Learning-Based Resource Allocation: Efficient Content Delivery Enabled by Convolutional Neural Network

Lei Lei, Yaxiong Yuan, Thang X. Vu, Symeon Chatzinotas, and Björn Ottersten

Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, Luxembourg
Emails: {lei.lei; yaxiong.yuan; thang.vu; symeon.chatzinotas; bjorn.ottersten@uni.lu}

***Abstract*—In practical content delivery, when the time-frequency resources are limited, it is a challenging task to satisfy terminals' data demand in a heavy-traffic and mutual-interfered scenario. In this paper, we investigate time-efficient and energy-efficient solutions for content delivery at the network edge. We formulate two resource allocation problems, aiming at minimizing the total transmission time/energy in content delivery. The problems are formulated as mixed-integer linear programming. We obtain the global optimal solution by the branch-and-bound algorithm which typically incurs long computational time. To enable a computationally-efficient solution for fast and high-quality decision making, we resort to learning-based approaches to tackle the difficult combinatorial-optimization part. We investigate two deep-learning approaches, i.e., fully-connected deep neural network (FC-DNN) and convolutional neural network (CNN), to solve the problems. The FC-DNN and CNN are trained to learn and predict the discrete decisions. We compare the performance among FC-DNN, CNN, and the optimal solution. The numerical results illustrate that the proposed learning-based resource allocation approaches can achieve significant time-saving gains in computation and have promising performance in optimality approximation.**

***Index Terms*—Resource optimization, convolutional neural network, machine learning, content delivery networks.**

I. Introduction

In a content delivery network, the system can usually be fully-loaded, e.g., in presence of severe interference among a large number of terminals with excessive data traffic requested [1], [2]. In this scenario, the use of limited available resources to efficiently serve the terminals at the cell edge is challenging. As one of the solutions, popular contents can be cached at the edge, such that most of the edge-terminals can be served directly from the local cache. Otherwise, more resources, e.g., energy, time, are consumed if the terminals have to get the service from the macro base station (MBS) remotely [3].

In the literature, extensive studies have been devoted to develop advance algorithmic solutions to enable efficient content-delivery schemes, e.g., aiming at reducing transmission delay [4], network energy consumption [5], and transmit power [2]. As a matter of fact, most of the solutions are offline, that is, the algorithms need long time to output the optimized results with satisfactory performance. This considerable computational delay impairs the algorithms' applicability for practical systems. The issue is that if the adopted algorithm in a real-time environment cannot provide the optimization results timely, when the new inputs or requests arrive, the system has to wait, which is undesirable for network resource management. As an emerging research area, integrating machine learning to resource optimization has received considerable research attention [6]–[9], [11]. In [7], [11], the authors investigated machine-learning based approaches, e.g., training a full-connected deep neural network (FC-DNN) or a logistic-regression model, to address the resource scheduling problems in caching, multi-antenna, and non-orthogonal multiple access systems, respectively. As shown in [8], considerable efforts have been devoted to apply reinforcement learning to resource management in complex wireless networks. Recently, the authors in [9] used convolutional neural network (CNN) as a heuristic method to provide a fast solution for transmit power control.

In this paper, content caching at the edge is firstly performed. We then focus on two content-delivery problems, minimum-time and minimum-energy, and solve the two problems under a unified learning-based framework. The optimization decisions to be made in both problems are the same, i.e., determining the best strategy for grouping mobile terminals (MTs) and allocating the time resources among the selected groups. The problems are mixed-integer linear programming (MILP). Conventionally, the global optimum can be obtained by the branch-and-bound (B&B) algorithm but it is typically time consuming. To significantly reduce the computational time, we develop a learning-based resource allocation approach. Firstly by analysis, we identify the features to be learned in resource allocation. Secondly, two deep-learning models, CNN and FC-DNN, are trained to learn the mapping between the channel gains and the optimal discrete decisions.

Unlike [9], the outputs from the CNN or FC-DNN cannot directly provide a complete and feasible solution for solving the formulated MILP. In addition, from the literature, how to use machine learning to address constrained combinatorial optimization problems is challenging and studied in a limited extent [10]. We then rely on the deep-learning model to tackle the combinatorial part in optimization which is most difficult and computational-heavy in resource allocation. We combine the predictions from CNN or FC-DNN with the optimal B&B algorithm to enable a near-optimal, feasible, and

978-1-5386-6528-2/19/$31.00

fast solution. The numerical results show that for large-scale instances, the designed CNN achieves better performance than FC-DNN, in terms of prediction accuracy and computational time. Compared to the conventional iterative algorithm B&B, both the learning-based approaches demonstrate satisfactory performance in computational efficiency and optimality approximation.

II. System Model

A. Cellular Systems and Edge Caching

We consider downlink transmission in a multiple-input single-output (MISO) cellular system, where a cache-enabled small base station (SBS) equipped with L antennas is deployed at the cell edge serving up to U single-antenna MTs. Due to the limited storage capacity, the SBS follows standard caching policy [1], [4], and proactively cache part of the most popular files at the edge. The SBS can directly transmit a requested file to its associated MT if the file is currently available in the cache. Otherwise, the MT has to request the file from the core network via the MBS remotely or from other nearby cache-enabled devices at the edge. According to the cached files at the SBS and the MTs' requested files, the set of MTs served by the SBS and the MBS can be divided as $\mathcal{U}$ and $\bar{\mathcal{U}}$, respectively.

In data transmission, the time domain is slotted. We suppose that there are T time slots, i.e., a transmission frame, available in the system. All the content delivery tasks should be finished within T time slots in order to avoid extra transmission delay at the MTs side. To reduce the signaling overhead, the channel state information is collected once at each frame. We assume the SBS occupies a dedicated channel with bandwidth B Hz, which is orthogonal to the channel allocated to the MBS. All the transmission links SBS-to-MTs share the same channel but are sequentially scheduled in different time slots. We consider quasi-static block fading channels, such that the channel fading coefficients are fixed during a transmission frame. We remark that the SBS are allocated by a dedicated channel in this work, then the content delivery as well as the optimization task for serving the MTs in $\mathcal{U}$ and $\bar{\mathcal{U}}$ can be carried out independently. We assume that the majority of the content delivery tasks are from the SBS, thus we focus on the transmission in SBS-to-MTs. The scheduling for the MTs in $\bar{\mathcal{U}}$ follows analogously.

B. Transmission Model

At each time slot, one or multiple MTs can be scheduled simultaneously. Let g denote a group, and $\mathcal{U}_g$ denote the MTs included in group g, where $|\mathcal{U}_g| \leq L$ on each time slot. In total, we consider $G = C_U^1 + \ldots, +C_U^L$ possible candidate groups by enumeration. For example, when $U = 3$ and $L = 2$, all the candidate groups are $\{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}$, and the group $\{1,2,3\}$ is excluded. The channel vector of MT u is denoted by $\mathbf{h}_u \in \mathbb{C}^{L\times 1}$. We assume $\mathbf{h}_u$ follows circular-symmetric complex Gaussian distribution $\mathbf{h}_u \sim \mathcal{CN}(\mathbf{0}, \sigma_{h_u}^2 \mathbf{I}_L)$, where $\sigma_{h_u}^2$ is the parameter of path-loss between the SBS and MT u. The SBS performs precoding before transmitting data to MTs. Denote x_u^g as the modulated signal and $\mathbf{w}_u^g \in \mathbb{C}^{L\times 1}$ as the precoding vector for MT u in group g. The received signal for MT $u \in \mathcal{U}_g$ can be expressed as

$$y_u^g = \mathbf{h}_u^H \mathbf{w}_u^g x_u^g + \sum_{i\in\mathcal{U}_g\setminus\{u\}} \mathbf{h}_u^H \mathbf{w}_i^g x_i^g + n_u, \tag{1}$$

where the first and second terms in (1) represent the desired signal and the inter-MT interference respectively. n_u is Gaussian noise with zero mean and variance σ^2. The signal-to-interference-plus-noise ratio (SINR) for MT $u \in \mathcal{U}_g$ is given by

$$\mathrm{SINR}_{ug} = \frac{|\mathbf{h}_u^H \mathbf{w}_u^g|^2}{\sum\limits_{i\in\mathcal{U}_g\setminus\{u\}} |\mathbf{h}_u^H \mathbf{w}_i^g|^2 + \sigma^2}. \tag{2}$$

The achievable data rate can be expressed as

$$R_{ug} = B\log_2\left(1 + \mathrm{SINR}_{ug}\right), u \in \mathcal{U}_g. \tag{3}$$

We consider minimum mean square error (MMSE) precoding for each group. We collect all the channels vectors $\mathbf{h}_u$ for the MTs in group g, and form a $|\mathcal{U}_g| \times L$ matrix $\mathbf{H}_g$. Under MMSE, the beamformer vector for MT u in group g is of the form $\mathbf{w}_u^g = \sqrt{p_u}\hat{\mathbf{h}}_u$, where p_u is the transmit power for MT u and $\hat{\mathbf{h}}_u$ is the column corresponding to MT u in $\mathbf{H}_g^H(\sigma^2\mathbf{I} + \mathbf{H}_g\mathbf{H}_g^H)^{-1}$. In this work, a suboptimal algorithm, iterative water-filling [12], is adopted to obtain the power $p_1, \ldots, p_U$ among MTs. Denote $\alpha_{u,i}^g = |\mathbf{h}_u^H \hat{\mathbf{h}}_i|^2$, $\forall u, i \in \mathcal{U}_g$, by the interference factor caused to MT u from the MT i's beamforming vector. The total power consumption in group g is $p_g = \sum_{u\in\mathcal{U}_g} \alpha_{u,u}^g p_u$, which is predefined for each group.

III. Problem Formulation

In this section, we formulate two resource allocation problems, aiming at efficiently delivering all the required data in a resource-constrained scenario. The optimization task amounts to determining which MTs should be scheduled on a time slot, and the time-slot allocation in content delivery. We introduce integer variables $x_g \in \{0, 1, \ldots, T\}$ to indicate the number of used time slots for group g. We formulate two content delivery problems with different objectives. To save time resources for the system, a time-efficient content delivery problem is formulated in P1.

$$\text{P1: } \min_{x_g\in\mathbb{Z}} \sum_{g=1}^{G} x_g \tag{4a}$$

$$\text{s.t.} \sum_{g=1}^{G} x_g R_{ug} \geq Q_u, \ \forall u = \{1, \ldots, U\} \tag{4b}$$

$$x_g \in \{0, 1, \ldots, T\}, \ \forall g \in \{1, \ldots, G\} \tag{4c}$$

P1 is to minimize the number of used time slots, such that each MT's data traffic Q_u can be delivered in a timely manner. Next, following the same structure, we consider an energy-efficient delivery problem in P2 to minimize the total energy

consumption with limited time resources, such that all the content delivery tasks can be completed within T.

$$\text{P2:} \min_{x_g \in \mathbb{Z}} \sum_{g=1}^{G} x_g p_g \tag{5a}$$

$$\text{s.t.} \sum_{g=1}^{G} x_g B \log_2(1+\text{SINR}_{ug}) \geq Q_u, \ \forall u = \{1,\ldots,U\} \tag{5b}$$

$$\sum_{g=1}^{G} x_g \leq T \tag{5c}$$

$$x_g \in \{0,1,\ldots,T\}, \ \forall g \in \{1,\ldots,G\} \tag{5d}$$

Both P1 and P2 are mixed-integer linear programming problems. Their NP-hardness has been discussed in [5], [11]. Conventionally, the B&B algorithm is a straightforward way to obtain the global optimum, where a linear relaxation problem is solved at each node of the branch-and-bound tree. Since the computational complexity of the optimal algorithm increases exponentially with the input size, the required computational time also dramatically increases. For dealing with this issue in real-time applications, next, we propose a learning-based approach to provide a fast, feasible, and near-optimal solution for P1 and P2.

IV. Proposed Learning-Based Approaches

A. Features to be Learned

The optimization decisions in P1 and P2 consist of two parts. One is to select the best MT groups from an exponential number of candidates. The other is to determine how many time slots to be allocated to these selected groups. We remark that the major difficulties are from the first part which is computationally heavy. Once the scheduled groups have been decided, the remaining problem is relatively easy to solve. By analyzing the optimal decisions, we observe that there exists a pattern between the spatial features of channel coefficients and the decisions of optimal groups. For example, two MTs located distantly with weak mutual interference, or with significant difference in channel coefficients, are more likely to be grouped together in the same time slot at the optimum. Thus, we treat the grouping information, i.e., the most promising groups with high probability to be scheduled, as the feature to be learned and predicted by the deep-learning models.

In training-data generation, we obtain the optimal groups by applying the optimal B&B algorithm. we organize the optimal grouping information in a binary vector with number of U elements, i.e., $\boldsymbol{v} = [v_1,\ldots,v_i,\ldots,v_U]$, where "1" in the i-th element stands for that at least one i-cardinality group (the groups containing exact i MTs) is scheduled, otherwise "0". For example, if $U = 3$ and the optimal scheduled groups are {1,3}, {2}, then the vector reads $\boldsymbol{v} = [1,1,0]$. In the output layer of FC-DNN or CNN, the predicted information is organized as same as the U-dimension vector $\boldsymbol{v}$.

B. CNN-Based Structure

To establish a predicting system to produce $\boldsymbol{v}$, in this paper we adopt CNN to learn the relations between channel coefficients and the optimal groups. CNN has been widely used to extract spatial features for image classification [13]. We use CNN to exploit the spatial features from channel coefficients. FC-DNN can be effective to capture the nonlinear input-output relations. However, with the increased network scale, the computational efficiency of FC-DNN might decrease. Moreover, parameter explosion due to its fully-connected structure may result in over-fitting issues [14]. Unlike FC-DNN, in CNN, only part of nodes are connected between two adjacent convolution layers. As a result, CNN is able to use fewer parameters to extract relevant features at a low computational cost.

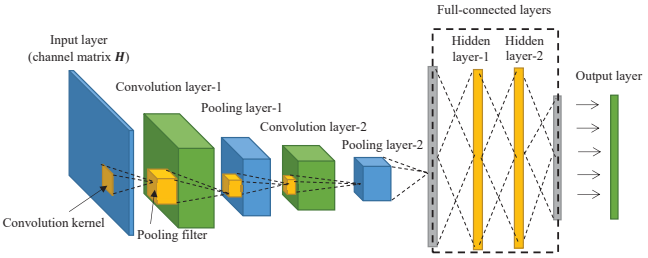


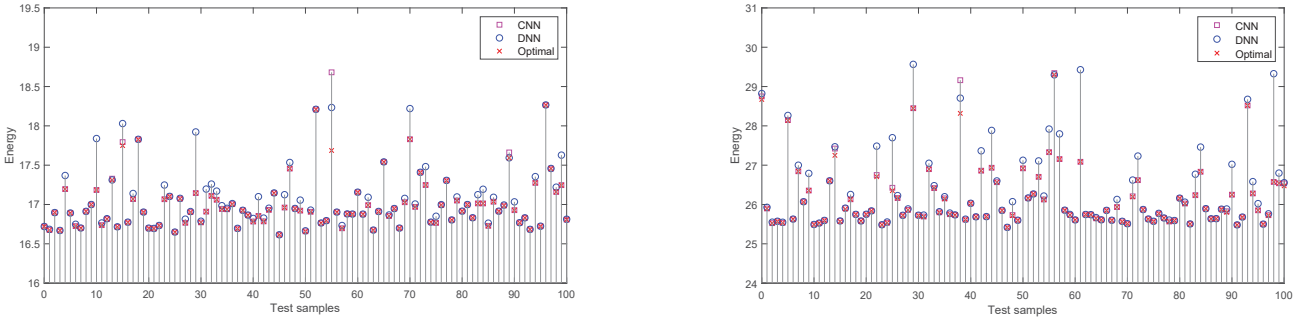
Figure 1. The designed CNN's structure

Fig. 1 illustrates the structure of the adopted CNN which consists of the following five main components.

- *Input layer*. In CNN, the input data is reorganized as an image-like 3-dimension matrix. The first two dimensions represent the image's length and width while the last dimension refers to the depth. For P1 and P2, the input data refers to the channel matrix $\boldsymbol{H}$ below. To facilitate training process, the channel coefficients can be further normalized and converted to dB [9].

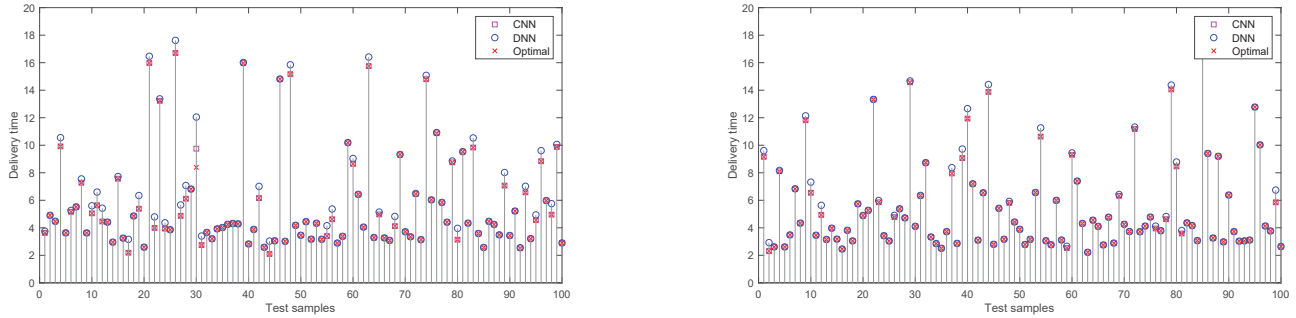
$$\boldsymbol{H} = \begin{bmatrix} h_{1,1} & \cdots & h_{1,U} \\ \vdots & \ddots & \vdots \\ h_{L,1} & \cdots & h_{L,U} \end{bmatrix}. \tag{6}$$

  As the depth of the matrix $\boldsymbol{H}$ is 1, the size of the input is $L \times U \times 1$.
- *Convolution layer*. As shown in Fig. 1, each neuron in the convolution layer only connects a squared part of the previous layer. This squared core is called filter or convolution kernel. By our design, we use two convolution layers with a 3×3 and a 2×2 convolution kernel respectively. The processing depth is set to 3. It means that the kernel enables to transfer the node matrix into a 3-tier unit node by convolution. Then the kernel will move around to cover all the image with a fixed step. The convolution layers try to analyze the data of each kernel for obtaining the features with a higher level of abstraction. More than that, since the weights are shared via the convolution kernel, the number of parameters can be significantly reduced in the neural network.



(a) U=10 (b) U=15

Figure 2. Energy consumption in P2: Comparisons among CNN, FC-DNN, and the optimum



(a) U=10 (b) U=15

Figure 3. Delivery time in P1: Comparisons among CNN, FC-DNN, and the optimum

- *Pooling layer*. Pooling layer is used to further decrease the size of the output matrix from the previous convolution layer. Similar to the convolution layer, pooling layer also adopts a filter to convert a node matrix as a unit node. The pooling filter applies the maximizing or averaging operations instead of convoluting operations.
- *Fully-connected layer*. The convoluting and pooling can be regarded as a process of automatic feature extraction. After that, full-connected (FC) layers are also needed to generate the final output. The structure of FC layer is identical with DNN.
- *Output layer*. By design, the output carries the estimated information for the optimal grouping decisions, which is referred to as binary vector $\boldsymbol{v}$.

Once the predicted vector $\boldsymbol{v}$ is obtained from the CNN's output layer, we round the fractional values to binary. Then the predicted groups can be derived by reading the "1" elements from the rounded $\boldsymbol{v}$. We use set $\mathcal{G}'$ to denote the union of the predicted groups from CNN, where $|\mathcal{G}'| \leq U$. Replacing the original groups $\{1, \ldots, G\}$ by restricted set $\mathcal{G}'$ in P1 and P2, we can efficiently solve the restricted problems to enable a feasible solution since the number of variables is now reduced from an exponential number G to a small number which is computationally light in general.

Table I
SYSTEM SETTINGS

Parameter	Value
Number of edge MTs	5 – 15
Cell radius	300 m
Power allocation	Iterative water-filling [12]
Dimension in input layer	$L \times U \times 1$
Convolution layer-1 kernel	3×3
Convolution layer-2 kernel	2×2
Convolution layer step	1
Convolution layer depth	3
Pooling layer filter	2×2
Pooling layer step	2
Pooling method	Max pooling [6]
Nodes in hidden layer-1	200
Nodes in hidden layer-2	200
Nodes in output layer	U
Active function	ReLU [6]
Optimizer	Adam optimization [14]
Training set size	5000
Test set size	100
Optimization Solver	Python, Gurobi 8.0
DNN Implementation	Python, TensorFlow

V. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the CNN and FC-DNN based approaches for P1 and P2. Following the same structure in Fig. 1, we extract an FC-DNN consisting of an input layer, two hidden layers, and an output layer. By comparing with the optimal B&B algorithm, we show the performance of the DNN- and CNN-based methods in terms of prediction accuracy and computation time. The simulation

parameters are summarized in Table I.

A. Comparison in Prediction Accuracy

Fig. 2(a) and 2(b) demonstrate the energy consumption by performing the CNN-based approach, the DNN-based approach and the optimal algorithm for the cases of U =10 and 15 in P2, respectively. As shown in the results, 100 test sets are used for evaluation. The average accuracy of the predictions in CNN and FC-DNN is evaluated as follows,

$$\frac{1}{100}\sum_{i=1}^{100}\frac{V_{dl}^{i}-V_{opt}^{i}}{V_{opt}^{i}}, \tag{7}$$

where V_{dl} and V_{opt} are the derived objective values, i.e., energy in P2 and time in P1, from the deep-leaning approaches, i.e., FC-DNN and CNN, and the optimal algorithm, respectively. From Fig. 2(a) and 2(b), the consumed energy in all the cases increases with more MTs in the system. In average, the prediction accuracy of the CNN-based approach is 96.90% and 94.47%, while those of FC-DNN are 92.13%, 89.20%, for U =10 and 15, respectively. Next, we evaluate the performance for solving P1. The comparison for the content-delivery time among FC-DNN, CNN, and optimal B&B, are shown in Fig. 3(a), and 3(b), for the cases of U=10, and 15, respectively. The prediction accuracy of the CNN-based approach in P1 (analogous to the metric in Eq. (7)), achieves 97.27%, and 96.08% in the cases of U=10, and 15, respectively, whereas the performance in FC-DNN slightly drops to 95.16%, and 93.78%.

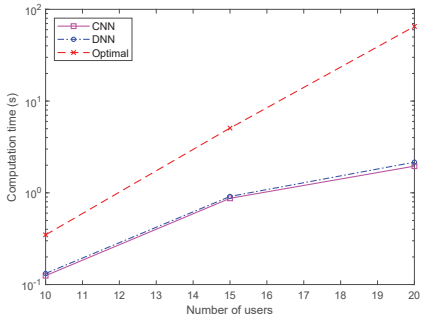


Figure 4. Computational time with respect to number of MTs

B. Comparison in Computational Time

The average computational time of optimal method exponentially increases in terms of the number of MTs. For CNN and FC-DNN based methods, the computational time consists of two parts, i.e., the time for CNN/DNN testing and the post-processing time. Specifically, the testing phase starts from giving a test set to the well-trained CNN/DNN, until obtaining the predicted vectors $\boldsymbol{v}$. The post-processing time counts for resolving the small-scale optimization problem by using restricted set $\mathcal{G}'$. We observe from Fig. 4 that the CNN and FC-DNN based approaches present much higher computational-efficiency than the optimal method which increases exponentially in computational time.

VI. Conclusions

We developed a CNN based approach for resource allocation to enable a fast, feasible, and near-optimal solution for both time-efficient and energy-efficient content delivery. We formulated two resource allocation problems for minimizing delivery time and energy consumption in serving MTs' requests. We adopted an optimal algorithm B&B as the performance benchmark. We designed FC-DNN and CNN based approaches to approximate the combinatorial decisions. Numerical results demonstrate the promising performance of the developed learning-based resource allocation, in terms of prediction accuracy and computational time.

VII. Acknowledgments

The work has been supported by the European Research Council (ERC) project AGNOSTIC (742648), by the FNR CORE projects ROSETTA (11632107) and ProCAST (C17/IS/11691338), and by the FNR bilateral project LARGOS (12173206).

References

[1] E. Bastug, M. Bennis, and M. Debbah, "Living on the edge: The role of proactive caching in 5G wireless networks," in *IEEE Communications Magazine*, vol. 52, no. 8, pp. 82–89, Aug. 2014.

[2] L. You, L. Lei , and D. Yuan, "Range assignment for power optimization in load-coupled heterogeneous networks," in Proc. *IEEE ICCS*, 2014.

[3] Z. Chang, L. Lei, Z. Zhou, S. Mao and T. Ristaniemi, "Learn to cache: Machine learning for network edge caching in the big data era," in *IEEE Wireless Communications*, vol. 25, no. 3, pp. 28–35, June 2018.

[4] T. X. Vu, L. Lei, S. Vuppala, A. Kalantari, S. Chatzinotas, and B. Ottersten, "Latency minimization for content delivery networks with wireless edge caching", in Proc. *IEEE ICC*, May 2018.

[5] L. Lei, D. Yuan, C. K. Ho, and S. Sun, "Optimal cell clustering and activation for energy saving in load-coupled wireless networks," in *IEEE Transactions on Wireless Communications*, vol. 14, no. 11, pp. 6150–6163, Nov. 2015.

[6] A. Zappone, M. Renzo, M. Debbah, "Wireless networks design in the era of deep learning: model-based, AI-based, or both?", Online Available: arXiv:1902.02647

[7] L. Lei, L. You, G. Dai, T. X. Vu, D. Yuan, and S. Chatzinotas, "A deep learning approach for optimizing content delivering in cache-enabled HetNet," in Proc. *IEEE ISWCS*, Aug. 2017.

[8] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y. Liang, D. I. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," Online Available: arXiv:1810.07862

[9] W. Lee, M. Kim and D. Cho, "Deep power control: Transmit power control scheme based on convolutional neural network," in *IEEE Communications Letters*, vol. 22, no. 6, pp. 1276-1279, June 2018.

[10] Y. Sun, M. Peng, Y. Zhou, Y. Huang, S. Mao, "Application of Machine Learning in Wireless Networks: Key Techniques and Open Issues," Online available: arXiv:1809.08707.

[11] L. Lei, L. You, Q. He, T. X. Vu, S. Chatzinotas, D. Yuan, and B. Ottersten, "Learning-assisted optimization for energy-efficient scheduling in deadline-aware NOMA systems," in *IEEE Transactions on Green Communications and Networking*, accepted, 2019.

[12] W. Yu, G. Ginis, and J. M. Cioffi, "Distributed multiuser power control for digital subscriber lines," in *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 5, pp. 1105–1115, 2002.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Proc. *NIPS*, 2012, pp. 10971105.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.