifically, as shown in Fig. 1, a user who is located in the coverage area of the F-AP can send the content request to the local F-AP. The requested content will be firstly searched whether it is stored in its cache [27]. If so, the requested content would be directly delivered to the user. Otherwise, the request will be searched in a caching content list for all neighbour F-APs maintained. The caching content list of each F-AP is stored at the cache manager in the MBS. According to the list, if the requested content is cached at any other F-APs, the found F-AP would transmit the requested content to the user. In this way, the user does not need to connect the BBU pool to fetch the requested content, which relieves the burden on fronthaul between the BBU pool and F-APs. If the requested content is not stored in any F-APs, the local F-AP will send the request to the BBU pool. Upon receiving a request from the F-AP, it conducts a search to look for the content. If the content can be found in the BBU pool, the BBU pool delivers the corresponding content to the requesting F-AP via fronthaul link. The requested content can be provided by caching at either F-APs or the BBU pool, therefore, it greatly reduces the traffic between the core network and the cloud. However, if the requested content is neither cached at F-APs nor BBU pool, the request would be forwarded to the Internet to obtain the content from the source (*i.e.*, content provider) in the cloud.

IV. Privacy-Preserving Federated Deep Learning Models for Edge Caching

In this section, the proposed privacy-preserving federated learning based edge caching is introduced in detail. It consists of three parts: privacy-preserving FL, one-class collaborative variational autoencoder (OCC-VAE) and cooperative caching. The privacy-preserving FL framework is utilised to train the content popularity model while protecting users' privacy. The content popularity model we applied in FL is OCC-VAE, which can obtain the relationship between users and contents, and find its hidden representations. The historical requests of users together with their contextual information (*e.g.*, time, location, age), as the training data for the OCC-VAE, are exploited to learn the popularity of content in order to make smart caching decisions. The training data is binary where 1 (positive examples) and 0 (negative examples) represent requested contents and unrequested contents, respectively, but the negative examples are often absent. Marking all missing examples as negative examples is a bias prediction. One-class collaborative filtering with a random sampling mechanism is used to correct this bias in OCC-VAE.

A. Privacy-Preserving Federated Learning for Edge Caching

Most of the existing learning-based caching schemes need to collect the data from users to train the prediction model. However, these data may contain private and sensitive information (*e.g.*, gender, age, location and occupation). Uploading users' data to the central server will result in the users' privacy risk. By leveraging the local computation capacity and dataset of users, FL can perform distributed training. This training needs to run multiple FL communication rounds. In our proposed FL communication round, the F-AP firstly sends a global model to selected users. Next, selected users compute their updated models independently, which is based on their local resources. Then, the updated model from each selected user will be collected by the associated F-AP. All updated models are used to construct an improved global model by the weighted averaging method [4]. The weight of each user depends on their data size. Users who contain more training data make more contributions to the global OCC-VAE prediction model in FL .

As shown in Fig. 2, a FL communication round executes at users and their F-APs, which has five steps [22]:

- **Step A:** The users who have good network connections, enough energy and computing resources are chosen to attend the FL communication round. Their associated F-APs will send the current global model to them.
- **Step B:** Each selected user utilises its local dataset to compute the updated global model.
- **Step C:** Local differential privacy mechanism applies to calculated models.
- **Step D:** The updated model from selected users are sent to their associated F-APs.
- **Step E:** Each F-AP aggregates received models to generate a new global model.

All the above steps are repeated until the achieved results are stable. After that, **Step F** in Fig. 2 can be executed. Which contents can be cached at F-APs and BBU are determined.

In our FL setting, we assume the whole data size of the training dataset is d which is formed by the training examples over K selected users with C contents. The training goal is to minimise the following objective of FL [4]:

$$\min_{w} f(w) = \frac{1}{d}\sum_{i=1}^{d} f_i(w), \tag{1}$$

where $f_i(w) = \ell(u_i, c_i; w)$ is the loss function of the prediction on the example (u_i, c_i) made with model parameters w. Based on the resource situation of each user, we select K users to participate in the FL training, where K users are indexed by k. Denote the set of indexes of training examples on user k is D_k, with $d_k = |D_k|$. Thus, the objective can be re-written as

$$\min_{w} f(w) = \sum_{k=1}^{K} \frac{d_k}{d} F_k(w), \quad \text{where} \quad F_k(w) = \frac{1}{d_k}\sum_{i \in D_k} f_i(w), \tag{2}$$

Typically, stochastic gradient descent (SGD) is implemented to optimise the OCC-VAE model. In the FL setting, with applying SGD optimisation, each communication round only calculates a single batch of gradients. It may cause large communication costs, since FL requires plenty of communication rounds to train a high-quality OCC-VAE model. Therefore, we use the federated stochastic gradient descent method (FedSGD) [4] for optimisation. FedSGD allows each user to iterate multiple rounds and then takes the average of gradients $\triangledown F_k(w_r)$ on its local data at the parameters of model w_r in the r^{th} round. These average gradients are sent to the server and then applied to generate the global OCC-VAE model. Additionally, the local dataset in each selected user k is generated from the usage of its device, *e.g.*, content requests in daily life. Some users make heavy use of certain specific applications, which causes the data imbalance problem for the federated training. To handle this problem, we implement the weighted aggregation method to aggregate the model in the edge server. The equations of updates of the model and aggregation are given by

$$w_{r+1} \leftarrow w_r - \eta \sum_{k=1}^{K} \frac{d_k}{d} \triangledown F_k(w_r), \tag{3}$$

$$w_{r+1} \leftarrow \sum_{k=1}^{K} \frac{d_k}{d} w_{r+1}^k, \tag{4}$$

where η is the learning rate. The weighted sum is implemented in the aggregation method. Weights for parameter aggregation depend on the corresponding data size of users. More data on user k makes more contributions to update the shared OCC-VAE model.

During the FL training process, the parameters of OCC-VAE model have been exchanged. Privacy leakage still may happen by analysing the parameters of the model. Therefore, we utilise local differential privacy mechanism into the proposed FLCH to prevent privacy leakage and model attacks.

B. Differential Privacy Mechanism

In the FL framework, each user computes an updated OCC-VAE model based on the current global model. Then, a differential privacy mechanism will be used to perturb the original local models, to prevent privacy leakage.

Data Distortion: Given the parameters of model w_r^k, a Gaussian mechanism is exploited to distort the w. A scaled version of w is generated by $\triangledown F_k\left(w_r^{k*}\right) = \triangledown F_k\left(w_r^k\right) / max\left(1, \frac{\left\|\triangledown F_k\left(w_r^{k*}\right)\right\|_2}{S}\right)$ [28], where S is the upper bound of the sensitivity of the scaled model, defined as $S = \text{median}\left\{\triangledown F_k\left(w^k\right)\right\}$. Choosing the value of S is a trade-off. The larger S makes the larger noise variance and thus the accuracy of model may decrease. With scaled to S, noise is added to the model. The updated model at user side is:

$$w_{r+1} = w_r - \triangledown F_k\left(w_r^k\right) / max\left(1, \frac{\left\|\triangledown F_k\left(w_r^{k*}\right)\right\|_2}{S}\right) + \mathcal{N}\left(0, \sigma^2 S^2\right), \tag{5}$$

where the probability distribution of noise follows a normal distribution $\mathcal{N}\left(0, \sigma^2 S^2\right)$, which is scaled to S [28].

The proposed privacy-preserving federated learning scheme is outlined in Algorithm 1. The integrated model of the one-class collaborative variational autoencoder (OCC-VAE) is the model trained in privacy-preserving FL.

C. One-Class Collaborative Variational Autoencoder

The OCC-VAE is proposed to predict content popularity ahead of time. VAE is a powerful unsupervised learning method, which aims to realise the data distribution $p(x)$ from the training set. Moreover, VAE is naturally suitable for estimating content popularity, which can effectively cluster data in the latent space [29]. An overview of VAE is depicted in Fig. 3. An inference neural network $q(z \mid x)$ maps the input x to a distribution (*i.e.* Gaussian distribution) with estimating the latent variable z. $p(z \mid x)$ is a generative neural network, which is used to decode the sampled latent variable z back into an observed data x.

In our setting, we employ the VAE to learn deep latent representations from user-by-content request matrix X and implicit relationship between users and contents. The X consists of samples of variable x_u^c, where $X \in \mathbb{N}^{U \times C}$, $1 < u < U$ and $1 < c < C$. u and c represent the index of users and contents,

Algorithm 1 ***FLCH***:

Server Execution:

Initialise the global model w_0
for each round r = 1,2,... **do**:
  K: a set of users, $k \in K$.
  U_r: a set of selected users
  for each user k **in parallel do**:
    if k meet requirement of training model **then**
      add k to U_r
    end if
  end for
  Get w_r
  for each user $k \in U_r$ in parallel **do**:
    $w_{r+1}^k \leftarrow$ **UserUpdate**(w_r, k)
    for each weight $p \in w_{r+1}^k$ **do**:
      Sample a Bernoulli variable a
      if $a == 1$ **then**
        $p* = k + r \cdot \frac{e^{\epsilon}+1}{e^{\epsilon}-1}$
      else $a == 1$
        $p* = k - r \cdot \frac{e^{\epsilon}+1}{e^{\epsilon}-1}$
      replace p as $p*$
    $w_{r+1}^k* \leftarrow w_{r+1}^k$
  end for
  $w_{r+1}^k \leftarrow \sum_{k=1}^{K} \frac{d_k}{d} w_{r+1}^{k*}$
for end
Return w_{r+1}

User Execution:

Input Data: w_r, X
UserUpdate(w, k)**:**
  for each local epoch from 1 to E **do**
    for batch $b \in B$ **do**
      $w_{r+1} \leftarrow w_r - \eta \nabla l\,(w_r; b) + \mathcal{N}\left(0, \sigma^2 S^2\right)$
    end for
  end for
Return w_{r+1}

respectively. The $x_u = \left[x_u^1, ..., x_u^c\right]^T \in \mathbb{N}^C$ is a vector with the request number for each content from user u. Moreover, our proposed FLCH is the context-aware content caching scheme. The users' content retrieval history and context are utilised to learn the context-specific content popularity, hence, the context of user $\hat{a}$ is appended to X.

However, the value of elements in X in our setting is only 1 or 0. The value of 1 represents the positive example of the user's interests. The value of 0 indicates an unknown positive or negative example corresponding to missing, as it is impossible for users to request all the contents. In fact, the input matrix is sparse so that it is hard to identify the negative examples. Furthermore, all negative examples and missing positive examples are mixed which makes it difficult to distinguish them [30]. For example, if we simply solve the issue by marking all the missing examples as negative, then the result may be inaccuracy. The reason is that the positive examples may be included in the missing examples. Therefore, we use a random sampling mechanism to mark the negative

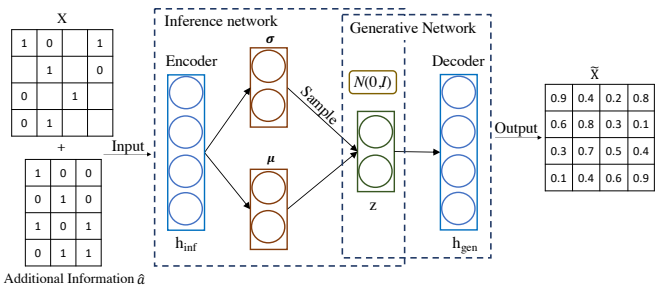


Fig. 3. Variational Autoencoder

examples in the input matrix X.

The probability of random sampling is determined by the preference of users for contents. The probability of content c requested by the user u is

$$Pr_{(u,c)} = r_1 \sum_{i=0}^{C} x_i^u + r_2 \frac{1}{\sum_{j=0}^{U} x_c^j}, \tag{6}$$

where $\sum_{i=0}^{C} x_i^u$ represents the number of contents requested by user u. $\sum_{j=0}^{U} x_c^j$ means how many times that content c has been requested by users. r_1,r_2 are coefficients.

In general, VAE assumes that for every $x_u \in X$, there are one or many settings of the latent variables $z_u \sim p\,(z_u)$ which causes the model to generate something very similar to x_u. Here, $p\,(z_u)$ is the probability distribution of z_u. Mathematically speaking the goal is to maximise the probability of x_u in the input data under the generative process, which is formally defined as:

$$p\,(x_u) = \int p\,(x_u \mid z_u)\, p\,(z_u)\, dz_u. \tag{7}$$

In general, $p\,(x_u \mid z_u)$ is typically parameterised with a highly flexible function approximator such as neural networks. While both prior $p\,(z_u)$ and likelihood $p\,(x_u \mid z_u)$ can be formulated exactly, the posterior $p\,(z_u \mid x_u) = \frac{p(x_u, z_u)}{\int p(x_u, z_u) dz_u}$ needs an intractable integral over the latent space. Thus, instead of calculating the posterior $p\,(z_u \mid x_u)$, VAE takes an advantage of a parametrized variational approximation $q\,(z_u \mid x_u)$ to provide a distribution over the latent variables that are more likely to produce the input data x. This is done by minimizing the Kullback-Leibler (KL) divergence between $q\,(z_u \mid x_u)$ and $p\,(z_u \mid x_u)$:

$$\begin{aligned} &KL\left[q\,(z_u \mid x_u) \parallel p\,(z_u \mid x_u)\right] = \\ &\quad \mathbb{E}_{z_u \sim q(z_u \mid x_u)} \left[\log q\,(z_u \mid x_u) - \log p\,(z_u \mid x_u)\right]. \end{aligned} \tag{8}$$

By applying Bayesian inference to $p\,(z_u \mid x_u)$, we achieve

$$\begin{aligned} \mathrm{KL}\left[q(z_u \mid x_u) \parallel p(z_u \mid x_u)\right] &= \mathbb{E}_{z_u \sim q(z_u \mid x_u)} \left[\log \frac{q(z_u \mid x_u) p(x_u)}{p(x_u \mid z_u)}\right] \\ &= \mathbb{E}_{z_u \sim q(z_u \mid x_u)} \left[\log q\,(z_u \mid x_u)\right] + \mathbb{E}_{z_u \sim q(z_u \mid x_u)} \left[\log p\,(x_u)\right] \\ &- \mathbb{E}_{z_u \sim q(z_u \mid x_u)} \left[\log p\,(x_u \mid z_u)\right] - \mathbb{E}_{z_u \sim q(z_u \mid x_u)} \left[\log p\,(z_u)\right]. \end{aligned} \tag{9}$$

Then, to maximise $\mathbb{E}_{z_u \sim q(z_u|x_u)}[\log p(x_u)]$, Eq. 9 can be rewritten as follow:

$$\begin{aligned}&\mathbb{E}_{z_u \sim q(z_u|x_u)}[\log p(x_u)] - \mathrm{KL}[q(z_u \mid x_u) \parallel p(z_u \mid x_u)] = \\ &\mathbb{E}_{z_u \sim q(z_u|x_u)}[\log p(x_u \mid z_u)] + \\ &\mathbb{E}_{z_u \sim q(z_u|x_u)}[\log p(z_u) - \log q(z_u \mid x_u)] \\ &= \mathbb{E}_{z_u \sim q(z_u|x_u)}[\log p(x_u \mid z_u)] - \mathrm{KL}[q(z_u \mid x_u) \parallel p(z_u)]. \end{aligned} \tag{10}$$

The lower bound of $\mathbb{E}_{z_u \sim q(z_u|x_u)}[\log p(x_u)]$ is as:

$$\begin{aligned}\log p(x_u) \geq &\mathbb{E}_{z_u \sim q(z_u|x_u)}[\log p(x_u \mid z_u)] \\ &- \mathrm{KL}[q(z_u \mid x_u) \parallel p(z_u)]. \end{aligned} \tag{11}$$

The right hand-side of Eq. 10 is the variational lower bound of VAE. The $q(z_u \mid x_u)$ follows a Gaussian distribution, which is $N(\mu, \mathrm{diag}(\sigma^2))$. σ^2 means variance and μ represents the mean. To maximise the variational lower bound, $p(x_u \mid z_u)$ and $q(z_u \mid x_u)$ can be trained. Also, the reparameterisation trick [31] $z_u = \mu + \sigma \odot \epsilon$ is exploited to obtain the unbiased estimate of low variance bound. $\sigma(x_u)$ is covariance, $\mu(x_u)$ is the mean, and ϵ follows the standard normal distribution $\mathcal{N}(0, 1)$. The update equation is the following:

$$\begin{aligned}&\mathbb{E}_{q(z_u|x_u)}[\log p(x_u \mid z_u)] = \\ &\qquad \mathbb{E}_{\epsilon \sim N(0,I)}[\log p(x_u \mid z_u = \mu + \sigma \odot \epsilon)], \end{aligned} \tag{12}$$

where ϵ is drawn from $\epsilon \sim N(0, 1)$. By utilising the reparameterisation trick, $p(x_u \mid z_u)$ and $q(z_u \mid x_u)$ are trained by SGD.

Therefore, we feed X with incomplete rows (resp. columns) into VAE to learn the latent representation Z. X can be recovered from Z, where the outputs are a matrix containing predicted missing values. The highest score contents in outputs are caching contents in the cache-enabled server. The cache decisions are made by inference through the encoder and decoder networks that consist of fully connected layers. Therefore, the computational complexity of the OCC-VAE model should be the same as the forward propagation of fully connected layers, $O(n)$, where n is the size of the input data.

V. Experimental Results

In this section, we evaluate our proposed proactive content caching scheme using a real world datasets and compare its performance to six reference algorithms. We conduct experiments via a fog computing-like testbed with 10 nodes.

A. Testbed and Dataset

The testbed consists of three HP Z440 workstations with 64G memory and 10 Raspberry Pi devices. This represents a simple fog computing environment. Two workstation as the F-APs are applied to aggregate the parameters of the OCC-VAE model and one workstation is the MBS. All Raspberry Pi devices as users conduct the training of the OCC-VAE model [21]. Keras is employed as the framework of VAE with tensorflow as backend.

To evaluate the performance of the proposed FLCH caching scheme, the dataset we used is a real-world dataset, MovieLens 1M [32]. It is collected by GroupLens Research, which has about 1 million ratings of 3883 movies from 6040 users. UserID, MovieID, Rating, Timestamp and user information (such as gender, age, Zip-code and occupation) are included in the dataset. In our experiments, MovieLens dataset is divided into 10 users. To be specific, the dataset for each user consists of 604 users' data in the MovieLens' dataset. A real scenario is considered in this paper that users have unbalanced data due to their different activities and behaviours.

Our proposed FLCH can also be adapted to more complex environments (*e.g.*, Internet of Vehicles, IoT, and Multi-access Edge Computing) by modifying the proposed deep learning model and iterative optimisation algorithm for FL.

B. Performance Evaluation

We firstly compare FLCH with six reference algorithms which are Oracle, Random, Least Recently Used, Least Frequently Used, AutoEncoder and Stacked AutoEncoder. We then evaluate the performance of hierarchical caching with cooperation between the BBU pool and F-APs. The four reference algorithms are presented as follows:

- **Oracle**: Oracle algorithm gets the best cache hit ratio, as it knows the perfect prior knowledge of requests from users.
- **Random**: Random algorithm randomly selects the contents to cache.
- **Least Recently Used (LRU)**: When the caching storage is limited, LRU removes the content based upon the time of usage. The content will be replaced if it has not been requested for a long time. If the content has not been requested in the most recent period of time, it may not be requested in the future as well.
- **Least Frequently Used (LFU)**: LFU keeps tracking the number of times that a content has been requested and replaces the caching content based on the historical request frequency of cache content. If a content has been requested for multiple times in the past, the frequency of this requested content may be higher in the future. LFU evicts the least popular content.
- **AutoEncoder (AE)**: AE attempts to copy its input data to its output data by reconstructing input data from the latent representation, which is an unsupervised learning model.
- **Stacked AutoEncoder (SAE)**: Stacked AE is a deep structure, which stacks multiple AEs. It aims to learn a compressed hidden representation from the input.

Fig. 4 shows the impact of cache size on the cache hit ratio for an F-AP without the hierarchical cooperative caching (N-FLCH). With increasing cache size, the overall cache hit ratios of all caching algorithms rise. Oracle has the perfect prior knowledge about the user demands in future that provides an upper bound to other caching algorithms. Whereas, random gives the worst cache hit ratio which is the lower bound. The cache performance of N-FLCH, Stacked AE, AE based caching scheme outperform LFU and LRU caching schemes, as they learn the latent relationship between users and contents to predict content popularity, while LFU and LRU do not

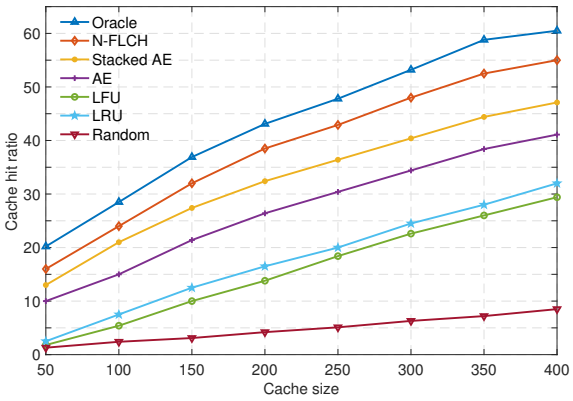


Fig. 4. Cache hit ratio: N-FLCH vs Reference caching schemes

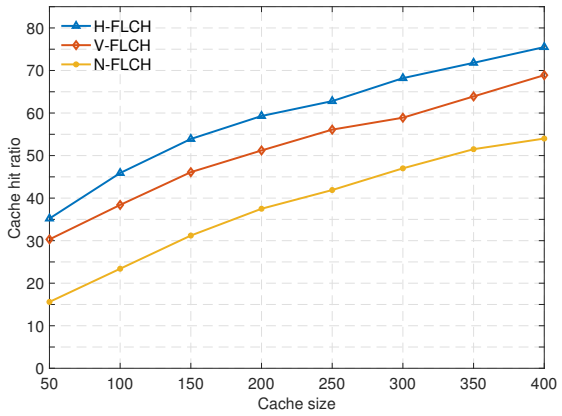


Fig. 5. Cache hit ratio of the FLCH with different cooperative strategies (H-FLCH, V-FLCH, N-FLCH)

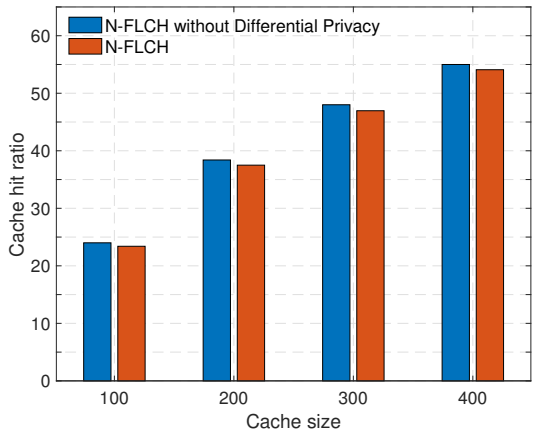


Fig. 6. Cache hit ratio: N-FLCH without differential privacy vs. N-FLCH

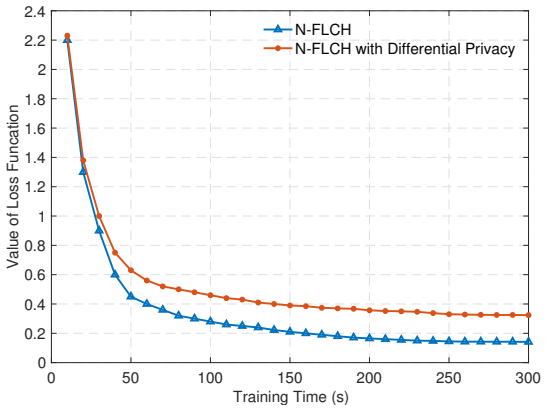


Fig. 7. Training loss vs. Training time

consider the changing trend of content popularity. The N-FLCH shows a better performance compared to SAE and AE, because N-FLCH clusters the request of users in the latent space. LFU obtains a high cache hit ratio than LRU. Both of them study from the historical requests of users. The order of users' requests influences the cache hit ratio of LRU and LFU caching scheme as they make caching decision by observing local recent user request patterns.

Fig. 5 reveals that using the hierarchical cooperative caching mechanism can achieve the higher cache hit ratio. Compared with the hierarchical cooperative caching schemes V-FLCH and H-FLCH, N-FLCH only caches contents at F-APs, which gives the lowest cache hit ratio. The cache hit ratio of H-FLCH is higher than V-FLCH. In H-FLCH, if the requested content is not stored in the local F-AP, the request will be sent to neighbour F-APs, which will deliver the content to the requester if found in their caches. V-FLCH allows a content to be fetched from the BBU cache through cooperation between BBU and F-APs, which obtains the highest cache hit ratio as expected. When the cache size of F-APs is set to 50, the cache hit ratios for N-FLCH, H-FLCH and V-FLCH reach 15.6%, 30.3% and 35.2%, respectively. With the cache size of F-AP increasing to 400, the cache hit ratios of N-FLCH, H-FLCH and V-FLCH are 54%, 68.9% and 75.5%, respectively.

Fig. 6 indicates the influence of the proposed differential privacy mechanism to the cache hit ratio. The experiment results compare the impact of varying cache size on cache hit ratio with differential privacy mechanism and without differential privacy mechanism. Fig. 6 also shows the achieved cache hit ratios for N-FLCH and N-FLCH without differential privacy mechanism are similar. N-FLCH without differential privacy mechanism slightly outperforms N-FLCH. When the cache size is 100 and the number of users is 10, the average cache hit ratio of N-FLCH achieves 23.4%, while N-FLCH without differential privacy mechanism get the cache hit ratio of 24%. The same trend has been exhibited for the other cache sizes. This experiment results demonstrate the differential privacy mechanism keeps high accuracy during the OCC-VAE model training as well as protecting user's privacy.

Fig. 7 shows the results of the loss function in FLCH. Along with the training time increasing, values of the loss function in FLCH are decreasing. The loss values of N-FLCH and N-FLCH without differential privacy decrease quickly at the beginning. After 150s, the loss values become stable, but they continue reducing. Both two approaches converge reasonably quickly, but the speed of convergence of N-FLCH without differential privacy is quicker than N-FLCH, and N-FLCH without differential privacy can achieve a lower loss value. When the training time is 300s, the loss value of N-FLCH

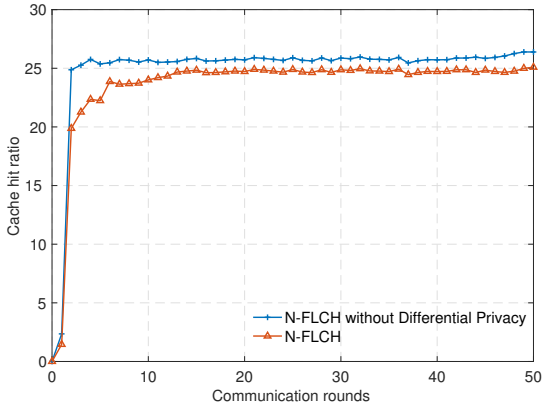


Fig. 8. Cache hit ratio vs Communication rounds

without differential privacy is 0.14, while N-FLCH can only get 0.34. This is because some noises are added to N-FLCH, which makes it hard to train and influence the training results, but the achieved results are not much different.

Fig. 8 compares the cache hit ratio of the FLCH with and without differential privacy mechanism against the number of FL communication rounds. Two approaches show the same trend. To achieve the optimal cache hit ratio, more communication rounds are needed. Fig. 8 shows that N-FLCH reaches the optimal cache hit ratio after 10 FL communication rounds, while FLCH without the differential privacy mechanism achieves it after 5 rounds. However, when the number of communication round is 50, two approaches can achieve similar cache hit ratios that are 26.39% and 25.9%, respectively. Thus, this experiment demonstrates that our proposed differential privacy mechanism can get a similar cache hit ratio while further protecting user's privacy.

VI. Conclusion

In this paper, we have proposed a novel privacy-preserving federated learning based cooperative hierarchical caching scheme (FLCH) for F-RANs. FLCH protects users' privacy by utilising the emerging federated learning framework with differential privacy. Through integrating the one-class collaborative variational autoencoder, FLCH is able to predict the context-specific content popularity by utilising the request history of users and their context information. To further enhance the cache hit ratio and reduce users' latency, FLCH leverages the vertical and horizontal cooperations between the BBU pool and F-APs. The experimental results demonstrate that FLCH achieves the higher cache hit ratio than other caching schemes, such as stacked autoencoder and LRU. The differential privacy mechanism in FLCH can achieve a strict privacy guarantee while maintaining the performance of learning models. Moreover, the proposed hierarchical cooperative caching mechanism can further improve the caching performance. In our future work, we will investigate blockchain empowered federated learning methods for content caching to further enhance security. Moreover, we will conduct research into a theoretical framework to analyse the convergence of the proposed algorithm.

References

[1] C. V. N. Index, "Global mobile data traffic forecast update, 2017–2022 white paper," *Cisco: San Jose, CA, USA*, 2019.

[2] H. Zhang, Y. Qiu, X. Chu, K. Long, and V. C. Leung, "Fog radio access networks: Mobility management, interference mitigation, and resource optimization," *IEEE Wireless Communications*, vol. 24, no. 6, pp. 120–127, 2017.

[3] P. Liu, Y. Ding, and T. Fu, "Optimal throwboxes assignment for big data multicast in vdtns," *Wireless Networks*, 2019. [Online]. Available: https://doi.org/10.1007/s11276-019-01974-z

[4] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 2016.

[5] M. Jindal, J. Gupta, and B. Bhushan, "Machine learning methods for iot and their future applications," in *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*. IEEE, 2019, pp. 430–434.

[6] T. Fu, P. Liu, K. Liu, and P. Li, "Privacy-preserving vehicle assignment in the parking space sharing system," *Wireless Communications and Mobile Computing*, vol. 2020, pp. 1–13, 2020.

[7] Y. E. Sagduyu, Y. Shi, and T. Erpek, "Iot network security from the perspective of adversarial deep learning," in *2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 2019, pp. 1–9.

[8] M. Salimitari, M. Joneidi, and M. Chatterjee, "AI-enabled blockchain: An outlier-aware consensus protocol for blockchain-based iot networks," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[9] S. Yang, S. Fan, G. Deng, and H. Tian, "Local content cloud based cooperative caching placement for edge caching," in *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. IEEE, 2019, pp. 1–6.

[10] Q. Li, X. Wang, and D. Wang, "Optimal D2D cooperative caching system in sdn based wireless network," in *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. IEEE, 2019, pp. 1–7.

[11] T. Hou, G. Feng, S. Qin, and W. Jiang, "Proactive content caching by exploiting transfer learning for mobile edge computing," *International Journal of Communication Systems*, vol. 31, no. 11, p. e3706, 2018.

[12] S. Mehrizi, S. Chatterjee, S. Chatzinotas, and B. Ottersten, "Online spatiotemporal popularity learning via variational bayes for cooperative caching," *IEEE Transactions on Communications*, vol. 68, no. 11, pp. 7068–7082, 2020.

[13] S. Zhang, W. Sun, and J. Liu, "Spatially cooperative caching and optimization for heterogeneous network," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11 260–11 270, 2019.

[14] G. Qiao, S. Leng, S. Maharjan, Y. Zhang, and N. Ansari, "Deep reinforcement learning for cooperative content caching in vehicular edge computing and networks," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 247–257, 2019.

[15] W. Huang, T. Song, Y. Yang, and Y. Zhang, "Cluster-based cooperative caching with mobility prediction in vehicular named data networking," *IEEE Access*, vol. 7, pp. 23 442–23 458, 2019.

[16] Y. Mo, J. Bao, S. Wang, Y. Ma, H. Liang, J. Huang, P. Lu, and J. Chen, "CCPNC: a cooperative caching strategy based on content popularity and node centrality," in *2019 IEEE International Conference on Networking, Architecture and Storage (NAS)*. IEEE, 2019, pp. 1–8.

[17] D. Wu, B. Liu, Q. Yang, and R. Wang, "Social-aware cooperative caching mechanism in mobile social networks," *Journal of Network and Computer Applications*, vol. 149, p. 102457, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804519303170

[18] S. Gao, P. Dong, Z. Pan, and G. Y. Li, "Reinforcement learning based cooperative coded caching under dynamic popularities in ultra-dense networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5442–5456, 2020.

[19] Z. Yang, Y. Liu, Y. Chen, and L. Jiao, "Learning automata based Q-learning for content placement in cooperative caching," *IEEE Transactions on Communications*, 2020.

[20] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. of 2019 ICC*. IEEE, 2019, pp. 1–7.

[21] 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.

[22] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.
[23] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," *arXiv preprint arXiv:1903.03934*, 2019.
[24] M. R. Sprague, A. Jalalirad, M. Scavuzzo, C. Capota, M. Neun, L. Do, and M. Kopp, "Asynchronous federated learning for geospatial applications," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2018, pp. 21–28.
[25] J. Chen, R. Monga, S. Bengio, and R. Jozefowicz, "Revisiting distributed synchronous sgd," in *International Conference on Learning Representations Workshop Track*, 2016. [Online]. Available: https://arxiv.org/abs/1604.00981
[26] H. Zhang, Y. Qiu, K. Long, G. K. Karagiannidis, X. Wang, and A. Nallanathan, "Resource allocation in NOMA-based fog radio access networks," *IEEE Wireless Communications*, vol. 25, no. 3, pp. 110–115, 2018.
[27] Q. Li, W. Shi, X. Ge, and Z. Niu, "Cooperative edge caching in software-defined hyper-cellular networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2596–2605, 2017.
[28] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," *arXiv preprint arXiv:1712.07557*, 2017.
[29] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou, "Variational deep embedding: An unsupervised and generative approach to clustering," *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017.
[30] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang, "One-class collaborative filtering," in *Proceedings of the 8th IEEE International Conference on Data Mining, ICDM*. IEEE, 2008, pp. 502–511.
[31] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.
[32] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems*, vol. 5, no. 4, p. 19, 2016.

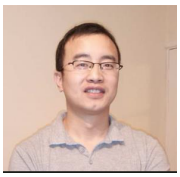
Zhengxin Yu is a Ph.D. student in the Department of Computer Science within College of Engineering, Maths and Physical Science at the University of Exeter, UK. She received an MSc in Information Technology Management for Business from the University of Exeter in 2016. Her research interests focus on deep learning, federated learning and mobile edge computing.

Jia Hu is a Senior Lecturer in Computer Science at the University of Exeter. He received his Ph.D. degree in Computer Science from the University of Bradford, UK, in 2010, and M.Eng. and B.Eng. degrees in Electronic Engineering from Huazhong University of Science and Technology, China, in 2006 and 2004, respectively. His research interests include edge-cloud computing, resource optimization, applied machine learning, and network security.

Geyong Min is a Professor of High Performance Computing and Networking in the Department of Computer Science within the College of Engineering, Mathematics and Physical Sciences at the University of Exeter, United Kingdom. He received the PhD degree in Computing Science from the University of Glasgow, United Kingdom, in 2003, and the B.Sc. degree in Computer Science from Huazhong University of Science and Technology, China, in 1995. His research interests include Computer Networks, Wireless Communications, Parallel and Distributed Computing, Ubiquitous Computing, Multimedia Systems, Modelling and Performance Engineering.

Zi Wang is currently a computer science Ph.D. student in the College of Engineering, Mathematics and Physical Science at the University of Exeter, UK. He received his M.Sc. and B.E. in computer science from the University of Electronic Science and Technology of China (UESTC) in 2018 and 2015, respectively. His research interests focus on deep learning, applied machine learning, cloud and edge computing and computer networks.

Wang Miao received his Ph.D. degree in Computer Science from the University of Exeter, United Kingdom in 2017. He is currently a Postdoctoral Research Associate at the College of Engineering, Mathematics, and Physical Sciences of the University of Exeter. His research interests focus on Network Function Virtualization, Software Defined Networking, Unmanned Aerial Networks, Wireless Communication Networks, Wireless Sensor Networks, and Performance Modelling and Analysis.



Shancang Li received the Ph.D. degree in computer science from Xi'an Jiaotong University, Xi'an, China, in 2008. He is currently a Senior Lecturer with the Network Forensics, University of the West of England (UWE Bristol), Bristol, United Kingdom. Over the last few years, he has been working on a few research projects funded by EU, EPSRC, A4B (Academic expertise for Business), Technology Strategy Board, and industry. He has authored and co-authored several papers based on these research projects. His current research interests include network forensics, device security, wireless sensor networks, Internet of Things, and lightweight cryptography over IoT.