Learn to Cache: Machine Learning for Network Edge Caching in the Big Data Era

Zheng Chang, Lei Lei, Zhenyu Zhou, Shiwen Mao, and Tapani Ristaniemi

Abstract

The unprecedented growth of wireless data traffic not only challenges the design and evolution of the wireless network architecture, but also brings about profound opportunities to drive and improve future networks. Meanwhile, the evolution of communications and computing technologies can make the network edge, such as BSs or UEs, become intelligent and rich in terms of computing and communications capabilities, which intuitively enables big data analytics at the network edge. In this article, we propose to explore big data analytics to advance edge caching capability, which is considered as a promising approach to improve network efficiency and alleviate the high demand for the radio resource in future networks. The learning-based approaches for network edge caching are discussed, where a vast amount of data can be harnessed for content popularity estimation and proactive caching strategy design. An outlook of research directions, challenges, and opportunities is provided and discussed in depth. To validate the proposed solution, a case study and a performance evaluation are presented. Numerical studies show that several gains are achieved by employing learning-based schemes for edge caching.

Introduction

The proliferation of smartphones has substantially enriched the mobile user experience, leading to a vast amount of emerging multimedia services, including video streaming, Internet and mobile gaming, social network applications, and so on. Such dramatic changes of different types of contents result in an interesting phenomenon on data and content distribution, that is, the same popular contents may be requested many times at different time instants but at a similar location, which has motivated mobile operators to rethink the current network architecture, and seek more sophisticated and advanced techniques to bring contents closer to end users with low latency and in a cost-efficient way [1]. In this context, moving contents' proximity to the network edge and proactively caching popular contents, such as at base stations (BSs), namely infrastructure caching, or at user equipments (UEs), namely infrastructureless caching, are recognized as promising solutions for enabling data services with low latency and alleviating heavy traffic load at cellular backhaul [2]. For example, in a vehicular network, a roadside unit can cache popular content, such as traffic or weather information, and distribute it to the vehicles in proximity in infrastructure caching, while the vehicles can pre-cache and disseminate the content for other nearby vehicles in infrastructureless caching. In general, two closely related problems need to be addressed for edge caching: content placement and content delivery. The content placement problem is to determine what, where, and when to cache, and the content delivery problem is to find a way to deliver the content to end users. In wireless networks, content delivery can be realized via the access scheme, such as cellular downlink or device-to-device (D2D) communications. The content placement problem, however, heavily relies on the accuracy of the prediction of user requirement and content popularity, and caching strategy design, which draws great efforts in network edge caching research.

To accurately predict the demand for data content, users' demand profiles can be tracked, recorded, and built by leveraging the massive amount of available data. Moreover, the widely deployed online social networks have become an enabler for content sharing and distribution. As a matter of fact, users who have similar backgrounds and interests or close social relationships tend to rank the data content in a similar way [3]. Thus, the correlation of social and geographic data, as well as the history data of users, can be utilized for better prediction of user demand. However, the large-scale data also poses a major obstacle for efficiently utilizing an intelligent edge caching mechanism. Thanks to the recent advances in the computing and storage of BSs and UEs, big data analytics [4], for example, machine learning schemes, can be explored and implemented even to the network edge to analyze and extract the features of the collected data from end users and make more accurate caching decisions.

In fact, as most traditional approaches for addressing an unexpected growth of data traffic are becoming ineffective in terms of scalability and flexibility, big data analytics has been recognized as an innovative way to manage future wireless networks and cope with the challenges brought by the data explosion [4]. In today's networks, wireless data is generated on a very large scale from various sources, with different quality and trust levels. The induced 4V features (volume, velocity, variety, veracity) not only pose immediate challenges to conventional network management operations, but also bring profound opportunities for estab-

Digital Object Identifier: 10.1109/MWC.2018.1700317

Zheng Chang and Tapani Ristaniemi are with the University of Jyväskylä; Lei Lei is with the University of Luxembourg; Zhenyu Zhou (corresponding author) is with North China Electric Power University; Shiwen Mao is with Auburn University.

1536-1284/18/$25.00 © 2018 IEEE

lishing a smart and intelligent network. In the context of edge caching, as it is unlikely to provide an accurate caching prediction based on one dimension of data from one single end user, big data analytics schemes, in particular, machine learning mechanisms for large scale multi-dimensional data from various resources, are indispensable [5]. In this work, we examine the potentials and challenges of utilizing machine learning in network edge caching design.

In this article, we first overview the concepts and architectures of network edge caching and the need for sophisticated big data analytics approaches. Then we briefly introduce the big data analytics schemes proposed for wireless networks. Moreover, we leverage machine-learning-based approaches for enabling big-data-enabled edge caching, and provide detailed discussions on its potential for performance gain and future research directions and architectures to accommodate machine learning schemes in caching development. To validate the proposed solution, we consider two case studies and present the corresponding performance evaluation.

Network Edge Caching: Concepts and Challenges

Caching at the edge of the wireless network is a promising way to boost network throughput, improve energy efficiency (EE), decrease service latency, and reduce traffic load of the cellular backhaul. These improvements are rooted in the fact that popular contents are brought to the network edge to be reused by many UEs.

Infrastructure Caching

As shown in Fig. 1, it is expected that a data center/fog node with data storage can be deployed at the BS level, for example, at existing macro BSs (MBSs) and small BSs (SBSs). Compared to data caching or fetching in the core network or even at a higher level, edge caching at the BSs essentially alleviates backhaul congestion. Moreover, it is also possible to deploy new dedicated caching entities with cabled backhaul or dedicated wireless backhaul to enable a flexible and cost-effective method of content distribution. As caching more data can generally increase the cache-hit probability and alleviate the required backhaul capacity, but at the cost of distributed storage, the trade-off should be investigated. Moreover, as the BS usually has more powerful computing units, it is also able to provide an accurate prediction of the data demand.

Infrastructureless Caching

Today's smart devices, such as smartphones and tablets, usually have large storage capacities that are typically underutilized. Infrastructureless caching, that is, caching at the device level, can be implemented efficiently and effectively by utilizing these storage spaces. By device caching, the traffic load of the BS and core network can be further alleviated, and are made available for other operations [2]. As illustrated in Fig. 1, there are two kinds of device caching. One is that with known or estimated content popularity or user demand, popular contents can be pushed to the UEs via broadcast or unicast. Such a process can also be referred to as content push, self-caching, or pre-fetching (Fig. 1). In practice, due to the involved energy consumption, a user may not be willing to cache contents for others. Therefore, energy consumption investigation is important in this context and has been studied in [3, 6]. In addition, how to motivate users to cache contents is also of research interest [6].

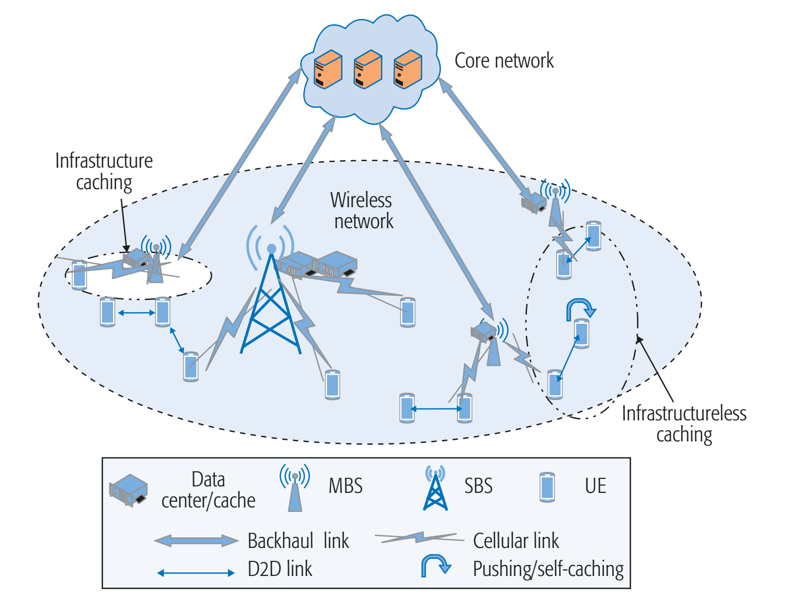

FIGURE 1. Caching at the network edge.

If the contents are not cached at a local device but in the UEs in proximity, a D2D communication link can be established to deliver the contents. In D2D caching, as shown in Fig. 1, a BS can utilize the available information of data popularity and user location, and cache popular contents at given UEs that are willing to share their storage with others. Recent studies have shown a profound performance gain in terms of throughput and EE achieved by D2D caching [6].

Related Works and Challenges

The investigation of caching strategies basically focuses on the core issues: when, how, and what to cache [1]. In addition, an edge caching mechanism needs to address another challenging issue: where to cache. Most existing research works focus on the content placement and content delivery problems in edge caching. For the infrastructure caching design, there are several key features to be explored, that is, content popularity, social relations [3, 7], user preference, cache size, as well as estimation or data uncertainty. For infrastructureless caching, the authors of [1] discuss the importance of exploring where, what, and when to cache and share data. The authors of [6] focus on extracting the inherent social relations of the devices to encourage D2D caching and pushing. Based on the above observation, we summarize the key features, challenges, and approaches of the edge caching schemes in Fig. 2.

In fact, content popularity, which is one of the key parameters for caching accuracy, is time-varying and usually unknown in advance. Moreover, how to utilize different features of the previously requested data to improve the tracking and esti-

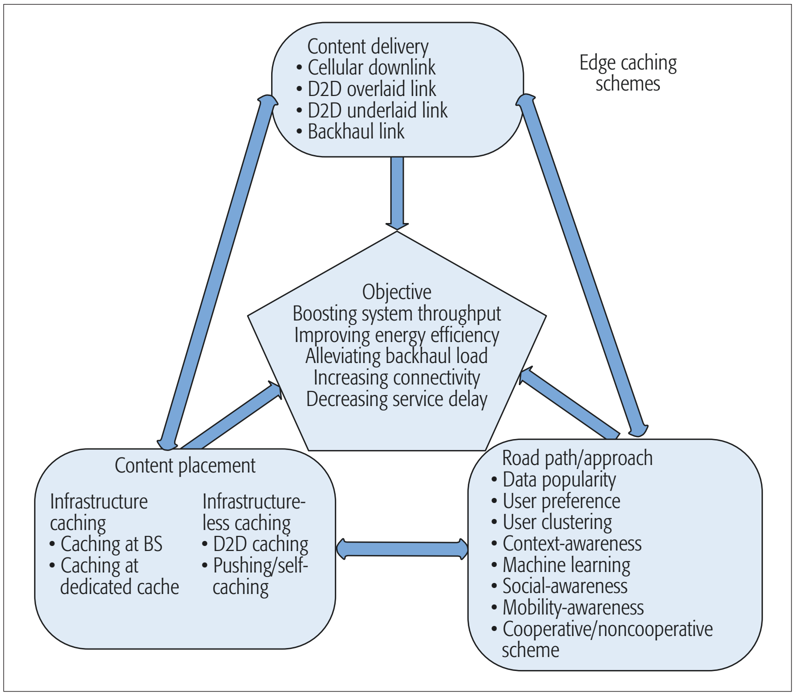

FIGURE 2. Summary of the edge caching schemes.

mation of timely content popularity is an important issue. Moreover, the locations of the UEs are also time-varying and the prediction of spatio-temporal users' behavior requires dedicated efforts. All such data uncertainties may pose great obstacles for next-step analysis, and also make the proposed schemes vulnerable to the upcoming data and device explosion. Therefore, employing machine learning technology in the design of caching policies has great potential to improve the system performance in the emerging big data era.

Learning for Caching at the Network Edge

In the following, we first briefly summarize the typical big data analytics schemes in the design of wireless networks as shown in Fig. 3 and discuss the machine learning scheme for edge caching development. It is worth noticing that some big data analytics schemes may have inherent connections. For example, some of the machine learning schemes can be classified as general data mining schemes or vice versa, and some statistical modeling schemes can be considered as broad data mining schemes.

Big Data Analytics for Future Networks

Stochastic and Statistical Modeling: Using probabilistic models, stochastic-modeling-based methods can capture the explicit features and dynamics of the data traffic and the deployment of network elements. Commonly used stochastic models include the K-Markov model (KMM), hidden Markov model (HMM), stochastic geometric model, time series, linear/nonlinear random dynamic systems, and so on, which have been applied to address the problems of energy harvesting analysis, data traffic control, prediction of BS sleeping and user association, and so on [4].

Statistical modeling is a simplified mathematical method to approximate reality and make predictions from the approximation. Statistical modeling is a popular tool for channel modeling, measurements, deployment and traffic analysis, multiple-input multiple-output (MIMO) systems, and so on [8].

Data Mining: Data mining focuses on extracting and exploiting the implicit structures in the datasets. Data-mining-based schemes have been widely applied to solving the security problems, such as intrusion and anomaly detection, and those of self-organizing networks (SONs), such as self-optimization, self-healing, and many others [5].

Distributed and Dynamic Optimization: Distributed optimization techniques, such as primal/dual decomposition and alternating direction method of multipliers (ADMM), are useful to decouple large-scale data transmission and analysis problems into several small subproblems for parallel computing so as to relieve both the computational burden at the fog node or in the cloud, and to alleviate bandwidth pressures at the fronthaul/backhaul links.

Machine Learning: The main objective of machine learning is to establish a functional relationship between input data and output actions in order to obtain an auto-processing capability for patterns of data inputs. Based on whether the data is labeled or not, machine learning can be generally categorized into two groups: supervised and unsupervised learning. In supervised learning, the goal is to establish a function from labeled training data (input and output data), while unsupervised learning is to infer a function to describe the hidden structure from unlabeled data.

In addition, based on how learning is performed, there are several other learning schemes, such as transfer learning, deep learning, and reinforcement learning. In the following, we break the traditional categorization of machine learning and introduce some learning schemes that have been or have the potential to be applied for edge caching.

Machine Learning Schemes for Edge Caching

Classification and Regression Analysis: Among the many useful techniques in supervised learning, classification and regression analysis are two common methods that have been applied to context identification of mobile usage and prediction of traffic levels (classification) and content demand. Regression analysis relies on a statistical process for estimating the relationships among the variables. The goal of regression analysis is to predict the value of one or more continuous-valued estimation(s). Although supervised learning may obtain relatively good caching decisions, some pre-knowledge is required to label the data, which in practice may not be possible when there is not sufficient information about the users in the network.

Clustering: In unsupervised learning, clustering is used to identify the different patterns in the datasets. It can be applied to edge caching design by clustering numbers of UEs into different groups based on their behavioral and data request history information [7]. Then the edge node can predict the data demand based on the interests or social relations of the entire group and cache the content that attracts the most UEs in the group. It can be found that proximity measure among groups should be carefully chosen for implementing clustering schemes and has a great impact on the algorithm performance.

Reinforcement Learning: Reinforcement learning (RL) focuses on how a machine or agent determines the proper actions automatically to optimize its performance. In RL, reward feedback is required for the machine to adapt its behavior from the environment. Basically, RL is about the decision making process instead of simply learning from the data. There are some attempts to apply RL in caching design [9, 10]. Typical examples are applying Q-learning to perceive the data request probability or popularity distribution, and the statistics of the random arrival of data or UEs by finding the Q-value. The multi-armed bandit learning scheme [10] has also been applied to edge caching design by properly designing the reward of caching.

Transfer Learning: Transfer learning (TL) focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. In other words, TL allows one to deal with some problems by leveraging the already existing data of some related tasks. Generally, by leveraging user-content correlations and the information from some other domains, such as social networks or location, the aim of utilizing transfer-learning-based schemes is to enhance the estimation of content popularity [11]. However, a TL-based scheme may face difficulty when source and target problems have few relations. Therefore, when the information from another domain is not as related to content demand, the TL-based scheme may not provide accurate decisions for caching.

Deep Learning Approach: Deep learning (DL) investigates a deep, multi-layered, and hierarchical architecture of data learning and distributed representation, where higher-level, more abstract features are defined by lower-level features [12]. Due to its hierarchical architecture, DL schemes enable automatic abstraction and feature extraction from the underlying data. As for edge caching, DL-based schemes are able to make accurate caching decisions in some cases. Among DL architectures, we have used a deep neural network (DNN) for the caching design and provide a DNN-based scheme for optimizing content delivery in edge caching. A DNN is an artificial neural network with several hidden layers between the input and output layers. Compared to the conventional iterative optimization methods, the DL-based approach can provide a good approximation to the optimal content delivery solution with significant complexity reduction. However, It is worth noticing that it may require a large amount of data for training.

Similarity Learning Approach: Generally, similarity learning has been applied to supervised learning. In similarity learning, the learning machine is given pairs of examples that are considered similar and pairs of less similar objects. It needs to learn a similarity function (or a distance metric function) that can predict whether new objects are similar. Similarity learning can be applied to edge caching by identifying the similarity among UEs with similar data demands and selecting the UEs who can act as edge caches. However, the system should have sufficient knowledge of the UEs in order to perform similarity learning for accurate decisions. We provide a case study and examine the effect of similarity learning below.

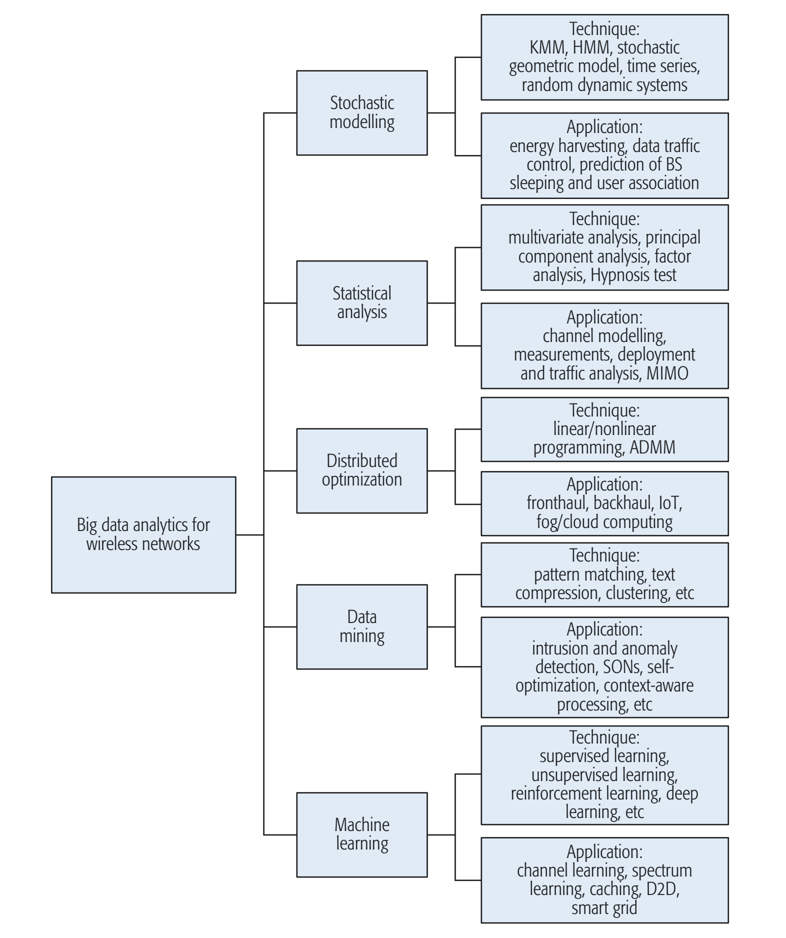
Big data analytics for wireless networks
Stochastic modelling
Technique: KMM, HMM, stochastic geometric model, time series, random dynamic systems
Application: energy harvesting, data traffic control, prediction of BS sleeping and user association
Statistical analysis
Technique: multivariate analysis, principal component analysis, factor analysis, Hypnosis test
Application: channel modelling, measurements, deployment and traffic analysis, MIMO
Distributed optimization
Technique: linear/nonlinear programming, ADMM
Application: fronthaul, backhaul, IoT, fog/cloud computing
Data mining
Technique: pattern matching, text compression, clustering, etc
Application: intrusion and anomaly detection, SONs, self-optimization, context-aware processing, etc
Machine learning
Technique: supervised learning, unsupervised learning, reinforcement learning, deep learning, etc
Application: channel learning, spectrum learning, caching, D2D, smart grid

FIGURE 3. Big data analytics for wireless networks: approaches and applications.

Developing Learning-Based Edge Caching: Challenges and Future Directions

The performance of the edge caching algorithm heavily depends on the knowledge of content popularity among a number of users, which is usually observed in a large area and over a very long period. However, the temporary content popularity in practice varies largely from time to time and is usually not in line with certain distribution. Therefore, knowing temporary content popularity is of importance to design efficient proactive caching algorithms. Investigating a machine-learning-based scheme may bring a new way of edge caching development. However, there are still many challenges ahead concerning the amount of data and computational resources, learning process, accuracy, efficiency, privacy, and security. In the following, we introduce the obstacles that may prevent leaning-based caching design and point out possible research directions.

Cold-Start UE and Data Sparsity

The cold-start problem is very prevalent in the machine-learning-based system. To implement the learning-based scheme for edge caching, data or information on the UEs within range is necessary.

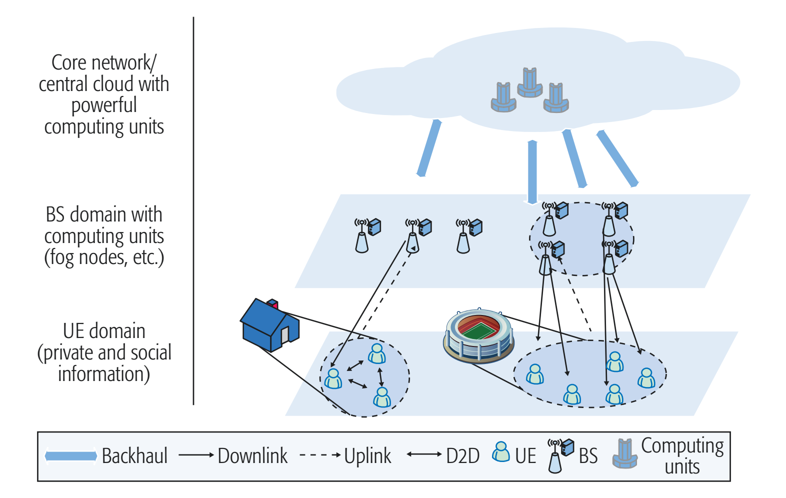
Core network/ central cloud with powerful computing units
BS domain with computing units (fog nodes, etc.)
UE domain (private and social information)
Backhaul
Downlink
Uplink
D2D
UE
BS
Computing units

FIGURE 4. Hierarchical edge caching structure.

However, this requirement sometimes cannot be satisfied due to the mobile and dynamic nature of wireless networks. When the UEs enter a new cell, the BS has no sufficient prior knowledge of the new UEs. Thus, it cannot obtain accurate estimation of the demand and cache the possible requested data by leveraging machine learning without sufficient data. There are some possible solutions for addressing the cold-start UE problem. One is to explore protocol design for collaborative edge caching. By designing protocol, the edge node can request information from the central cloud/core network or other edge nodes regarding the new UEs and execute the caching decisions. Moreover, the edge node may also be able to check the cache or its record of the UEs provided that the security and privacy can be guaranteed. In addition, it is of profound importance to investigate effective machine learning schemes to analyze the limited available data, and estimate the usefulness of the data, to overcome the data sparsity.

Integration of Machine Learning at the Edge: Efficiency and Accuracy

In edge caching, the machine-learning-based schemes face many new challenges concerning data processing and analysis. Both data sparsity and high density pose challenges for the learning and training process. Moreover, some learning-based schemes can be computationally prohibitively expensive. Therefore, it is worth investigating whether data preprocessing is needed to extract knowledge from the raw data before the machine learning process. In addition, limited computing resources also pose stringent constraints on operating the learning process in a sustainable way. The computing resources may be insufficient to process the high-dimensional data and make accurate caching decisions. Thus, how to tightly integrate machine learning at the network edge for great consolidation to improve the intelligent functionalities of the edge, from both the academic and industrial perspectives, is significant. Efficient learning schemes for massive high-dimensional data should be developed in order to provide accurate prediction of the cached data at the network edge. Radio resources, computational efficiency, and EE should be seriously considered when designing a machine learning scheme. In addition, investigation of the trade-off between the consumed resources, such as computing units, spectrum, and energy, and accuracy of prediction, is highly practical and vital for the caching strategy design at the network edge.

Security and Privacy

In order to provide an accurate caching strategy, a large volume of data should be collected and processed at the network edge or even at the central core network. A large amount of data that are collected for strategy design may be exposed to active attackers or passive eavesdroppers. Moreover, by applying machine learning schemes, the outcome and extracted information contain much sensitive and critical personal information, and any leakage can cause serious confidentiality, security, and privacy concerns. Utilizing the social network also poses security and privacy issues as the learned data is also from other UEs. For example, in vehicular networks, the route and destination of a vehicle may be predicted, and the map and transportation status can be pre-cached. However, if such information is not secure, the safety of the vehicle may face some problems. To secure the edge caching system, security and privacy-preserving schemes should be developed not only in the transmission/collection domain, but also in the data processing, access, and storage domains, and to both edge nodes and UEs. In particular, the sophisticated cryptographic protocol, authentication/access control, secure interface design, anomaly detection, and prevention mechanisms should be explored.

Hierarchical Collaborative Edge Caching Structure for a Learning-Based Scheme

As discussed, the inherent features create many obstacles toward efficiently utilizing machine learning approaches in edge caching. For example, a UE may not have authority to obtain others' data and information. Meanwhile, the computing capabilities of the UEs may not be sufficient to learn from the data. Thus, the UEs may not be able to make D2D caching decisions themselves. Moreover, due to the limited computing resources at the edge node and the mobility of the UEs, one single BS or other edge nodes may not execute the entire learning and caching process. Based on the above observations, a hierarchical edge caching structure as shown in Fig. 4 should be considered for edge caching.

To cope with the problems of caching at the device level, the caching decision should be made in the BS domain. The BS can utilize the data (at both the individual and social levels) from UEs, execute the learning process, and identify the UE that should act as the cache for other UEs and the content in which the other UEs are interested. For example, as shown in Fig. 4, when considering a home environment, the UE may be willing to cache the content and share it with other UEs. In addition, when the BS acts as the cache (e.g., in a stadium), it should be able to obtain the information or data from many UEs, and leverage the computing capabilities and UE information from other BSs or core networks to perform collaborative network caching via learning schemes. Within such a hierarchical structure, the backhaul between the BS and core networks, and radio resources among the BSs should be utilized to carry out (abstracted) data transmission. As such, the limitation caused by data sparsity and insufficiency computing resources may be overcome, along with limited radio resources, the overhead of signaling, and sophisticated algorithm design.

Case Study and Numerical Results

In this section, two case studies are conducted to evaluate the learning performance in infrastructure and infrastructureless caching. We use the datasets from a real-life cellular network, Alexanderplatz in the city of Berlin, provided by European project MOMENTUM (publicly available data: www.zib.de/momentum). The entire serving area is divided by thousands of pixels. Each pixel represents a small square area, where the signal strength between each pixel and BS is derived from real measurements. The BSs' locations, antenna heights, mechanical tilts, electrical tilts, and azimuths are pre-optimized.

Machine/Deep Learning for Infrastructure Caching

In the first case study, we integrate unsupervised learning, deep learning, and optimization algorithms to edge caching. A sub-area is extracted from the dataset consisting of seven cache-enabled BSs with orthogonal channel allocation. Each cell serves 100–200 randomly distributed UEs in its converge area/pixels. The objective is to optimize the energy consumption in data transmission, such that all the UEs' file/data requests can be satisfied in a timely manner. Conventional optimization algorithms may fail to support online decision making in real-time systems due to the high computational complexity in optimal content delivery. By introducing learning approaches, we aim at providing an efficient solution with competitive performance. The whole procedure can be organized in two phases. In the first phase, we use unsupervised learning (e.g., *K*-means clustering) to partition the UEs in each cell into 10–20 clusters based on their channel conditions and history information. In the second phase, based on the clustering result, we enumerate all the groups among the clusters, then selectively and sequentially schedule these cluster groups to transmit data to serve UEs. The optimal solution can be obtained by some iterative algorithms (e.g., see the linear programming formulations and the exact algorithms in [12–14]), but the process is time-consuming. We then train a DNN, and let it learn how the optimal decisions behave with input parameters (channel conditions and UEs' file requests). After training, the well-trained DNN helps us to establish a predicting system to tackle the most difficult and time-consuming part of optimization. The resilient back-propagation scheme is advocated as the learning heuristic for supervised learning in the DNN training stage. The DNN's output design is tailored. For example, the DNN returns a *K*-dimension binary vector. The *k*th element of the vector indicates whether the *k*th cluster should be scheduled alone or simultaneously transmitted with other clusters. Relying on these types of output information, one can significantly reduce the searching space in the optimization process, for example, excluding a large number of non-optimal groups. Thus, an overall efficient solution for content delivery can be expected.

CPU time (s) in computations					
Number of UEs per cell	LBS		Alg.1	Alg.2	Alg.3
100 (10 clusters)	0.046		0.16	0.15	0.11
150 (15 clusters)	0.052		0.97	0.73	0.48
200 (20 clusters)	0.085		226.7	142.9	82.4
DNN performance in LBS					
Training set size	500	1000	2000	3000	5000
Time (s) in DNN training	4.9	7.2	11.85	15.6	25.3
DNN predict accuracy	52%	61%	85%	89%	92%
LBS in approximating the optimum (100 UEs per cell)					
Training set size	500	1000	2000	3000	5000
Energy (optimum: 336.8J)	481.6	454.7	383.9	367.1	363.7
Gap to optimum	43%	35%	14%	9%	8%

TABLE 1. Performance of the proposed learning-based solution.

We compare the performance of the proposed learning-based solution ("LBS" in Table 1) with three content delivery algorithms, that is, simplex algorithm ("Alg.1") [12], column generation algorithm [13] ("Alg.2"), and a near-optimal algorithm [14] ("Alg.3"), where Alg.1 and Alg.2 are optimal, and Alg.3 is heuristic. First, we evaluate the average CPU time in computations (seconds per instance) in Table 1. The computing time is counted from the moment of giving a new input to a well-trained DNN or to the algorithms until obtaining the final (feasible) solution. From the results, the average computation time in LBS is much less than all the other algorithms, and is insensitive to the network scale. Second, we evaluate the DNN performance in terms of training time and prediction accuracy. In general, the training time linearly increases with the training set size. For training a mature DNN, the process can be completed in around dozens of seconds. When the training is sufficient (e.g., training by thousands of datasets), one can expect a high-quality prediction by the DNN. On average, over 90 percent of tested cases, the DNN's predicted results are consistent with the optimal results. Third, we show the LBS's capability in approximating the optimal energy. We use the 100-UE instances for illustration. Based on the accurate DNN predictions, LBS is able to progressively improve its energy saving performance in the training, around 8 percent gaps to the optimum (336.8 J). The near-optimal solution (Alg.3) has similar energy saving performance, around 5–13 percent gaps to the optimum, but with much more CPU time. Therefore, toward online optimization in edge caching, adopting learning approaches in content delivery is promising to achieve competitive performance and meanwhile enables less computation time.

Similarity Learning for D2D Caching

In this case, we consider applying similarity learning for D2D caching design and evaluate the user satisfaction based on the hit probability. It is assumed that there are several transmit UEs

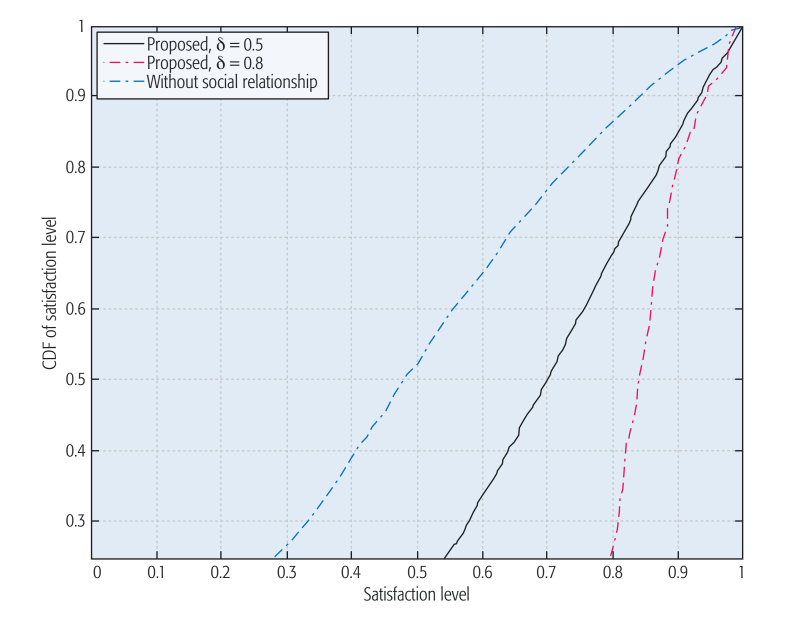

FIGURE 5. Satisfaction level of D2D caching.

(TX-UEs) who act as caches and several receive UEs (RX-UEs) who ask for data. The BS can utilize the similarity learning to find the social tie of two UEs, and we define the similarity of the UEs as their common interests over a large number of data items. A Kullback-Leibler (KL)-divergence-based metric is used for obtaining the similarity. After finding the social tie via learning, we also consider the physical relations, that is, the link quality of D2D communications. We incorporate a one-to-one matching scheme for pairing the TX and RX UEs, and the final pairing decision is made based on both social and physical relations, and the goal of finding the TX-RX pair so that the social throughput (defined as the combination of similarity ranking and data rate of the D2D link) can be maximized.

We examine the impact of social relations on user satisfaction. Figure 5 shows the cumulative distribution function (CDF) of the satisfaction for D2D RX-UEs, that is, the similarity of users' preferences on the data between the matched D2D pairs. To investigate the impact of the social relations on D2D RX-UEs' satisfaction, we compare our proposed socially aware matching algorithm with the one without consideration of social information. In addition, the threshold of social relationships $0 < \delta < 1$ (a lager δ indicates that a stronger social tie is needed for pairing) is also varied to see its impact. It is shown that when compared to the one without social relation consideration, the proposed scheme can obtain better satisfaction for users. It can also be seen that when δ decreases, the satisfaction performance also becomes worse. This is mainly due to the fact that for a higher threshold, it is more difficult for D2D TX-UEs and RX-UEs to form a pair, which in turn achieves a better satisfaction performance.

Conclusion

In this article, big data analytics techniques, particularly machine learning mechanisms, are proposed to advance edge caching capability. We review and categorize the current edge caching schemes and introduce big data analytics techniques. The major families of machine learning algorithms are examined in the context of their potential applications in edge caching. The challenges, along with a long-term view of research directions, and opportunities are provided and discussed in depth. A hierarchical collaborative edge caching structure for implementing learning schemes is also introduced. To validate the proposed solution, a case study and a performance evaluation are presented. Numerical studies show that several performance gains can be achieved.

Acknowledgment

This work is partially supported by the Academy of Finland (No. 284748), and National Science Foundation of China (NSFC) under grant No. 61601181, Fundamental Research Funds for the Central Universities under grant No. 2017MS13, Beijing Natural Science Foundation (4174104), Beijing Outstanding Young Talent under Grant No. 2016000020124G081. L. Lei's work has been supported by the Luxembourg National Research Fund (FNR) CORE project ROSETTA (11632107) and the European Research Council (ERC) project AGNOSTIC (742648). S. Mao's work is supported in part by the NSF under Grants DMS-1736470 and CNS-1702957, and by the Wireless Engineering Research and Education Center (WEREC) at Auburn University.

References

[1] X. Wang *et al.*, "Cache in the Air: Exploiting Content Caching and Delivery Techniques for 5G Systems," *IEEE Commun. Mag.*, vol. 52, no. 2, Feb. 2014, pp. 131–39.

[2] M. Zhang, H. Luo, and H. Zhang, "A Survey of Caching Mechanisms in Information-Centric Networking," *IEEE Commun. Surveys & Tutorials*, vol. 17, no. 3, 2015, pp. 1473–99.

[3] B. Bai *et al.*, "Caching Based Socially-Aware D2D Communications in Wireless Content Delivery Networks: A Hypergraph Framework," *IEEE Wireless Commun.*, vol. 23, no. 4, Aug. 2016, pp. 74–81.

[4] S. Bi *et al.*, "Wireless Communications in the Era of Big Data," *IEEE Commun. Mag.*, vol. 53, no. 10, Oct. 2015, pp. 190–99.

[5] E. Zeydan *et al.*, "Big Data Caching for Networking: Moving from Cloud to Edge," *IEEE Commun. Mag.*, vol. 54, no. 9, Sept. 2016, pp. 36–42.

[6] Z. Chang *et al.*, "Collaborative Mobile Clouds: An Energy Efficient Paradigm for Content Sharing," *IEEE Wireless Commun.*, 2017.

[7] M. S. ElBamby *et al.*, "Content-Aware User Clustering and Caching in Wireless Small Cell Networks," *Proc. 11th Int'l. Symp. Wireless Commun. Systems*, Barcelona, Spain, Aug. 2014.

[8] Z. Han, M. Hong, and D. Wang, *Signal Processing and Networking for Big Data Applications*, Cambridge Univ. Press.

[9] K. Guo, C. Yang, and T. Liu, "Caching in Base Station with Recommendation via Q-Learning," *2017 IEEE Wireless Commun. Networking Conf.*, San Francisco, CA, Mar. 2017.

[10] J. Song *et al.*, "Learning Based Content Caching and Sharing for Wireless Networks," *IEEE Trans. Commun.*, 2017.

[11] B. N. Bharath, K. G. Nagananda, and H. V. Poor, "A Learning-Based Approach to Caching in Heterogenous Small Cell Networks," *IEEE Trans. Commun.*, vol. 64, no. 4, Apr. 2016, pp. 1674–86.

[12] L. Lei *et al.*, "A Deep Learning Approach for Optimizing Content Delivering in Cache-Enabled HetNet," *Proc. IEEE Int'l. Symp. Wireless Commun. Systems*, Bologna, Italy, Aug. 2017.

[13] A. Khreishah, J. Chakareski, and A. Gharaibeh, "Joint Caching, Routing, and Channel Assignment for Collaborative Small-Cell Cellular Networks," *IEEE JSAC*, vol. 34, no. 8, Aug. 2016, pp. 2275–84.

[14] L. Lei *et al.*, "Optimal Cell Clustering and Activation for Energy Saving in Load-Coupled Wireless Networks," *IEEE Trans. Wireless Commun.*, vol. 14, no. 11, Nov. 2015, pp. 6150–63.

Additional Reading

[1] C. Jiang *et al.*, "Machine Learning Paradigms for Next-Generation Wireless Networks," *IEEE Wireless Commun.*, vol. 24, no. 2, Apr. 2017, pp. 98–105.

Biographies

Zheng Chang [SM] (zheng.chang@jyu.fi) received his B.Eng. degree from Jilin University, Changchun, China, in 2007, his M.Sc. (Tech.) degree from Helsinki University of Technology (now Aalto University), Espoo, Finland, in 2009, and his Ph.D degree from the University of Jyväskylä, Finland, in 2013. From June to August 2013, he was a visiting student at Tsinghua University, and from April to May 2015 he was a visiting researcher at the University of Houston, Texas. Currently he is working at the University of Jyväskylä, and his research interests include IoT, machine learning, and green communications.

Lei Lei [M] (lei.lei@uni.lun) received his Ph.D. degree in 2016 from Linköping University, Sweden. Since November 2016, he has been a research associate at the Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg. He was a research assistant at the Institute for Infocomm Research, A*STAR, Singapore, from June 2013 to December 2013. He received the IEEE Sweden Vehicular Technology-Communications-Information Theory (VT-COM-IT) Joint Chapter best student journal paper award in 2014. His current research interests include resource allocation and optimization in 5G-satellite networks, wireless caching, energy-efficient communications, and machine learning in wireless communications.

Zhenyu Zhou [SM] (zhenyu_zhou@ncepu.edu.cn) received his M.E. and Ph.D. degrees from Waseda University, Tokyo, Japan, in 2008 and 2011, respectively. Since March 2013, he has been an associate professor at School of Electrical and Electronic Engineering, North China Electric Power University. He received the Beijing Outstanding Young Talent award in 2016. He is an editor of *IEEE Access* and *IEEE Communications Magazine*. His research interests include green communications and smart grid.

Shiwen Mao [SM] (smao@ieee.org) received his Ph.D. in electrical and computer engineering from Polytechnic University, Brooklyn, New York, in 2004. He is the Samuel Ginn Distinguished Professor and director of the Wireless Engineering Research and Education Center (WEREC) at Auburn University, Alabama. His research interests include wireless networks, multimedia communications, and smart grid. He received the NSF CAREER Award in 2010. He was a co-recipient of the Best Demo Award from IEEE SECON 2017, the Best Paper Awards from IEEE GLOBECOM 2016 and 2015, IEEE WCNC 2015, and IEEE ICC 2013, and the recipient of the 2004 IEEE Communications Society Leonard G. Abraham Prize in the Field of Communications Systems.

Tapani Ristaniemi [SM] (tapani.ristaniemi@jyu.fi) received his M.Sc. in 1995 (mathematics), Ph.Lic. in 1997 (applied mathematics), and Ph.D. in 2000 (wireless communications), all from the University of Jyväskylä, Jyväskylä. Currently, he is a full professor at the University of Jyväskylä. He is also an adjunct professor at Tampere University of Technology. In 2013 he was a visiting professor at the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. He is also a co-founder of Magister Solutions Ltd. His research interests are in the areas of brain and communication signal processing and wireless communication systems research.