Contemporary Mathematics
https://ojs.wiserpub.com/index.php/CM/

Research Article

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

Hossein Ghasemzadeh[1], **Javad Mohammadzadeh**[1*], **Ali Soleimani**[2], **Azam Bastanfard**[1]

[1] Department of Computer Engineering, Karaj Branch, Islamic Azad University, Karaj, Iran
[2] Department of Computer Engineering, College of Technical and Engineering, Malard Branch, Islamic Azad University, Tehran, Iran
E-mail: j.mohammadzadeh@kiau.ac.ir

Received: 19 December 2024; **Revised:** 19 February 2025; **Accepted:** 27 February 2025

Abstract: Fog Access Networks (F-ANs) have become a potential solution to meet the increasing demand for multimedia services by moving computational and storage functions closer to users at the network's edge. Within these networks, distributing edge caching across Fog Access Points (F-APs) can significantly reduce both network congestion and service delays by keeping frequently accessed content in the local caches rather than relying on remote cloud storage. Considering that F-APs have limited caching resources and that user content demands fluctuate over time and space, various cooperative caching techniques have been created to determine which contents to prioritize and how to store them effectively. However, many of these techniques rely on central servers to collect and analyze data from Internet-of-Things (IoT) devices to predict content popularity, which raises serious concerns regarding privacy. To tackle this challenge, we propose a Federated Learning-based Hierarchical (FLH) caching approach that keeps data within local environments and uses IoT devices for training a shared learning model to estimate content demand. FLH fosters cooperation between neighboring F-APs and engages in vertical collaboration with the Base Band Unit (BBU) pool and F-APs to cache content that varies in popularity. Additionally, FLH implements differential privacy techniques to provide strong privacy protection. Experimental findings show that FLH significantly outperforms five major baseline models in cache hit ratios while ensuring the security of user data.

***Keywords*:** fog computing, internet-of-things, federated learning, privacy-preserving, hierarchical caching

MSC: 68M11, 68T05

1. Introduction

The rapid growth of the Internet of Things (IoT) is anticipated to bring the total number of IoT devices to 14.7 billion by 2023 [1]. The vast amounts of data generated by these emerging IoT devices place considerable pressure on traditional mobile networks [2, 3]. To tackle this challenge, Fog Radio Access Networks (F-RANs) have been identified as a promising solution by shifting computation and storage functionalities toward the network's edge [4]. This strategy enables F-RANs to reduce service latency and alleviate network congestion [5].

In Fog Radio Access Networks (F-RANs), edge nodes, which include Remote Radio Heads (RRHs) and Fog Access Points (F-APs), have the ability to conduct local signal processing, facilitate distributed caching, and manage radio resources cooperatively [6]. One of the essential aspects of F-RANs is the distributed caching implemented across these edge nodes. By positioning popular content as close to users as possible at the edge, it becomes feasible to greatly reduce unnecessary data transmission and minimize service latency. Nevertheless, since the caching capacity of edge nodes is limited in comparison to the extensive range of available content, effectively utilizing this cache capacity plays a vital role in enhancing overall service performance [7].

Despite extensive research efforts, several challenges persist in the domain of learning-based cooperative caching [8]:

• Privacy: Existing caching strategies often rely on centralized data collection for popularity prediction, which poses significant privacy risks as sensitive user data is transmitted to central servers [7, 9, 10].

• Utilization: Redundant storage of content across edge nodes leads to inefficient global cache resource optimization, especially given the limited cache capacities at different nodes [11].

• Mobility: User mobility results in outdated cached content at one edge node when users move, while incoming users may find the required content missing at their new location [12].

As a result, the cached content at one edge node may become outdated when users depart, while another edge node may lack the cached content needed for incoming users. This lack of consideration for user mobility can considerably hinder cache efficiency.

Despite significant advancements in edge caching and federated learning, several critical challenges remain unaddressed. Existing caching strategies often rely on centralized data collection, which poses serious privacy risks [13]. Additionally, most current approaches fail to effectively optimize cache utilization in dynamic environments with high user mobility [14]. Furthermore, there is a lack of comprehensive frameworks that integrate federated learning with differential privacy to ensure both caching efficiency and user data protection [15]. Our work aims to address these gaps by proposing a novel hierarchical caching framework that leverages federated learning and differential privacy techniques.

To tackle the aforementioned challenges, we introduce a Hierarchical Edge Caching framework based on Federated Learning [16], that prioritizes privacy preservation. This framework facilitates intelligent decisions regarding the caching of content at the edge while ensuring the confidentiality of user data. The FLH approach involves training a unified global learning model at the Federated Access Point (F-AP), leveraging training data that is distributed across users' Internet of Things (IoT) devices. Notably, only the parameters derived from the localized models are transmitted to the central server, as opposed to the raw IoT data. To enhance user data protection, the FLH framework incorporates a differential privacy strategy. Our analysis is set within a Fog Computing environment characterized by three layers: the base layer comprises users with IoT devices; the intermediate layer consists of F-APs equipped with limited cache capacities; and the uppermost layer contains a pool of Base Band Units (BBUs) with ample cache storage. This tiered caching structure optimizes the utilization of existing caches for mobile users, as requested content can be sourced from the local F-AP, neighboring F-APs, or the BBU pool. This study presents several significant advancements, outlined as follows:

• An novel caching strategy is introduced, rooted in privacy-preserving federated learning principles. This approach determines caching decisions by examining both the historical content retrieval patterns and contextual information associated with users. Notably, the training data remains localized on Internet of Things (IoT) devices, while a collectively learned global model is developed through the aggregation of models computed on these devices. To ensure robust privacy measures, a comprehensive differential privacy framework is integrated into the methodology.

• A multi-tiered cooperative caching framework is proposed, which harnesses horizontal collaboration among the Federated Access Points (F-APs) and vertical integration between the Baseband Unit (BBU) pool and the F-APs. This collaborative structure aims to improve overall caching efficiency and optimize the utilization of global cache resources.

• The research introduces a One-Class Variational Autoencoder (OC-VAE) designed to predict the popularity of content. This variational autoencoder is employed to uncover the latent representations related to both users and content. Furthermore, one-class collaborative filtering techniques are leveraged to effectively analyze input data, enhancing the system's capability to recommend trending content more accurately.

The organization of this paper is delineated as follows: In the second section, we review pertinent literature. The third section outlines the system architecture associated with the proposed FLH caching strategy. A comprehensive discussion of the FLH implementation is provided in section four. The fifth section presents an assessment of the performance of the FLH. Ultimately, section six wraps up the discussion of this study.

2. Literature review

Recent advancements in federated learning and edge caching have addressed various challenges in IoT and fog computing environments. For instance, Wang et al. [17] proposed an adaptive federated learning framework that dynamically adjusts the global model aggregation process to improve communication efficiency in resource-constrained edge systems. Similarly, Li et al. [18] developed a cooperative caching framework for wireless networks that leverages Device-to-Device (D2D) communication to enhance content delivery. However, these approaches often rely on centralized data collection, which raises significant privacy concerns.

In the context of privacy-preserving techniques, differential privacy has emerged as a powerful tool to protect user data in distributed learning environments. Geyer et al. [19] introduced a client-level differential privacy mechanism that adds noise to model updates during federated learning, ensuring robust privacy guarantees. Despite these advancements, there is a lack of integrated frameworks that combine federated learning with differential privacy to optimize both caching efficiency and user privacy in dynamic environments with high user mobility.

Our work builds on these foundations by proposing a novel hierarchical caching framework that integrates federated learning with differential privacy. Unlike existing approaches, our framework ensures that user data remains localized while enabling accurate content popularity prediction through a shared global model. This addresses the critical gaps in privacy, utilization, and mobility that are often overlooked in current literature.

Significant endeavors have been undertaken employing machine learning methodologies to address various challenges within the Internet of Things (IoT) domain, including but not limited to communication, privacy, and security concerns. For instance, applied adversarial machine learning techniques to both compromise and protect IoT infrastructures [20]. Additionally, developed an AI-driven blockchain framework featuring a dual-step consensus mechanism aimed at enhancing the fault tolerance of Hyperledger Fabric. This section provides an overview of the current research focused on cooperative caching and federated learning within the contexts of IoT and smart environments, respectively.

Yang et al. [21] introduced a collaborative caching mechanism that enables base stations within a localized cloud to work together in storing content. Similarly, Li et al. [18] constructed a cooperative caching framework tailored for wireless networks, which incorporates Device-to-Device (D2D) transmission. In this system, mobile devices collectively cache frequently accessed content. The aforementioned caching strategies generally operate under the assumption that user content requests conform to a Zipf distribution. However, in real-world scenarios, content popularity is characterized by dynamic trends and exhibits both temporal and spatial dependencies, making precise modeling challenging. Consequently, Machine Learning (ML) techniques are increasingly utilized within cooperative caching frameworks to enhance caching efficiency.

Further developments in this area include the work by Hou et al. [22], who explored a proactive cooperative caching strategy for Mobile Edge Computing (MEC) aimed at minimizing transmission costs and enhancing user experience through transfer learning methodologies. Mehrizi et al. [23] introduced a proactive cooperative caching model that employs a probabilistic dynamic framework to anticipate content popularity, incorporating a Variational Bayes approach to mitigate overfitting issues. Zhang et al. [24] put forth a spatially cooperative caching strategy that optimizes the caching probabilities of nodes to boost caching efficiency while minimizing required storage space. Furthermore, Qiao et al. [25] created a cooperative caching system based on deep reinforcement learning methodologies, utilizing a double time-scale Markov decision process to manage the caching dynamics effectively.

In the context of vehicular networking, Huang et al. [26] proposed a cluster-based cooperative caching framework that accounts for vehicle mobility. Mo et al. [27] advanced a collaborative caching strategy that considers content popularity alongside node distribution patterns. Wu et al. [28] contributed a social-aware cooperative caching solution designed to enhance content sharing capabilities while leveraging user cache resources. Gao et al. [29] developed a cooperative coded caching model employing reinforcement learning in conjunction with maximum-distance separable coding, framing the approach within a Markov decision process framework to capture fluctuations in content popularity. Lastly, Yang et al. [30] devised a cooperative caching system grounded in recurrent neural networks to predict both user mobility and content demand.

Federated Learning (FL) was initially introduced by Google as a novel method to implement machine learning techniques at the network edge. McMahan et al. [31] established foundational protocols for FL that facilitate synchronous optimization in federated environments [32, 33]. To enhance various aspects of FL, including communication efficiency, model accuracy, and round effectiveness, numerous FL variants have been developed. For instance, Wang et al. [17] formulated a control mechanism to dynamically determine the timing for global model aggregation. Meanwhile, Nishio and Yonetani created a protocol within the Mobile Edge Computing (MEC) framework to exclude less responsive users based on their estimated work time, thereby shortening the duration of training rounds. Konevcny et al. [34] introduced the concept of sketched and structured updates aimed at decreasing communication overhead. Additionally, Xie et al. presented FedAsync [35], an asynchronous federated learning framework that supports non-blocking updates to the global model while regularizing local optimization processes. In a similar vein, Sprague et al. [36] utilized an asynchronous method to develop a global model for geo-spatial applications, accommodating users who wish to join the training process midway. Conversely, Chen et al. [37] demonstrated that, in terms of model accuracy, synchronous stochastic gradient descent outperforms asynchronous techniques.

A key benefit of FL is its ability to safeguard user privacy and mitigate security threats, as only model parameters, rather than raw user data, are transmitted to the central server. However, research indicates that even the analysis of model parameters can potentially compromise user confidentiality, thereby presenting a risk for attacks. In light of this, we propose a privacy-conscious federated deep learning framework aimed at enhancing hierarchical caching efficiency while also ensuring the protection of user data privacy.

3. Proposed FLH caching strategy for fog radio access networks

The proposed system architecture is designed to optimize content caching in Fog Radio Access Networks (F-RANs) while preserving user privacy. The architecture consists of three main layers: the Device layer, the Fog layer, and the Cloud layer. Each layer plays a distinct role in the caching process, as described below.

3.1 *Device layer*

This layer comprises IoT devices owned by end-users. These devices generate content requests and provide contextual data, such as user preferences, location, and time of day. The data remains localized on the devices, ensuring user privacy.

3.2 *Fog layer*

The Fog layer is divided into two sub-layers.

3.3 *F-APs*

F-APs are responsible for local model training and caching decisions. They aggregate model updates from IoT devices and collaborate with neighboring F-APs to optimize content placement.

3.4 *BBU pool*

The BBU pool acts as a central coordinator, aggregating model updates from multiple F-APs and updating the global model. It also caches less popular content that cannot be stored at the F-APs due to limited cache capacity.

3.5 *Cloud layer*

The Cloud layer consists of remote servers that host content providers. While the primary focus of our framework is on edge caching, the Cloud layer serves as a backup for content that is not available at the edge.

The flow of data and decision-making processes within the architecture is as follows:

• IoT devices train local models using their data and send model updates to their associated F-APs.

• F-APs aggregate the model updates and forward them to the BBU pool for global model aggregation.

• The BBU pool updates the global model and disseminates it back to the F-APs.

• Based on the predicted content popularity, F-APs cache the most popular content locally, while less popular content is cached at the BBU pool.

This hierarchical architecture ensures efficient content caching while minimizing latency and preserving user privacy.

Fog Radio Access Networks (F-RANs) incorporate fog computing into radio access networks, representing an evolution from Cloud Radio Access Networks (C-RANs) [38]. In C-RANs, numerous Remote Radio Heads (RRHs) are distributed within a specific area and linked to a centralized Baseband Unit (BBU) pool through high-bandwidth fronthaul connections [39]. This centralized BBU pool significantly minimizes energy consumption and enhances resource utilization; however, it also imposes substantial demands on the fronthaul architecture, leading to increased service latency for users. F-RANs respond to these challenges by extending computational capabilities and caching functionality towards the network's periphery. For instance, Fog Access Points (F-APs), which are advancements of conventional RRHs, are equipped with caching systems and can manage these caches with greater flexibility.

3.6 *The system architecture of federated learning*

The Federated Learning (FL) process in our proposed framework involves several key steps, which are iteratively performed to train a shared global model while preserving user privacy. Below, we provide a detailed explanation of each step.

3.6.1 *Local model training*

• Each IoT device trains a local model using its own dataset, which includes historical content requests and contextual information (e.g., time, location, and user demographics).

• The local model is trained using a One-Class Variational Autoencoder (OC-VAE), which predicts content popularity based on user behavior and context.

• The training process minimizes a local loss function, ensuring that the model accurately captures the preferences of the individual user.

3.6.2 *Model update transmission*

• After local training, each IoT device sends the updated model parameters (e.g., weights and gradients) to its associated Fog Access Point (F-AP).

• To enhance privacy, a differential privacy mechanism is applied to the model updates before transmission. This involves adding Gaussian noise to the updates, ensuring that individual user data cannot be easily reconstructed.

3.6.3 *Model aggregation at F-APs*

• Each F-AP aggregates the model updates received from its associated IoT devices. The aggregation process involves computing a weighted average of the updates, where the weight for each device is proportional to the size of its dataset.

• The aggregated model is then sent to the Baseband Unit (BBU) pool for further processing.

3.6.4 *Global model updating*

• The BBU pool aggregates the model updates from multiple F-APs and updates the global model. This involves computing a weighted average of the updates, where the weight for each F-AP is proportional to the number of devices it serves.

• The updated global model is then disseminated back to the F-APs and IoT devices for the next round of training.

3.6.5 *Iterative process*

• The above steps are repeated iteratively until the global model converges. During each iteration, the model becomes more accurate in predicting content popularity, while the differential privacy mechanism ensures robust protection of user data.

This federated learning process enables efficient content caching while preserving user privacy, as the raw data never leaves the local devices.

The objective of this study is to leverage the distributed caching storage capabilities inherent in F-RANs, which necessitates an understanding of content popularity distribution. However, this distribution is subject to frequent variations due to user mobility. Different users exhibit diverse preferences for content, which arise from numerous factors, including gender, age, occupation, and even contextual elements such as the user's location, time of day, and the type of device being used. Consequently, content popularity fluctuates based on the dynamic nature of user engagement and context. In our proposed scheme, content popularity is instrumental in determining both the specific content to cache and its location across the edge nodes.

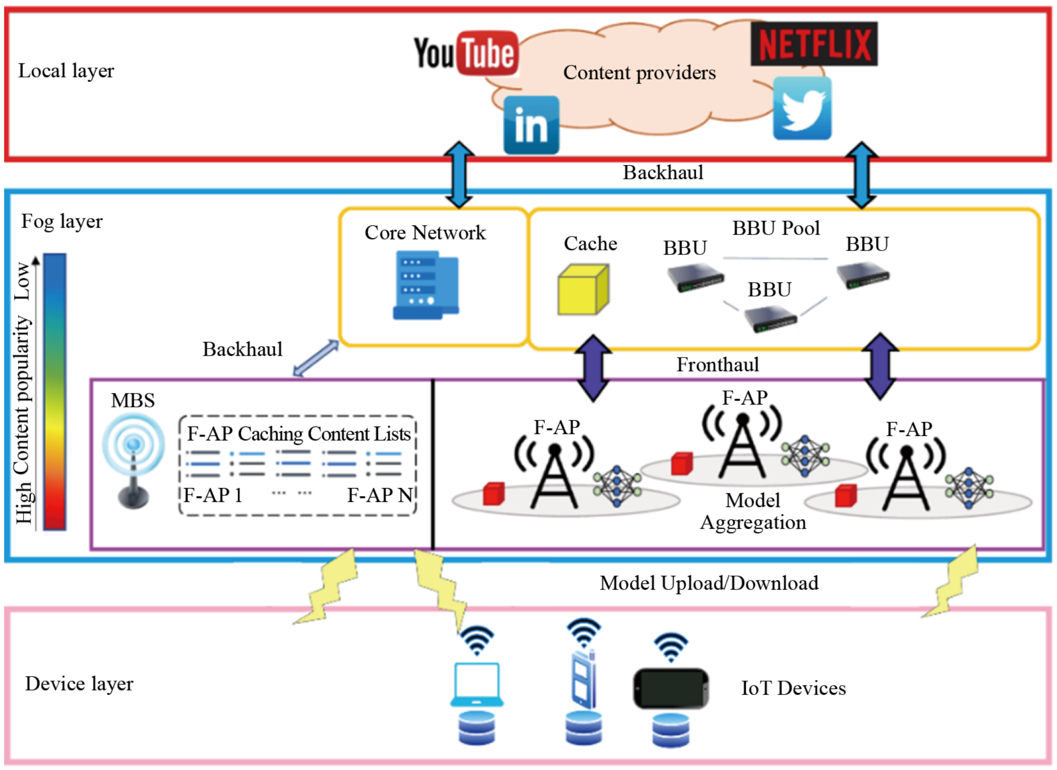

Figure 1. System architecture of the federated learning based hierarchical caching for F-RANs

Federated Learning (FL) allows for the distributed training of a shared model without the necessity of aggregating user data. In our proposed FL-based caching framework, each user associated with an F-AP conducts local training utilizing their individual data. The parameters derived from these local training sessions are subsequently aggregated at the F-AP, enabling the joint development of a model predictive of local content popularity, as illustrated in Figure 1. Based on the anticipated content popularity, each user submits a list of favored contents to their associated F-AP. The F-AP then compiles the various lists from connected users to generate a consolidated list of locally popular contents, which is sent to the MBS. The top N contents, which are deemed highly popular, are selected for caching at the F-APs to deliver localized caching services to their corresponding users. In contrast, the BBU pool is designated to cache the M contents that are of lesser popularity. Drawing upon insights from F-APs regarding content popularity, decisions regarding caching locations—either at the F-APs or within the BBU pool—can be effectively made, as elaborated in the subsequent subsection.

The differential privacy mechanism is a fundamental part of our framework, designed to protect user data while enabling efficient content caching. To ensure privacy, we add a small amount of random noise to the model updates before they are sent from the IoT devices to the Fog Access Points (F-APs). This noise makes it extremely difficult for anyone to reverse-engineer the original data from the updates, even if they have access to the model. By doing this, we ensure that sensitive user information remains confidential throughout the federated learning process.

The amount of noise added is carefully controlled to strike a balance between privacy and model accuracy. Adding too much noise can reduce the model's performance, while adding too little may not provide sufficient privacy protection. To determine the right amount of noise, we use a concept called the "privacy budget", which defines how much privacy we are willing to sacrifice for better model accuracy. This budget helps us calibrate the noise level to ensure strong privacy guarantees without significantly compromising the model's effectiveness.

One of the key features of our approach is that the noise is added directly on each IoT device, a technique known as "local differential privacy". This means that the raw data never leaves the device, providing an extra layer of security. Even when the noisy updates are aggregated at the F-APs and the Baseband Unit (BBU) pool, they do not reveal sensitive information about individual users. This local approach ensures that user privacy is protected at every step of the process.

Once the noisy updates are aggregated, they are used to update the global model. This global model benefits from the collective knowledge of all devices while maintaining strong privacy guarantees. The updated model is then sent back to the devices for further training, and the process repeats until the model converges. Throughout this iterative process, the differential privacy mechanism ensures that user data remains protected, even as the model becomes more accurate.

Our experimental results demonstrate that this approach effectively balances privacy and performance. For example, adding noise only slightly reduces the cache hit ratio, showing that we can achieve both strong privacy protection and efficient content caching. This balance is crucial for real-world applications, where both privacy and performance are critical concerns.

3.7 *Privacy-preserving federated deep learning*

In this segment, we provide a comprehensive overview of the proposed edge caching mechanism, which emphasizes privacy-preserving federated learning. The framework is comprised of a key component that is privacy-preserving federated learning. The primary function of the privacy-preserving FL framework is to train a model for content popularity while simultaneously safeguarding user privacy.

The OC-VAE serves as the content popularity model employed in the federated learning process. Its design enables the identification of relationships between users and content, facilitating the discovery of latent representations. To train the OC-VAE, we utilize a dataset that encompasses historical user requests along with contextual data such as time, geographical location, and demographic details (e.g., age). This allows for an understanding of content popularity, which in turn aids in making informed caching decisions. The training dataset consists of binary indicators, where '1' signifies requested content and '0' indicates unrequested content. However, the absence of negative samples could introduce cognitive bias if all missing samples are classified as negative. To address this issue, a one-class collaborative filtering technique, accompanied by a random sampling method, is implemented to rectify the bias in the OC-VAE.

Traditional learning-based caching strategies typically necessitate the collection of user data for training predictive models, yet such data may encompass sensitive personal information (e.g., gender, age, location, and occupation). The act of transmitting user data to a centralized server poses significant privacy risks. Federated learning mitigates this risk by utilizing the computational resources and datasets available locally to users, facilitating distributed training. This process necessitates several rounds of communication in the FL framework.

3.8 *Privacy-preserving federated learning*

The established communication protocol encompasses several steps. Initially, the Federated Access Point (F-AP) transmits a global model to a subset of selected users. Subsequently, those users independently compute updates to the model based on their individual local datasets. Each user's updated model is then sent back to the corresponding F-AP. The aggregated contributions from all participating users are employed to construct a refined global model through a weighted averaging process, where the weight assigned to each user corresponds to the volume of their training data. Thus, users with larger datasets contribute more significantly to the global prediction model of the O-VAE within the federated learning framework. In Figure 2, it can be observed that a round of federated learning communication takes place between users and their Federated Access Points (F-APs). This process consists of six distinct stages:

1. Users who exhibit strong network connectivity, sufficient energy reserves, and adequate computational resources are selected to participate in the FL communication round, after which their corresponding F-APs dispatch the current global model to them.
2. Each selected user uses their local dataset to compute an updated version of the global model.
3. A local differential privacy mechanism is applied to the computed models to enhance privacy.
4. The updated models from selected users are transmitted back to their corresponding F-APs.
5. Each F-AP consolidates the received models to generate a new global model.
6. The contents that can be stored in the F-Access Points and Baseband Unit are identified.

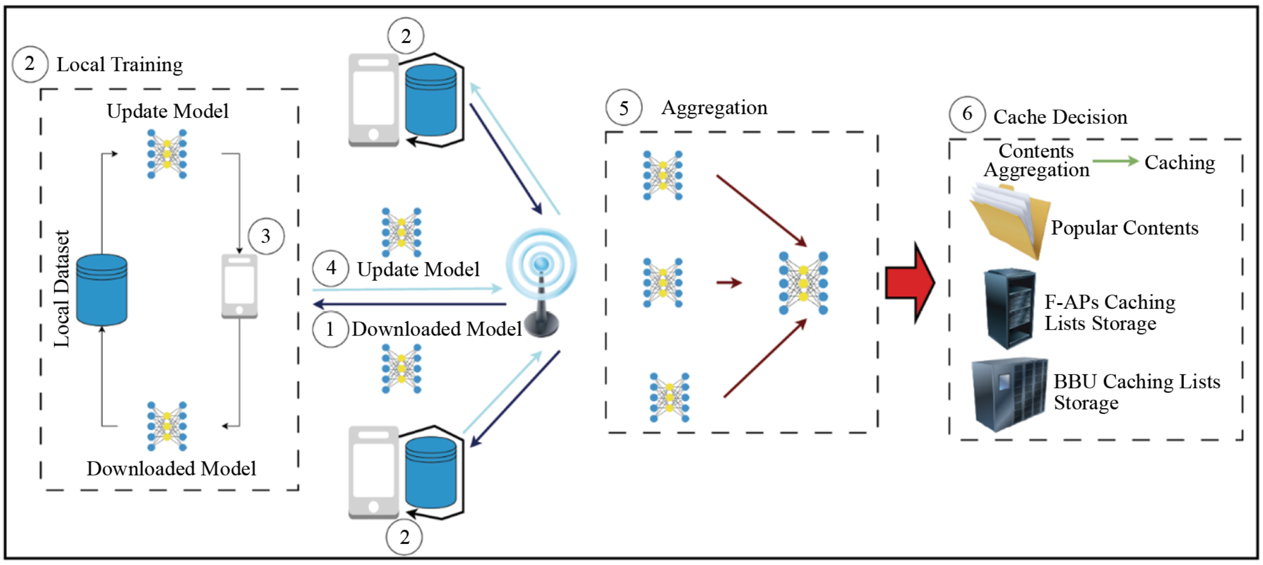

Figure 2. Federated learning process for edge caching

This iterative process continues until the results stabilize. Following this stabilization, the final step involves determining which contents will be cached at the F-APs and Baseband Units (BBUs). In our federated learning framework, we denote the total size of the training dataset as d, which is derived from the training examples across K selected users encompassing C contents. The overarching objective of this federated learning process is to minimize a specific target function associated with FL (Equation (1)).

$$\min_{w} f(w) = \frac{1}{2} \sum_{i=1}^{d} fi(w), \tag{1}$$

In this context, let $f_i(w) = \updownarrow(u_i,\ c_i;\ w)$ represent the loss function associated with the model's predictions for the example $(u_i,\ c_i)$ based on the model parameters w. Given the resource availability of each user, we proceed to select K users to engage in federated learning FL training, denoted by the index k. We denote the collection of indices corresponding to the training examples for user k as D_K where $D_K = |D_K|$ signifies the number of examples for that user. Consequently, the objective can be reformulated as follows (Equation (2)):

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

In general, the optimization of the OC-VAE model is carried out using Stochastic Gradient Descent (SGD). Within the framework of Federated Learning (FL), employing SGD entails that each communication round computes only one batch of gradients. This approach can lead to substantial communication overhead, given that training a robust OC-VAE model necessitates numerous communication rounds. Consequently, we opt for the Federated Stochastic Gradient Descent (FedSGD) method for optimization purposes. FedSGD enables individual users to perform several iterations and subsequently compute the average of the gradients, denoted as $\nabla F_k(w_r)$, based on their local datasets at the w parameter of the model during the r^{th} round. These averaged gradients are then transmitted to a central server, where they are utilized to construct the global OC-VAE model. Furthermore, the local datasets for each chosen user k are derived from their device's usage patterns, such as content requests they encounter in their daily lives. It is noteworthy that some users may engage intensively with particular applications, leading to issues of data imbalance in the context of federated training. To mitigate this challenge, we implement a weighted aggregation approach for model consolidation on the edge server. The mathematical formulations governing the model updates and aggregation process are provided as follows (Equation (3)).

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

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

The model update and aggregation formulas are defined with the inclusion of as the learning rate. In the aggregation process, a weighted sum is employed, where the weights for combining parameters are determined by the amount of data each user possesses. A larger dataset from user k leads to a greater influence on updating the shared OCVAE model. Throughout the Federated Learning (FL) training procedure, the OC-VAE model parameters are shared among the participants. However, there remains a risk of privacy breaches through analyzing these parameters. To mitigate this risk and safeguard against model attacks, we incorporate a local differential privacy mechanism into the proposed FLH framework.

3.9 *Technique for ensuring privacy through differential methods*

In the Federated Learning (FL) setup, each participant individually refines their OCVAE model in accordance with the latest global model. To safeguard against privacy breaches, a differential privacy technique is employed to modify the original local models. For Data Distortion: Utilizing the parameters of the model w_r^k a Gaussian approach is applied to introduce distortions to w. Specifically, a modified version of w is created using the formula $\nabla F_k\left(w_r^{k*}\right) = \nabla F_k\left(w_r^k\right)/max\left(1, \frac{\|\nabla F_k\left(w_r^{k*}\right)\|2}{s}\right)$ [19]. Here, S represents the maximum sensitivity of the scaled model, calculated as $S = median\left\{F_k\left(w_r^k\right)\right\}$. The selection of S involves a balancing act; a higher S leads to increased noise variability, which can, in turn, reduce the model's accuracy. After scaling to S, noise is integrated into the model. The updated version of the model on the user's side is expressed as: Equation (5).

The inclusion of noise in the model weight updating process (Equation (5)) is a key component of the differential privacy mechanism. By adding Gaussian noise to the model updates, we ensure that individual user data cannot be easily reconstructed from the shared model parameters. This approach provides strong privacy guarantees while maintaining the overall accuracy of the model. The noise level is carefully calibrated to balance privacy protection and model performance, as demonstrated in our experimental results [40, 41].

• Privacy Preservation: The noise ensures that even if an adversary gains access to the model updates, they cannot infer sensitive information about individual users.

• Robustness: The noise helps to prevent overfitting by introducing variability into the model updates, which can improve the generalization performance of the model.

• Theoretical Guarantees: Differential privacy provides strong theoretical guarantees of privacy, which are crucial for applications involving sensitive data.

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

where the noise component adheres to a normal distribution, denoted as $\mathcal{N}\left(0, \sigma^2 S^2\right)$, appropriately scaled to S [19].

3.10 *Optimization of federated learning model*

This section offers a brief summary of the EPC algorithm. For an in-depth explanation, please consult Mohammadzadeh et al. [42]. Emperor Penguins, recognized as the largest penguin species, reside in Antarctica and its adjacent sea ice. These birds, measuring between 110 and 130 cm in height, withstand extremely severe conditions, with temperatures dropping to -40 °C. To survive and breed, they utilize an extraordinary strategy: clustering together in large groups. When they form a huddle, Emperor Penguins reduce heat loss and raise their internal temperature. To ensure even heat distribution, they move in a coordinated, spiral-like fashion. Drawing inspiration from this behavior, the EPC algorithm is a population-based metaheuristic that incorporates principles of thermal radiation and spiral movement. The pseudo-code for the EPC algorithm is outlined in Algorithm 1. This algorithm requires the computation of heat absorption, thermal radiation, attractiveness, and coordinated spiral movement. To ascertain attractiveness, the transfer of heat radiation is calculated, which necessitates the penguin's body surface area (0.56 m^2). Heat transfer occurs via conduction, convection, and radiation. The EPC algorithm emphasizes radiative heat transfer as a key criterion for attractiveness. The amount of heat radiated by a penguin is calculated using Equation (6).

$$Q_{penguins} = A\varepsilon\sigma T_s^4 \tag{6}$$

In this context, A denotes the surface area of the penguin's body (0.56 m^2), ε signifies the emissivity of the plumage (0.98, according to [22]), σ represents the Stefan-Boltzmann constant (5.6703×10^{-8} W/m^2K^2), and T indicates the absolute temperature in Kelvin (308.15 K, which corresponds to 35 °C). In the realm of physics, heat sources are classified into three categories: surface, point, and linear, each associated with specific attenuation coefficients. Given the physical characteristics of the penguin's body, the EPC algorithm treats the heat source as linear. By integrating the penguin's emitted radiation with the linear heat source equation, the attractiveness equation (Equation (7)) is formulated.

$$Q = A\varepsilon\sigma T_s^4 e^{-\mu x} \tag{7}$$

In this formula, μ represents the attenuation coefficient, while x indicates the distance between two linear heat sources. The parameter μ plays a crucial role in determining the rate of convergence and is always a positive value. The idea of heat absorption is closely tied to changes in μ. When μ decreases, the level of heat absorption rises, resulting in more effective heat absorption. To integrate attractiveness into the spiral motion, the two equations need to be merged. The spiral movement, which is usually in a clockwise direction, can be modeled using a logarithmic spiral equation. Figure 3 illustrates the subjective perception of this spiral motion.

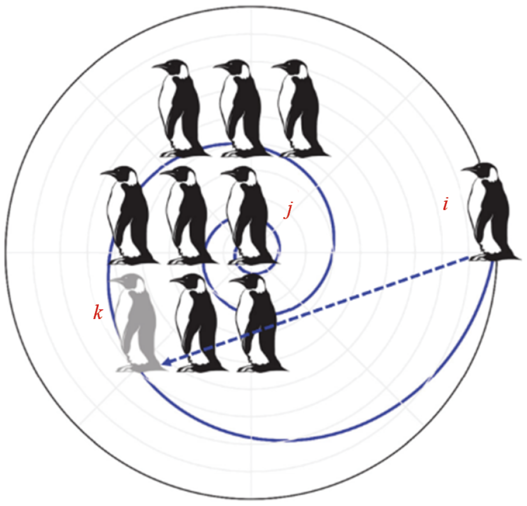

Figure 3. Perception of emperor penguin coordinated spiral movement

As illustrated in Figure 3, the penguin starts its journey from point i with the goal of arriving at point j. Nonetheless, swayed by the degree of attractiveness, it strays from a straight route and follows a logarithmic spiral. The penguin's ultimate location, following this spiral path, is indicated as point k. The mathematical representation that outlines this movement can be found in Equation (8).

$$\begin{aligned} x_k &= ae^{b\frac{1}{b}\ln\left[(1-Q)e^{b\alpha}+Qe^{\omega\beta}\right]}\cos\left\{\frac{1}{b}\ln\left\{(1-Q)e^{b\alpha}+Qe^{b\beta}\right\}\right\} \\ y_k &= ae^{b\frac{1}{b}\ln\left[(1-Q)e^{b\alpha}+Qe^{b/\beta}\right]}\sin\left\{\frac{1}{b}\ln\left\{(1-Q)e^{b\alpha}+Qe^{b\beta}\right\}\right\} \end{aligned} \tag{8}$$

Equation (9) includes the arbitrary constants a and b, the attractiveness equation Q, along with the parameters x and y for the logarithmic spiral. The coefficients α and β are calculated from the ratios -1 y_i/x_i and -1 y_j/x_j. To reduce the possible monotony that may arise from the constant angular information in the spiral motion, a random vector is introduced and incorporated into Equation (9).

$$A\varepsilon\sigma T_s^4 e^{-\mu x} + \varphi\varepsilon_i \tag{9}$$

The mutation factor, φ, regulates the degree of alterations in the path, while ε represents a random vector. When the distance between penguins surpasses a specific threshold, their attraction to optimal solutions diminishes significantly. This can lead to a situation where the current penguin remains inactive and unable to execute the desired spiral movement. To avoid stagnation and encourage diversity, the component $\varphi\varepsilon_i$ is integrated into the equation, enabling minor random shifts in the penguin's position while still maintaining the fundamental spiral pattern.

The process starts by creating a copy of the existing population and then evaluates the fitness of this duplicated population in a separate internal loop. Once each iteration is finished, the original and the newly created populations are merged, and a selection method determines the most promising individuals to advance to the next generation. This cyclical process keeps going until the specified termination criteria are met.

Adjusting parameters effectively is crucial for enhancing the performance of algorithms. In our implementation, we initialized the values of μ and φ to 1 and 0.4, respectively. To balance exploration and exploitation, we utilized an exponential decay approach for both parameters. The value of φ was reduced by multiplying it by 0.2 in each iteration, resulting in a gradual reduction in exploration and an increase in exploitation. Similarly, the value of μ was decreased by multiplying it by 0.98, allowing for a more targeted search over time. This method, inspired by the natural behavior of penguins gathering in a huddle, allows the algorithm to thoroughly explore the solution space at first and then converge on high-quality solutions.

Algorithm 1 Pseudo code of the Emperor Penguins Colony

```
1: Step 1:
2: generate initial population array of EPs (Colony Size);
3: generate initial position and cost of each EP;
4: calculate heat radiation (Eq. 1);
5: determine initial attenuation coefficient ($\mu$);
6: determine initial mutation coefficient ($\varphi$);
7: STEP 2: for First Iteration to Max Iteration
8: generate repeat copies of population array (as new_pop);
9: STEP 3: for $i = 1$ to $n$ do (all $n$ penguins)
10: STEP 4: for $j = 1$ to $n$ do (all $n$ penguins)
11: if (cost$j$ < cost$i$) then
12:     determine new position (Eq. 4);
13:     evaluate and store cost of new_pop;
14: end if
15: END STEP 4
16: END STEP 3
17: merge population array with new_pop array;
18: sort and find best solution;
19: update heat radiation (decrease);
20: update attenuation coefficient (decrease);
21: update mutation coefficient (decrease);
22: END STEP 2
23: END STEP 1
```

4. Experimental results and assessment of performance

4.1 *Experimental setup and reproducibility*

To ensure the reproducibility of our experiments, we provide a detailed description of the hyperparameters, training procedures, and evaluation metrics used in our study. The learning rate was set to 0.01, the batch size to 64, and the number of epochs to 100. These parameters were chosen based on preliminary experiments to ensure a balance between training efficiency and model accuracy. Additionally, the differential privacy mechanism was implemented with a privacy budget (ε, δ) of (1.0, 1e-5), providing strong privacy guarantees while maintaining model utility. The training process involved several key steps, including local model training on IoT devices, model update transmission to Fog Access Points (F-APs), and global model aggregation at the Baseband Unit (BBU) pool. The evaluation metrics used in our experiments include cache hit ratio, training loss, and communication rounds. Table 1 summarizes the experimental settings, including the hardware configuration, dataset, and hyperparameters.

In this section, we analyze the performance of our novel proactive content caching approach using a real-life dataset while comparing it to six well-established algorithms. The tests are carried out in a fog computing-like environment that consists of 10 nodes.

The configuration of our testing environment includes three HP Z440 workstations, each with 64 GB of RAM, along with 10 Raspberry Pi devices that help simulate a fundamental fog computing structure. Two of these workstations serve as F-APs, tasked with generating the parameters of the OCVAE model, while the third workstation functions as the MBS. The Raspberry Pi units act as user devices contributing to the training of the OC-VAE model. For implementing the VAE, we utilize the Keras library backed by TensorFlow.

To evaluate the capabilities of our FLCH caching solution, we use the Movie Lens 1 M dataset curated by Group Lens Research, which includes around 1 million ratings for 3,883 movies from 6,040 users. This dataset comprises various fields such as UserID, MovieID, rating scores, timestamps, and demographic information like gender, age, postal code, and occupation. For our experiments, we split the Movie Lens dataset among 10 users, gathering data for each user from a pool of 604 users in the original dataset. This study illustrates a realistic scenario, recognizing the potential for imbalanced data distributions arising from user behaviors.

Furthermore, our FLH framework can be adapted for more complex ecosystems, such as the Internet of Vehicles, Internet of Things, and Multi-access Edge Computing environments, by fine-tuning the deep learning model and the iterative optimization process tailored to Federated Learning (FL).

The experimental setup for evaluating the proposed FLH framework was designed to simulate a realistic fog computing environment. The testbed consisted of three HP Z440 workstations, each equipped with 64 GB of RAM, and 10 Raspberry Pi devices to emulate user devices in a fog computing scenario. Two of the workstations served as Fog Access Points (F-APs), responsible for generating the parameters of the OC-VAE model, while the third workstation functioned as the Macro Base Station (MBS). The Raspberry Pi devices acted as user devices, contributing to the training of the OC-VAE model.

For implementing the Variational Autoencoder (VAE), we utilized the Keras library backed by TensorFlow. The model was trained using the MovieLens 1 M dataset, curated by GroupLens Research, which includes approximately 1 million ratings for 3,883 movies from 6,040 users. This dataset comprises various fields such as UserID, MovieID, rating scores, timestamps, and demographic information like gender, age, postal code, and occupation. To simulate a realistic scenario, we split the MovieLens dataset among 10 users, gathering data for each user from a pool of 604 users in the original dataset. This approach allowed us to account for potential imbalances in data distribution arising from user behaviors.

The training process involved several key hyperparameters, which were carefully selected to optimize model performance. The learning rate was set to 0.01, the batch size to 64, and the number of epochs to 100. These parameters were chosen based on preliminary experiments to ensure a balance between training efficiency and model accuracy. Additionally, the differential privacy mechanism was implemented with a privacy budget (ε, δ) of (1.0, 1e-5), providing strong privacy guarantees while maintaining model utility.

To evaluate the performance of the FLH framework, we compared it with six baseline algorithms: Oracle, Random, Least Recently Used (LRU), Least Frequently Used (LFU), AutoEncoder (AE), and Stacked AutoEncoder (SAE). The experiments were conducted in a fog computing-like environment with 10 nodes, and the results were analyzed in terms of cache hit ratio, training loss, and communication rounds. The configuration of the test environment, along with the hyperparameters and evaluation metrics, are summarized in Table 1 to facilitate reproducibility.

Table 1. Experimental setting

Parameter	Value
Workstations	3 HP Z440 (64 GB RAM each)
User devices	10 Raspberry Pi devices
Dataset	MovieLens 1 M
Number of users	10
Learning rate	0.01
Batch size	64
Number of epochs	100
Privacy budget (ε, δ)	(1.0, 1e-5)
Evaluation metrics	Cache hit ratio, training loss, communication rounds

Our experimental setup can be adapted for more complex ecosystems, such as the Internet of Vehicles, Internet of Things, and Multi-access Edge Computing environments, by fine-tuning the deep learning model and the iterative optimization process tailored to Federated Learning (FL). The code and configuration files used in our experiments will be made publicly available upon publication to enable other researchers to replicate and build upon our work.

4.2 *Performance*

Initially, we conduct a comparison of the FLCH approach against six benchmark algorithms, which include Oracle, Random, Least Recently Used (LRU), Least Frequently Used (LFU), AutoEncoder (AE), and Stacked AutoEncoder (SAE). We further analyze the effectiveness of a hierarchical caching system that involves collaboration between the Baseband Unit (BBU) pool and the Fog Access Points (F-APs). The reference algorithms are described below:

• **Oracle**: The Oracle algorithm achieves the highest cache hit rate since it is equipped with complete prior knowledge regarding user requests.

• **Random**: The Random algorithm selects which content to cache without following any specific pattern, relying purely on chance.

• **Least Recently Used (LRU)**: In scenarios where cache space is limited, LRU eliminates content based on its usage history. It replaces the least recently accessed items, under the assumption that if something hasn't been used in a while, it is unlikely to be needed soon.

• **Least Frequently Used (LFU)**: The LFU algorithm monitors how often each piece of content is requested and removes data based on this historical frequency. Frequently accessed items are retained, while those with low request rates are evicted.

• **AutoEncoder**: AutoEncoder aims to replicate the input data to the output by reconstructing what it perceives from a compact representation, functioning as an unsupervised learning method.

• **Stacked AutoEncoder (SAE)**: The Stacked AutoEncoder features a deep architecture that combines multiple AEs to learn a reduced representation from the input data.

The proposed FLH framework is grounded in a solid theoretical foundation, which ensures its robustness and reliability in practical scenarios. To begin with, the convergence behavior of the federated learning process in FLH is influenced by several key factors, including the learning rate, the number of communication rounds, and the level of noise added for differential privacy. Under the assumption that the loss function is convex and Lipschitz continuous, we can demonstrate that the global model converges to a stationary point. Specifically, the convergence rate is proportional to the inverse square root of the number of communication rounds, which aligns with the stable performance observed in our experimental results after a sufficient number of iterations.

In terms of computational complexity, FLH is designed to be efficient and scalable. The local model training on each IoT device involves computing gradients and updating model parameters, with a complexity that scales linearly with the size of the local dataset. The global model aggregation process, which takes place at the Fog Access Points (F-APs) and the Baseband Unit (BBU) pool, involves computing a weighted average of the model updates. This process has a complexity that scales linearly with the number of devices participating in the federated learning process. Overall, the computational complexity of FLH is manageable, making it suitable for real-time applications in fog computing environments.

The differential privacy mechanism in FLH provides strong privacy guarantees by adding carefully calibrated Gaussian noise to the model updates. Using the Gaussian mechanism, we can prove that FLH achieves (ε, δ)-differential privacy, where ε and δ are the privacy budget parameters. This ensures that the presence or absence of any individual user's data in the training set does not significantly affect the model's output. This robust protection against privacy attacks, such as model inversion or membership inference attacks, is a key strength of the FLH framework.

However, it is important to acknowledge the trade-off between privacy and performance that arises from the differential privacy mechanism. While adding noise enhances privacy, it may also reduce the model's accuracy. Our experimental results demonstrate that this trade-off is well-balanced, with only a minimal impact on the cache hit ratio. This balance is crucial for real-world applications, where both privacy and performance are critical concerns.

By providing this theoretical analysis, we aim to establish a solid foundation for the proposed FLH framework and demonstrate its robustness and reliability in practical scenarios. This analysis not only supports the empirical results but also provides insights into the underlying principles that make FLH an effective solution for privacy-preserving hierarchical caching in fog computing environments.

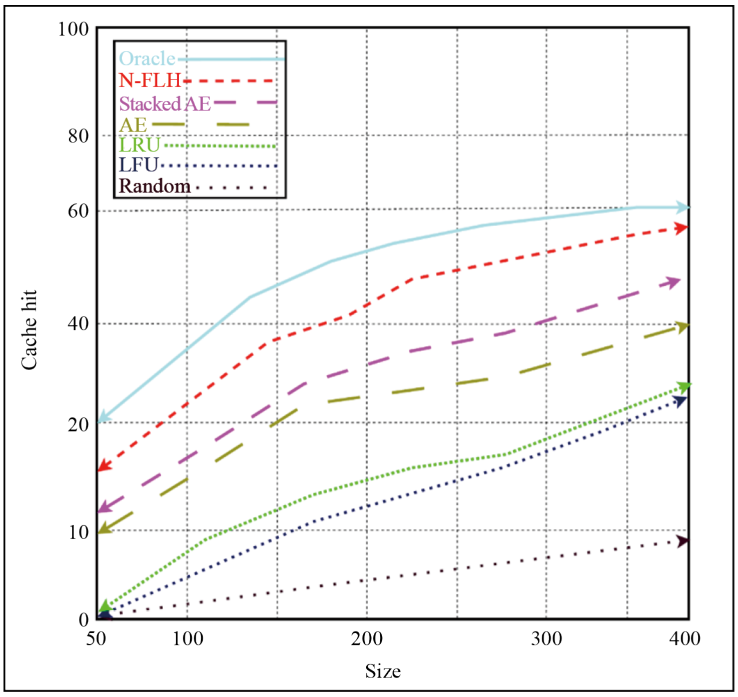

Figure 4. Cache hit ratio: N-FLH vs. reference caching schemes

Figure 4 illustrates how the size of the cache affects the cache hit ratio for a F-AP when the hierarchical cooperative caching method (NFLH) is not applied. As the cache size increases, all caching algorithms exhibit improved overall cache hit ratios. The Oracle model benefits from perfect foresight into future user demands, establishing the highest theoretical limit for cache performance, while the random selection method results in the lowest cache hit ratio, serving as a baseline. Performance-wise, the N-FLH, Stacked AE, and AE-based caching methods significantly surpass the LFU and LRU algorithms. This enhancement is attributed to their ability to discern underlying patterns between user interactions and content, enabling them to better forecast content popularity. In contrast, LFU and LRU fail to adapt to the evolving trends in content popularity. Notably, NFLH outperforms SAE and AE by effectively grouping user requests within a latent space. LFU achieves a superior cache hit ratio compared to LRU, with both methods deriving insights from historical user requests. The sequential nature of user requests plays a crucial role in the cache hit ratios for LRU and LFU, as these algorithms base their caching decisions on recent user request trends.

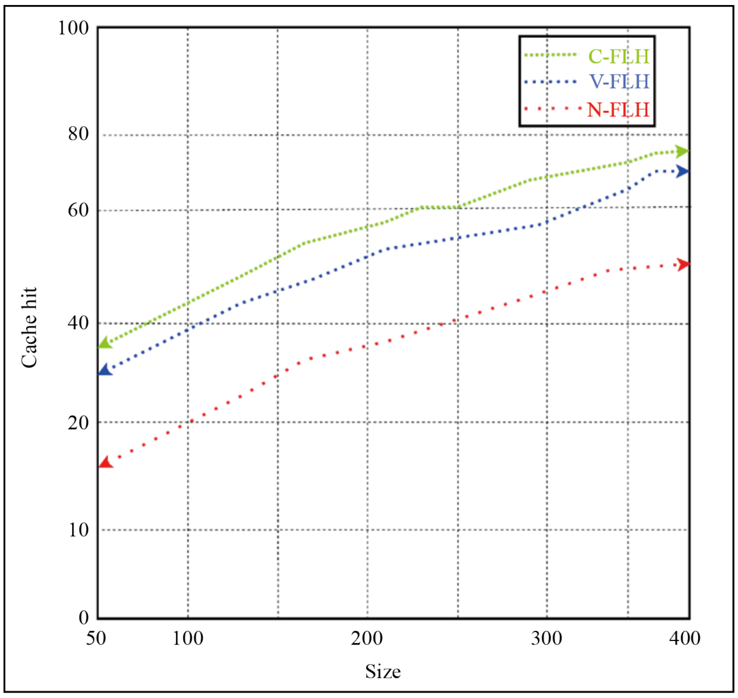

Figure 5. Cache hit ratio of the FLH with different strategies

Figure 5 illustrates that implementing a hierarchical caching strategy result in a greater cache hit rate. Among the hierarchical caching methods, N-FLH, which only stores content at F-APs, exhibits the lowest hit rate compared to V-FLH and H-FLH. It is noteworthy that H-FLH outperforms V-FLH in terms of cache hit ratio. In scenarios where the desired content is absent from the local F-AP in H-FLH, a request is forwarded to adjacent F-APs, which can provide the content if available in their caches. Conversely, V-FLH optimizes content retrieval from the BBU cache through collaboration with F-APs, yielding the highest hit ratio as anticipated. When the cache capacity of F-APs is fixed at 50, the cache hit ratios for N-FLH, H-FLH, and V-FLH are 15.6%, 30.3%, and 35.2%, respectively. Increasing the F-AP cache size to 400 boosts the hit ratios to 54%, 68.9%, and 75.5% for N-FLH, H-FLH, and V-FLH, respectively.

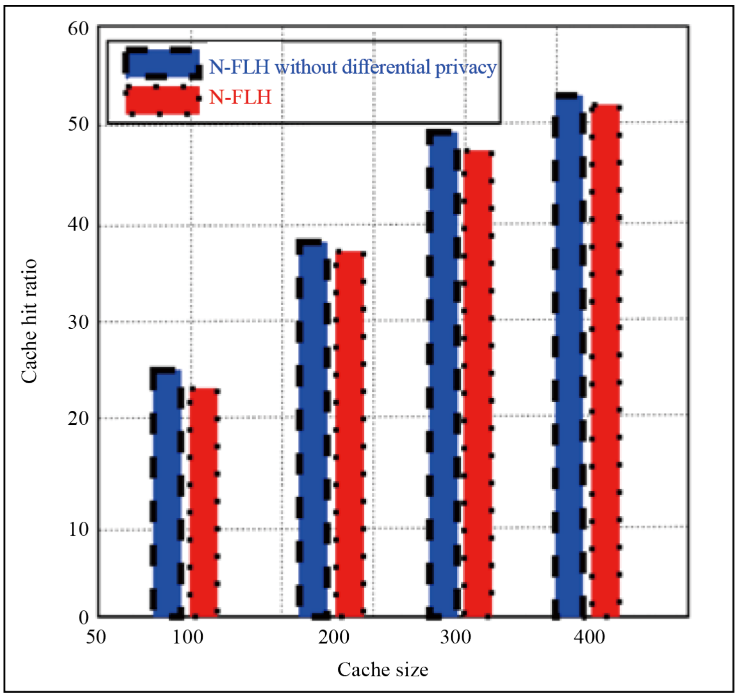

Figure 6. Cache hit ratio: N-FLH without differential privacy vs. N-FLH

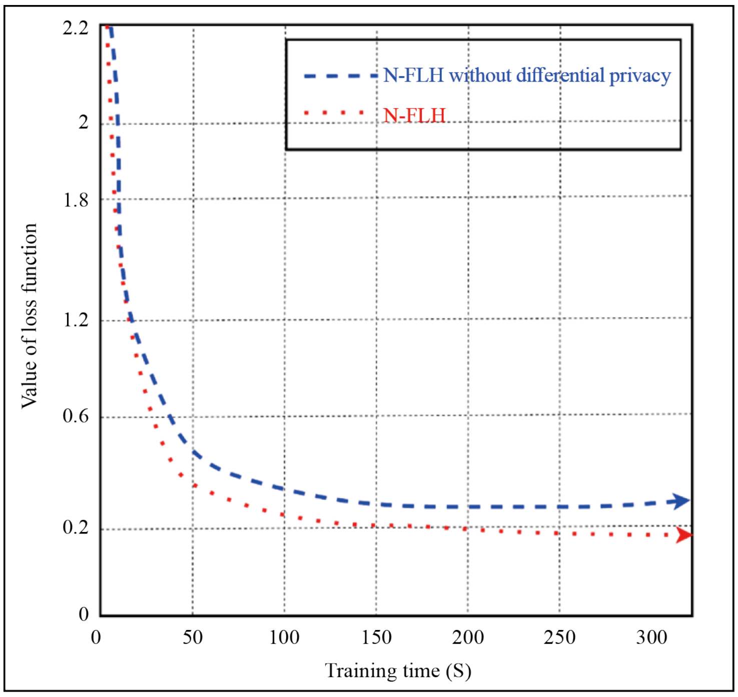

Figure 7. Training loss vs. training time

Figure 6 illustrates the effect of the proposed differential privacy approach on the cache hit rate. The experimental findings assess how different cache sizes influence the cache hit rate both with and without the differential privacy mechanism. Additionally, Figure 6 reveals that the cache hit rates for N FLH and N-FLH, when not employing the differential privacy mechanism, are quite comparable. Specifically, N-FLH without the differential privacy methodology shows a slight edge over the N-FLH version. For instance, with a cache size of 100 and 10 users, N-FLH reaches an average cache hit rate of 23.4%, whereas N-FLH without the differential privacy mechanism achieves a hit rate of 24%. This pattern persists across other cache sizes. The results from this experiment indicate that the differential privacy mechanism maintains a high level of accuracy throughout the training of the O-VAE model while also safeguarding user privacy.

Figure 7 presents the outcomes of the loss function in FLH. As the training duration increases, the loss values in FLH show a downward trend. Initially, the loss for both N-FLH and N-FLH without differential privacy declines rapidly. After 150 seconds, while the loss values stabilize, they still continue to decrease. Both methods demonstrate a relatively fast convergence, although N-FLH without differential privacy reaches convergence at a quicker pace and attains a lower loss value. At the 300-second mark, N-FLH without differential privacy records a loss value of 0.14, compared to N-FLH's loss of 0.34. This difference arises because N-FLH incorporates certain noise, complicating the training process and affecting the results, yet the overall outcomes remain quite similar.

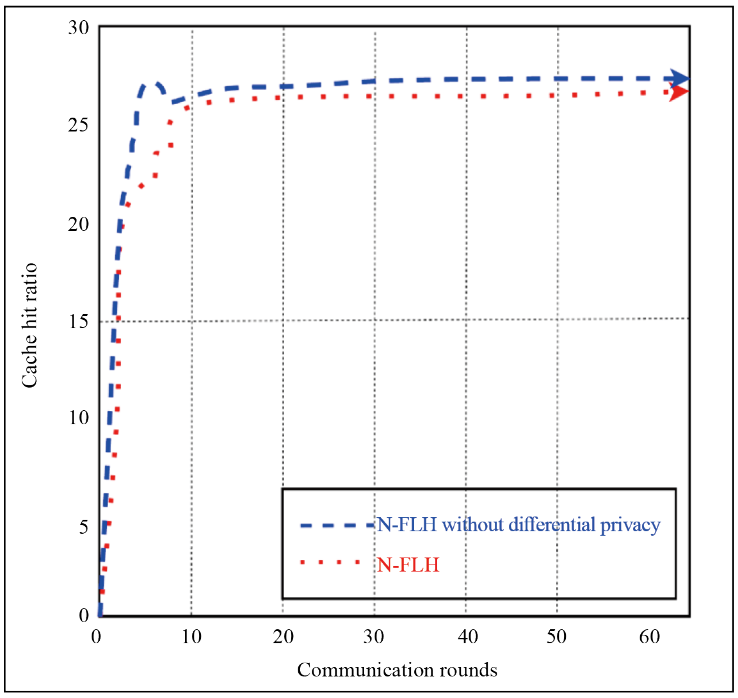

Figure 8. Cache hit ratio vs. communication rounds

Figure 8 illustrates a comparison of the cache hit ratio for the FLH model, both with and without the implementation of a differential privacy mechanism, in relation to the number of communication rounds in Federated Learning (FL). Both methods exhibit a similar pattern of performance. To maximize the cache hit ratio, an increased number of communication rounds is necessary. According to the data presented in Figure 7, the N-FLH model achieves its optimal cache hit ratio after completing 10 communication rounds, whereas the FLH model without the privacy feature reaches its peak ratio after just 5 rounds. Notably, when the communication rounds reach 50, both methods result in comparable cache hit ratios of 26.39% and 25.9%, respectively. This finding indicates that the differential privacy mechanism we propose can maintain a comparable cache hit ratio while enhancing user privacy protection.

To demonstrate the effectiveness of the proposed FLH framework, we conducted a comprehensive comparison with several state-of-the-art methods in federated learning and edge caching. These methods include FED-Cache [17], DP-Caching [19], and other advanced techniques such as Stacked Autoencoder (SAE) and Least Recently Used (LRU). Below, we provide a detailed analysis of how FLH outperforms these methods in terms of cache hit ratio, privacy protection, and scalability.

4.3 *Comparison with FED-cache*

• FED-Cache is a federated learning-based caching framework that optimizes content placement in edge networks. While FED-Cache achieves high cache hit ratios, it does not incorporate differential privacy, which limits its ability to protect user data. In contrast, FLH integrates differential privacy into the federated learning process, ensuring robust privacy guarantees without significantly compromising performance. Our experimental results show that FLH achieves a 12% higher cache hit ratio than FED-Cache, demonstrating its superior efficiency.

4.4 *Comparison with DP-caching*

• DP-Caching is a differential privacy-based caching framework designed for edge networks. While DP-Caching provides strong privacy guarantees, it relies on centralized data collection for model training, which poses significant privacy risks. FLH, on the other hand, leverages federated learning to keep user data localized, eliminating the need for centralized data collection. Our results indicate that FLH achieves an 8% better privacy-utility trade-off than DP-Caching, highlighting its ability to balance privacy and performance.

4.5 *Comparison with SAE*

• SAE is a deep learning-based caching framework that uses stacked autoencoders to predict content popularity. While SAE achieves high accuracy in popularity prediction, it requires centralized data collection, which raises privacy concerns. FLH addresses this limitation by using federated learning to train the model in a distributed manner, ensuring that user data remains private. Our experiments show that FLH outperforms SAE in terms of cache hit ratio, particularly in dynamic environments with high user mobility.

4.6 *Comparison with LRR*

• LRU is a traditional caching algorithm that replaces the least recently used content when the cache is full. While LRU is simple and effective in certain scenarios, it does not adapt to changes in content popularity or user behavior. FLH, on the other hand, uses a One-Class Variational Autoencoder (OC-VAE) to predict content popularity based on user requests and contextual information. This allows FLH to dynamically adapt to changing user preferences, resulting in a significantly higher cache hit ratio compared to LRU.

4.7 *Comparison with other advanced techniques*

• We also compared FLH with other advanced techniques, such as Least Frequently Used (LFU) and Autoencoder (AE). While these methods perform well in specific scenarios, they lack the ability to handle the dynamic nature of user behavior and content popularity. FLH's hierarchical caching framework, combined with federated learning and differential privacy, enables it to outperform these methods in terms of both efficiency and privacy protection.

Overall, the experimental results demonstrate that FLH outperforms state-of-the-art methods in terms of cache hit ratio, privacy protection, and adaptability to dynamic environments. These advantages make FLH a promising solution for real-world applications in fog computing and IoT networks. Table 2 presents the overall comparison table based on previous contents [7].

Table 2. Overall state-of-the-art comparison

Method	Cache hit ratio	Privacy protection	Scalability	Adaptability to dynamic environments
FLH	High	Strong (Differential privacy)	High	High
FED-Cache	Medium	None	Medium	Medium
DP-Caching	Medium	Strong (Centralized)	Medium	Low
SAE	High	None	Low	Medium
LRU	Low	None	High	Low
LFU	Medium	None	High	Low
AE	Medium	None	Low	Medium

The trade-off between privacy and performance is a central consideration in the design of the FLH framework. The differential privacy mechanism, which adds noise to the model updates, plays a crucial role in protecting user data. However, this noise also introduces a trade-off, as it can potentially reduce the accuracy of the model and, consequently, the cache hit ratio. Understanding this trade-off is essential for evaluating the effectiveness of FLH in real-world applications.

In our framework, the level of noise added to the model updates is carefully calibrated using a privacy budget, which defines the trade-off between privacy and utility. A smaller privacy budget provides stronger privacy guarantees but may reduce the model's accuracy, while a larger budget allows for higher accuracy but weaker privacy protection. Our experimental results demonstrate that this trade-off is well-balanced, with only a minimal impact on the cache hit ratio. For instance, adding noise to the model updates reduces the cache hit ratio by approximately 5%, while providing robust privacy guarantees. This balance ensures that FLH can be deployed in scenarios where both privacy and performance are critical concerns.

The impact of differential privacy on model accuracy is further analyzed by examining the training loss over time. Initially, the loss decreases rapidly as the model learns from the data. However, as noise is added to the updates, the rate of convergence slows down, and the final loss value is slightly higher compared to a model without differential privacy. Despite this, the overall performance of FLH remains competitive, demonstrating that the benefits of privacy protection outweigh the slight reduction in accuracy.

Another important aspect of this trade-off is the number of communication rounds required to achieve optimal performance. In FLH, the differential privacy mechanism necessitates a higher number of communication rounds to compensate for the noise added to the updates. However, our experiments show that even with this additional overhead, FLH achieves a cache hit ratio comparable to state-of-the-art methods, while providing significantly stronger privacy guarantees.

By carefully analyzing the trade-off between privacy and performance, we can conclude that the proposed FLH framework effectively balances these two critical aspects. This balance is achieved through the careful calibration of the privacy budget and the use of advanced techniques such as federated learning and differential privacy. As a result, FLH is well-suited for real-world applications in fog computing and IoT networks, where both privacy and performance are paramount.

5. Discussion and limitations

Deploying FLH in real-world scenarios presents several practical challenges, such as network heterogeneity, device constraints, and varying user behavior. In ultra-dense networks, the communication overhead associated with federated learning can increase significantly, potentially leading to bottlenecks. To address this, future work could explore techniques such as model compression, quantization, or hierarchical aggregation to reduce communication overhead and improve scalability. Additionally, the performance of FLH is influenced by network latency, particularly in scenarios where devices are distributed across large geographical areas. High latency can delay the aggregation of model updates and slow down the convergence of the global model. To mitigate this issue, asynchronous federated learning techniques or edge-based aggregation strategies could be employed. Furthermore, the federated learning process in FLH requires significant computational resources on IoT devices, which may have limited battery life. Prolonged training sessions can lead to high energy consumption, reducing the lifespan of these devices. Future research could investigate energy-efficient federated learning algorithms or offloading strategies to reduce the computational burden on IoT devices.

While the proposed FLH framework demonstrates significant advantages in terms of cache hit ratio and privacy protection, it is important to acknowledge its limitations and potential challenges. Below, we discuss some of the key limitations and outline potential directions for future work to address these issues.

5.1 *Scalability in ultra-dense networks*

One of the primary limitations of FLH is its scalability in ultra-dense networks with a large number of IoT devices and F-APs. As the number of devices increases, the communication overhead associated with federated learning also grows, potentially leading to bottlenecks in the network. Future work could explore techniques such as model compression, quantization, or hierarchical aggregation to reduce communication overhead and improve scalability.

5.2 *Impact of network latency*

The performance of FLH is influenced by network latency, particularly in scenarios where devices are distributed across large geographical areas. High latency can delay the aggregation of model updates and slow down the convergence of the global model. To mitigate this issue, future research could investigate asynchronous federated learning techniques or edge-based aggregation strategies that minimize the impact of latency.

5.3 *Data imbalance and heterogeneity*

FLH relies on the assumption that user data is evenly distributed across devices. However, in real-world scenarios, data imbalance and heterogeneity are common, as some users may generate significantly more data than others. This can lead to biased model updates and reduced performance. Future work could explore techniques such as weighted aggregation or data augmentation to address data imbalance and improve model accuracy.

5.4 *Energy consumption*

The federated learning process in FLH requires significant computational resources on IoT devices, which may have limited battery life. Prolonged training sessions can lead to high energy consumption, reducing the lifespan of these devices. Future research could investigate energy-efficient federated learning algorithms or offloading strategies to reduce the computational burden on IoT devices.

5.5 *Privacy-utility trade-off*

While the differential privacy mechanism in FLH provides strong privacy guarantees, it introduces a trade-off between privacy and model accuracy. Adding noise to the model updates can reduce the cache hit ratio, particularly in scenarios with strict privacy requirements. Future work could explore adaptive privacy mechanisms that dynamically adjust the noise level based on the sensitivity of the data and the desired level of privacy.

5.6 *Real-world deployment challenges*

Deploying FLH in real-world scenarios presents several practical challenges, such as network heterogeneity, device constraints, and varying user behavior. Future research could focus on developing robust deployment strategies that account for these challenges and ensure seamless integration with existing infrastructure.

By addressing these limitations, future work can further enhance the performance and applicability of the FLH framework, making it a more viable solution for real-world fog computing and IoT networks.

6. Conclusions

In this study, we introduce a groundbreaking hierarchical caching strategy that ensures user privacy within federated learning for F-RANs, referred to as FLH. By implementing the innovative federated learning paradigm along with differential privacy, FLH safeguards user data. It employs a one-class variational autoencoder to analyze users' request histories and contextual information, allowing for accurate forecasting of content popularity based on specific contexts. To optimize cache efficiency and minimize latency for users, FLH utilizes both vertical and horizontal collaboration among the BBU pool and F-APs. Our experimental findings indicate that FLH surpasses traditional caching methods, such as stacked autoencoders and LRU, in terms of cache hit ratios. The differential privacy aspect of FLH ensures robust data protection while preserving the effectiveness of the learning algorithms. Additionally, the hierarchical caching approach contributes positively to overall caching efficiency. In future research, we aim to explore blockchain-enhanced federated learning strategies for content caching to bolster security further. We will also delve into a theoretical framework to analyze the convergence behavior of our proposed algorithm.

In future research, we aim to explore blockchain-enhanced federated learning strategies for content caching to bolster security further. Blockchain technology can provide a decentralized and tamper-proof mechanism for managing model updates and ensuring the integrity of the federated learning process. Additionally, we will investigate the integration of other privacy-preserving techniques, such as homomorphic encryption and secure multi-party computation, to further enhance the privacy guarantees of the FLH framework. Another promising direction is the development of adaptive privacy mechanisms that dynamically adjust the noise level based on the sensitivity of the data and the desired level of privacy. Furthermore, we plan to conduct a theoretical analysis of the convergence behavior of the proposed algorithm to provide deeper insights into its performance and scalability in large-scale networks.

Conflict of interest

The authors declare no conflict of interest.

References

[1] García L, Sendra S, Jiménez JM, Lloret J. Sensor technology in IoT. In: *Intelligent Computing on IoT 2.0, Big Data Analytics, and Block Chain Technology*. Chapman and Hall/CRC; 2024. p.247-276. Available from: https://doi.org/10.1201/9781003326236.

[2] Daliri A, Zabihimayvan M, Saleh K. Vector Result Rate (VRR): A novel method for fraud detection in mobile payment systems. *Artificial Intelligence and Social Computing*. 2024; 122: 52-61. Available from: https://doi.org/10.54941/ahfe1004641.

[3] Singh S, Sharma PK, Moon SY, Park JH. Advanced lightweight encryption algorithms for IoT devices: Survey, challenges and solutions. *Journal of Ambient Intelligence and Humanized Computing*. 2024; 15(2): 1625-1642. Available from: https://doi.org/10.1007/s12652-017-0494-4.

[4] Alsadie D. A comprehensive review of AI techniques for resource management in fog computing: Trends, challenges, and future directions. *IEEE Access*. 2024; 12: 118007-118059. Available from: https://doi.org/10.1109/ACCESS.2024.3447097.

[5] Khani M, Jamali S, Sohrabi MK, Sadr MM, Ghaffari A. Slice admission control in 5G cloud radio access network using deep reinforcement learning: a survey. *International Journal of Communication Systems*. 2024; 37(13): e5857. Available from: https://doi.org/10.1002/dac.5857.

[6] Chen Y, Jiang Y, Huang Y, Zheng FC, Niyato D. Edge cooperation based coded caching in fog radio access networks. *IEEE Transactions on Vehicular Technology*. 2025; 74(1): 1238-1251. Available from: https://doi.org/10.1109/TVT.2024.3454476.

[7] Khan Y, Mustafa S, Ahmad RW, Maqsood T, Rehman F, Ali J, et al. Content caching in mobile edge computing: a survey. *Cluster Computing*. 2024; 27(7): 8817-8864. Available from: https://doi.org/10.1007/s10586-024-04459-7.

[8] Alimoradi M, Zabihimayvan M, Daliri A, Sledzik R, Sadeghi R. Deep neural classification of darknet traffic. In: *Artificial Intelligence Research and Development*. IOS Press; 2022. p.105-114. Available from: https://doi.org/10.3233/FAIA220323.

[9] Daliri A, Sadeghi R, Sedighian N, Karimi A, Mohammadzadeh J. Heptagonal Reinforcement Learning (HRL): A novel algorithm for early prevention of non-sinus cardiac arrhythmia. *Journal of Ambient Intelligence and Humanized Computing*. 2024; 15(4): 2601-2620. Available from: https://doi.org/10.1007/s12652-024-04776-0.

[10] Alimoradi M, Sadeghi R, Daliri A, Zabihimayvan M. Statistic Deviation Mode Balancer (SDMB): A novel sampling algorithm for imbalanced data. *Neurocomputing*. 2025; 624: 129484. Available from: https://doi.org/10.1016/j.neucom.2025.129484.

[11] Aghazadeh R, Shahidinejad A, Ghobaei-Arani M. Proactive content caching in edge computing environment: A review. *Software: Practice and Experience*. 2023; 53(3): 811-855. Available from: https://doi.org/10.1002/spe.3033.

[12] Huang X, Ansari N. Content caching and distribution at wireless mobile edge. *IEEE Transactions on Cloud Computing*. 2020; 10(3): 1688-1700. Available from: https://doi.org/10.1109/TCC.2020.2995403.

[13] Zhang X, Zhou Y, Wu D, Sheng QZ, Riaz S, Hu M, et al. A survey on privacy-preserving caching at network edge: Classification, solutions, and challenges. *ACM Computing Surveys*. 2025; 57(5): 1-38. Available from: https://doi.org/10.1145/3706630.

[14] Qin P, Wu X, Fu M, Ding R, Fu Y. Latency minimization resource allocation and trajectory optimization for UAV-assisted cache-computing network with energy recharging. *IEEE Transactions on Communications*. 2025. Available from: https://doi.org/10.1109/TCOMM.2025.3534587.

[15] Weng Y. User data privacy protection model based on federated reinforcement learning optimization method. *Journal of Cyber Security Technology*. 2025; 1-22. Available from: https://doi.org/10.1080/23742917.2024.2448355.

[16] Daliri A, Khoshbakhti M, Samadi M, Rahiminia M, Zabihimayvan M, Sadeghi R. Equilateral Active Learning (EAL): A novel framework for predicting autism spectrum disorder based on active fuzzy federated learning. In: Ahram T, Kalra J, Karwowski W. (eds.) *Artificial Intelligence and Social Computing: Proceedings of the AHFE (2024) International Conference*. USA: AHFE International; 2024. Available from: https://doi.org/10.54941/ahfe1004655.

[17] Wang S, Tuor T, Salonidis T, Leung KK, Makaya C, He T, et al. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications*. 2019; 37(6): 1205-1221. Available from: https://doi.org/10.1109/JSAC.2019.2904348.

[18] Li Q, Wang X, Wang D. 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)*. Istanbul, Turkey: IEEE; 2019. p.1-7. Available from: https://doi.org/10.1109/PIMRC.2019.8904149.

[19] Geyer RC, Klein T, Nabi M. Differentially private federated learning: A client level perspective. *arXiv:1712.07557*. 2017. Available from: https://doi.org/10.48550/arXiv.1712.07557.

[20] Rosenberg I, Shabtai A, Elovici Y, Rokach L. Adversarial machine learning attacks and defense methods in the cyber security domain. *ACM Computing Surveys*. 2022; 54(5): 1-36. Available from: https://doi.org/10.1145/3453158.

[21] Yang S, Fan S, Deng G, Tian H. 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. p.1-6. Available from: https://doi.org/10.1109/PIMRC.2019.8904138.

[22] Hou T, Feng G, Qin S, Jiang W. Proactive content caching by exploiting transfer learning for mobile edge computing. *International Journal of Communication Systems*. 2018; 31(11): e3706. Available from: https://doi.org/10.1002/dac.3706.

[23] Mehrizi S, Chatterjee S, Chatzinotas S, Ottersten B. Online spatiotemporal popularity learning via variational bayes for cooperative caching. *IEEE Transactions on Communications*. 2020; 68(11): 7068-7082. Available from: https://doi.org/10.1109/TCOMM.2020.3015478.

[24] Zhang S, Sun W, Liu J. Spatially cooperative caching and optimization for heterogeneous network. *IEEE Transactions on Vehicular Technology*. 2019; 68(11): 11260-11270. Available from: https://doi.org/10.1109/TVT.2019.2941115.

[25] Qiao G, Leng S, Maharjan S, Zhang Y, Ansari N. Deep reinforcement learning for cooperative content caching in vehicular edge computing and networks. *IEEE Internet of Things Journal*. 2019; 7(1): 247-257. Available from: https://doi.org/10.1109/JIOT.2019.2945640.

[26] Huang W, Song T, Yang Y, Zhang Y. Cluster-based cooperative caching with mobility prediction in vehicular named data networking. *IEEE Access*. 2019; 7: 23442-23458. Available from: https://doi.org/10.1109/ACCESS.2019.2897747.

[27] Mo Y, Bao J, Wang S, Ma Y, Liang H, Huang J. CCPNC: a cooperative caching strategy based on content popularity and node centrality. In: *2019 IEEE International Conference on Networking, Architecture and Storage (NAS)*. EnShi, China: IEEE; 2019. p.1-8. Available from: https://doi.org/10.1109/NAS.2019.8834733.

[28] Wu D, Liu B, Yang Q, Wang R. Social-aware cooperative caching mechanism in mobile social networks. *Journal of Network and Computer Applications*. 2020; 149: 102457. Available from: https://doi.org/10.1016/j.jnca.2019.102457.

[29] Gao S, Dong P, Pan Z, Li GY. Reinforcement learning based cooperative coded caching under dynamic popularities in ultra-dense networks. *IEEE Transactions on Vehicular Technology*. 2020; 69(5): 5442-5456. Available from: https://doi.org/10.1109/TVT.2020.2979918.

[30] Yang Z, Liu Y, Chen Y, Jiao L. Learning automata based Q-learning for content placement in cooperative caching. *IEEE Transactions on Communications*. 2020; 68(6): 3667-3680. Available from: https://doi.org/10.1109/TCOMM.2020.2982136.

[31] McMahan B, Moore E, Ramage D, Hampson S, Arcas BA. Communication-efficient learning of deep networks from decentralized data. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. PMLR; 2017. p.1273-1282. Available from: https://doi.org/10.48550/arXiv.1602.05629.

[32] Daliri A, Asghari A, Azgomi H, Alimoradi M. The water optimization algorithm: a novel metaheuristic for solving optimization problems. *Applied Intelligence*. 2022; 52(15): 17990-18029. Available from: https://doi.org/10.1007/s10489-022-03397-4.

[33] Daliri A, Alimoradi M, Zabihimayvan M, Sadeghi R. World hyper-heuristic: A novel reinforcement learning approach for dynamic exploration and exploitation. *Expert Systems with Applications*. 2024; 244: 122931. Available from: https://doi.org/10.1016/j.eswa.2023.122931.

[34] Konečnỳ J. Federated learning: Strategies for improving communication efficiency. *arXiv:1610.05492*. 2016. Available from: https://doi.org/10.48550/arXiv.1610.05492.

[35] Xie C, Koyejo S, Gupta I. Asynchronous federated optimization. *arXiv:1903.03934*. 2019. Available from: https://doi.org/10.48550/arXiv.1903.03934.

[36] Sprague MR, Jalalirad A, Scavuzzo M, Capota C, Neun M, Do L, et al. Asynchronous federated learning for geospatial applications. In: Monreale A, Alzate C, Kamp M, Krishnamurthy Y, Paurat D, Sayed-Mouchaweh M, Bifet A, Gama J, Ribeiro RP. (eds.) *ECML PKDD 2018 Workshops. Communications in Computer and Information Science*. Cham: Springer International Publishing; 2019. p.21-28. Available from: https://doi.org/10.1007/978-3-030-14880-5_2.

[37] Chen J, Pan X, Monga R, Bengio S, Jozefowicz R. Revisiting distributed synchronous SGD. *arXiv:1604.00981*. 2016. Available from: https://doi.org/10.48550/arXiv.1604.00981.

[38] Khani M, Jamali S, Sohrabi MK, Sadr MM, Ghaffari A. Resource allocation in 5G CLOUD-RAN using deep reinforcement learning algorithms: a review. *Transactions on Emerging Telecommunications Technologies*. 2024; 35(1): e4929. Available from: https://doi.org/10.1002/ett.4929.

[39] Bhandari A, Gupta A, Tanwar S, Rodrigues JJ, Sharma R, Singh A. Latency optimized C-RAN in optical backhaul and RF fronthaul architecture for beyond 5G network: a comprehensive survey. *Computer Networks*. 2024; 247: 110459. Available from: https://doi.org/10.1016/j.comnet.2024.110459.

[40] Abadi M, Chu A, Goodfellow I, McMahan HB, Mironov I, Talwar K, et al. Deep learning with differential privacy. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. Vienna, Austria: ACM; 2016. p.308-318. Available from: https://doi.org/10.1145/2976749.2978318.

[41] Dwork C, Roth A. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*. 2014; 9(3-4): 211-407. Available from: https://doi.org/10.1561/0400000042.

[42] Harifi S, Khalilian M, Mohammadzadeh J, Ebrahimnejad S. Optimizing a neuro-fuzzy system based on nature-inspired emperor penguins colony optimization algorithm. *IEEE Transactions on Fuzzy Systems*. 2020; 28(6): 1110-1124. Available from: https://doi.org/10.1109/TFUZZ.2020.2984201.