sensors

Article

Edge Caching Based on Collaborative Filtering for Heterogeneous ICN-IoT Applications

Divya Gupta [1,†], Shalli Rani [1,*], Syed Hassan Ahmed [2], Sahil Verma [3], Muhammad Fazal Ijaz [4,†] and Jana Shafi [5,*]

1 Chitkara University Institute of Engineering and Technology, Chitkara University, Rajpura 140401, Punjab, India; divya.gupta@chitkara.edu.in or divya190789gupta@gmail.com
2 Independent Researcher, Corona, CA 13088, USA; sh.ahmed@ieee.org
3 Department of Computer Science and Engineering, Chandigarh University, Mohali 140413, India; sahilverma@ieee.org
4 Department of Intelligent Mechatronics Engineering, Sejong University, Seoul 05006, Korea; fazal@sejong.ac.kr
5 Department of Computer Science, College of Arts and Science, Prince Sattam Bin Abdul University, Wadi Ad-Dwasir 11991, Saudi Arabia
* Correspondence: shalli.rani@chitkara.edu.in (S.R.); j.jana@psau.edu.sa (J.S.)
† Divya Gupta and Muhammad Fazal Ijaz contributed equally to this work and are first co-authors.

Abstract: The substantial advancements offered by the edge computing has indicated serious evolutionary improvements for the internet of things (IoT) technology. The rigid design philosophy of the traditional network architecture limits its scope to meet future demands. However, information centric networking (ICN) is envisioned as a promising architecture to bridge the huge gaps and maintain IoT networks, mostly referred as ICN-IoT. The edge-enabled ICN-IoT architecture always demands efficient in-network caching techniques for supporting better user's quality of experience (QoE). In this paper, we propose an enhanced ICN-IoT content caching strategy by enabling artificial intelligence (AI)-based collaborative filtering within the edge cloud to support heterogeneous IoT architecture. This collaborative filtering-based content caching strategy would intelligently cache content on edge nodes for traffic management at cloud databases. The evaluations has been conducted to check the performance of the proposed strategy over various benchmark strategies, such as LCE, LCD, CL4M, and ProbCache. The analytical results demonstrate the better performance of our proposed strategy with average gain of 15% for cache hit ratio, 12% reduction in content retrieval delay, and 28% reduced average hop count in comparison to best considered LCD. We believe that the proposed strategy will contribute an effective solution to the related studies in this domain.

Keywords: information centric networking; internet of things; collaborative filtering; edge cloud; content caching

check for updates

Citation: Gupta, D.; Rani, S.; Ahmed, S.H.; Verma, S.; Ijaz, M.F.; Shafi, J. Edge Caching Based on Collaborative Filtering for Heterogeneous ICN-IoT Applications. *Sensors* **2021**, *21*, 5491. https://doi.org/10.3390/s21165491

Academic Editor: Roberto Beraldi

Received: 21 July 2021
Accepted: 13 August 2021
Published: 15 August 2021

1. Introduction

In recent years, the advancements in the internet of things (IoT) technology has gained a lot of popularity. Today, investigation on different forms for providing IoT as a solution is attracting both industry and academia, as well as seeking attention more than ever [1,2]. Unlike Wireless Sensor networks (WSNs), IoT offers tremendous scope for nodes and their connections. The recent progress ensures easy utilization of IoT in various applications, which include, but are not limited to, smart grids, smart education, intelligent transportation systems, e-healthcare, smart industries, smart agriculture, smart cities, smart homes, and wearables [3]. Due to the plethora of applications support from various disciplines, IoT has now become a bridge between human and real world for information communication. These extensive IoT applications, with almost different nature, have introduced complications in existing wireless communication systems and have, therefore, formed a heterogeneous IoT for today's real information world. Although the current developments in mobile communication, with its newly launched 5G, offer enhanced mobile broadband (eMBB), massive machine type communication (mMTC), and ultra-reliable and low latency communication (URLLC), still, these heterogeneous IoT always possess higher requirements in terms of reliable connection, low cost, high speed, minimum delay, and scalable communication [4]. Due to limited resource constraints, such as storage and computing with IoT devices, these complex and diverse requirements of heterogeneous IoT tasks can be computed by effective utilization of cloud computing technology, where a network cloud has abundance of these resources. The ever-increasing growth in the IoT technology has offered various advanced functions to several IoT devices. For example, a smart phone today can perform various computations that was earlier possible only using computers or laptops. This simply corresponds to cloud computing in close proximity to users. However, with the massive data being generated by these heterogeneous IoT devices and extremely high latency offered during IoT to cloud communication, the simultaneous access by several IoT devices have demanded high bandwidth requirements. Therefore, the conventional single cloud computing model cannot satisfy all these requirements to meet quality of experience (QoE) [5]. The edge computing has been extended as a layer between cloud and IoT in this process of advancements to provide significant solution [6]. The edge cloud offers sufficient resources for computation and storage requirements [7]. Indeed, the different features offered by this edge computing model has provided various solutions in different domains, but it still faces different challenges, and research in this field is in its infancy [8]. The edge computing itself lacks intelligent computation and communication. The intelligent decision making for different computations based on different scenarios can be implemented by deploying artificial intelligence (AI) technology in the edge cloud and central cloud. The intelligent algorithms were generally deployed on these clouds to make them work smarter. The various fields where AI has proved its effectiveness include robotics, image processing, natural language processing, speech recognition, and so on [9]. Recently, cloud computing has started using AI's cognitive services to offer better QoE. Moreover, the delays involved in current internet architecture for IoT content distribution due to its IP address-based approach where content has to be fetched from target device offers various challenges. This can be managed efficiently by leveraging information centric networking (ICN) communication in IoT and is referred as ICN-IoT [10,11].

To facilitate enhanced network performance, this study aims to achieve faster content distribution for IoT device user's requested content in support to higher cache hit ratio, reduced delay, and low path stretch. The integration of ICN with IoT would support fast searching based on content names, as well as caching of content, within network nodes. The edge cloud would offer low overhead to central cloud for request processing, as well as low delay to user requests. The AI deployed in edge cloud would be utilized for making intelligent decision in regards to requested content caching on edge nodes. Our proposed solution for effectively managing the massive traffic flows on the central cloud mainly considers caching of frequently requested contents at edge clouds which are in near proximity to IoT end users. The proposed solution leverages collaborative filtering and k-means clustering techniques for making intelligent caching decisions at edge nodes. Indeed, various studies in literature do exist for dealing with these challenges. Similar to this, the authors in Reference [12,13] proposed use of small distributed data centers in entire network to reduce burden at core cloud network, and some works [14–16] offer prefetching of content at edge nodes to alleviate and control back-haul traffic. Although the concept of collaborative filtering was utilized by various online business applications, websites and live streaming services for generating recommendations for user's preferences, yet, found its application in networking domain after the proposal presented in Reference [17]. The author presented effective benefit for content placement decision based on collaborative filtering. However, this could also be used as a improved solution for various other challenges, such as traffic bottlenecks, bandwidth wastage, and timely content delivery. Despite its several uses in various applications [18–20], item-based collaborative filtering in support to better QoE is still not properly utilized in networking and communication. In addition, this concept has been mainly implemented in the networks supporting TCP/IP architecture, which further degrades system performance due to its target-based delivery approach.

Considering benefits associated with the combination of all these technologies, such as ICN-IoT, cloud computing, and edge computing, along with the deployment of AI, our main focus is on designing AI-enabled edge model for intelligent content caching strategy which is suitable for heterogeneous IoT architectures to effectively manage massive traffic flow on central clouds.

To this end, following are the contributions made in this paper:

- We propose an enhanced ICN-IoT content caching strategy by enabling AI's collaborative filtering within edge cloud to support heterogeneous IoT architectures for traffic management at conventional cloud computing model. An architecture is designed by combining ICN-IoT, edge, cloud, and AI for heterogeneous IoT applications to provide an enhanced hardware model which support user's QoE.
- We propose a content-based collaborative filtering caching technique for intelligently caching content on edge nodes. Through the combination of a wireless communication model, and collaborative filtering caching model of edge nodes, a content fetching algorithm is designed to retrieve user's data efficiently.
- We perform extensive simulations to validate the effectiveness of our proposed scheme over state-of-the-art caching strategies, such as LCE, LCD, ProbCache, and CL4M. The results obtained from experimentation prove that our proposed scheme significantly achieves higher cache hit ratio. In addition, the proposed scheme is efficient to achieve lower content delay and reduced path stretch when compared to these strategies.

The rest of this paper is structured as follows: Section 2 provides brief discussion on the related work in this domain. The system model of our proposed strategy, along with its major components, is presented in Section 3. In Section 4, the detailed description of our proposed design, while considering content caching based on collaborative filtering, is given. The performance of the proposed approach is evaluated in Section 5. Finally, the conclusion of this study is presented in Section 6.

2. Related Work

This section represents some recent studies in relation to IoT, ICN, edge computing, and artificial intelligence, either as an individual technology or in combination for designing efficient caching strategies. The studies of these works would be beneficial to justify our motivation for conducting this study and support academic achievements in the research field.

2.1. IoT and ICN

With ever-increasing traffic on the internet due to several connected IoT devices, the management of each IP-based content request following conventional network architecture has started to impose a challenge for its performance. Combining IoT with ICN, the work in Reference [21] focused on incorporating ICN in IoT. The benefits gained by IoT from ICN and various challenges being addressed through this combined architecture is introduced in this study. The works in Reference [21] continued research in this combined field and addressed various issues in this combined architecture by introducing ICN-based IoT architecture. Further, work in Reference [22] listed comparison of IoT integration with different CCN standards and offered some improvements towards data traffic management. In order to reduce the energy consumption in IoT networks, an ICN-based forwarding strategy for data transmission was proposed by Reference [23]. The work in Reference [24] investigated a unique naming scheme for smooth interest packet flow between NDN-IoT-based regions. Similarly, management of the data packet overhead was discussed in Reference [25]. Similarly, working towards energy efficiency, the work in Reference [26] proposed an ICN by combining with specific IoT approach called TSCH.

Further, the authors in Reference [27] proposed a context-based approach for IoT data communication utilizing ICN architecture. The approach mainly focused on correct routing and forwarding of information with management of FIB and PIT data structures. Moreover, Quevedo et al., in Reference [28], presented various caching strategies while considering bandwidth and energy issues faced by this integration. Generally, cache schemes focus on providing solution to three issues namely which location is best suited to cache data, what content should be cached and how the content to be stored. The default caching scheme is Cache Everything Everywhere (CEE) [29], where each intermediate node locally caches each piece of content which passes through it. The policy is straightforward but results in high content redundancy. The simple location-based caching is Leave Copy Down (LCD) [30], which locates content at the node, i.e., one level below where cache hit occurs along the delivery path. However, due to frequent requests of popular contents, at some instance, all nodes will have a copy of same content, leading to cached content duplication, as well as cache full. The prob(p) [30] is another simple yet stateless caching scheme where content on routers are cached based on some probability. The scheme does not consider the router's location while making a caching decision. To resolve the location issue, ProbCache provides high probability for caching when content is near to the consumer [31]. The scheme manages to reduce redundancy but at the cost of low cache utilization of the node far from the consumer. To support chunk level caching, WAVE, based on content popularity and nodes co-ordination, was proposed [32]. The approach is similar to LCD and differs by caching content exponentially at the neighbor based on frequency of content. The scheme stores frequent requests near the edge router but does not consider data packet caching time.

The various broad caching strategies, such as probability-based [30,31], popularity aware [33,34], reactive caching [30,32], proactive caching [35,36], and non-cooperative [29,36], were designed for various areas, such as mobile devices, IoT, 5G, vehicular networking, ad hoc networks, etc. However, these strategies do not fit for an environment with limited resources due to their own limitations, in one way or another. For non-cooperative caching, every node individually makes a caching decision on whether to cache content or not, hence resulting in cache redundancy and no effective utilization of resources.

To address the problems in non-cooperative caching, researchers focused on design of cooperative caching schemes for ICN. In cooperative caching, network nodes work in collaboration with others for making caching decision.

2.2. IoT and Edge Computing

The support for IoT applications using edge computing model was presented by authors in Reference [37]. The performance comparison in terms of energy efficiency and content fetching delay was conducted to highlight importance of edge in IoT. Further, researchers in Reference [38] focused on challenges faced by current approaches of IoT and recommended use of fog computing in IoT scenarios based on several reasons. The study in Reference [39] analyzed the security aspects offered by IoT-fog computing when compared with IoT-clouds. Sarkar et al., in Reference [40], investigated the suitability of fog computing for meeting the requirements from various heterogeneous IoT applications which, in reality, are not feasible to accomplish with use of traditional cloud model. The work in Reference [41] presented different new approaches for combining in IoT architecture and discussed the benefit of incorporating mobile edge computing (MEC) in IoT, which itself adopted the fog computing model. In addition, the researchers in Reference [42] proposed a model for agreement of resources. The work was mainly focused around achieving efficient resource management and demonstrating its effectiveness after evaluation on the cloudSIM toolkit. The study in Reference [43] proposed a model for supporting reasonable and effective communication among IoT devices. The algorithm used the concept of matching theory for accomplishment of node pairs. Further, the feasibility while combining fog computing with smart gateways was analyzed in Reference [44]. The authors in Reference [45] proposed home-box networks for efficient content delivery in peer to peer overlay networks. The design architecture considered several delays to improve the service performance in the IoT domain.

Moreover, given limitations offered due to inflexible design of Fog-IoT architecture to meet current demands, the work in Reference [21] proposed smart collaborative caching by leveraging ICN for IoT in a fog environment. the solution was designed to achieve content caching, node location tracing, and resource sharing. Further, the authors in Reference [46] proposed a joint optimization solution for fog-IoT networks which basically deals with issues related to content caching, computation offloading, and resource sharing. The paper proposed a solution based on actor-critic-reinforcement learning to solve joint optimization issues. Similarly, working on 3C, i.e., computation, caching, and communication, Luo et al. [47] proposed an efficient algorithm based on an iterative task team formulation method for solving these issues as a subproblem, with minimum possible cost. The researchers in Reference [11] proposed a fog-based caching scheme in an IoT environment by utilizing ICN. The proposed solution worked towards offering minimum latency to user content requests by providing content near to edge networks based on its popularity. Working in the same direction to offer minimum service delay to IoT nodes while reducing energy consumption, work in Reference [48] proposed smart clustering mechanism by utilizing both fog nodes (FNs) and terminal nodes (TNs).

2.3. *Artificial Intelligence in Edge Caching*

The artificial intelligence in terms of machine learning and deep learning can be applied to wireless edge networks for deciding what content to cache and where to cache so that caching objectives are optimized. The authors in Reference [49] proposed a caching scheme which predicts the popularity of the new video based on the similarity of features which are already present in the published video. The work in Reference [50] proposed popularity-based supervised and deep learning framework for caching at base stations in mobile edge computing networks. Further, the work in Reference [51] considered learn-to-rank algorithm and k-means clustering for caching content in small networks. The scheme aims to maximize cache hit ratio (CHR), and the solution of optimization problem is NP-hard if content popularity is not known. The algorithm was designed based on historical content requests. The works in References [52,53] presented proactive learning-based caching at small base stations and user equipment to meet user satisfaction ratio. Chang et al. [54] presented a big data and ML-based framework for caching content inside edge networks. The smart caching in edge networks was explained using two case studies, where the first was designed by combining unsupervised learning and deep learning, and the second was using social ties between end users. The reinforcement learning is also used by many studies for deciding content caching. The learning process of an RL agent can be modeled as optimal control of a Markov Decision Process (MDP). Based on this, the work in Reference [55] presented base station-based distributed caching and delivery framework. The cache replacement transmission was minimized using an MDP optimization solution based on variables, such as popularity and transmission cost of cache replacement from one base station to another. The model uses the Q-learning approach for transmission cost minimization. The authors in Reference [56] proposed deep reinforcement learning approach for content caching and network slicing in vehicular environment. The work in Reference [57] proposed multi-tier content caching based on deep Q-learning to support improved performance in radio access networks. The comparison of different works based on several parameters is represented in Table 1.

Table 1. AI-based caching techniques.

Ref	Machine Learning Technique	Algorithm	Objective	Caching Strategy	Caching Location	Network
[49]	Supervised	CNN	High computation offloading ratio	Proactive	Base station	wireless cellular
[50]	Supervised	DNN	Reduced content retrieval delay	Proactive	Base station	Mobile edge computing
[51]	Supervised and unsupervised	Learn-to-rank	Improved cache hit ratio	Reactive	Small base station	Small cell network
[52,53]	Supervised	NA	High user satisfaction ratio	Proactive	Base station, user equipment	Small cell network
[54]	Unsupervised	DNN	Minimize latency, High data rate	Proactive	Mobile base station	Hetnet
[55]	Reinforcement learning	Q-learning	minimum cache replacement transmission cost	Proactive	Base station, user equipment	Macro cellular
[56]	Reinforcement learning	Deep RL	maximum cache hit ratio	Reactive	V2I	Radio access networks
[57]	Reinforcement learning	Deep Q-learning	maximum cache hit ratio	Proactive	Base station, user equipment, access point	Radio access networks

3. System Model

This paper considers an ICN-based heterogeneous IoT environment. The users in this network can send message to content providers to receive required content [4]. A three-tier network architecture serving various IoT applications is presented in Figure 1. Layer 1, also known as the top layer, represents the core network comprising various cloud servers. These servers are assumed to have all the data demanded by the users. In addition, these servers maintain records of all access history from various edge nodes (ENs) that it serves. The middle layer (layer 2) is mainly comprised of several ENs, where each EN is connected with number of end users. The several users from diversified ICN-IoT applications constitute the bottom layer of this architecture. Here, we represent all ENs as set $EN = \{EN_1, EN_2, \dots, EN_M\}$ for M number of total available ENs in the network. The users in the network are represented as set $U = \{u_1, u_2, \dots, u_K\}$ for K number of total users, where each user may have varying preferences. The cloud server stores total Ncontents and can be represented as set $C = \{c_1, c_2, \dots, c_N\}$, where each piece of content is assumed to have equal size of T Mbits. Moreover, each piece of content is supposed to have set F of J attributes, $F = \{f_1, f_2, \dots, f_J\}$, where each attribute denotes some feature. For example, if content represents clothes, the attribute may specify the genre of the clothes, such as western, fashion, ethnic, indo-western, etc., with each piece of clothing having an index value between 0 and 1 for each genre. For instance, a western piece of clothing with lots of cuts and short length may be assigned larger index value for its style and fashion attributes.

Based on Tables 2 and 3, for each EN and content, the sum of all the values of the attributes is equal to 1, i.e., $\sum_{i=1}^{J} f_i = 1$ for every C and EN.

To calculate the number of times a specific content C_q has been requested by any EN_p, a history of requests is created, as shown in Table 4. The value of particular cell $REQ_{p,q}$ is, therefore, calculated as :

$$REQ_{p,q} = \frac{EN_p * C_q}{J} \leq 1. \tag{1}$$

Here, EN_p represents rows in Table 2, and C_q represents columns in Table 3. On the other hand, the value for cell $REQ_{p,q}$ can also be retrieved from history by setting $REQ_{p,q}$ to the fraction of times content C_q has been requested by EN_p among all content requests. This REQ is important and would be useful for deciding the content placement inside ENs.

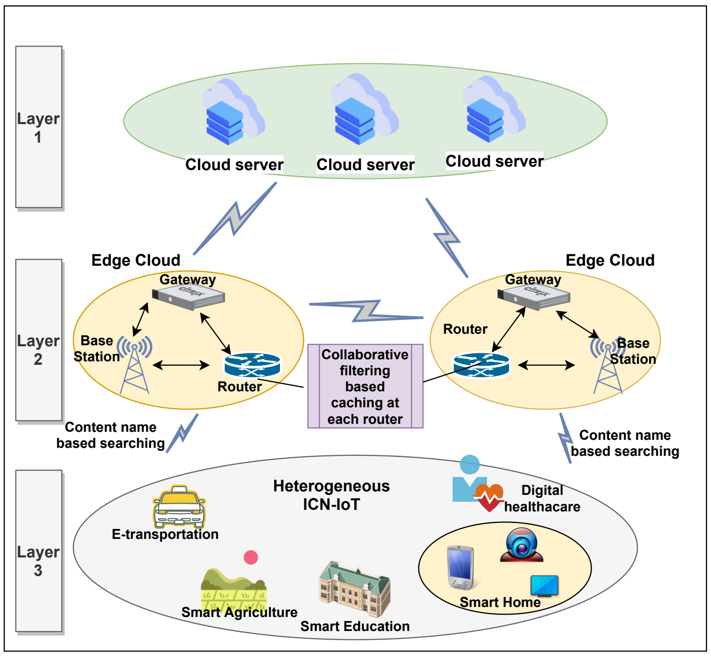

Figure 1. Three-tier network architecture.

Table 2. *EN* features.

Edge Nodes/Features	f_1	f_2	$\cdots$	f_J
EN_1	...	...	...	...
EN_2	...	...	...	...
...	...	...	...	...
EN_M	...	...	...	...

Table 3. Content features.

Features/Contents	C_1	C_2	$\cdots$	C_N
f_1	...	...	...	...
f_2	...	...	...	...
...	...	...	...	...
f_J	...	...	...	...

The cloud center is a huge database with capacity equal to *vol*. All the content requested by IoT user can be provided through a traditional cloud data center but with longer delays due to congestion bottleneck. The edge nodes on edge clouds can cache some content frequently requested by users to offer reduced latency. To provide simplicity to our design approach, we consider homogeneous cache capacity for each edge node. Therefore, we assume each EN is equipped with cache of equal size, say $Size_L$, such that

$$Size_L = vol * \rho, \tag{2}$$

where ρ contains a value between 0 and 1, i.e., $0 < \rho < 1$, and vol represents the total capacity of the cloud server database. $Size_L$ denotes a very small cache space available to each EN as compared to total volume vol.

Table 4. REQ: History of requests.

Edge Nodes/Contents	C_1	C_2	$\cdots$	C_N
ED$_1$	...	...	...	...
ED$_2$	...	...	...	...
...	...	...	...	...
ED$_M$	...	...	...	...

In our design model, we consider dividing the cache size of each EN into two equal halves, where one half, i.e., $Size_{L1} = \frac{Size_L}{2}$, is used for caching the content based on $EN's$ local popularity Lp for each EN. $Size_{L1}$ contains a set of Z most popular contents $C_1, C_2, \ldots, C_Z$ arranged in descending order of their popularities. The content ranking depends on the value of Pop, where Pop_{EN_p,C_q} represents the popularity of C_q in edge node EN_p. This Pop value can be obtained from REQ (history of requests). The other half of cache space, i.e., $Size_{L2} = Size_L - Size_{L1}$, is used to cache V contents based on content-based collaborative filtering technique. Both cache spaces are utilized to perform content placement during off-peak hours so as to minimize network traffic in peak hours.

4. Proposed Framework

The AI-enabled content placement and caching among edge nodes to support heterogeneous IoT applications is proposed using collaborative filtering technique.

4.1. Edge Clustering

In order to obtain benefit from intelligent content caching strategy, all the edge nodes are partitioned into several clusters [48]. Based on the entries present in REQ (Table 4), where values can be obtained using Equation (1), we apply k-means clustering to group the ENs into G clusters (CLs), i.e., $CL_1, CL_2, \ldots, CL_G$ [58]. We use cosine distance metric (refer to Equation (3)) to calculate the distance between any pair of ENs, say EN_i and EN_j [52]. ENs which belong to the same cluster are going to have distance values approximately near to 0, whereas ENs belonging to different clusters will have distance value near to 1. Moreover, the zero value represents the EN is locating exactly in the middle of two CLs.

$$dist(EN_i, EN_j) = 1 - \frac{\sum_{m=1}^{N} REQ_{i,m} REQ_{j,m}}{\sqrt{\sum_{m=1}^{N} (REQ_{i,m})^2}\sqrt{\sum_{m=1}^{N} (REQ_{j,m})^2}}. \tag{3}$$

Here, $REQ_{i,m}$ represents the number of requests EN_i has made for content C_m.

Further, to decide the optimal number of CLs, there are various methods already available in the literature. We used Silhouette coefficient method as a metric for deciding the efficiency of clustering method [59,60]. This method is generally used to estimate how much certain observation fits to its cluster. To decide the optimal number of clusters, the average silhouette coefficient is calculated for every possible number of clusters, and the one with the highest average silhouette coefficient value is chosen.

4.2. Edge Caching

As discussed in the previous section, each ED is equipped with certain cache space, i.e., $Size_L$. Out of total available space, one portion, i.e., $Size_L1$, is used to cache contents based on the local popularity of the content. However, the rest of the portion $Size_L2$ is used to cache content based on the highest probability of content to be requested in future. To predict the probability of future requests for available contents, we apply content-based collaborative filtering. To proceed in this direction, firstly, the similarity index between any two pair of contents says C_i and C_j is being calculated using cosine coefficient as given in Equation (4) [53].

$$Sim(C_i, C_j) = \frac{C_i * C_j}{\parallel C_i \parallel * \parallel C_j \parallel} = \frac{\sum_{b=1}^{J} C_{b,i} * C_{b,j}}{\sqrt{\sum_{b=1}^{J}(C_{b,i})^2}\sqrt{\sum_{b=1}^{J}(C_{b,j})^2}}. \tag{4}$$

Here, C_i and C_j are the ith and jth contents being represented as a column in Table 3, and b is the bth feature from total available J number of features. In contrast to the cosine distance coefficient, cosine similarity coefficient has value 1 when two contents are similar. The value 0 represents no similarity between two contents. The similarity index between the same file will always result to value 0, i.e., $sim(C_i, C_i) = 0$ for all i (refer to Table 5).

Using Tables 4 and 5, we construct a content prediction table CP (Table 6), where $CP_{i,j}$ can be calculated using formula:

$$CP_{i,j} = \sum_{d=1, d \neq j}^{N'} sim(C_d, C_j) * REQ_{i,d}. \tag{5}$$

Here, N' represents set of all the contents requested by ED_i. The higher value of $CP_{i,j}$ represents high probability of content C_j being requested by edge node ED_i. Based on the prediction results, ED_i will cache contents with high future request probability in to its $Size_{L2}$ space.

Table 5. Content similarity.

Similarity	C_1	C_2	⋯	C_i	⋯	C_j	⋯	C_N
C_1	0	⋯	⋯	⋯	⋯	⋯	⋯	⋯
C_2	⋯	0	⋯	⋯	⋯	⋯	⋯	⋯
⋯	⋯	⋯	0	⋯	⋯	⋯	⋯	⋯
C_i	⋯	⋯	⋯	0	⋯	⋯	⋯	⋯
⋯	⋯	⋯	⋯	⋯	0	⋯	⋯	⋯

Table 6. CP: Content prediction.

Prediction	C_1	C_2	⋯	C_i	⋯	C_j	⋯	C_N
ED$_1$	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯
ED$_2$	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯
⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯
ED$_M$	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯

4.3. Content Fetching

Based on the content caching scheme discussed above, each ED caches content in its allocated space. Each user U_x must be associated with at least one ED, based on the nearest distance rule. Considering a request for a content C_i by user U_x from an edge device ED_j, the local cache of ED_j would be checked first to determine if it contains requested content C_i. In case of a match, C_i would be delivered to U_x locally. On the other hand, if ED_j does not cache a copy of C_i, it will start searching C_i among all the EDs belonging to its own cluster CL_s. If C_i is cached by any of the EDs belonging to CL_s, it would be delivered to U_x without caching in to ED_j. Otherwise, in case of search failure inside CL_s, the content would be requested from central cloud server and would be cached locally inside ED_j, based on LRU replacement policy. Algorithm 1 explains the content fetching procedure. The likelihood of the content requests generated by users is based on the history of requests (refer to Table 4). Each user would be connected to same ED for a duration that is long enough to allow full content delivery.

Algorithm 1: Content Searching C_i.

```
Input: $U_x$ associated with $ED_j$
Output: $C_i$
Begin:
 1  For each $C_i$ requested by $U_x$ do
 2  Check $C_i$ in $ED_j$'s cache
 3  if ((($Size_{L1}, Size_{L2}$) $\leftarrow$ check ($C_i$) ) $\neq$ 1)
 4      if(($CL_s$ $\leftarrow$ check ($C_i$)) $\neq$ 1)
 5          cloud Database $\leftarrow$ check($C_i$)
 6              $ED_j$ caches $C_i$
 7              return $C_i$ to $U_x$
 8      else
 9          return $C_i$ to $U_x$
10      endif
11  else
12      return $C_i$ to $U_x$
13  endif
End
```

5. Evaluation Scenario

The efficiency of the proposed strategy has been evaluated against ICN-IoT benchmark caching schemes by implementing simulations in Icarus simulator. The simulator with four building blocks, such as scenario generation, scenario orchestration, experiments execution, and result collection, is specifically designed for research in field of ICN caching and routing. Before evaluation, network is initially warm up with 3×10^5 number of requests. The warm-up requests are initial messages which are sent to network caching nodes to perform content caching before analyzing system performance. To measure performance evaluation, measured requests are set to 6×10^5 and are sent in network after completion of warm-up phase. Each user sends request messages that follow Poisson distribution as this is the most widely used distribution by various caching strategies during implementation. The network request rate is set to default value as in Icarus settings, i.e., 12 requests per second for the whole network. The content popularity follows Zipfian distribution, with skewness parameter α ranges from 0.6 to 1.2. In Zipfian distribution, the value α corresponds to concentration of user preference. The large α value signifies higher concentration of user preferences; in case of α = 1.25, more users are interested in same request in contrast to α = 0.8. The experiments are performed using tree network topology as this is highly preferred for performance evaluation in recent works. To maintain the fairness, the uniform storage capacity is allocated to each node by dividing total network caching capacity with the number of contents. For experimentation purpose, this study considers different scenarios by varying network cache from 0.04% to 5%. In addition to content placement scenario generation, the replacement of content is also important. The proposed strategy considers LRU for cache replacement due to its low complexity and high consistency with already available ubiquitous caching schemes. Based on all the aforementioned settings, the experiments were performed for 4 different content popularity (α) values with consideration of 4 network cache capacities for each network topology. The simulations was performed twice, and the average value for each performance metric was considered to compare the proposed strategy with other benchmark schemes.

5.1. Performance Metrics

From the wide variety of metrics available for computing significance of caching strategies [61], this work evaluates the performance of proposed strategy based on the following:

5.1.1. Cache Hit Ratio (CHR)

CHR determines the ratio of number of requests processed by edge caching nodes rather than cloud servers. Assuming total M requests being served by network, if request for any content k is cached inside an edge node and can be served to requester, then it is counted as cache hits ($Cache_{hits}$). In case of requested content not found in edge node, it is served by a cloud server and is considered as cache miss ($Cache_{miss}$) The formula for calculating CHR is:

$$Cache\ \ Hit\ \ Ratio\ \ (CHR) = \frac{Cache_{hits}}{Cache_{hits} + Cache_{miss}}. \tag{6}$$

5.1.2. Content Retrieval Delay (CRD)

CRD refers to the total delay incorporated in getting the requested content k by an end user. In order to calculate total delay, the forwarding operation of both request message for content k and response message with content object K are considered. Therefore, CRD is computed using the formula given in the following equation.

$$Content\ \ Retrieval\ \ Delay\ \ (CRD)\ \ = request\ \ travel\ \ delay\ \ +\ \ response\ \ travel\ \ delay. \tag{7}$$

5.1.3. Average Hop Count (AHC)

AHC defines the average number of hops a content request needs to travel in order to be satisfied when normalized over total hops until reaching original server. The 0 value of AHC tends to requests that are served more closely to user, hence the caching strategy shows its highest efficiency. The value of AHC can be computed using the formula given in the following equation.

$$Average\ \ Hop\ \ Count\ \ (AHC)\ \ = \frac{Number\ \ of\ \ hops\ \ travelled}{Number\ \ of\ \ hops\ \ to\ \ server}. \tag{8}$$

5.2. Simulation Results

The evaluation results of the proposed strategy has been compared against various benchmark caching schemes, which include LCE, LCD, CL4M, and ProbCache. This section represents the results obtained after performing simulations on aforementioned caching strategies for different performance metrics.

5.2.1. Cache Hit Ratio (CHR)

The cache hit ratio is the most important metric to examine the performance of any caching strategy. It determines the percentage of requests being served out of total generated requests. The high CHR always implies reduced burden on the core network as most of the requests are served by intermediate nodes. The initial round of experiments examine the performance of the proposed strategy for CHR. Figures 2 and 3 show the cache hit ratio results of various caching strategies for different cache size and different popularity parameter α, respectively. From the results obtained, it can be observed that the proposed strategy outperforms existing benchmark caching strategies for higher cache hit operations. The reason behind the better performance of the proposed strategy, among others, is due to caching content on edge devices based on content popularity and future prediction. With the increasing popularity of requested content, the more content would be cached at edge nodes, and, hence, higher CHR can be attained.

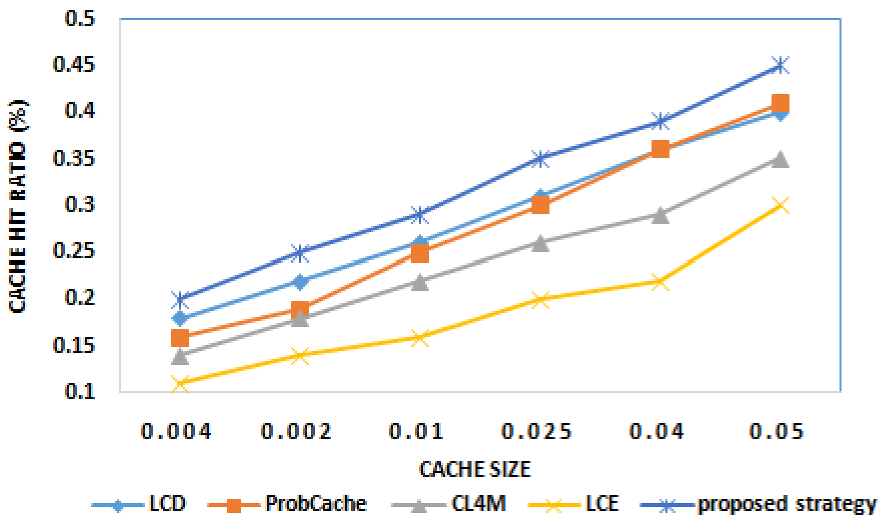

Figure 2. Cache hit ratio for different cache sizes.

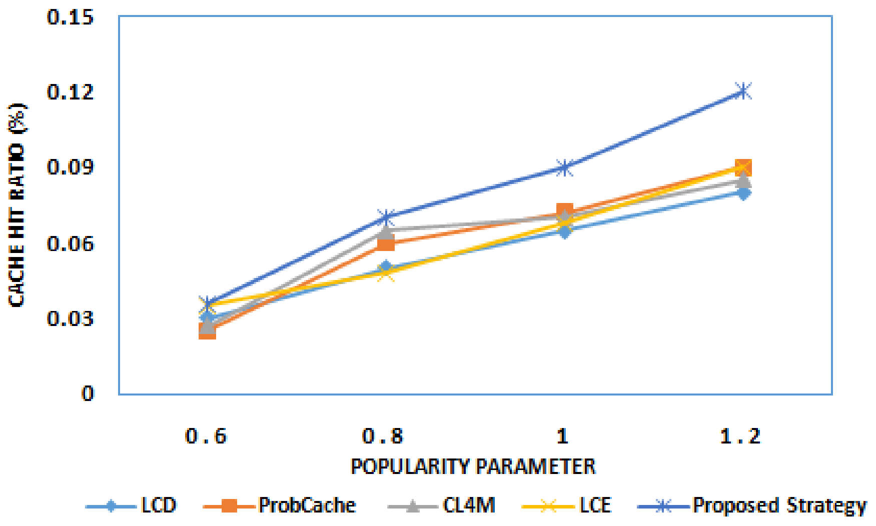

Figure 3. Cache hit ratio for different popularity parameter α.

5.2.2. Content Retrieval Delay (CRD)

The next step of experiments work for interpreting the results of content retrieval delay offered by proposed strategy in comparison to benchmark caching strategies. This metric is mainly used to compute network latency. Figures 4 and 5 show the result of content retrieval delay of various caching strategies for different cache size and different popularity parameter α, respectively. The results clearly reveal the outstanding performance of the proposed strategy among other considered strategies for reduced retrieval delay with increase in cache size, as well as content popularity. The reduction in retrieval delay with gradually increasing cache size is due to higher cache capacity of edge nodes to cache content and provides data packets for user requests. Similar to this, less delay is observed for content having higher popularity because of its caching near the user.

5.2.3. Average Hop Count (AHC)

The last round of experiments investigated the performance of proposed strategy among aforementioned caching strategies, for comparison by considering another challenging metric, i.e., average hop count. Figures 6 and 7 show the average hop count results of various caching strategies for different cache sizes and different popularity parameter α, respectively. From the results obtained, it can be observed that the proposed strategy outperforms existing state of art caching strategies by reducing average number of hops traversed during content delivery operations. For the average hop count, the maximum improvement is recorded in case of proposed strategy (with varying cache size). This is due to the design model of the proposed strategy where caching at intermediate nodes is always aimed to reduce hop count with minimum delay. From the results in Figure 6, the continuous fall in average hop count value with growing cache size can be clearly observed for all caching strategies. This represents the direct significance of cache size on number of hops traversed. The decrease in count of hops with increase in popularity for all strategies can be seen in Figure 7. The caching of the most popular and future predicted content near the user is the reason behind the obtained results.

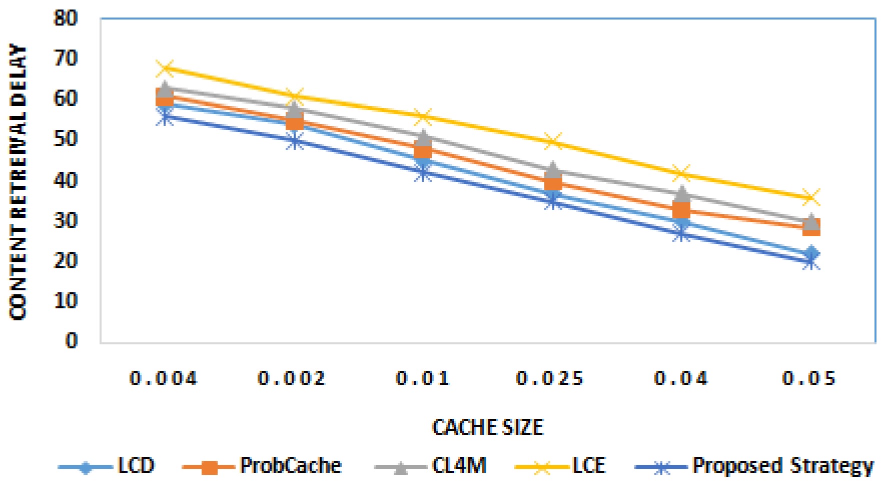

Figure 4. Content retrieval delay for different cache sizes.

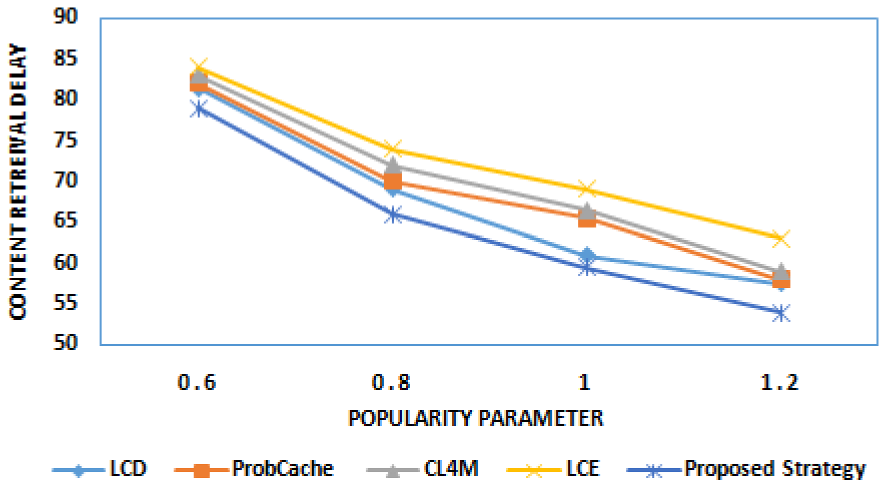

Figure 5. Content retrieval delay for different popularity parameter α.

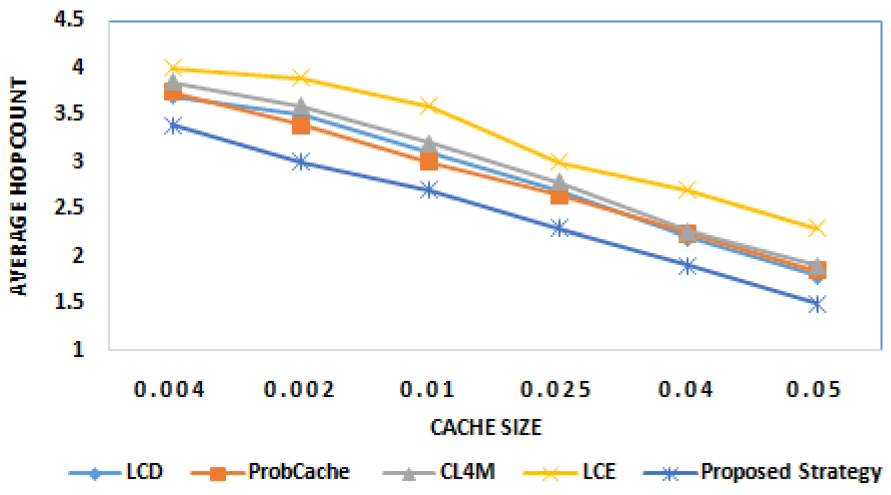

Figure 6. Average hop count for different cache sizes.

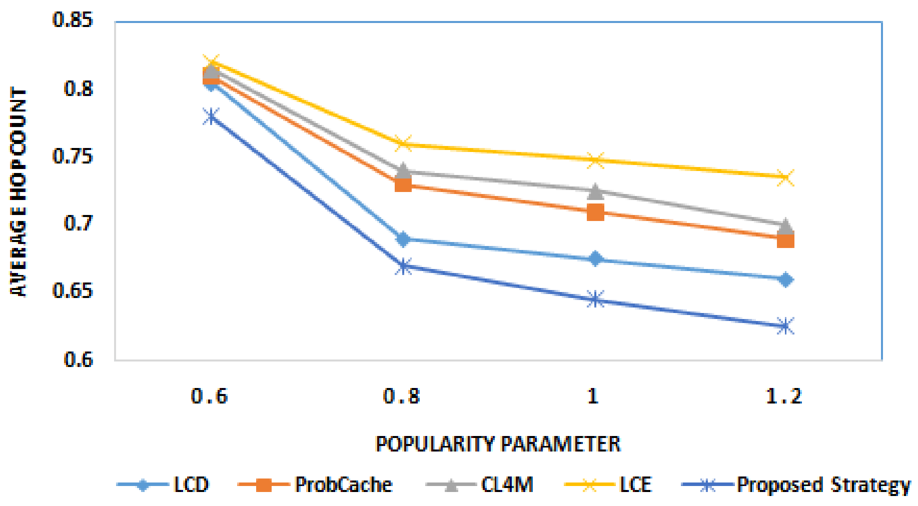

Figure 7. Average hop count for different popularity parameter α.

6. Conclusions

In this paper, we aimed at providing better QoE to user by offering minimum content retrieval delay, as well as reduced average number of hops utilized to obtain desired content and to enhance cache hit ratio on each edge node. To this end, we first considered an edge-enabled heterogeneous ICN-IoT network architecture to satisfy user's latest demands. Further, to support an intelligent caching, we proposed an collaborative filtering-based content caching strategy on each edge cloud where contents would be cached based on its local popularity and predicted future demands. Afterwards, utilizing both edge clustering and caching, an algorithm was designed for fetching content by user from the network to meet QoE. Numerical results revealed the effectiveness of our proposed strategy over proposed strategy over various benchmark strategies, such as LCE, LCD, CL4M, and ProbCache, for achieving considered cache hit ratio and content retrieval delay. The reason behind the improved performance of our strategy in comparison to existing considered benchmark strategies is caching content based on some request history, thereby predicting future demands of users. As the work is carried out in collaborative filtering with various cache sizes, the proposed technique still needs to be checked with various NDN schemes. For the future work, we will design a caching strategy while considering large set of attributes available with the requested content and varying caching capacity of all the edge nodes to further support scalable network with reduced content retrieval latency.

Author Contributions: Conceptualization, D.G. and S.R.; methodology, S.H.A., S.R. and S.V.; validation, M.F.I. and S.R.; formal analysis, S.H.A. and S.R.; investigation, D.G. and J.S.; resources, S.R.; data curation, D.G., S.R. and S.H.A.; writing—original draft preparation, D.G., S.R. and S.H.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Sejong University research fund.

Acknowledgments: Jana Shafi would like to thank the Deanship of Scientific Research, Prince Sattam bin Abdul Aziz University for supporting this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Afzal, B.; Umair, M.; Shah, G.A.; Ahmed, E. Enabling IoT platforms for social IoT applications: Vision, feature mapping, and challenges. *Future Gener. Comput. Syst.* **2019**, *92*, 718–731. [CrossRef]
2. Aldowah, H.; Rehman, S.U.; Umar, I. Trust in IoT Systems: A Vision on the Current Issues, Challenges, and Recommended Solutions. *Adv. Smart Soft Comput.* **2021**, *1188*, 329–339.
3. Hajjaji, Y.; Boulila, W.; Farah, I.R.; Romdhani, I.; Hussain, A. Big data and IoT-based applications in smart environments: A systematic review. *Comput. Sci. Rev.* **2021**, *39*, 100318. [CrossRef]
4. Hao, Y.; Miao, Y.; Hu, L.; Hossain, M.S.; Muhammad, G.; Amin, S.U. Smart-Edge-CoCaCo: AI-enabled smart edge with joint computation, caching, and communication in heterogeneous IoT. *IEEE Netw.* **2019**, *33*, 58–64. [CrossRef]
5. Forestiero, A.; Mastroianni, C.; Papuzzo, G.; Spezzano, G. A proximity-based self-organizing framework for service composition and discovery. In Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, Melbourne, Australia, 17–20 May 2010; pp. 428–437.
6. Zhao, L.; Wang, J.; Liu, J.; Kato, N. Optimal edge resource allocation in IoT-based smart cities. *IEEE Netw.* **2019**, *33*, 30–35. [CrossRef]
7. Markakis, E.K.; Karras, K.; Sideris, A.; Alexiou, G.; Pallis, E. Computing, caching, and communication at the edge: The cornerstone for building a versatile 5G ecosystem. *IEEE Commun. Mag.* **2017**, *55*, 152–157. [CrossRef]
8. Naveen, S.; Kounte, M.R. Key technologies and challenges in IoT edge computing. In Proceedings of the 2019 Third International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 12–14 December 2019; pp. 61–65.
9. Huh, J.H.; Seo, Y.S. Understanding edge computing: Engineering evolution with artificial intelligence. *IEEE Access* **2019**, *7*, 164229–164245. [CrossRef]
10. Arshad, S.; Azam, M.A.; Rehmani, M.H.; Loo, J. Recent advances in information-centric networking-based Internet of Things (ICN-IoT). *IEEE Internet Things J.* **2018**, *6*, 2128–2158. [CrossRef]
11. Hua, Y.; Guan, L.; Kyriakopoulos, K.G. A Fog caching scheme enabled by ICN for IoT environments. *Future Gener. Comput. Syst.* **2020**, *111*, 82–95. [CrossRef]
12. Bittencourt, L.F.; Diaz-Montes, J.; Buyya, R.; Rana, O.F.; Parashar, M. Mobility-aware application scheduling in fog computing. *IEEE Cloud Comput.* **2017**, *4*, 26–35. [CrossRef]
13. Wang, M.; Wu, J.; Li, G.; Li, J.; Li, Q.; Wang, S. Toward mobility support for information-centric IoV in smart city using fog computing. In Proceedings of the 2017 IEEE International Conference on Smart Energy Grid Engineering (SEGE), Oshawa, ON, Canada, 14–17 August 2017; pp. 357–361.
14. Sinky, H.; Khalfi, B.; Hamdaoui, B.; Rayes, A. Adaptive edge-centric cloud content placement for responsive smart cities. *IEEE Netw.* **2019**, *33*, 177–183. [CrossRef]
15. Sinky, H.; Khalfi, B.; Hamdaoui, B.; Rayes, A. Responsive content-centric delivery in large urban communication networks: A LinkNYC use-case. *IEEE Trans. Wirel. Commun.* **2017**, *17*, 1688–1699. [CrossRef]
16. Vallero, G.; Deruyck, M.; Meo, M.; Joseph, W. Base Station switching and edge caching optimisation in high energy-efficiency wireless access network. *Comput. Netw.* **2021**, *192*, 108100. [CrossRef]
17. Niu, Y.; Gao, S.; Liu, N.; Pan, Z.; You, X. Clustered small base stations for cache-enabled wireless networks. In Proceedings of the 2017 9th International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, China, 11–13 October 2017; pp. 1–6.
18. Yu, C.; Tang, Q.; Liu, Z.; Dong, B.; Wei, Z. A recommender system for ordering platform based on an improved collaborative filtering algorithm. In Proceedings of the 2018 International Conference on Audio, Language and Image Processing (ICALIP), Shanghai, China, 16–17 July 2018; pp. 298–302.
19. Cui, Z.; Xu, X.; Fei, X.; Cai, X.; Cao, Y.; Zhang, W.; Chen, J. Personalized recommendation system based on collaborative filtering for IoT scenarios. *IEEE Trans. Serv. Comput.* **2020**, *13*, 685–695. [CrossRef]
20. Wang, W.; Chen, J.; Wang, J.; Chen, J.; Liu, J.; Gong, Z. Trust-enhanced collaborative filtering for personalized point of interests recommendation. *IEEE Trans. Ind. Inform.* **2019**, *16*, 6124–6132. [CrossRef]
21. Song, F.; Ai, Z.Y.; Li, J.J.; Pau, G.; Collotta, M.; You, I.; Zhang, H.K. Smart collaborative caching for information-centric IoT in fog computing. *Sensors* **2017**, *17*, 2512. [CrossRef] [PubMed]

22. Baccelli, E.; Mehlis, C.; Hahm, O.; Schmidt, T.C.; Wählisch, M. Information centric networking in the IoT: Experiments with NDN in the wild. In Proceedings of the 1st ACM Conference on Information-Centric Networking, Paris, France, 24–26 September 2014; pp. 77–86.
23. JSM, L.M.; Lokesh, V.; Polyzos, G.C. Energy efficient context based forwarding strategy in named data networking of things. In Proceedings of the 3rd ACM Conference on Information-Centric Networking, Kyoto, Japan, 26–28 September 2016; pp. 249–254.
24. Datta, S.K.; Bonnet, C. Interworking of NDN with IoT architecture elements: Challenges and solutions. In Proceedings of the 2016 IEEE 5th Global Conference on Consumer Electronics, Kyoto, Japan, 11–14 October 2016; pp. 1–2.
25. Siris, V.A.; Thomas, Y.; Polyzos, G.C. Supporting the IoT over integrated satellite-terrestrial networks using information-centric networking. In Proceedings of the 2016 8th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Larnaca, Cyprus, 21–23 November 2016; pp. 1–5.
26. Hahm, O.; Adjih, C.; Baccelli, E.; Schmidt, T.; Wählisch, M. Time Slotted Channel Hopping and Information-Centric Networking for IoT. Available Online: https://reposit.haw-hamburg.de/handle/20.500.12738/4227 (accessed on 21 July 2021).
27. Dong, L.; Wang, G. Support context-aware IoT content request in Information Centric networks. In Proceedings of the 2016 25th Wireless and Optical Communication Conference (WOCC), Chengdu, China, 21–23 May 2016; pp. 1–4.
28. Quevedo, J.; Corujo, D.; Aguiar, R. A case for ICN usage in IoT environments. In Proceedings of the 2014 IEEE Global Communications Conference, Austin, TX, USA, 8–12 December 2014; pp. 2770–2775.
29. Jacobson, V.; Smetters, D.K.; Thornton, J.D.; Plass, M.F.; Briggs, N.H.; Braynard, R.L. Networking named content. In Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies, Rome, Italy, 1–4 December 2009; pp. 1–12.
30. Laoutaris, N.; Che, H.; Stavrakakis, I. The LCD interconnection of LRU caches and its analysis. *Perform. Eval.* **2006**, *63*, 609–634. [CrossRef]
31. Psaras, I.; Chai, W.K.; Pavlou, G. Probabilistic in-network caching for information-centric networks. In Proceedings of the Second Edition of the ICN Workshop on Information-Centric Networking, Helsinki, Finland, 17 August 2012; pp. 55–60.
32. Cho, K.; Lee, M.; Park, K.; Kwon, T.T.; Choi, Y.; Pack, S. WAVE: Popularity-based and collaborative in-network caching for content-oriented networks. In Proceedings of the 2012 Proceedings IEEE INFOCOM Workshops, Orlando, FL, USA, 25–30 March 2012; pp. 316–321.
33. Li, J.; Wu, H.; Liu, B.; Lu, J.; Wang, Y.; Wang, X.; Zhang, Y.; Dong, L. Popularity-driven coordinated caching in named data networking. In Proceedings of the 2012 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), Austin, TX, USA, 29–30 October 2012; pp. 15–26.
34. Ren, J.; Qi, W.; Westphal, C.; Wang, J.; Lu, K.; Liu, S.; Wang, S. Magic: A distributed max-gain in-network caching strategy in information-centric networks. In Proceedings of the 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Toronto, ON, Canada, 27 April–2 May 2014; pp. 470–475.
35. Zhang, Z.; Lung, C.H.; St-Hilaire, M.; Lambadaris, I. Smart proactive caching: Empower the video delivery for autonomous vehicles in ICN-based networks. *IEEE Trans. Veh. Technol.* **2020**, *69*, 7955–7965. [CrossRef]
36. Khelifi, H.; Luo, S.; Nour, B.; Sellami, A.; Moungla, H.; Naït-Abdesselam, F. An optimized proactive caching scheme based on mobility prediction for vehicular networks. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018; pp. 1–6.
37. Sarkar, S.; Misra, S. Theoretical modelling of fog computing: a green computing paradigm to support IoT applications. *IET Netw.* **2016**, *5*, 23–29. [CrossRef]
38. Yannuzzi, M.; Milito, R.; Serral-Gracià, R.; Montero, D.; Nemirovsky, M. Key ingredients in an IoT recipe: Fog Computing, Cloud computing, and more Fog Computing. In Proceedings of the 2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Athens, Greece, 1–3 December 2014; pp. 325–329.
39. Lee, K.; Kim, D.; Ha, D.; Rajput, U.; Oh, H. On security and privacy issues of fog computing supported Internet of Things environment. In Proceedings of the 2015 6th International Conference on the Network of the Future (NOF), Montreal, QC, Canada, 30 September–2 October 2015; pp. 1–3.
40. Sarkar, S.; Chatterjee, S.; Misra, S. Assessment of the Suitability of Fog Computing in the Context of Internet of Things. *IEEE Trans. Cloud Comput.* **2015**, *6*, 46–59. [CrossRef]
41. Salman, O.; Elhajj, I.; Kayssi, A.; Chehab, A. Edge computing enabling the Internet of Things. In Proceedings of the 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), Milan, Italy, 14–16 December 2015; pp. 603–608.
42. Aazam, M.; Huh, E.N. Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT. In Proceedings of the 2015 IEEE 29th International Conference on Advanced Information Networking and Applications, Gwangju, Korea, 24–27 March 2015; pp. 687–694.
43. Abedin, S.F.; Alam, M.G.R.; Tran, N.H.; Hong, C.S. A Fog based system model for cooperative IoT node pairing using matching theory. In Proceedings of the 2015 17th Asia-Pacific Network Operations and Management Symposium (APNOMS), Busan, Korea, 19–21 August 2015; pp. 309–314.
44. Aazam, M.; Huh, E.N. Fog computing and smart gateway based communication for cloud of things. In Proceedings of the 2014 International Conference on Future Internet of Things and Cloud, Barcelona, Spain, 27–29 August 2014; pp. 464–470.

45. Markakis, E.; Negru, D.; Bruneau-Queyreix, J.; Pallis, E.; Mastorakis, G.; Mavromoustakis, C.X. A p2p home-box overlay for efficient content distribution. In *Emerging Innovations in Wireless Networks and Broadband Technologies*; IGI Global: Hershey, PA, USA, 2016; pp. 199–220.
46. Wei, Y.; Yu, F.R.; Song, M.; Han, Z. Joint optimization of caching, computing, and radio resources for fog-enabled IoT using natural actor–critic deep reinforcement learning. *IEEE Internet Things J.* **2018**, *6*, 2061–2073. [CrossRef]
47. Luo, S.; Chen, X.; Zhou, Z. F3C: Fog-enabled Joint Computation, Communication and Caching Resource Sharing for Energy-Efficient IoT Data Stream Processing. In Proceedings of the 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), Dallas, TX, USA, 7–10 July 2019; pp. 1019–1028.
48. Abkenar, F.S.; Khan, K.S.; Jamalipour, A. Smart Cluster-based Distributed Caching for Fog-IoT Networks. *IEEE Internet Things J.* **2020**, *8*, 3875–3884. [CrossRef]
49. Doan, K.N.; Van Nguyen, T.; Quek, T.Q.; Shin, H. Content-aware proactive caching for backhaul offloading in cellular network. *IEEE Trans. Wirel. Commun.* **2018**, *17*, 3128–3140. [CrossRef]
50. Thar, K.; Tran, N.H.; Oo, T.Z.; Hong, C.S. DeepMEC: Mobile edge caching using deep learning. *IEEE Access* **2018**, *6*, 78260–78275. [CrossRef]
51. Zhang, C.; Ren, P.; Du, Q. Learning-to-rank based strategy for caching in wireless small cell networks. In *International Conference on Internet of Things as a Service*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 111–119.
52. Baştuğ, E.; Bennis, M.; Debbah, M. Proactive caching in 5G small cell networks. In *Towards 5G: Applications, Requirements and Candidate Technologies*; Wiley: Hoboken, NJ, USA, 2016; pp. 78–98.
53. Bastug, E.; Bennis, M.; Debbah, M. Living on the edge: The role of proactive caching in 5G wireless networks. *IEEE Commun. Mag.* **2014**, *52*, 82–89. [CrossRef]
54. Chang, Z.; Lei, L.; Zhou, Z.; Mao, S.; Ristaniemi, T. Learn to cache: Machine learning for network edge caching in the big data era. *IEEE Wirel. Commun.* **2018**, *25*, 28–35. [CrossRef]
55. Wang, W.; Lan, R.; Gu, J.; Huang, A.; Shan, H.; Zhang, Z. Edge caching at base stations with device-to-device offloading. *IEEE Access* **2017**, *5*, 6399–6410. [CrossRef]
56. Xiang, H.; Yan, S.; Peng, M. A deep reinforcement learning based content caching and mode selection for slice instances in fog radio access networks. In Proceedings of the 2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall), Honolulu, HI, USA, 22–25 September 2019; pp. 1–5.
57. Jiang, F.; Yuan, Z.; Sun, C.; Wang, J. Deep Q-learning-based content caching with update strategy for fog radio access networks. *IEEE Access* **2019**, *7*, 97505–97514. [CrossRef]
58. Liu, Y.; Ma, Z.; Yan, Z.; Wang, Z.; Liu, X.; Ma, J. Privacy-preserving federated k-means for proactive caching in next generation cellular networks. *Inf. Sci.* **2020**, *521*, 14–31. [CrossRef]
59. De Amorim, R.C.; Hennig, C. Recovering the number of clusters in data sets with noise features using feature rescaling factors. *Inf. Sci.* **2015**, *324*, 126–145. [CrossRef]
60. Thinsungnoena, T.; Kaoungkub, N.; Durongdumronchaib, P.; Kerdprasopb, K.; Kerdprasopb, N. The clustering validity with silhouette and sum of squared errors. *Learning* **2015**, *3*. [CrossRef]
61. Ali, A.S.; Mahmoud, K.R.; Naguib, K.M. Optimal caching policy for wireless content delivery in D2D networks. *J. Netw. Comput. Appl.* **2020**, *150*, 102467. [CrossRef]