Optimal UAV Caching and Trajectory in Aerial-Assisted Vehicular Networks: A Learning-Based Approach

Huaqing Wu, *Graduate Student Member, IEEE*, Feng Lyu, *Member, IEEE*, Conghao Zhou, *Graduate Student Member, IEEE*, Jiayin Chen, Li Wang, *Senior Member, IEEE*, and Xuemin Shen, *Fellow, IEEE*

***Abstract*—In this article, we investigate the UAV-aided edge caching to assist terrestrial vehicular networks in delivering high-bandwidth content files. Aiming at maximizing the overall network throughput, we formulate a joint caching and trajectory optimization (JCTO) problem to make decisions on content placement, content delivery, and UAV trajectory simultaneously. As the decisions interact with each other and the UAV energy is limited, the formulated JCTO problem is intractable directly and timely. To this end, we propose a deep supervised learning scheme to enable intelligent edge for real-time decision-making in the highly dynamic vehicular networks. In specific, we first propose a clustering-based two-layered (CBTL) algorithm to solve the JCTO problem offline. With a given content placement strategy, we devise a time-based graph decomposition method to jointly optimize the content delivery and trajectory design, with which we then leverage the particle swarm optimization (PSO) algorithm to further optimize the content placement. We then design a deep supervised learning architecture of the convolutional neural network (CNN) to make fast decisions online. The network density and content request distribution with spatio-temporal dimensions are labeled as channeled images and input to the CNN-based model, and the results achieved by the CBTL algorithm are labeled as model outputs. With the CNN-based model, a function which maps the input network information to the output decision can be intelligently learnt to make timely inference and facilitate online decisions. We conduct extensive trace-driven experiments, and our results demonstrate both the efficiency of CBTL in solving the JCTO problem and the superior learning performance with the CNN-based model.**

***Index Terms*—Edge intelligence, UAV caching, trajectory design, deep supervised learning, graph decomposition.**

Manuscript received December 4, 2019; revised March 13, 2020; accepted April 28, 2020. Date of publication June 29, 2020; date of current version November 20, 2020. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 91638204 and Grant 61871416, in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada, in part by the Fundamental Research Funds for the Central Universities under Grant 2018XKJC03, and in part by the Beijing Municipal Natural Science Foundation under Grant L192030. This article was presented at the 2019 IEEE Wireless Communications and Signal Processing (WCSP). *(Corresponding author: Feng Lyu.)*

Huaqing Wu, Conghao Zhou, Jiayin Chen, and Xuemin Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: h272wu@uwaterloo.ca; c89zhou@uwaterloo.ca; j648chen@uwaterloo.ca; sshen@uwaterloo.ca).

Feng Lyu is with the School of Computer Science and Engineering, Central South University, Changsha 410083, China (e-mail: fenglyu@csu.edu.cn).

Li Wang is with the Beijing Key Laboratory of Work Safety Intelligent Monitoring, School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: liwang@bupt.edu.cn).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSAC.2020.3005469

I. INTRODUCTION

TO ACCOMMODATE ever-increasing vehicular traffic demands especially in the future driverless era, the Internet of Vehicles (IoV) is envisioned to be vigorously robust and deliver high-bandwidth contents limitlessly [2]–[4]. As the vehicular network resource is highly dynamic while the terrestrial network resources (4G/5G) are fixed and rigid, it is a challenging task to guarantee satisfactory network performance anywhere at any time, such as at urban busy roads during rush hours. Unmanned aerial vehicle (UAV) caching [5] is a promising paradigm to assist the terrestrial network. By proactively caching popular and repetitively requested content files with large size (e.g., HD map or the video streaming of football match, concert, etc.), UAV caching can significantly alleviate the traffic burden of the terrestrial network. Particularly, the caching-enabled UAVs are physically free from the backhaul limitation, which makes the implementation of UAV communications more feasible considering the high agility of UAVs. With fully controllable mobility and high altitude, UAVs can support fast reconfiguration with high line-of-sight (LoS) probability for UAV-to-vehicle (U2V) wireless links [6]. When content requests are hit by the caching, the files can be delivered directly without traversing wireless backhaul, reducing the response delay significantly [7]. Besides, on-demand UAV communications can be dispatched when the terrestrial network is overloaded, the manner of which is flexible and cost-effective.

Significant research efforts have been put on the UAV-assisted communications. In [8], spectrum trading among the selfish macro BS manager and UAV operators is investigated to optimize pricing strategy and bandwidth allocation. In [9], UAV-assisted edge computing is studied to serve remote IoT users by jointly optimizing the communication and computing resources. The problem of UAV deployment is investigated in [10], where UAVs are dispatched over geographical areas with intensive service demands. Leveraging the controllable mobility of UAVs, a hybrid network architecture utilizing

UAVs as moving base stations (BSs) is proposed in [11], where a UAV flies cyclically along the cell edge to offload traffic from cellular BS (CBS). UAV trajectory optimization is studied in [12]–[14]. In [12], the authors aim to minimize the number of deployed UAVs to serve the vehicles in disaster situations by jointly optimizing trajectory and radio resource allocation. In [13], the UAV trajectory optimization is investigated to minimize the expired data packets in UAV-assisted wireless sensor system. To maximize the minimum throughput over all ground users, multiple UAVs are used as aerial BSs in [14] with jointly optimized multi-user scheduling and association as well as the UAV trajectory and power control. UAV-assisted caching is studied in [15]–[17]. In [15], a UAV with limited energy is dispatched to facilitate proactive caching at the ground nodes. In [16], user-centric information is leveraged for efficient cache-enabled UAV deployment to maximize the users' quality of experience using a minimum total transmit power of the UAVs. One cache-enabled UAV is adopted in [17] and an online wireless caching scheme is designed via jointly optimizing UAV trajectory, transmission power and caching content scheduling.

Despite the extensive existing works, various technical challenges associated with UAV-assisted caching in vehicular networks remain. First, most existing works consider UAV caching in networks with low/no user mobility, which cannot be directly applied to the vehicular scenarios. The high vehicle mobility causes spatio-temporal variation in vehicle density and content request distribution, and further affects the UAV caching performance. Second, the joint decision of content placement, UAV trajectory, and content delivery in UAV-assisted vehicular networks has not been well addressed. The joint optimization is essential to improve the UAV content delivery performance in vehicular networks as the three decisions interact with each other. Third, as the vehicular network condition varies significantly with uncertainties, online decisions should be constantly made to keep pace with the dynamic vehicular environments, posing real-time requirements to the optimization solution.

In this article, we focus on the joint design of UAV caching (including content placement and content delivery) and UAV trajectory in urban vehicular networks that have substantial content demands. Our objective is to find the optimal solution to the joint optimization problem in real time to maximize the overall network throughput under the UAVs' energy constraints. By partitioning the target area into small regular areas and representing each area by a point, we construct a topology graph to find the optimal paths of the UAVs, where the edge weights are affected by the content placement and content delivery scheme. As the formulated joint caching and trajectory optimization (JCTO) problem is intractable directly and timely due to the coupling of variables, we propose a learning-based scheme named *LB-JCTO* to enable edge intelligence (EI) [18] and make real-time decisions in the highly dynamic vehicular environments.

LB-JCTO is an offline optimization and learning for online decision framework, in which deep supervised learning is conducted to facilitate online decisions under the supervision of offline optimized target. Particularly, in the first stage of *LB-JCTO*, we propose a clustering-based two-layered (CBTL) algorithm to solve the JCTO problem. Given a content placement strategy, we jointly optimize UAV trajectory and content delivery by a time-based graph decomposition method, where a directed graph is constructed in accordance with the spatial-temporal variant vehicle densities and content requests. Resource constrained shortest path (RCSP) algorithms [19] are then used to find the optimal path, representing the optimal content delivery and trajectory solution. Based on the achieved optimal results, we then leverage the particle swarm optimization (PSO) algorithm to optimize content placement, further enhancing the network throughput. Although the CBTL algorithm can achieve a satisfactory performance, the algorithm complexity still cannot meet the real-time requirements for online decisions. To this end, in the second stage of *LB-JCTO*, we adopt a deep learning architecture of convolutional neural network (CNN) [20] to conduct supervised learning at the intelligent edge. Particularly, the spatio-temporal variant network density and content requests are labeled as channeled images and input to the learning model, and the optimized solutions obtained by the CBTL algorithm are labeled as model outputs. With the well-trained CNN model, a function which maps the input network information to the output decision can be learnt to enable timely decision-making.

The merits of *LB-JCTO* are three-fold: 1) at the offline optimization stage, we can use a complicated algorithm with relatively high computation complexity to obtain the optimized results, which are good learning targets; 2) for offline training, as EI-based *LB-JCTO* is trained to learn from a good target, its performance can be well guaranteed; and 3) at the online stage, with the well-trained CNN-based model, *LB-JCTO* simply runs a matching function in the UAVs for the up-to-date collected information, which can output high-quality and real-time decisions. To evaluate the performance of *LB-JCTO*, we conduct extensive trace-driven experiments based on DiDi Chuxing GPS Dataset [21]. Performance results show that: 1) the offline optimization algorithm CBTL can achieve near-optimal performance and outperform the benchmark scheme significantly; and 2) guided by the CBTL algorithm, the CNN-based deep learning approach can also achieve superior performance, and more importantly, it can react to the network input rapidly. We highlight our major contributions in this article as follows.

- We study the joint design of UAV caching and trajectory in vehicular networks, which is of significant importance for future IoVs to deliver high-bandwidth contents robustly. Specifically, we formulate the JCTO problem to investigate the interplay between the caching scheme and UAV trajectory design, where the analysis and derivation of UAV energy consumption, achievable throughput of UAV-based moving cells and terrestrial networks are respectively presented.
- To solve the JCTO problem in real time, we propose a learning-based scheme named *LB-JCTO* to make online inferences in response to the dynamic vehicular networks. Particularly, in the *LB-JCTO* scheme, the CBTL algorithm is devised to optimize the JCTO problem offline,

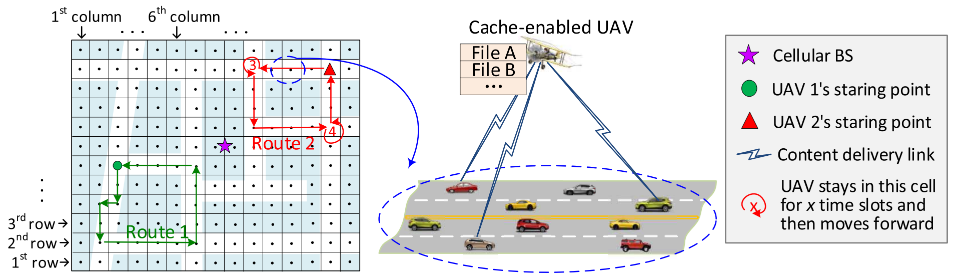


Fig. 1. Overview of UAV-aided edge caching in vehicular networks.

and then a CNN-based learning scheme is designed to facilitate online decisions.

- Extensive trace-driven experiments are carried out, and results demonstrate that the CBTL algorithm can efficiently solve the joint optimization problem, and the CNN-based learning model can well emulate the capability of CBTL while satisfying the real-time requirements.

We organize the remainder of this article as follows. System model and problem formulation are given in Section II. Section III presents the proposed *LB-JCTO* scheme, which includes the CBTL-based offline optimization as presented in Section IV, and the offline model training and online decision as investigated in Section V. Trace-driven experimental results are carried out in Section VI. In Section VII, we conclude the article and direct our future work.

II. System Scenario and Problem Formulation

A. Scenario Description

We investigate UAV-assisted edge caching in vehicular networks by utilizing UAVs to cache content files and then serve the ground vehicles. Without loss of generality, we focus on a rectangle area covered by a single CBS, as shown in Fig. 1. A set $\mathcal{K}$ of K rotary-wing UAVs equipped with caching storage units are dispatched into the system to act as moving BSs to serve the ground vehicles along with the CBS. Let $\mathcal{F} = \{f_1, f_2, \cdots, f_F\}$ be the set of F files requested in the scenario. The size of the content files is assumed to be the same[1] and denoted by $\varsigma_f, \forall f_f \in \mathcal{F}$. The caching storage capacity of UAV k is C_k, i.e., UAV k can cache no more than C_k content files. In addition, the UAV flies horizontally at a constant altitude H to achieve a lower level of energy consumption [22].

The ground vehicular networks are highly dynamic with spatio-temporal variance for vehicle densities and content request distributions. Constrained by the street-layout, vehicle densities on the roads (white areas in Fig. 1) are generally larger than those on the other areas, so as the content request probabilities. Basically, vehicle density can be estimated based on position information collected from vehicles equipped with GPS devices. Content popularity can be modeled as Zipf distributions according to the analysis for many real datasets [23]. With Zipf-based content popularity, the request distributions still vary spatially and temporally due to different user preferences. Although there exist many works modeling and predicting a user's preference by using learning methods [24], the statistical modeling of the preference of a certain user set based on real-world datasets has not been well investigated [25]. Inspired by the model in [25], we model the file preference in each grid at time slot t as follows:

- A grid v and a content file f_f are respectively associated with feature values $\phi_{v,t}$ and ϑ_f, where $\phi_{v,t}, \vartheta_f \in [0,1]$.[2]
- The probability that file f_f is requested by users in grid v at time t is calculated as:

$$r_{v,t,f} = p_f \frac{g(\phi_{v,t}, \vartheta_f)}{\sum_{v' \in \mathcal{V}} g(\phi_{v',t}, \vartheta_f)}, \tag{1}$$

where $p_f = \frac{1/f^\xi}{\sum_{m\in\mathcal{F}} 1/m^\xi}$ is the popularity of content f_f, and $g(\phi_{v,t}, \vartheta_f) = (1 - |\phi_{v,t} - \vartheta_f|)^{\frac{1}{\alpha^3}-1}$ is a kernel function representing the correlation between grid v and file f_f.[3]

Basically, $r_{v_1,t_1,f} \neq r_{v_2,t_2,f}$ for $v_1 \neq v_2$ or $t_1 \neq t_2$ due to its spatio-temporal variance, despite that vehicles may have similar preferences over some popular content files.

Aiming at maximizing the overall network throughput, the UAVs should fly to different locations to keep pace with the network dynamics. To better illustrate the time-varying locations of the UAVs, we partition the target area into small square grids with side length w. Each grid square is represented by its central point and denoted by (i,j), $i \in [1, N_{\text{row}}], j \in [1, N_{\text{col}}]$. This means that the grid locates in the i-th row and j-th column, and N_{row} and N_{col} are the numbers of rows and columns in the target area. We can then construct a topology graph $G = (\mathcal{V}, \mathcal{E})$ representing the whole map, where $\mathcal{V}$ is the set of central points of the grid squares and $\mathcal{E}$ contains the edges connecting the central points. For two points $u = (i,j)$ and $v = (m,n)$, $(u,v) \in \mathcal{E}$ if and only if $u, v \in \mathcal{V}, |i-m| \le 1, |j-n| \le 1$.[4]

The UAV endurance is equally discretized into T_U time slots, each with a time length of Δ_t. The UAVs can fly along

[1]In reality, for files with different sizes, the analysis can be easily extended by dividing each file into chunks of equal size.

[2]Features of a location may include weather characteristics, historical park, upcoming famous events, etc.; a content file is associated with features such as file type and size, metadata, keywords or tags. To simplify the model, we use a random value to represent the features of a grid/file, but this basic model can be easily extended to multi-dimensional feature vectors.

[3]The grids and files are correlated due to the underlying correlation between location and content features.

[4]In the remainder of this article, v and (i,j) are used interchangeably to represent a grid square.

the points and edges in graph G, where a UAV can move from point u to v in two continuous time slots only if $(u, v) \in \mathcal{E}$. Notice that the starting and ending points of a trajectory are the same, i.e., the location where the UAV is dispatched and collected for battery charging. The starting and ending points for multiple UAVs can be different. As shown in Fig. 1, two UAVs fly along different trajectories, which are respectively marked in red and green, with different starting points. Along the flying trajectory, the UAVs can keep changing locations at different time slots (Route 1) or choose to hover above a certain location for multiple time slots (Route 2).

B. Notation Definition

To facilitate analysis, we define necessary notation and symbols below.

1) $\mathbf{D}_{N_{\text{row}}\times N_{\text{col}}\times T_U}$ is a three-dimensional array showing the spatial- and temporal-varying vehicle densities. The (i,j,t)-th entry $d_{i,j,t}$ (or $d_{v,t}$ if $v=(i,j)$) is the average number of vehicles in the grid represented by point (i,j) $((i,j)\in\mathcal{V})$ at time slot t $(t \le T_u)$.

2) $\mathbf{R}_{N_{\text{row}}\times N_{\text{col}}\times T_U\times F}$ is a four-dimensional array representing the spatial- and temporal-varying content request distributions. The (i,j,t,f)-th entry $r_{i,j,t,f}$ (or $r_{v,t,f}$ if $v=(i,j)$) is the request probability of file f_f in the grid represented by point (i,j) at time slot t.

3) $\mathbf{X}_{K\times T_U} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_K]$ is the flying trajectories of K UAVs. $\mathbf{x}_k = [x_{k,1}, x_{k,2}, \cdots, x_{T_U}]$ is the k-th UAV's trajectory within its endurance time, where $x_{k,t}$ is its location at time slot t. Thus we have $(x_{k,t}, x_{k,t+1}) \in \mathcal{E}$ and $x_{k,1} = x_{k,T_U} = v_{0,k}$, where $v_{0,k}$ is the starting point where UAV k is released.

4) $\mathbf{A}_{K\times F} = [\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_K]$ is an indicator matrix showing the caching of content files in the UAVs, where $\mathbf{a}_k = [a_{k,1},\, a_{k,2}, \cdots a_{k,F}]$ represents the k-th UAV's caching status. $a_{k,f} = 1$ if content file f_f is cached in UAV k; otherwise, $a_{k,f} = 0$.

5) $\mathbf{S}_{K\times T_U} = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_K]$ denotes the content delivery decision of the UAVs along their trajectories, where $\mathbf{s}_k = [s_{k,1}, s_{k,2}, \cdots s_{k,T_U}]$. $s_{k,t} = 1$ if UAV k chooses to serve the vehicular requests at time t, whereas $s_{k,t} = 0$ means that UAV k flies without content delivery at time t.

C. Communication Models

In this part, we introduce the models for U2V and CBS-to-vehicle (C2V) communications.

1) U2V Communications: In this work, the caching-enabled UAVs work in Wi-Fi spectrum with a constant transmission power, which is denoted by P_U. Notice that NLoS links can severely degrade the communication performance and cannot support efficient U2V content delivery. Therefore, the UAVs' coverage radius, denoted by r_{UAV}, can be defined by constraining the LoS probability and free-space pathloss [26]. With a fixed flying altitude H, we have

$$r_{\text{UAV}} = \min\left\{ \frac{H}{\tan\left(a_1 - \frac{1}{a_2}\ln\left(\frac{1-\xi_{LoS}}{a_1\xi_{LoS}}\right)\right)}, \sqrt{\left(\frac{c\gamma_{\max}}{4\pi f_c}\right)^2 - H^2} \right\}, \tag{2}$$

where a_1 and a_2 are constant values determined by environment. f_c, c, ξ_{LoS} and $\gamma_{\max}$ respectively represent the carrier frequency, light speed, the LoS probability requirement, and U2V free space pathloss threshold. When partitioning the target area into grid squares, we can set the side length of each square as $w = \sqrt{2}r_{\text{UAV}}$. With this setting, when a UAV hovers above a grid square, its coverage area can approximately equal the square area.

According to IEEE 802.11 standard, the Wi-Fi coverage area can be divided into zone areas based on the achieved signal-to-noise ratio (SNR) levels, and the data rates are calculated based on the Wi-Fi modulation and coding schemes [27], [28]. Therefore, the coverage area of a Wi-Fi-based UAV can be divided into L zones. The j-th zone has a distinct annulus area with width l_j and a data rate of R_j. Thus, a vehicle within a grid area can achieve a mean throughput of:

$$\overline{R}_{UAV} = \rho\left(\frac{\sum_{j=1}^{j=L}\left(R_j\left[(l_j + l_{j-1})^2 - l_{j-1}^2\right]\right)}{\left(\sum_{j=1}^{j=L} l_j\right)^2}\right), \tag{3}$$

where $l_0 = 0$ and ρ is Wi-Fi throughput efficiency factor.

The Wi-Fi channel is shared by associated vehicles under a contention-based mechanism. With N vehicles sharing the Wi-Fi channel in a grid area, each vehicle achieves a data rate of $\overline{R}_{UAV}/N$. When the associated vehicles have a throughput requirement of R_{req}, the number of vehicles that can be simultaneously served by a UAV should be no more than $N_{U,\max}$, where

$$N_{U,\max} = \left\lfloor \overline{R}_{UAV}/R_{req} \right\rfloor. \tag{4}$$

2) C2V Communications: In this work, channel inversion power control is adopted for the CBS, where transmit power is allocated based on the channel conditions to ensure equal average SNR for all the associated vehicles. The C2V channel gain is considered as $h_{i,C} = \varrho_{i,C} d_{i,C}^{-\alpha}$, where $\varrho_{i,C}$ is the channel fading and follows an exponential distribution with unit mean, $d_{i,C}$ is the distance between the vehicle and CBS, and α is the pathloss exponent. Shadowing and Doppler effects [29] are not involved in this work for analysis simplicity. Let B_C be the total available cellular bandwidth, which is equally shared by vehicles using C2V communications. $P_{C,\max}$ is the maximum available transmission power of the CBS. Given that the C2V channel gains keep changing due to vehicle mobility, it is costly to perform real-time power allocation based on every vehicle's instantaneous channel condition. Thus, the average C2V pathloss is used for power allocation for vehicles in the same grid. The average pathloss is dependent on the average distance to the CBS, which can be calculated referring to *Lemma 1* in [1]. When there are no UAVs in the system, the overall cellular network throughput at time slot t is

$$R_C(t) = \frac{B_C}{\sum_{v\in\mathcal{V}} d_{v,t}} \sum_{v\in\mathcal{V}} d_{v,t} \log\left(1 + \frac{P_{v,C}(t) h_{v,C}}{\frac{B_C \sigma^2}{\sum_{v\in\mathcal{V}} d_{v,t}}}\right), \tag{5}$$

where $P_{v,C}(t)$ and $h_{v,C}$ are the transmission power and average channel gain from the CBS to the vehicles in v ($v \in \mathcal{V}$), and σ^2 is the noise power density. Thus, we have

$$\sum\nolimits_{v\in\mathcal{V}} d_{v,t}P_{v,C}(t) = P_{C,\max}, \\ P_{v_1,C}(t)h_{v_1,C} = P_{v_2,C}(t)h_{v_2,C},\ \forall v_1, v_2 \in \mathcal{V}. \quad (6)$$

Assuming that K UAVs have caching status $\mathbf{A}$, delivery decision $\mathbf{S}$, and trajectories $\mathbf{X}$, the set of positions of the K UAVs at time slot t is $\mathcal{V}_t = \{x_{1,t}, x_{2,t}, \cdots, x_{K,t}\}$. In grid $x_{k,t}$, the average number of vehicles that need to be served by the CBS is derived as:

$$n_{C,k,t}^{\mathbf{A},\mathbf{X},\mathbf{S}} = (1-s_{k,t})d_{x_{k,t},t} + s_{k,t}d_{x_{k,t},t}\left(1-\sum_{f_f\in\mathbf{a_k}} r_{x_{k,t},t,f}\right). \quad (7)$$

The cellular spectrum bandwidth allocated to each associated vehicle, denoted by $B_{C,t}^{\mathbf{A},\mathbf{X},\mathbf{S}}$, is

$$B_{C,t}^{\mathbf{A},\mathbf{X},\mathbf{S}} = \frac{B_C}{\sum_{k=1}^{K} n_{C,k,t}^{\mathbf{A},\mathbf{X},\mathbf{S}} + \sum_{u\in\mathcal{V},u\notin\mathcal{V}_t} d_{u,t}}. \quad (8)$$

The average throughput achieved by the CBS (denoted by $R_{C,t}^{\mathbf{A},\mathbf{X},\mathbf{S}}$) and UAV k (denoted by $R_{U,k,t}^{\mathbf{A},\mathbf{X},\mathbf{S}}$) can be respectively expressed as:

$$R_{C,t}^{\mathbf{A},\mathbf{X},\mathbf{S}} = B_{C,t}^{\mathbf{A},\mathbf{X},\mathbf{S}}\left(\sum_{u\in\mathcal{V},u\notin\mathcal{V}_t} d_{u,t}\log\left(1+\frac{P_{u,C}(t)h_{u,C}}{B_{C,t}^{\mathbf{A},\mathbf{X},\mathbf{S}}\sigma^2}\right) + \sum_{k=1}^{K} n_{C,k,t}^{\mathbf{A},\mathbf{X},\mathbf{S}}\log\left(1+\frac{P_{x_{k,t},C}(t)h_{x_{k,t},C}}{B_{C,t}^{\mathbf{A},\mathbf{X},\mathbf{S}}\sigma^2}\right)\right),$$

$$R_{U,k,t}^{\mathbf{A},\mathbf{X},\mathbf{S}} = \overline{R}_{UAV}\cdot\varepsilon\left(d_{x_{k,t},t}\cdot s_{k,t}\cdot\sum\nolimits_{f_f\in\mathbf{a_k}} r_{x_{k,t},t,f}\right), \quad (9)$$

where $\varepsilon(x)$ is a unit step function, $\varepsilon(x) = 1$ if $x > 0$; otherwise, $\varepsilon(x) = 0$. Then we can derive the overall system throughput as:

$$R(\mathbf{A},\mathbf{X},\mathbf{S}) = \sum\nolimits_{t=1}^{T_U}\left(R_{C,t}^{\mathbf{A},\mathbf{X},\mathbf{S}} + \sum\nolimits_{k=1}^{K} R_{U,k,t}^{\mathbf{A},\mathbf{X},\mathbf{S}}\right). \quad (10)$$

D. UAV Energy Consumption Models

The energy consumption of the caching-enabled UAVs mainly includes two parts: the propulsion energy required to support its movement and the communication energy for content delivery.

1) Propulsion Energy: According to [30] and [1], for a rotary-wing UAV with speed V, the propulsion power consumption and the propulsion energy consumption during one time slot can be respectively expressed as:

$$P(V) = P_0\left(1+\frac{3V^3}{U^2}\right) + P_1\left(\left(1+\frac{V^4}{4v_r^4}\right)^{\frac{1}{2}} - \frac{V^2}{2v_r^2}\right)^{\frac{1}{2}} + \frac{1}{2}AV^3,$$

$$E_p(x) = \frac{x}{V}P(V) + \max\left\{\Delta_t - \frac{x}{V}, 0\right\}\cdot(P_0+P_1), \quad (11)$$

where $x \in \{0, w, \sqrt{2}w\}$ is the flying distance within one time slot determined by the UAV trajectory planning, P_0, P_1, U, v_r, and A are constant parameters related to the UAV's weight, wing area, air density, etc.

2) Communication Energy: When UAV k flies above grid v at time slot t, the probability that there are n vehicles requesting file f_f (with request probability $r_{v,t,f}$) is:

$$\Pr(f_f, d_{v,t}, n) = \binom{d_{v,t}}{n} r_{v,t,f}^n (1-r_{v,t,f})^{d_{v,t}-n}. \quad (12)$$

Let $\mathbf{a}_k$ be the caching status of UAV k and $\mathcal{F}_k = \{f_{k,1}, f_{k,2}, \cdots, f_{k,m}\}$ be the set of m content files satisfying $a_{k,f} = 1\ \forall f_f \in \mathcal{F}_k$. Assuming that the requests for different files are independent, the probability that there are n requests for files in $\mathcal{F}_k$ is

$$\Pr(\mathcal{F}_k, d_{v,t}, n) = \sum_{n_1=0}^{n}\Pr(f_{k,1}, d_{v,t}, n_1)\sum_{n_2=0}^{n-n_1}\Pr(f_{k,2}, d_{v,t}, n_2)\cdots \sum_{n_{m-1}=0}^{n-\sum_{j=1}^{m-2} n_j}\Pr(f_{k,m-1}, d_{v,t}, n_{m-1})\Pr(f_{k,m}, d_{v,t}, n - \sum_{j=1}^{m-1} n_j). \quad (13)$$

Note that one extreme case is that every vehicle requests all the cached files, in which case there are $m\cdot d_{v,t}$ requests for files in $\mathcal{F}_k$. For $n > m\cdot d_{v,t}$, we have $\Pr(\mathcal{F}_k, d_{v,t}, n) = 0$. Therefore, given UAV k's caching status $\mathbf{a}_k$ and its position $x_{k,t}$ at time t, the average communication energy consumption to serve the requesting vehicles is:

$$E_c(\mathbf{a}_k, x_{k,t}) = \sum_{n=1}^{m\cdot d_{v,t}}\Pr\left(\mathcal{F}_k, d_{x_{k,t},t}, n\right)P_U\min\left\{\Delta_t, \frac{n\cdot\varsigma_f}{\overline{R}_{\text{UAV}}}\right\}. \quad (14)$$

E. Problem Formulation

To maximize the overall network throughput in the UAV-assisted edge caching system under UAV energy constraints, $\mathbf{A}$, $\mathbf{S}$, and $\mathbf{X}$ should be jointly optimized since they interact with each other. Based on the network throughput and UAV energy consumption analysis given in Sections II-C and II-D, the JCTO problem can be formulated as:

$$(\text{JCTO}):\ \max_{\mathbf{A},\mathbf{X},\mathbf{S}}\ R(\mathbf{A},\mathbf{X},\mathbf{S}) \quad (15)$$

$$s.t.\ \sum\nolimits_{f_f\in\mathcal{F}} a_{k,f} \le C_k,\quad \forall f_f\in\mathcal{F}, \forall k\in\mathcal{K}, \quad (15a)$$

$$\sum_{t=1}^{T_U} E_p(\|x_{k,t}-x_{k,t-1}\|) + E_c(\mathbf{a}_k, x_{k,t})\cdot s_{k,t} \le E_{k,\max}, \quad (15b)$$

$$x_{k,1} = x_{k,T_U} = v_{0,k},\quad (x_{k,t-1}, x_{k,t})\in\mathcal{E}, \quad (15c)$$

$$a_{k,f} = \{0,1\},\quad s_{k,t} = \{0,1\}, \quad (15d)$$

where $E_{k,\max}$ is the overall on-board energy of the k-th UAV. Constraint (15a) restricts the maximum number of files cached in the UAVs and constraint (15b) regulates the maximum allowable UAV energy consumption. Constraint (15c) represents that UAVs can only fly along the edges in the

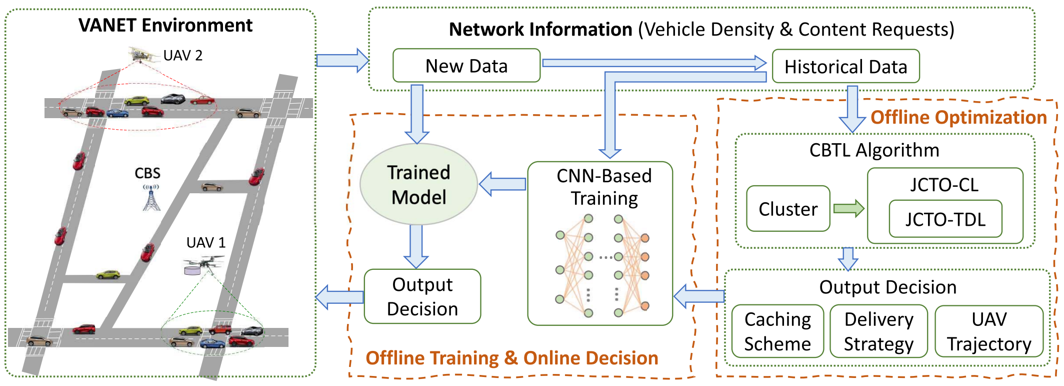


Fig. 2. Working diagram of the proposed *LB-JCTO* scheme.

graph and need to finally return to the starting points. The JCTO problem in (15) is non-convex and intractable since $\mathbf{A}$, $\mathbf{X}$, and $\mathbf{S}$ should be jointly optimized for all the K UAVs under spatio-temporal network variations and the energy constraints of UAVs.

III. Design of *LB-JCTO*

In this work, we propose a learning-based framework named *LB-JCTO* to solve the JCTO problem in (15). As shown in Fig. 2, the *LB-JCTO* scheme includes two major stages: 1) offline optimization; and 2) offline model training and online decision.

Offline Optimization: In this stage, network information (including vehicle density and content request distribution) is obtained either from collected historical data or from predictions. Then we propose a CBTL algorithm to effectively solve the JCTO problem and achieve near-optimal solutions to the content placement, delivery decisions, and UAV trajectories.

Offline Model Training and Online Decision: We leverage a CNN-based deep learning scheme to learn from the CBTL algorithm and make fast decisions. The CBTL algorithm works as a labeler (or supervisor) for the CNN-based learning model. In specific, the network information and the solution obtained by the CBTL algorithm work together as labeled data, based on which a function that maps the input network information to the output decisions can be learnt with the CNN-based model. After well-trained, the learning model can be utilized to output the corresponding JCTO decisions rapidly to react to new network information. In the meantime, the new network information can be collected and used to further train and update the learning model.

Notice that in the *LB-JCTO scheme*, network information collection, CBTL-based offline optimization, and CNN-based offline model training can be conducted at the UAV control center or edge server with powerful computing and processing capabilities. Then the well-trained learning model can be transferred and implemented on the UAVs to perform model inference locally and make fast response to the dynamic network information.

A. Offline Optimization

In the offline optimization stage, the proposed CBTL algorithm first groups vehicles into $K+1$ clusters, each served by a UAV or the CBS. The clustering process ensures that each UAV only needs to fly within a certain area rather than traveling a long distance. This helps save the limited on-board energy and prevent potential collisions among different UAVs. When clustering the vehicles, a new metric combining three different types of similarities is considered in this work. More specifically, cellular performance similarity, physical location similarity, and content preference similarity are considered to ensure that vehicles in the same cluster have similar C2V channel conditions, physical locations, and content interests.

After the vehicle clustering, the JCTO problem needs to be solved for the UAV for each cluster. Since the JCTO problem is non-convex and difficult to solve, the proposed CBTL algorithm adopts a vertical decomposition that leads to the following two-layered structure of the problem:

- *Caching-Layer (CL) Optimization:* The CL optimization problem can be reformulated as:

$$\text{(JCTO-CL)}: \max_{\mathbf{A}} \sum\nolimits_{t=1}^{T} R(\mathbf{A},\mathbf{X},\mathbf{S}) \tag{16}$$
$$s.t. \text{ Constraint (15a).} \tag{16a}$$

- *Trajectory-and-Delivery-Layer (TDL) Optimization:* The TDL optimization problem can be reformulated as:

$$\text{(JCTO-TDL)}: \max_{\mathbf{X},\mathbf{S}} \sum\nolimits_{t=1}^{T} R(\mathbf{A},\mathbf{X},\mathbf{S}) \tag{17}$$
$$s.t. \text{ Constraints (15b-15d).} \tag{17a}$$

In this work, the JCTO-CL problem is solved by leveraging the PSO algorithm, and a time-based graph decomposition method is devised to solve the JCTO-TDL problem, which will be elaborated in Sections IV-C and IV-D. Notice that when solving the JCTO-CL problem, the quality of a caching policy (i.e., the achievable $R(\mathbf{A},\mathbf{X},\mathbf{S})$) is determined by the optimal achievable performance with the JCTO-TDL problem. In other words, the JCTO-TDL optimization is embedded within the JCTO-CL problem in our CBTL-based offline optimization algorithm.

B. Offline Model Training and Online Decision

Given network information, the CBTL-based offline optimization algorithm can achieve satisfied throughput performance. However, in practice it might be difficult to precisely predict the vehicle mobility and content request distributions within each grid. For example, a vehicle collision can easily change the vehicle density in the accident location as well as in the surrounding grids. When the network condition changes unexpectedly after dispatching the UAVs, the achievable system throughput might decrease if the UAVs keep moving along the pre-designed trajectories. Under such circumstances, re-calculating a good trajectory is critical to guarantee efficient UAV service provision. Since UAVs are generally energy-constrained and have limited computing power, our proposed CBTL algorithm is not suitable to be implemented in the UAVs to continuously update the trajectories and delivery decisions in real time. Therefore, in this work, we design a CNN-based deep learning model that is trained offline under the supervision of the CBTL-based algorithm. Then the UAVs can obtain the trained model from the edge server and perform real-time model inference locally [18].

CNN is an effective image processing algorithm and has been widely applied in many fields [31], [32]. With superior learning ability in image understanding, CNN is suitable for our problem for the following reasons. First, the convolutional layers can effectively capture the local dependencies and extract important features, e.g., network features like the road layout or dense areas with frequent requests. Second, CNN also introduces pooling mechanisms to reduce data dimension while preserving dominant features. With these two characteristics, the CNN-based model is not only good at learning features but also scalable to large-scale problems.

In the offline training of the CNN-based model, supervised learning is conducted to learn the matching function from the input data to the output decisions. Specifically, the input network information is labeled as channeled images and normalized before fed into the learning model. The optimized solutions obtained by the CBTL algorithm are labeled as model outputs to provide supervision to the learning model. The detailed learning model structure will be introduced in Section V.

IV. CBTL-Based Offline Optimization

A. Determining the Number of UAVs

With UAV-assisted communications, the network throughput can be improved and thus each vehicle's satisfaction with respect to perceived QoS is enhanced. The optimal number of UAVs dispatched into the system depends on user service satisfaction and required resource. In general, users prefer more UAVs to achieve higher performance satisfaction levels. The service providers, on the other hand, tend to use the least resource to achieve the best gain and focus more on cost efficiency, e.g., the performance enhancement introduced by each UAV. Fig. 3 shows the impact of the number of UAVs on users' satisfaction level (approximated by using sigmoid function [33]) and throughput improvement by each UAV. It can be seen that, with more UAVs launched into the system, the average user satisfaction level increases while the throughput improvement efficiency decreases. Thus, the optimal K varies in different scenarios with diverse user satisfaction requirements, the number of UAVs a provider has, and/or the deployment cost efficiency.

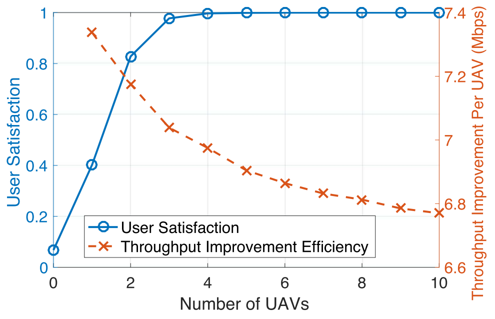


Fig. 3. Impact of K on user satisfaction level and throughput improvement efficiency. $H = 35$ m, $P_{C,\max} = 43$ dBm, $B_C = 20$ MHz, $P_U = 28$ dBm.

In our UAV-based edge caching system, we aim to minimize the number of UAVs required to satisfy the vehicles' throughput requirement of R_{req}. Recall that channel inversion power control is adopted for the CBS and the bandwidth is equally allocated to all associated vehicles. All vehicles served by the CBS have the same received signal strength γ_C, which can be calculated based on (6):

$$\gamma_C = P_{C,\max} \Big/ \left(\sum_{v \in \mathcal{V}} \frac{d_{v,t}}{h_{v,C}} \right). \tag{18}$$

When guaranteeing the vehicles' throughput requirement, the CBS can serve more vehicles if it serves close vehicles rather than remote vehicles, since the close ones have better C2V links and require smaller transmit power. For this reason, we sort the grids in descending order based on C2V channel power gain. Let v_i be the i-th closest grid to the CBS. When the CBS serves users in the first N_C nearest grids, the achievable throughput per vehicle should satisfy

$$\frac{B_C}{\sum_{i=1}^{N_C} d_{v_i,t}} \log \left(1 + \frac{P_{C,\max} \sum_{i=1}^{N_C} d_{v_i,t}}{B_C \sigma^2 \sum_{i=1}^{N_C} \frac{d_{v_i,t}}{h_{v_i,C}}} \right) \geq R_{req}. \tag{19}$$

The left side of Eq. (19) decreases monotonously with N_C. Let $N_{C,\max}$ denote the maximum value of N_C that satisfies Eq. (19). Although the closed-form expression of the optimal N_C cannot be derived, $N_{C,\max}$ can be determined by using approaches like bisection method. Therefore, the minimum number of UAVs required to ensure vehicle throughput requirement is expressed as:

$$K_{\min} = \left\lceil \frac{\sum_{v \in \mathcal{V}} d_{v,t} - \sum_{i=1}^{N_{C,\max}} d_{v_i,t}}{N_{U,\max}} \right\rceil. \tag{20}$$

B. Vehicle Clustering

With K UAVs dispatched into the system, the vehicles can be clustered into $K+1$ groups to be served by the UAVs and the CBS. In this work, a widely used clustering method, K-means clustering [34], is adopted to cluster the vehicles by considering the following similarities:

1) Cellular Performance Similarity: When a UAV is dispatched into the system, the throughput gain increases if it serves vehicles with poor cellular performance [35]. Thus, the cellular throughput similarity is an important factor to be considered. Vehicles with good C2V channels prefer to be in the same cluster and served by the CBS, while other vehicles need to be served by UAVs. When assigned the same cellular bandwidth B_0 and transmit power P_0, vehicles in grid v achieve a throughput of $R_{C,v} = B_0 \log(1 + P_0\ h_{v,C}/\sigma^2)$. The similarity between vehicles in grids v and u is evaluated by $\mathrm{sim}_{u,v,1} = \min\{R_{C,v}/R_{C,u}, R_{C,u}/R_{C,v}\} \in [0,1]$.

2) Physical Location Similarity: Basically, the grid squares in the same cluster served by a UAV should be in proximity to avoid extra UAV propulsion energy consumption. The physical location similarity among grids is evaluated by $\mathrm{sim}_{u,v,2} = 1 - \frac{\mathrm{dist}_{u,v}}{\max_{u,v\in\mathcal{V}} \mathrm{dist}_{u,v}} \in [0,1]$, where $\mathrm{dist}_{u,v}$ is the average distance between grids u and v.

3) Content Preference Similarity: To improve the utilization efficiency for the limited UAV caching resources, vehicles with similar content interests should be grouped and served by the same UAV. Utilizing cosine similarity to evaluate the file preference similarity [25], we can express the average interest similarity between grids u and v during T_U time slots as:

$$\mathrm{sim}_{u,v,3} = \frac{1}{T_U} \sum\nolimits_{t=1}^{T_U} \frac{\mathbf{r}_{v,t} \cdot \mathbf{r}_{u,t}}{\|\mathbf{r}_{v,t}\| \cdot \|\mathbf{r}_{u,t}\|}, \tag{21}$$

where $\mathbf{r}_{v,t} = [r_{v,t,f_1}, \cdots, r_{v,t,f_F}]$ is the request distribution in grid v at time t and $\mathrm{sim}_{u,v,3} \in [0,1]$.

Taking the above-mentioned three metrics into account, the overall similarity between grids u and v can be evaluated by

$$\mathrm{sim}^{all}_{u,v} = \mathrm{sim}^{\alpha_1}_{u,v,1} \cdot \mathrm{sim}^{\alpha_2}_{u,v,2} \cdot \mathrm{sim}^{\alpha_3}_{u,v,3}, \tag{22}$$

where parameters α_1, α_2 and α_3 control the relative importance of the three metrics. For example, with $\alpha_1 > 1$, we give a higher priority to the cellular performance similarity, while $\alpha < 1$ indicates that we put more emphasis on the other two metrics. Targeting at maximizing $\mathrm{sim}^{all}_{u,v}$ within a cluster, the K-means based clustering algorithm can be summarized as given in **Algorithm 1.**

Algorithm 1 K-Means-Based Vehicle Clustering

```
Let $\overline{d}_v = \frac{1}{T_U}\sum_{t=1}^{T_U} d_{v,t}$, $\overline{\mathbf{r}}_v = \frac{1}{T_U}\sum_{t=1}^{T_U} \mathbf{r}_{v,t}$,
$\alpha_1, \alpha_2, \alpha_3 = 1$.
Step 1: Centroid Initialization: The first cluster centroid
  $v_C^0$ is the grid where the CBS is located. Besides,
  randomly choose $K$ grids, denoted by $v_C^1, \cdots, v_C^K$,
  as the centroids for the remaining $K$ clusters.
Step 2: Grid Clustering: $K+1$ clusters, denoted by
  $\mathcal{C}_0, \mathcal{C}_1, \cdots, \mathcal{C}_K$, are created by associating every grid with
  the centroid with maximum similarity based on (22).
Step 3: Centroid Update: Update the $K+1$ cluster
  centroids:
  $v_C^0 = v_C^0, \quad v_C^k = \frac{1}{\sum_{v\in\mathcal{C}_k}\overline{d}_v}\sum_{v\in\mathcal{C}_k}\overline{d}_v v,$
  $R_{C,v_C^k} = \frac{1}{\sum\limits_{v\in\mathcal{C}_k}\overline{d}_v}\sum_{v\in\mathcal{C}_k}\overline{d}_v R_{C,v}, \ \mathbf{r}_{v_C^k} = \frac{1}{\sum\limits_{v\in\mathcal{C}_k}\overline{d}_v}\sum_{v\in\mathcal{C}_k}\overline{d}_v \overline{\mathbf{r}}_v.$
Step 4: Repeat Steps 2-3 until converging.
Step 5: Repeat Steps 1-4 and choose the best for
  multiple runs.
```

C. JCTO-TDL Optimization in CBTL Algorithm

After vehicle clustering, the CBTL algorithm is applied to solve the JCTO problem for each UAV based on the vertical problem decomposition as mentioned in Section III-A. In this subsection, the JCTO-TDL problem is addressed by the proposed time-based graph decomposition method.

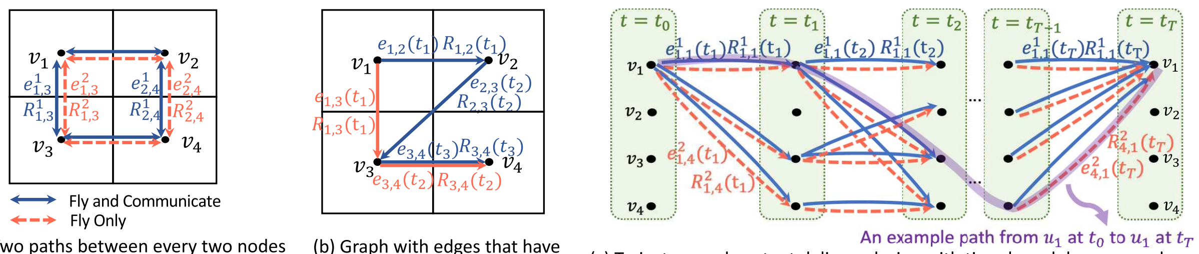
Given the caching policy and the current position of a UAV, when the UAV flies to any feasible position in the next time slot, the achievable network throughput and the corresponding energy consumption can be calculated based on Sections II-C and II-D. The JCTO-TDL sub-problem, which aims to find the optimal content delivery and UAV trajectory to maximize the network throughput under the energy constraint, is similar to the RCSP problem, which has been investigated in [19]. However, the RCSP algorithm cannot be directly applied to the JCTO-TDL problem for the following reasons: *1)* in each time slot, $s_{k,t}$ can be either 0 or 1, which corresponds to two edges between the adjacent grids with different weights (i.e., achievable network throughput) and costs (i.e., energy consumption), as shown in Fig. 4a; and *2)* due to the time-variant $\mathbf{D}$ and $\mathbf{R}$, the weights and costs of the edges change when visited by the UAVs at different time slots. Fig. 4b gives two examples of trajectories from v_1 to v_4, which are respectively marked in red and blue. The network throughput and energy consumption vary when the UAV flies from v_3 to v_4 at different times.

To address the above-mentioned issues, we propose a time-based decomposition method to expand graph $\mathcal{G}$ into a directed graph, as shown in Fig. 4c. The edge exists between v_i at time t_1 and v_j at time t_2 only if $t_2 = t_1 + \Delta_t$ and $(v_i, v_j) \in \mathcal{V}$. An example path is given as the purple curve in Fig. 4c, which represents a possible UAV trajectory and content delivery decision in each step from source (v_1 at time t_0) to destination (v_1 at time t_T[5]). To this end, the JCTO-TDL problem is equivalent to finding the optimal path in the expanded graph to maximize sum weights under the cost constraint. With given source and destination, RCSP algorithms can be leveraged to find the optimal path. Given that the UAV's energy consumption varies with different trajectories and delivery decisions, it is difficult to determine the destination t_k when the UAV exhausts its energy. To address this issue, we execute an energy-constrained line-search on t_k to find the optimal JCTO-TDL solution, as described in **Algorithm 2**.

D. JCTO-CL Optimization in CBTL Algorithm

The JCTO-TDL optimization provides the optimal $\mathbf{X}$, $\mathbf{S}$, and the corresponding $R(\mathbf{A}, \mathbf{X}, \mathbf{S})$ with given caching policy. In this subsection, the content placement is optimized to further improve the network throughput. However, conventional linear programming approaches cannot solve the JCTO-CL

[5]The UAV returns to the UAV center at time $t_T \le T_U$. From time slot t_{T+1} to T_U, the UAV stays in the UAV center without content delivery and charges its battery for the next flight.


v_1
v_2
v_3
v_4
$e^1_{1,3}$ $e^2_{1,3}$ $e^1_{2,4}$ $e^2_{2,4}$
$R^1_{1,3}$ $R^2_{1,3}$ $R^1_{2,4}$ $R^2_{2,4}$
Fly and Communicate
Fly Only
$e_{1,2}(t_1)$ $R_{1,2}(t_1)$
$e_{2,3}(t_2)$ $R_{2,3}(t_2)$
$e_{1,3}(t_1)$ $R_{1,3}(t_1)$
$e_{3,4}(t_3)R_{3,4}(t_3)$
$e_{3,4}(t_2)$ $R_{3,4}(t_2)$
$t = t_0$
$t = t_1$
$t = t_2$
$t = t_{T+1}$
$t = t_T$
$e^1_{1,1}(t_1)R^1_{1,1}(t_1)$
$e^1_{1,1}(t_2)$ $R^1_{1,1}(t_2)$
$e^1_{1,1}(t_T)R^1_{1,1}(t_T)$
$e^2_{1,4}(t_1)$
$R^2_{1,4}(t_1)$
$R^2_{4,1}(t_T)$
$e^2_{4,1}(t_T)$
An example path from u_1 at t_0 to u_1 at t_T

(a) Two paths between every two nodes showing different content delivery cases (b) Graph with edges that have time-variant weights and costs (c) Trajectory and content delivery design with time-based decomposed graph

Fig. 4. A simple example of trajectory and content delivery design with time-based graph decomposition. ($R^1_{i,j}(t)$ and $R^2_{i,j}(t)$: achievable throughput when flying from v_i to v_j at time t with and without content delivery. $e^1_{i,j}(t)$ and $e^2_{i,j}(t)$: corresponding energy consumption.)

Algorithm 2 JCTO-TDL Optimization in CBTL Algorithm

```
$v_k$: the starting and ending point of UAV $k$.
**Step 1:** For any $(u, v) \in \mathcal{E}$ and $t \in [1, T_U]$, calculate the weights (throughput) and costs (energy consumption).
**Step 2:** Use shortest path (SP) algorithms (e.g., Dijkstra's algorithm) to find the path with smallest cost. Let $E_{t_0,t_T}$ denote the sum cost from source to $v_k$ at time slot $t_T$. Find $t_T$ such that $E_{t_0,t_T} \le E_{k,all}$ and $E_{t_0,t_{T+1}} > E_{k,all}$. Let $t_T^{\max} = t_T$.
**Step 3:** Use SP algorithms to find the path with largest cost. Find $t_T$ such that $E_{t_0,t_T} \le E_{k,all}$ and $E_{t_0,t_{T+1}} > E_{k,all}$. Let $t_T^{\min} = t_T$.
**for** $t_T = [t_T^{\min}, t_T^{\max}]$ **do**
    Construct time-based decomposed graph as shown in Fig. 4(c).
    Apply RCSP algorithms to find the optimal path in the graph. Record the best path which leads to the maximum achievable network throughput.
**end**
Output the recorded best path.
```

problem because we are not able to provide a closed-form expression for $R(\mathbf{A}, \mathbf{X}, \mathbf{S})$. Compared to the exhaustive searching scheme with exorbitant time complexity, heuristic algorithms, especially the evolutionary heuristics, are considered as better alternative choices to approach the optima [36]. More specifically, we utilize the PSO algorithm in this work due to its low computational cost and fast convergence [37].

When applying the PSO algorithm to solve the JCTO-CL problem, we first generate a group of particles, each of which has a position indicating a potential caching scheme. Then the fitness values (achievable network throughput) of these particles are calculated based on the analysis in Section IV-C. Based on the particles' positions and fitness values, there exist a local optimal position ($\varpi^\ell_{local}(t)$) for each particle ℓ and a global optimal position ($\varpi_{global}(t)$) for the entire particle swarm at the t-th iteration. Then, at iteration $t+1$, the position $\varpi^\ell(t+1)$ and velocity $\nu^\ell(t+1)$ of particle ℓ are updated as:

$$\begin{aligned}\nu^\ell(t+1) &= \phi\nu^\ell(t) + c_1\phi_1(\varpi^\ell_{\text{local}}(t) - \varpi^\ell(t)) \\ &\quad + c_2\phi_2(\varpi_{\text{global}}(t) - \varpi^\ell(t)), \\ \varpi^\ell(t+1) &= \varpi^\ell(t) + \nu^\ell(t+1), \end{aligned} \tag{23}$$

where ϕ determines convergence speed, c_1 and c_2 are local and global learning coefficients, and ϕ_1 and ϕ_2 are positive random variables. The iteration terminates when a termination criterion (e.g., reaching the maximum iterations or minimum error criteria) is met.

To this end, with given ground vehicle densities and content request distributions, the JCTO problem can be solved effectively by our proposed CBTL algorithm.

V. CNN-Based Learning for Online Decision

Based on the offline optimized solutions provided by the CBTL algorithm, a CNN-based deep supervised learning scheme is designed in this section to make real-time decisions under dynamic network conditions.

A. Image-Like Input Data

As stated in Section IV, the JCTO problem is investigated with dynamic network information (e.g., $\mathbf{D}$ and $\mathbf{R}$). In this section, we adopt an image-based method to present the spatio-temporal network dynamics as images to facilitate the learning scheme.

Vehicle density $\mathbf{D}$ is a three-dimensional array, which consists of T_U two-dimensional matrices. Each two-dimensional matrix has $N_{\text{row}} \times N_{\text{col}}$ elements and can be viewed as a channel of an image. In this way, each pixel in the image corresponds to one element in the matrix. The input vehicle density can then be considered as an image with T_U channels, which differs from traditional images which commonly have three channels, i.e., RGB.

Content request distribution $\mathbf{R}$, on the other hand, is a four-dimensional array. To make the input dimension consistent without losing useful information about the request distribution, we use $\phi_{v,t}$ (as discussed in Section II-A, $v = (i,j), i \in [1, N_{\text{row}}], j \in [1, N_{\text{col}}]$) to represent the content request distribution in grid v at time t. Then the three-dimensional array $\mathbf{\Phi}_{N_{\text{row}} \times N_{\text{col}} \times T_U}$, with the (i,j,t)-th entry being $\phi_{i,j,t}$, can also be treated as an image with T_U channels.

Notice that $\mathbf{D}$ and $\mathbf{\Phi}$ are of different scales. Considering that neural networks are sensitive to the scaling and distribution of their inputs, proper normalization is critical for convergence [38]. In our CNN-based learning model, the input data is normalized before fed into the learning model by using the

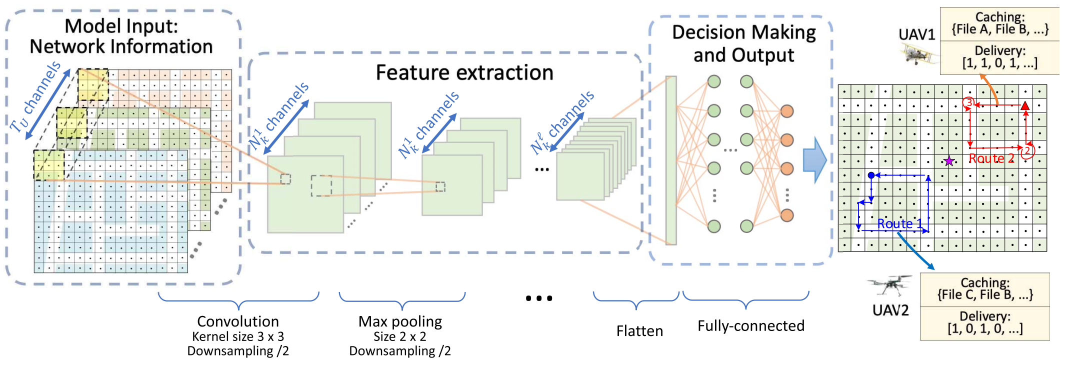


Fig. 5. Structure of the CNN-based deep supervised learning model.

min-max normalization as follows:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}. \tag{24}$$

With normalized range of the input data, the impact of $\mathbf{D}$ and $\mathbf{\Phi}$ are re-scaled to approximately the same level to facilitate the learning process.

B. CNN-Based Model Training

Fig. 5 shows the structure of our CNN-based learning model with three main parts, i.e., model input, feature extraction, and decision making and output.

1) Model input contains the image-like network information arrays with spatio-temporal characteristics, as explained in Section V-A. Thus, the l-th input data can be written as:

$$\text{IN}_l = \left[\mathbf{d}_1, \cdots, \mathbf{d}_t, \cdots, \mathbf{d}_{T_U}, \boldsymbol{\phi}_1, \cdots, \boldsymbol{\phi}_t, \cdots, \boldsymbol{\phi}_{T_U}\right], \tag{25}$$

where $\mathbf{d}_t$ and $\boldsymbol{\phi}_t$ are $N_{\text{row}} \times N_{\text{col}}$ matrices with (i,j)-th entry being $d_{i,j,t}$ and $\phi_{i,j,t}$, respectively.

2) The extraction of the network features is accomplished by the combination of convolutional and pooling layers, which is the core part of the CNN model. In the ℓ-th convolutional layer, there are N_k^ℓ kernels (or filters) of size $W_k^\ell \times H_k^\ell$ with stride s_k^ℓ. For instance, as shown in the 1st convolutional layer in Fig. 5, there are N_k^1 kernels of size 3×3 with stride 2, then the input network information of size $2N_{\text{row}} \times N_{\text{col}} \times T_U$ is fed into the 1st layer to convolve with the N_k^1 kernels, producing an output of size $\left(\lfloor\frac{2N_{\text{row}}-3}{2}\rfloor + 1\right) \times \left(\lfloor\frac{N_{\text{col}}-3}{2}\rfloor + 1\right) \times N_k^1$. This output is then added by biases and activated by an activation function (e.g., ReLU, Sigmoid, Softmax, etc), which introduces non-linearity into the system to improve the model's learning capability. After the convolutional layer, a pooling layer is introduced to downsample the convolved features by summarizing the presence of features, by using either a max-pooling function or an average-pooling function. For example, in the second layer (first pooling layer) in Fig. 5, we use max-pooling with size 2×2 and stride 2 to downsample the convolved features and the output is of size $\left(\left\lfloor\frac{\lfloor\frac{2N_{\text{row}}-3}{2}\rfloor - 1}{2}\right\rfloor + 1\right) \times \left(\left\lfloor\frac{\lfloor\frac{N_{\text{col}}-3}{2}\rfloor - 1}{2}\right\rfloor + 1\right) \times N_k^1$. With max-pooling layers, the data dimension can be reduced whereas dominant features can be preserved, and the level of distortion invariance can be improved. Basically, the CNN has more than one convolutional layer. With added layers, the CNN can capture not only some straightforward features (e.g., detection of road layout), but also sophisticated features (e.g., recognizing needy areas with intensive requests or undesired C2V links) of the model input.

3) In the decision making part, the extracted features are first flattened and concatenated into a column vector. Then some fully-connected layers are added to map the input data to the decision output, by learning the combinations of the extracted network features. Activation functions can be added after each fully-connected layer to introduce non-linearity and improve learning capability. Over a series of training epochs, a possibly nonlinear function between the input and output can be learnt with the CNN-based model. Notice that the output of the CNN-based model is represented by a column with length $\sum_k C_k + K \cdot (T_U + T_U \cdot 2)$, which includes:

- $\sum_k C_k$ numbers in the output show the caching status in the UAVs, with each number in range $[1, F]$ showing the index of the files being cached;
- $K \cdot T_U$ numbers in the output represent the K UAVs' delivery decisions, with each number in $\{0, 1\}$;
- $K \cdot T_U \cdot 2$ numbers in the output describe the trajectories of the K UAVs in T_U time slots, with each tuple of two numbers $(i,j) \in [1, N_{\text{row}}] \times [1, N_{\text{col}}]$ showing the location of a UAV.

VI. Performance Evaluation

A. Experiment Settings

In this section, we perform extensive trace-driven simulations to evaluate the proposed *LB-JCTO* scheme. We adopt the Didi Chuxing GAIA Initiative dataset, which includes taxi GPS traces within the second ring road in Xi'an [21]. The dataset logs key attributes of vehicular mobility including vehicle positions, vehicle ID, and corresponding timestamps. The traffic data is aggregated every 2-4 seconds from 1 October 2016 to 31 October 2016 (31 days). Specifically, we focus on a 2000 m $\times$ 2000 m square area within longitude range $(108.9169, 108.9300)$ and latitude range $(34.2290, 34.2466)$. For UAV communications, the zone parameters and corresponding data rates are calculated as shown in [1]. The default values of main parameters related to the UAVs are: $K = 2$, $H = 30$ m, $V = 15$ m/s, $E_{k,\max} = 50$ KJ, $\xi_{LoS} = 0.99$,

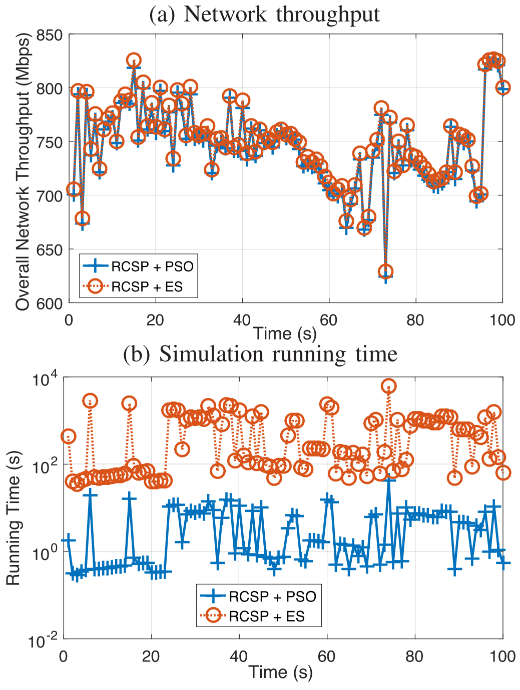


Fig. 6. Comparison between PSO- and ES-based algorithms.

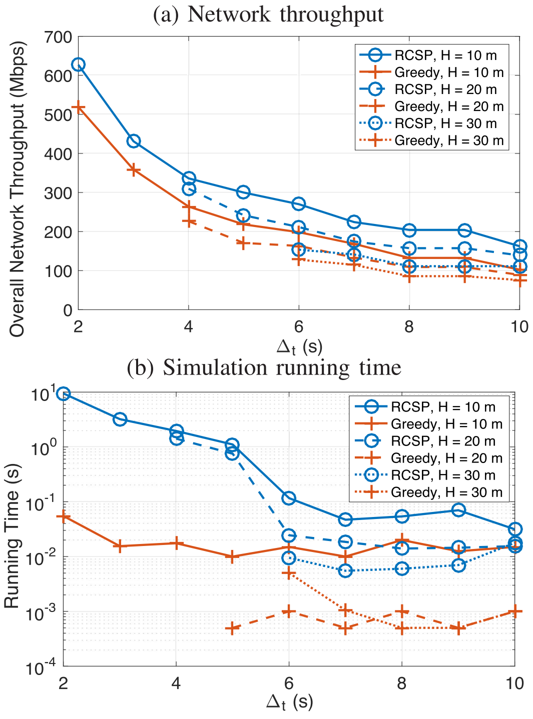


Fig. 7. Comparison between RCSP- and greedy-based algorithms.

and $\gamma_{\max} = 37.5$ dB. For cellular communications, the default parameters are: $\alpha = 3$, $B_C = 20$ MHz, $P_{C,\max} = 20$ W, and $\sigma^2 = 10^{-15}$ W/Hz. The Zipf exponent ξ is set to 0.7 and time slot length Δ_t is set to 5s unless otherwise specified. For learning model training, we implement a learning model with 11 layers detailed as follows. The model has three convolutional layers with channel sizes of 16, 32, and 64, respectively. The kernel size and strides for each convolutional layer are (3, 3) and (2, 2), respectively. After each convolutional layer, a max-pooling layer is added with pool size (2, 2). Four fully-connected layers are then added with 1024, 1024, 512, and 256 neurons, respectively, followed by one output layer. ReLU activation function is added after each layer to introduce non-linearity and improve learning capability.

B. Evaluation of CBTL-Based Offline Optimization

The following benchmark schemes are used for performance comparison to evaluate the performance of the offline optimization CBTL algorithm.

- *Exhaustive Search (ES) Algorithm:* An ES method is used in JCTO-CL to optimize the content placement decision.
- *Greedy Algorithm:* UAVs fly and deliver content greedily in JCTO-TDL optimization, where the UAVs always visit nearby locations with the best throughput performance in each step. The UAVs return to the starting points when the remaining energy is barely sufficient for returning.

Fig. 6 shows the network throughput and execution time with the PSO- and ES-based algorithms to solve the JCTO-CL problem. As can be seen in Fig. 6a, applying the PSO-based method in our CBTL algorithm achieves almost the same throughput performance as applying the ES method. However, the PSO-based method is much less time-consuming than the ES algorithm, as shown in Fig. 6b. Therefore, instead of learning from the optimal ES algorithm, the CNN-based model in this work utilizes the CBTL algorithm as the learning supervisor to save substantial time in data labelling such that more data can be used to train the model, which is more energy-efficient.

Fig. 7 shows the network throughput and corresponding execution time of applying RCSP-based and greedy-based algorithms with different values of Δ_t and H. A small Δ_t means a fine-grained CBTL optimization in time domain is conducted, whereas a large Δ_t (e.g., Δ_t equals the UAV endurance time and $T_U = 1$) is more related to the case with UAV deployment instead of trajectory design. Besides, with the simulation setting in Fig. 7, a lower UAV flying height corresponds to a smaller coverage area and indicates a refined division of the target area in spatial domain. It can be seen that, the network throughput and execution time both increase with smaller Δ_t and H, which can be attributed to the more sophisticated design in our CBTL algorithm. Focusing on the network throughput, we can conclude from Fig. 7a that applying RCSP-based method in our CBTL algorithm achieves a better performance. However, it has higher time complexity than the greedy algorithm, especially in the fine-grained optimization cases with small Δ_t and H, as shown in Fig. 7b. Concluding from Figs. 6-7, the proposed CBTL optimization algorithm can achieve near-optimal network throughput performance with time complexity slightly higher than the greedy-based algorithm.

Fig. 8 shows the impact of the number of UAVs K and the available UAV on-board energy $E_{k,\max}$ on the achievable network throughput. To further demonstrate the effectiveness of the proposed scheme, we also compare the case of UAV

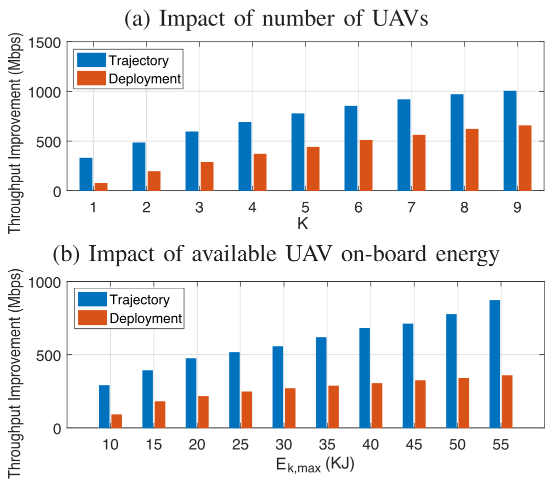


Fig. 8. Throughput performance vs. K and $E_{k,\max}$.

deployment. When UAVs are fixedly deployed, the optimal positions for UAVs are selected based on the network conditions at the first slot and the UAVs hover in those positions until energy depletion. As shown in Fig. 8a, the network throughput improvement increases with more UAVs dispatched into the system. However, throughput improvement introduced by each UAV diminishes with increasing K. The UAV trajectory design case outperforms the UAV deployment case, since the former is able to capture the network dynamics to ensure effective content delivery. In addition, as shown in Fig. 8b, a larger $E_{k,\max}$ leads to a higher network throughput for both UAV trajectory design and deployment cases since the UAVs can stay in the system longer. Notice that the throughput performance gap between the UAV trajectory and deployment cases increases with $E_{k,\max}$, since UAV trajectory design scheme is adaptive to the network variance and enables effective utilization of the energy to provide delivery services.

C. Evaluation of EI-Based CNN Learning Model

In this subsection, the achievable performance with the CNN-based learning model is evaluated. More specifically, we have trained multiple models by using different sets of trace data for performance comparison. For instance, "Model: [20, 50]", "Model: [50, 80]", "Model: $\geq$ 80", and "Model: General" are used to represent the models which are trained by using data where the number of vehicles in the target scenario is between 20 and 50, between 50 and 80, no less than 80, and unconstrained, respectively. For all the experiments, we adopt the trace data that is never used in the model training process to test the performance.

Fig. 9 shows the network throughput and execution time with the CNN-based learning model by using vehicle trace data during 9:15 AM $\sim$ 10:30 AM on 1 October 2016. The real-time number of vehicles in the target area is depicted in Fig. 9a. When using the general model to make online decisions, although the achieved network throughput outperforms that of the greedy-based algorithm, it is not as satisfactory as the RCSP-based offline CBTL algorithm. Therefore, it is not always the best option to train one model to incorporate all the features in different network conditions where the vehicle density varies from 40 to 120. To make comparison, in Fig. 9c, the "Model: $\geq$ 80" is used to make the JCTO decisions. Although the throughput performance is far from ideal in the beginning, the CNN-based learning model provides almost the same network throughput as the RCSP-based optimization algorithm when the vehicle density increases over 80 (shown within the red rectangle). More importantly, as shown in Fig. 9d, the CNN-based learning model takes much less time when compared with the CBTL-based offline optimization algorithms. Therefore, a well-trained CNN-based model can be utilized to make online decisions with a satisfactory network throughput performance and low-complexity. However, how to select and refine the models to apply to different network conditions requires further investigation, in order to achieve the best learning performance.

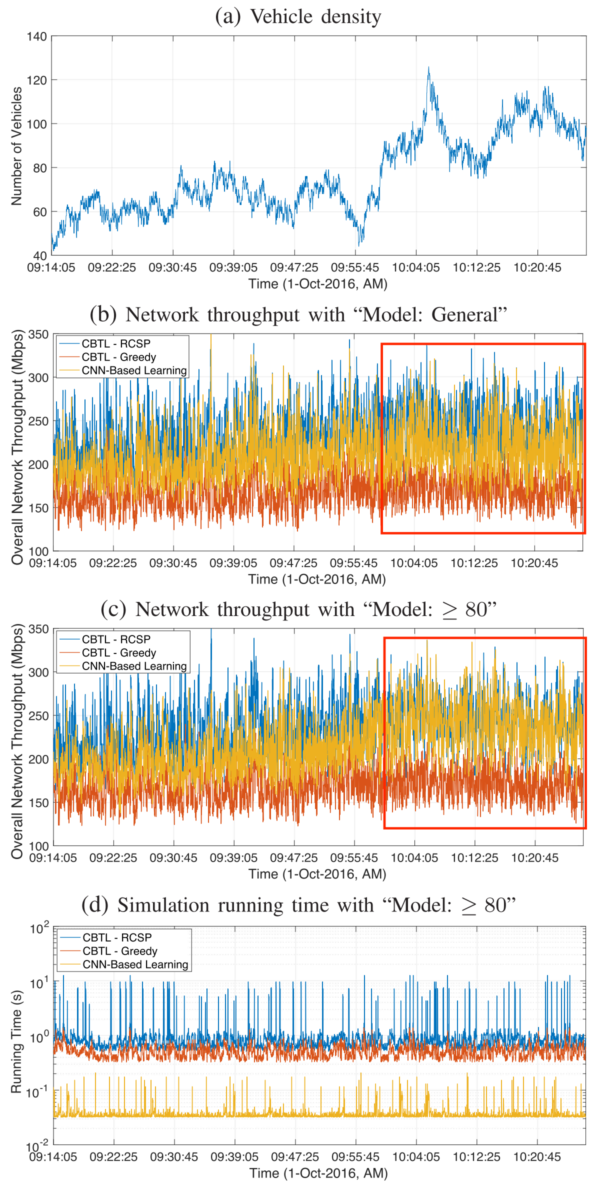


Fig. 9. Performance for CNN-based online decision model.

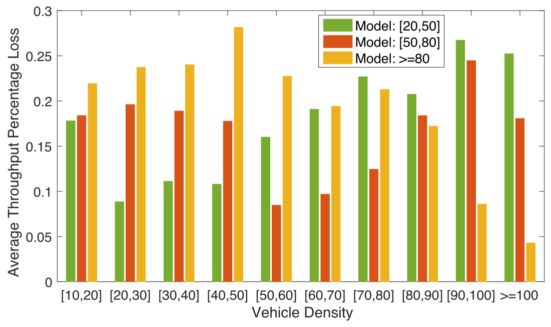


Fig. 10. Throughput performance with density-related CNN models.

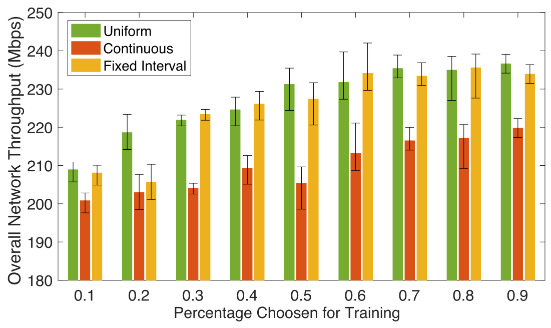


Fig. 11. Network throughput with different methods of training data selection.

Fig. 10 shows the achievable performance with different models under different density conditions. Specifically, average throughput percentage loss is used as a metric to evaluate the performance.[6] As shown in Fig. 10, "Model: [20, 50]" achieves the best performance (i.e., with the lowest throughput percentage losses) when applied to scenarios with vehicle densities falling in range [20, 50], but its performance is unsatisfactory in other cases. Similar results can be found for "Model: [50, 80]" and "Model: $\geq$ 80". Therefore, training multiple density-specified models and applying them in corresponding scenarios is a decent method to enhance model performance. However, a fine-grained model training can inflict significant training and storage cost. Besides, if the model granularity is too fine, the performance will be impacted since less training data is available. Therefore, the optimal number of learning models should be determined based on the throughput requirements, computing and storage capacities of devices, data availability, and so on.

Fig. 11 shows the impact of training data on the performance of the CNN-based models. The X axis represents the cases where 0.1 $\sim$ 0.9 of the available data is selected for model training. "Uniform" and "Continuous" indicate that the training data is selected uniformly and continuously from the available dataset, respectively. In "Fixed Interval" case, the training data is selected at fixed intervals, e.g., choosing the first two out of every ten pieces of data. As shown in Fig. 11, more data used for model training generally leads to a better throughput performance, since more information about the network features and regularities can be learnt with the CNN-based model. Among the three different methods of training data selection, the "Continuous" case achieves the worst performance since the training data is highly temporally correlated and it cannot learn all the network dynamics in time domain. On the other hand, the "Uniform" and "Fixed Interval" behave well since they are able to capture the network variance in different time intervals. Thus, to obtain a well-performed learning model, one should gather and select enough training data to learn as many potential features as possible.

[6]Throughput percentage loss is defined $(\hat{R} - \tilde{R})/\hat{R}$, where $\hat{R}$ and $\tilde{R}$ are the achievable network throughput of the CBTL-based offline optimization algorithm and the CNN-based learning model, respectively.

VII. Conclusion and Future Work

In this article, we have investigated the joint design of UAV caching and trajectory in highly dynamic vehicular networks. As the formulated JCTO problem is non-convex and difficult to solve in a timely manner, we have proposed *LB-JCTO* to offline optimize the JCTO problem and train a learning model at the edge to make online decisions. Particularly, in the offline stage, the CBTL algorithm has been devised to solve the JCTO problem. Then a CNN-based deep supervised learning model is trained to learn the CBTL algorithm, which can be used in the online stage to make fast decisions. Extensive trace-driven experiments have been carried out to demonstrate the efficiency of *LB-JCTO*. The problem formulation of JCTO and the optimization process of CBTL in this work can provide a theoretical basis for future studies related to the caching-enabled UAV systems. In addition, we believe the principle of offline optimization and learning for online decision can also be valuable for other complicated resource managements in future heterogeneous or space-air-ground integrated vehicular networks. For our future work, we will optimize the CNN-based model to further enhance the learning performance and make it adaptable to different vehicular environments.

References

[1] H. Wu, J. Chen, F. Lyu, L. Wang, and X. Shen, "Joint caching and trajectory design for cache-enabled UAV in vehicular networks," in *Proc. 11th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2019, pp. 1–6.
[2] R. He *et al.*, "Propagation channels of 5G millimeter-wave vehicle-to-vehicle communications: Recent advances and future challenges," *IEEE Veh. Technol. Mag.*, vol. 15, no. 1, pp. 16–26, Mar. 2020.
[3] F. Lyu *et al.*, "Characterizing urban vehicle-to-vehicle communications for reliable safety applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 6, pp. 2586–2602, Jun. 2020, doi: 10.1109/TITS.2019.2920813.
[4] C. Huang, A. F. Molisch, Y.-A. Geng, R. He, B. Ai, and Z. Zhong, "Trajectory-joint clustering algorithm for time-varying channel modeling," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 1041–1045, Jan. 2020.
[5] B. Jiang, J. Yang, H. Xu, H. Song, and G. Zheng, "Multimedia data throughput maximization in Internet-of-Things system based on optimization of cache-enabled UAV," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3525–3532, Apr. 2019.
[6] B. Li, Z. Fei, and Y. Zhang, "UAV communications for 5G and beyond: Recent advances and future trends," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2241–2263, Apr. 2019.
[7] L. Wang, H. Wu, Z. Han, P. Zhang, and H. V. Poor, "Multi-hop cooperative caching in social IoT using matching theory," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2127–2145, Apr. 2018.

[8] Z. Hu, Z. Zheng, L. Song, T. Wang, and X. Li, "UAV offloading: Spectrum trading contract design for UAV-assisted cellular networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 9, pp. 6093–6107, Sep. 2018.

[9] N. Cheng *et al.*, "Space/aerial-assisted computing offloading for IoT applications: A learning-based approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1117–1129, May 2019.

[10] V. Sharma, M. Bennis, and R. Kumar, "UAV-assisted heterogeneous networks for capacity enhancement," *IEEE Commun. Lett.*, vol. 20, no. 6, pp. 1207–1209, Jun. 2016.

[11] J. Lyu, Y. Zeng, and R. Zhang, "UAV-aided offloading for cellular hotspot," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 3988–4001, Jun. 2018.

[12] M. Samir, S. Sharafeddine, C. Assi, T. M. Nguyen, and A. Ghrayeb, "Trajectory planning and resource allocation of multiple UAVs for data delivery in vehicular networks," *IEEE Netw. Lett.*, vol. 1, no. 3, pp. 107–110, Sep. 2019.

[13] W. Li, L. Wang, and A. Fei, "Minimizing packet expiration loss with path planning in UAV-assisted data sensing," *IEEE Wireless Commun. Lett.*, vol. 8, no. 6, pp. 1520–1523, Dec. 2019.

[14] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-UAV enabled wireless networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 2109–2121, Mar. 2018.

[15] X. Xu, Y. Zeng, Y. L. Guan, and R. Zhang, "Overcoming endurance issue: UAV-enabled communications with proactive caching," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1231–1244, Jun. 2018.

[16] M. Chen, M. Mozaffari, W. Saad, C. Yin, M. Debbah, and C. S. Hong, "Caching in the sky: Proactive deployment of cache-enabled unmanned aerial vehicles for optimized quality-of-experience," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 5, pp. 1046–1061, May 2017.

[17] S. Chai and V. K. N. Lau, "Online trajectory and radio resource optimization of cache-enabled UAV wireless networks with content and energy recharging," *IEEE Trans. Signal Process.*, vol. 68, pp. 1286–1299, 2020.

[18] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," *Proc. IEEE*, vol. 107, no. 8, pp. 1738–1762, Aug. 2019.

[19] G. Y. Handler and I. Zang, "A dual algorithm for the constrained shortest path problem," *Networks*, vol. 10, no. 4, pp. 293–309, 1980.

[20] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[21] *Didi Chuxing GAIA Initiative*. Accessed: Jun. 2020. [Online]. Available: https://outreach.didichuxing.com/research/opendata/

[22] F. Ono, H. Ochiai, and R. Miura, "A wireless relay network based on unmanned aircraft system with rate optimization," *IEEE Trans. Wireless Commun.*, vol. 15, no. 11, pp. 7699–7708, Nov. 2016.

[23] E. Bastug *et al.*, "Big data meets telcos: A proactive caching perspective," *J. Commun. Netw.*, vol. 17, no. 6, pp. 549–557, Dec. 2015.

[24] Z. Huang, X. Xu, J. Ni, H. Zhu, and C. Wang, "Multimodal representation learning for recommendation in Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10675–10685, Dec. 2019.

[25] B. Chen and C. Yang, "Caching policy for cache-enabled D2D communications by learning user preference," *IEEE Trans. Commun.*, vol. 66, no. 12, pp. 6586–6601, Dec. 2018.

[26] W. Shi *et al.*, "Multi-drone 3-D trajectory planning and scheduling in drone-assisted radio access networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 8145–8158, Aug. 2019.

[27] W. Xu, W. Shi, F. Lyu, H. Zhou, N. Cheng, and X. Shen, "Throughput analysis of vehicular Internet access via roadside WiFi hotspot," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3980–3991, Apr. 2019.

[28] *IEEE Standard for Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, Standard 802.11, IEEE 802.11 Working Group, 2012.

[29] R. He, B. Ai, G. L. Stuber, G. Wang, and Z. Zhong, "Geometrical-based modeling for millimeter-wave MIMO mobile-to-mobile channels," *IEEE Trans. Veh. Technol.*, vol. 67, no. 4, pp. 2848–2863, Apr. 2018.

[30] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2329–2345, Apr. 2019.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, Stateline, NV, USA, Dec. 2012, pp. 1097–1105.

[32] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification," in *Proc. 29th AAAI Conf. Artif. Intell.*, Jan. 2015, pp. 2267–2273.

[33] E. B. Rodrigues, F. R. M. Lima, T. F. Maciel, and F. R. P. Cavalcanti, "Maximization of user satisfaction in OFDMA systems using utility-based resource allocation," *Wireless Commun. Mobile Comput.*, vol. 16, no. 4, pp. 376–392, Mar. 2016.

[34] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 881–892, Jul. 2002.

[35] F. Lyu *et al.*, "Online UAV scheduling towards throughput QoS guarantee for dynamic IoVs," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.

[36] L. Liu, Y. Song, H. Zhang, H. Ma, and A. V. Vasilakos, "Physarum optimization: A biology-inspired algorithm for the Steiner tree problem in networks," *IEEE Trans. Comput.*, vol. 64, no. 3, pp. 818–831, Mar. 2015.

[37] M. Clerc and J. Kennedy, "The particle swarm–explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. Evol. Comput.*, vol. 6, no. 1, pp. 58–73, Feb. 2002.

[38] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, vol. 37, Jul. 2015, pp. 448–456.

Huaqing Wu (Graduate Student Member, IEEE) received the B.E. and M.E. degrees from the Beijing University of Posts and Telecommunications, Beijing, China, in 2014 and 2017, respectively. She is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. Her current research interests include vehicular networks with emphasis on edge caching, resource allocation, and space-air-ground integrated networks.

Feng Lyu (Member, IEEE) received the B.S. degree in software engineering from Central South University, Changsha, China, in 2013, and the Ph.D. degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2018. From September 2018 to December 2019 and from October 2016 to October 2017, he worked as a Post-Doctoral Fellow and was a Visiting Ph.D. Student with the BBCR Group, Department of Electrical and Computer Engineering, University of Waterloo, Canada. He is currently a Professor at the School of Computer Science and Engineering, Central South University, Changsha. His research interests include vehicular networks, beyond 5G networks, big data measurement and application design, and cloud/edge computing. He is a member of the IEEE Computer Society, Communication Society, and Vehicular Technology Society.

Conghao Zhou (Graduate Student Member, IEEE) received the B.Sc. degree from Northeastern University, Shenyang, China, in 2017, and the M.Sc. degree from the University of Illinois at Chicago, Chicago, IL, USA, in 2018. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research interests include space-air-ground integration networks and machine learning in wireless networks.

Jiayin Chen received the B.E. and M.S. degrees from the School of Electronics and Information Engineering, Harbin Institute of Technology, Harbin, China, in 2014 and 2016, respectively. She is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. Her research interests include the area of vehicular networks and machine learning, with current focus on intelligent transport systems and big data.

Li Wang (Senior Member, IEEE) received the Ph.D. degree from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2009. She is currently a Full Professor with the School of Electronic Engineering, BUPT, where she heads High Performance Computing and Networking Laboratory. She is also a member of the Key Laboratory of the Universal Wireless Communications, Ministry of Education, China, and the Associate Dean of the School of Software Engineering, BUPT. She also held visiting positions with the School of Electrical and Computer Engineering, Georgia Tech, Atlanta, GA, USA, from December 2013 to January 2015, and with the Department of Signals and Systems, Chalmers University of Technology, Gothenburg, Sweden, from August 2015 to November 2015 and from July 2018 to August 2018. She has authored/coauthored almost 50 journal articles and two books. Her current research interests include wireless communications, distributed networking and storage, vehicular communications, social networks, and edge AI. She has served on TPC of multiple IEEE conferences, including the IEEE Infocom, the IEEE Globecom, International Conference on Communications, the IEEE Wireless Communications and Networking Conference, and the IEEE Vehicular Technology Conference in recent years. She was a recipient of the 2013 Beijing Young Elite Faculty for Higher Education Award, the best paper awards from several IEEE conferences, e.g., the IEEE ICCC 2017, the IEEE GLOBECOM 2018, and the IEEE WCSP 2019. She was also a recipient of the Beijing Technology Rising Star Award in 2018 and was selected as a Distinguished Young Investigator by the China Academy of Engineering in 2018. She was the Symposium Chair of the IEEE ICC 2019 on Cognitive Radio and Networks Symposium and the Tutorial Chair of the IEEE VTC 2019-fall. She also the Chair of the special interest group (SIG) on Social Behavior Driven Cognitive Radio Networks for the IEEE Technical Committee on Cognitive Networks. She currently serves on the Editorial Boards of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, the IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING, the IEEE ACCESS, *Computer Networks*, and *China Communications*.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990.

He is currently a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research focuses on network resource management, wireless network security, Internet of Things, 5G and beyond, and vehicular ad hoc and sensor networks. He is a registered Professional Engineer of Ontario, Canada, a Fellow of Engineering Institute of Canada, of Canadian Academy of Engineering, and of Royal Society of Canada, a Foreign Fellow of Chinese Academy of Engineering, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society. He received the R.A. Fessenden Award from the IEEE, Canada, in 2019, the Award of Merit from the Federation of Chinese Canadian Professionals (Ontario) presents in 2019, the James Evans Avant Garde Award from the IEEE Vehicular Technology Society in 2018, the Joseph LoCicero Award and the Education Award from the IEEE Communications Society in 2015 and in 2017, and the Technical Recognition Award from the Wireless Communications Technical Committee in 2019 and the AHSN Technical Committee in 2013. He has also received the Excellent Graduate Supervision Award from the University of Waterloo in 2006, and the Premier's Research Excellence Award (PREA) from the Province of Ontario, Canada, in 2003. He served as the Technical Program Committee Chair/Co-Chair for the IEEE Globecom'16, the IEEE Infocom'14, the IEEE VTC'10 Fall, the IEEE Globecom'07, the Symposia Chair for the IEEE ICC'10, and the Chair for the IEEE Communications Society Technical Committee on Wireless Communications. He is the Elected IEEE Communications Society Vice President for Technical & Educational Activities, the Vice President for Publications, the Member-at-Large on the Board of Governors, the Chair of the Distinguished Lecturer Selection Committee, and a member of IEEE Fellow Selection Committee. He was/is the Editor-in-Chief of the IEEE INTERNET OF THINGS JOURNAL, the IEEE NETWORK, *IET Communications*, and *Peer-to-Peer Networking and Applications*.