Proactive Edge Caching in Vehicular Networks: An Online Bandit Learning Approach

Proactive Edge Caching in Vehicular Networks: An Online Bandit Learning Approach

Qiao Wang, *Student Member, IEEE,* David Grace, *Senior Member, IEEE,*

***Abstract*—By bringing content close to end-users, proactive caching plays a vital role in improving the user experience in wireless networks. Caching content at the network edge proactively has been particularly effective in fast-changing vehicular networks. The objective of this paper is to address the proactive caching problem at the next roadside unit (RSU) in vehicular networks using reinforcement learning techniques. The paper proposes two proactive caching algorithms based on *multi-armed bandit* (MAB) learning, namely *non-contextual MAB-based* (MAB) and *contextual MAB-based* (cMAB). Additionally, the paper also investigates the uncertainty associated with proactive caching systems in the form of entropy with a specifically extended *Subjective Logic* framework, providing an insight into the underlying link between prediction accuracy and uncertainty. Two cities: Las Vegas, USA with grid road layout and Manchester, UK with more complex and historical layout, are considered in the simulation. Results have shown the generality of the proposed schemes in cities with different road layouts. Performance of the two proposed MAB-based systems is compared with two non-contextual baseline system: *Equal Probability-based* and *Probability-based*, and one contextual baseline system named *Compact Prediction Tree+ based*. Both proposed systems outperformed their counterparts. In terms of the prediction accuracy, cMAB has reached 75% and 80% accuracy in Las Vegas and Manchester respectively, and MAB reaches over 50% in both testing cities. Regarding the benefits to the vehicular network, cMAB and MAB perform similarly in both cities irrespective of the road layout. Particularly, the paper shows that on average 75% and 81% content fragments are proactively served with cMAB and over 50% with MAB in Las Vegas and Manchester, which is consistent with the prediction accuracy associated with the schemes.**

***Index Terms*—proactive edge caching, reinforcement learning, multi-armed bandit, mobility prediction, vehicular network, uncertainty, entropy, subjective logic**

I. Introduction

PAST decades have witnessed a rapid growth of the automobile industry and its economic and societal impacts continue to expand. With the rapid development in electronics and communications, vehicles will be able to communicate with each other, forming a large communication network, i.e., vehicular networks [1]. In addition, the upcoming era of autonomous vehicles means that vehicles will soon not only act as a simple means of transportation but also become moving entertainment centers where passengers are able to entertain themselves while traveling in the car [2]. However, on the other hand, this tendency also poses challenges. Content providers are facing unprecedented pressure on the quality of service (QoS) because of the link capacity and stability issues of general wireless networks [3], [4]. Another challenge of vehicular networks is their rapidly changing and dynamic nature. As fast-moving objects, vehicular users consequently experience short intermittent connectivity with roadside units (RSUs) more frequently than ordinary mobile users. This again is a factor giving rise to a degraded quality of experience (QoE). To improve QoS and QoE, caching has been a practical way applied by content providers to bring content closer to end users so that vehicles can fetch data from the nearest distributed server [5].

Recently, thanks to the development of mobile edge intelligence, mobile edge computing (MEC) units can be installed on RSUs, which enables them to perform both storage and computation functionalities [6], [7]. The immediate benefit by applying MEC-enabled RSUs is that vehicular users can reduce the frequency of accessing content from content providers, by directly accessing to caches in RSUs, hence lowering data services latency and alleviating backhaul traffic [8], [9]. Nevertheless, frequent link re-connections due to vehicles' high speed mobility as well as the fast fading of vehicle-to-roadside infrastructure (V2R) channels means that vehicles may not be able to finish consuming the content before leaving the connected RSU, meaning that they have to re-establish the connection to the remote server for the remaining parts at a drastically reduced data rate [4], [10]. This inevitably causes the user experience to be downgraded. Proactive caching has been recognized as a promising solution to the above issue. Pre-caching the desired content at the future RSU in advance allows vehicles to obtain immediate satisfaction after entering a new coverage area.

Effective proactive caching at the targeted RSU relies on effective prediction. For prediction purposes, the rapid development in machine learning has played an important role. In fact, predicting the next RSU as a proactive caching node is a direct application of reinforcement learning (RL) because every prediction is a decision to make. The goal of any RL problem is to map perceived *states* to *actions* by learning a *policy* function. Nevertheless, in systems that do not have to be represented by states, the learning problems become stateless decision problems and the learning agent becomes stateless, which significantly reduces the number of trials needed to learn a mature strategy and speed up the learning process [11]. This is of great help in a dynamically changing vehicular environment. *Multi-armed bandit* (MAB) problems [12], [13] are basic instances of RL problems or to be specific, single state model-free RL problems, where a learning agent does not have to build up a model of the environment. This feature makes it

Q. Wang and D. Grace are with the Communication Technologies Research Group, Department of Electronic Engineering, University of York, York YO10 5DD, United Kingdom (email: qiao.wang@york.ac.uk)

efficient in dealing with the variable vehicular environment. It has also attracted significant attention in various applications, from recommendation systems and information retrieval to healthcare and finance, thanks to its excellent performance combined with certain attractive properties, such as learning from less feedback [14]. In a bandit problem, the agent, i.e., the bandit, takes an action to achieve an immediate reward without states being involved, aiming to maximize the total amount of rewards.

Uncertainty is inextricably linked to learning algorithms and their models and is an important concept in machine learning methodology [15]. Assessing and quantifying uncertainty helps us to understand more precisely the benefits that models can bring. Reducing uncertainty will inevitably give us more accurate prediction results. *Subjective logic* [16], [17] has emerged as an effective method for uncertainty evaluation. This formalism allows us to express specific forms of probability distributions by generating a *multinomial opinion* over a discrete set of elements. It provides a concise formalism to represent Dirichlet-multinomial and Dirichlet-categorical models [18] and therefore, the opinion induces a categorical distribution over the element set that allows evaluation of the overall uncertainty as the entropy of the distribution. This model has also been recently used to assess uncertainties in deep networks [19], [20]. This will be further developed in this paper.

The purpose of the paper is to address the problem of effective proactive caching at the next RSU in vehicular networks through MAB problem model. We treat this as a decision-making problem and investigate the feasibility and prediction performance of bandit learning in such situations. To this end, we designed two algorithms: *non-contextual MAB* and *contextual MAB* in vehicular networks by modelling the problem as a bandit problem. The motivation of exploring these two algorithms is to further investigate the benefit on prediction performance by introducing *context* in contextual MAB. Another purpose of the work is to investigate the uncertainty behind the proposed proactive systems with Subjective Logic framework. The motivation behind this is that uncertainty is inseparably connected to learning algorithms, and we aim to verify and support the superiority of the proposed systems from the theoretical viewpoint of uncertainty. Our work fills the void of using multi-independent-agent MAB models to solve proactive caching problems in such scenarios. Specifically, the main contributions of the paper can be summarized as follows:

- We design non-contextual MAB-based and contextual MAB-based algorithms in an online learning way to address proactive caching at the next RSU. Despite the many applications of MAB in a range of fields such as ad placement and packets routing, we show how it can be used for the first time in pre-caching problems. We aim to attract more attention of the research community to use the model-free MAB technique in fast-changing vehicular environment.
- We implement MAB technique in a distributed way on individual RSUs to realize instant learning and prediction, whilst previous similar works were based on centralized approaches. Besides, the performance comparison with the baseline systems shows the advantages of using the MAB reinforcement learning technique in solving proactive caching problem. Particularly, the contextual MAB with only single context required shows a faster convergence and better accuracy than conventional sequence prediction model (i.e., Compact Prediction Tree+ based).
- We extend the subjective logic framework specifically to proactive caching systems to analyze, using entropy, the overall uncertainty behind the bandit problem based systems as well as two baseline systems. By doing this, we aim to investigate the uncertainty variation and its correlation with prediction accuracy of different proactive caching systems.
- We experiment with the test data of two cities with significantly differing characteristics, Las Vegas and Manchester, from USA and UK respectively. The results show the scalability and adaptability of MAB-based approaches in proactive caching problems with quite different road layouts.

The rest of the paper is structured as follows. The related works regarding proactive caching and MAB applications in vehicular networks are summarized in Section II. Section III mainly discusses the network architecture and system model. The proposed algorithms are introduced in Section IV, and in Section V, the uncertainty analysis model is provided. Section VI shows the simulations results. Section VII discusses the theoretical analysis, time complexity and convergence of the proposed algorithms. Section VIII concludes the paper.

II. Related Work

This section discusses some relevant studies and is divided into two parts: *Proactive Caching in Vehicular Networks* and *Reinforcement Learning and Uncertainty*.

A. Proactive Caching in Vehicular Networks

Proactive caching in the literature can roughly be categorized as what to cache and where to cache. The former mainly relies on content popularity prediction. For example, the authors in [21] proposed a two-level prediction model for predicting video popularity to pre-cache popular videos in the content delivery network and in the survey [22], the authors summarized the studies on popularity-based video caching techniques in cache-enabled networks. However, most of popularity prediction methods require RSUs to collect vehicles' data which may contain sensitive information. We believe that this will become increasingly difficult for network operators given the increasing restrictions on security and privacy. In addition, they are not very effective because vehicles are fast moving objects and this cause validity issues of the prediction. The latter, on the other hand, depends on how well the system is able to anticipate where a vehicle is going. This is more manageable and applicable for network operators and also essential in the rapidly changing vehicular environment. Therefore, we are interested in where to cache, as our proposed methods address the problem of proactive caching at the next possible RSU.

Predicting where to cache via mobility prediction may include caching at vehicle nodes and RSU nodes. The very recent work in [23] proposed a relay strategy where the RSU separate the connecting vehicles into clusters through trajectory prediction and pre-caching content at multiple vehicles in the cluster to increase the communication time with the requesting vehicle. An earlier work on vehicle node edge caching was by Yao *et al.* [24] used a tree-based Markov chain model - Prediction-based on Partial Matching (PPM) in a Vehicular Content-Centric Network to predict vehicles' probability of reaching different hot spot regions and they select a vehicle node with longer sojourn time in a hot spot region as the caching node. Whilst both works focused on vehicle node caching, they both require numerous data for offline training and the vehicles needed to send their trajectories to every RSU they visit in [24], which inevitably raised concerns on transmission overhead and privacy issues.

In terms of RSU node edge caching, Khelifi *et al.* [25] put forward a proactive caching scheme based on vehicular mobility prediction on top of a Named Data Networking architecture. The authors used Long Short Time Memory (LSTM) to first predict the moving direction of the vehicle and then estimated the next possible RSU, instead of directly predicting the next RSU. Similarly, in terms of the next RSU selection, the work in [4] also used LSTM to predict the direction of a vehicle by training realistic traffic data, after which Q-learning is applied to determine how much to cache. Both of the previous studies using LSTM require vast amount of offline training with labeled data for a proper model. This is the first fundamental difference from our work where we concentrate on online learning where prediction accuracy improves based on trial and error. Besides, the prediction model in these two works is considered in a centralized way, that is prediction is made by a central node for a vehicle after the offline training stage. In contrast, ours has considered a distributed system where RSUs are learning and predicting independently, which is the second substantial difference. The authors of [26] proposed a sequence-prediction based proactive caching system to address the problem. Their model is based on a sequence prediction algorithm, Compact Prediction Tree+ [27], by training vehicle-specified simulated traffic traces. Similarly, this work also requires training sequences of individual vehicles offline.

B. Reinforcement Learning and Uncertainty

One of the most widely used model-free RL techniques is Q-Learning proposed by Watkins [28]. It is an off-policy method where the policy is updated based on the best possible future scenario, in contrast to its on-policy counterpart State–action–reward–state–action (SARSA) [29] that takes into account what actually happens after an action is taken for policy updating. The Deep Q-Learning approach, combining Q-Learning with deep neural networks, has recently become a popular technique. Based on deep Q-networks [30], it resolves the impracticability to use the traditional tabular Q-Learning method in environments consisting of large state spaces with high dimensions, but it is also limited to supporting discrete action and state spaces [31]. Nevertheless, a significant issue of such methods is their adaptability and applicability in realistic wireless communications environments with features such as dynamic traffic levels, dynamic network topology, and so on.

Given a highly dynamic vehicular environment and discrete action set in our problem, it is applicable and practical to consider it as stateless instead of its classical counterpart, which can dramatically reduce the number of Q-values needed for estimation by the learning agent, thereby potentially reducing the number of trials needed for it to learn a mature strategy and improving the adaptability of RL-based cognitive devices (e.g., RSUs). The MAB model is a single-state model [13] with no state transitions (i.e., stateless). While it has been widely used and proven to be effective in areas such as ad placement, computer game-playing, etc., its application in vehicular networks seems to be more limited. Dai *et al.* [32] proposed a multi-armed bandit learning algorithm called Utility-table based Learning to solve the distributed task assignment problem in a MEC-empowered vehicular network. The work in [33] focused on task caching problems in the edge cloud. The authors proposed an intelligent task caching algorithm based on a multi-armed bandit algorithm and evaluated its benefits in task latency performance. Authors of [34] discussed the potential of using a MAB problem in future 5G small-cell networks as well as its applications and future research directions. A detailed example of using a MAB model for energy-efficient small cell activation in 5G networks has been provided in [34]. Xu *et al.* [35] investigated collaborative caching problems in small-cell networks by learning learn the cache strategies directly at small base stations online by utilizing multi-agent MAB.

The uncertainty associated with learning models has attracted significant attention. *Subjective logic* first proposed in [16] has emerged as an effective method for uncertainty evaluation. The work in [18] used a subjective logic framework to solve bandit problems, where the action selection is based on sampling the multinomial opinion over the action set. They quantified the overall uncertainty of the proposed system with the entropy of the categorical distribution. The authors in [36] argued that Beta distribution and subjective logic are isomorphic in terms of fusion, while finding the equivalence between uncertainty and entropy of Beta models. It has also been used for assessing uncertainty in deep networks as studied in [19] and [20].

Despite all the benefits of active caching for user experience and the potential of the MAB , to the best of our knowledge, there are no studies focused on the problem of applying MAB to select the next RSU for proactive caching. In addition, no one has studied the uncertainty of these systems so far. We believe this area is worth more investigations.

III. Network Architecture and Problem Statement

In this paper, we consider a vehicular network with caches enabled at edge RSUs as in Fig. 1. Vehicles that enters the network frequently request and download the content they are interested in. The RSUs are intelligent so they are able to learn and predict the next possible RSU a vehicle will connect to and send a pre-caching request to that RSU. In this way, the backhaul traffic can be reduced and the vehicular users may benefit from this in terms of their network experience.

Consider an area G in an urban area deployed with N RSUs in a set $\mathcal{R} = \{r_1, r_2, ..., r_N\}$ and a content database $\mathcal{C} = \{c_1, c_2, ..., c_K\}$ providing K pieces of content of various sizes, represented by f_{c_k}, $c_k \in \mathcal{C}$ fragments each of which is of size F_c. There may have $\mathcal{L} = \{1, 2, ..., L\}$ residential areas and workplaces in G where vehicular users from the set $\mathcal{V} = \{v_1, v_2, ..., v_M\}$ travel from and to on a daily basis. Each RSU $r_i \in \mathcal{R}$ is equipped with mobile edge computing (MEC) that is capable of both computing and caching so that the RSU can learn from vehicles' traces for next RSU prediction and proactively cache the proper content in need. Despite the functionalities, the computing resource consumption and content replacement in caches is out of the scope of the present work. There may be m neighboring RSUs of r_i and hence the next potential pre-caching node is selected from the neighbors. Besides, there is a central server that is responsible for a few connecting RSUs in a distributed way so that the result of proactive caching can be fed back to them to achieve accurate learning model.

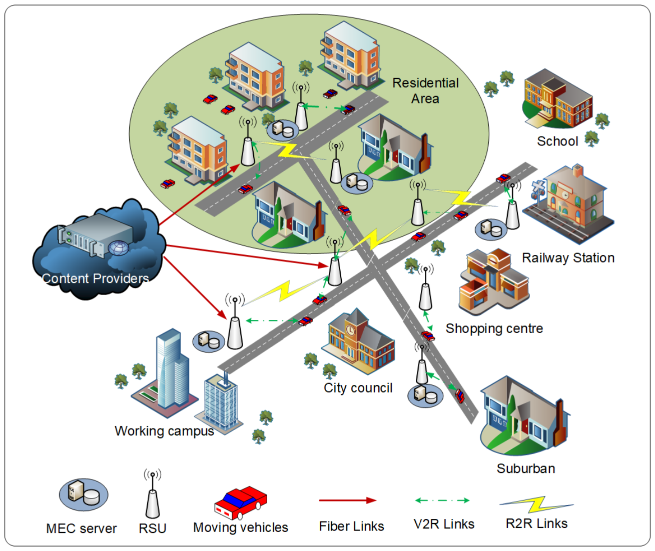


Fig. 1: Illustration of the architecture of cache-enabled vehicular network

The communication model characterizes only the key elements needed to study the problem, given that the interest of the work is to find where to cache accurately. Essentially, a vehicle $v_j \in \mathcal{V}$ in the network is associated with the closest RSU. v_j may request a type of content c_k from its currently connecting RSU r_i in a random way. r_i then starts to transmit c_k to v_j from its cache directly or through the content provider in the backhaul network or both, depending on the dwelling time and data rate. In order to focus on the proactive caching task at the next RSU, following assumptions are made: 1) the underlying issues arising at the physical and MAC layers e.g., packet loss, interference, re-transmissions are not considered

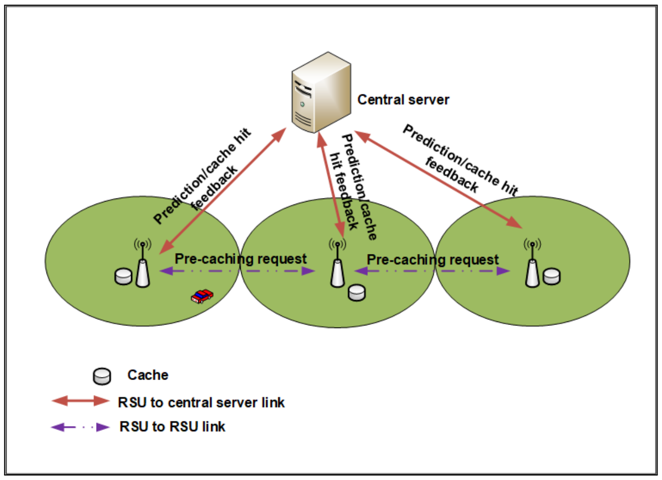

Central server
Prediction/cache hit feedback
Prediction/cache hit feedback
Prediction/cache hit feedback
Pre-caching request
Pre-caching request
Cache
RSU to central server link
RSU to RSU link

Fig. 2: Distributed Structure of Proactive Caching System

in vehicular communications so, the transmission rate e is a constant; 2) the dwell time of the vehicles in the coverage area of a RSU is extracted from the test trace being simulated and is known so that the number of content fragments can be derived; 3) the system is completely proactive, meaning that reactive caching is not enabled. When the RSU r_i evaluates that it is unlikely to transmit the entire content to v_j, it predicts the next RSU and requests it to perform proactive caching on the remaining fragments f_r. A transmission delay μ would be introduced if f_r is not found in the actual next RSU, via $\frac{f_r \times F_c}{\omega}$ where ω is the backhaul link rate. Additionally we assume that a vehicle does not request new content until it finishes consuming the current one [26] and the system keeps a record of content consumption so that when handover occurs, vehicles continue to download the remaining of its previously requesting content.

A typical proactive caching scenario is shown in a structure diagram in Fig. 2. The current associated RSU may face a prediction decision involving a few neighboring RSUs when a vehicle requests content from it. Assuming the vehicle $\hat{v}$ sends its request for a content $\hat{c}$ right after it enters the coverage of the RSU $\hat{r}$, $\hat{r}$ serves $\hat{v}$ with its cached fragments of $\hat{c}$ if available or through content provider in the backhaul network otherwise, which would cause some delay, or both ways. Whichever way, $\hat{r}$ evaluates how many fragments of $\hat{c}$ it can transmit to $\hat{v}$ before handover. If $\hat{c}$ has a relatively small size and/or $\hat{v}$ would stay connected rather long, $\hat{c}$ can be fully transmitted (consumed) and therefore, no proactive caching in the next RSU is required. Otherwise, $\hat{r}$ will have to coordinate one of its neighbors by the prediction/learning algorithm for pre-caching the remaining of $\hat{c}$ via *proactive caching request* message. This is where proactive caching happens. The role of the central server that connects multiple RSUs is to transmit *prediction/cache hit feedback* message which acts as *rewards* in the learning algorithms.

Problem Statement: In a vehicular network with proactive caching enabled, the goal of this feature is to enable seamless delivery of content to vehicular users. The effective cache hit is the way to achieve this goal and is based on accurate predictions of the next RSU. How to select the next possible

RSU node for a vehicle as accurately as possible through trial-and-error based reinforcement learning (i.e., the MAB algorithm) is the essential problem covered in this paper. Besides, how the uncertainty of prediction associated with proactive systems evolves during learning is another problem this paper studies.

IV. Algorithm Design

This section will briefly introduce some background of *multi-armed bandit problem*, following which the learning algorithms designed for proactive caching will be discussed.

A. Multi-armed Bandit Problem

The multi-armed bandit (MAB) problem, sometimes also known as *k*-armed bandit problem, is a special instance of reinforcement learning (RL). Different from a traditional or a full RL problem where a learning agent may have multiple states associated with the environment (e.g., positions in a game), it only has a single state in MAB problem [13] (i.e., no state transitions). From this perspective, MAB is essentially identical to *stateless Q-Learning* [37] and can also be treated as a model-free reinforcement learning technique. A well-known scenario of the bandit problem is where a gambler in a casino sits in front of a slot machine with one or multiple arms (referred to as a one-arm bandit and k-armed bandit respectively) and tries to get payoffs by pulling the arm(s). The ultimate goal of the gambler is to achieve the highest cumulative rewards through learning the inherent reward pattern of each lever and gradually concentrating on the best lever. During the learning process, the gambler will face the *exploration-exploitation* dilemma [38]: where the gambler tries out the potential arms that may return high payoffs (exploration) or pulls the arm that has yielded the highest reward from the past experiments (exploitation). This is a non-trivial process and carefully balancing exploration and exploitation is crucial in MAB problems.

A MAB problem can be formally given as a tuple [18]: $\langle \mathcal{A}, \mathcal{R} \rangle$, where $\mathcal{A} = \{a_1, a_2, ..., a_k\}$ is the a set of k actions (i.e., arms) and $\mathcal{R} = \{\theta_1, \theta_2, ..., \theta_k\}$ associates action a_i with its reward probability distribution defined by θ_i. There are a number of variants of MAB problems and it is out of the scope of the paper to cover all of them. Therefore, in the following a canonical example of MAB - the Bernoulli bandit problem and the contextual bandit will be discussed as they are closely related to the problem here and the proposed learning algorithm for proactive caching. In addition, the approaches to resolve exploration-exploitation dilemma in MAB problems are plenty such as ϵ-greedy, upper-confidence bound algorithm, Thompson sampling [38], etc. The aim of this paper is not to find out a sophisticated way to balance exploration and exploitation so the most straightforward ϵ-greedy is adopted here.

1) Bernoulli multi-armed bandit: Consider a *k*-armed bandit problem $\langle \mathcal{A}, \mathcal{R} \rangle$. The agent takes actions from action set $\mathcal{A}$ and any action played will generate a success (reward 1) or failure (reward 0). Action $a \in \mathcal{A}$ produces a success with probability $\theta \in \mathcal{R}$. In other words, for an action a,

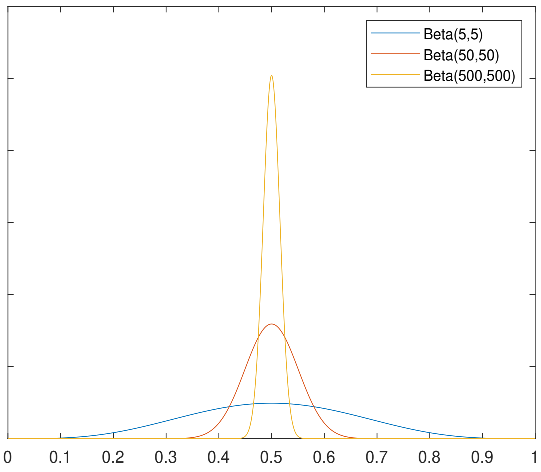


Fig. 3: An example of the sample-average process shown with Beta distribution

a reward $r = 1$ is produced with probability θ and $r = 0$ with probability $1 - \theta$. In this case, θ can be viewed as the expected reward of taking action a, is unknown to the agent and is invariant in a *stationary* MAB problem. One natural way to estimate such θ is to use *sample-average* method [13] by averaging the rewards actually received. The estimated value of θ at time step t can be denoted as:

$$\begin{aligned} Q_t(a) &= \frac{\text{sum of rewards when } a \text{ taken prior to t}}{\text{number of times } a \text{ taken prior to t}} \\ &= \frac{\sum_{i=1}^{t-1} r_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}} \end{aligned} \tag{1}$$

where A_i is the action taken at time i, $\mathbb{1}_{condition}$ is 1 if *condition* is true and 0 if not, and $r_i = \{1, 0\}$ is the reward of i-th selection of action a. According to the law of large numbers, Equation (1) converges to θ as the the denominator tends to infinity. A more intuitive way to illustrate this is through the probability density function of $Beta(\alpha = successes, \beta = failures)$ distribution as shown in Fig. 3. Consider a 95% confidence interval, in the late stage of sample-average process after 1000 trials with 500 successes and 500 failures, the range that captures the true probability θ is $[0.469,\ 0.531]$, i.e., $P\,(0.469 < \theta < 0.531) = 0.95$. However, the intermediate stage with 100 trials (50 successes and 50 failures) returns a much wider range of $[0.403,\ 0.597]$ for the same 95% confidence interval and the initial stage with only 10 trials gives an even wider range of $[0.212,\ 0.788]$. Thus, the more trials, the more certain one can be about the approximation to the true probability θ. By taking the proper action with associated action-selection strategy (e.g., ϵ-greedy), it is also to maximize the cumulative rewards $\sum_{t=0}^{T} r^t$ where T is the given time horizon.

2) Contextual bandits: As an extension of the above multi-armed problem, the contextual bandit problem associates actions with side information or context [39]. In such problems, the agent aims to learn a policy that maps contexts to actions, that is, $\pi(a_i \mid s_j)$ where s_j is one of the contexts. Another viewpoint is that it now consists of multiple independent MAB tasks associated with contexts, and the agent aims to learn the best policy under various contexts. Every time an agent is assigned a MAB task (possibly with a certain probability), it will be given a "clue" (i.e., context) and learn what the best action is under this clue. In general, the agent can do better with the presence of context information that distinguishes one bandit problem with another [13]. Despite the fact that contextual bandit problems involve learning policies, they still resemble the general MAB tasks, as the action taken only affects the immediate reward, and makes no difference to the next situations, as well as their rewards. Therefore, it is an intermediate between the MAB problem and the full RL problem.

B. MAB-based Proactive Caching Algorithms

The primary focus of the proactive caching problem in this paper is where to pre-store relevant content in the immediate future. Therefore, it is vital for a RSU to predict as accurately as possible the next potential RSU a vehicle is about to handover to. Intuitively, this may not seem to be closely related to a MAB problem so, in the following we will first demonstrate how to match them together.

1) Mapping proactive caching to MAB problems: As discussed in the last subsection, a MAB problem is composed of action set, rewards and agent as well as context in a contextual bandit problem. Here, an agent aims to maximize its cumulative rewards by taking appropriate actions from the action set in a given period time. Regarding the next-RSU proactive caching scenario in vehicular networks, a RSU assists a vehicle to successfully hit the content that was previously being transmitted. They resemble each other in terms of node selection and success or failure (reward). Therefore, we model the pre-caching problem as a MAB problem using the following mappings:

- **RSU as bandit learning agent**: Any RSU in the vehicular network acts as a learning agent, and its neighboring RSUs are equivalent to its actions. Predicting the next RSU as a proactive caching node is actually making a decision on one of the agent RSU's neighbors.
- **Stateless RSU**: In general, the state of a reinforcement learning agent is associated with the environment. Since the interaction of a RSU with the vehicular environment can be extremely dynamic and complicated to represent, the single state feature of MAB resolves this problem. In other words, an agent RSU is single state or stateless which means that it does not transfer to a new state by taking an action.
- **Action selection as next RSU prediction**: The agent RSU will either exploit its current knowledge to select the *greedy* action/neighbor or explore other non-greedy actions that may return a higher reward depending on the exploration-exploitation scheme adopted.
- **Reward generation**: When handover happens, the system will return a reward to the previous agent RSU. This is achieved by determining whether there is pre-cached content in the RSU after the handover, or alternatively whether the RSU is the previously predicted one. The reward in return helps an agent RSU compute the estimated values of its actions.
- **Previous RSU as context**: The agent RSU may also make use of contexts for its action selection as in a *contextual bandit problem*. By identifying the previous RSUs that the visiting vehicles coming from as contexts, it can map such contexts into various bandit tasks and perform more effective learning. Technical details about the contextual information will be covered shortly.

In a vehicular system with multiple RSUs, the problem becomes a *multi-player, multi-armed bandit* problem where each individual RSU is an independent player and learns its own best action or best policy. On this basis, we designed two algorithms to address proactive caching: *non-contextual (Bernoulli) MAB* and *contextual MAB*, and the detailed design will be elaborated in the following subsection.

2) Addressing Proactive Caching with bandit learning: We will elaborate the two bandit learning algorithms that address proactive caching from three aspects: *action selection and value estimation*, *reward function*, and *context information*.

a) **Action selection and value estimation** Two critical elements in MAB problems are action selection and action value update. Given the estimated action values $Q(a)$ of actions in $\mathcal{A}$, the ϵ-greedy method is used to make a selection: the best action is selected with probability of $1-\epsilon$; otherwise, actions will be selected randomly with a small probability ϵ regardless of their action values.

$$A_t = \begin{cases} \arg\max_a Q(a), & 1-\epsilon \\ random, & \epsilon \end{cases} \tag{2}$$

Another important method is the action value estimation, also known as *action-value method* in the literature. Recall in a Bernoulli multi-armed bandit, the true success probability θ of action a is its expected reward, defined as $\theta \doteq E\left[r \mid A=a\right]$. The sample-average approximation method for action-value estimation shown in Equation (1) can have a more compact representation with incremental implementation [13]. For action a which has been selected for n times, the estimated value is:

$$\begin{aligned} Q_{n+1} &= \frac{1}{n}\cdot\sum_{i=1}^{n} r_i \\ &= \frac{1}{n}\left(r_n + (n-1)\frac{1}{n-1}\sum_{i=1}^{n-1} r_i\right) \\ &= \frac{1}{n}\left(r_n + (n-1)Q_n\right) \\ &= Q_n + \frac{1}{n}\left(r_n - Q_n\right) \end{aligned} \tag{3}$$

An important parameter in the incremental value updating rule of Equation (3) is $\frac{1}{n}$, the *step-size*. As can be noted from the Equation 3, this step-size declines as n grows. In fact, this is fairly effective in a ***stationary*** bandit problem where the reward probabilities (i.e., θ) remain unchanged over time. Vehicular networks, however, are dynamic environment with varying traffic density and may result in a ***non-stationary*** bandit problem. Therefore, recent rewards should be given more weights when updating action values. This is often achieved using a constant step-size denoted with $\alpha \in [0, 1]$ and Equation (3) therefore becomes:

$$Q_{n+1} = Q_n + \alpha(r_n - Q_n) \tag{4}$$

A more general form of Equation (4) that is adopted in our algorithms is:

$$Q(a) \leftarrow (1 - \alpha)Q(a) + \alpha r \tag{5}$$

where $Q(a)$ is the quality value of action a, named **Q**-value as in **Q**-learning [11], [37], r is the reward associated with the most recent trial and is determined by a reward function, and $\alpha \in [0, 1]$ is the step-size or *learning rate* .

b) **Reward function** The reward function R is used to generate a reward associated with the action taken previously when an outcome is observed. Given an action a taken at time step t and the observed outcome as b (may occur immediately), its reward can be computed with $r_t = R(b)$. In a Bernoulli MAB problem discussed earlier, the reward function R is actually the outcome itself (1 or 0), meaning that $r_t = R(b) = b$. In order to introduce punishment for a caching miss, we referred to the reward function that has been successfully applied in the Dynamic Spectrum Access problem in [11]:

$$R(b) = \begin{cases} 1, & b = \text{True} \\ -1, & b = \text{False} \end{cases} \tag{6}$$

As mentioned in “Mapping proactive caching to MAB problems”, the outcome b is determined by observing whether a vehicle switches to the predicted RSU, equivalent to a cache hit or miss if pre-caching request was sent to the RSU. The relevant reward will then be generated with Equation (6) and fed back to the earlier decision-making RSU. With Equation (3), (5) and (6), the learning agent aims to update its estimate of each action $Q(a) = E[r_t]$, make an action selection and maximize its cumulative rewards $max \sum r_t$.

Notably, due to the constant α adopted in Equation (5) and the negative reward introduced in Equation (6), $Q(a) \in [-1, 1]$ is no longer a probability i.e., it is not an estimate of θ as in the sample-average method (Equation (1)), but directly represents the expected reward of the action a.

c) **Contextual information** The above methods for updating actions' **Q**-values, selection, and reward function can be applied to both non-contextual and contextual bandit problems. The difference is that the agent in a contextual MAB problem maps contextual situations to its specific action values. In other words, it associates a specific Q-table with each individual situation and aims to learn a policy under different contexts. There are limitations on the performance of the agent in a non-contextual MAB problem. For example, vehicles that visit one agent RSU might go to its neighbors in a similar proportion, which means that its actions might have similar values, and this results in back and forth selections between its actions. However, these vehicles may also come from various directions and if the agent RSU is able to distinguish and make use of such information, and split to separate bandit tasks, it is likely to improve the overall accumulated rewards. This is the major motivation to propose contextual MAB-based algorithm.

Specifically, the context we introduced on top of a non-contextual MAB-based algorithm is the previous RSU that a vehicle connected to prior to the current agent RSU. Once a vehicle connects to a RSU and starts to request content from it, the agent RSU needs to predict the next RSU (action selection) and inform it to pre-cache the needed content if necessary. In the non-contextual MAB, the agent RSU makes this decision according to the **Q**-values of its actions. In contextual MAB, however, it will first identify the previous RSU that the vehicle came from as context and learn the action values associated with it so that decisions are properly made under such context. The equivalent equations to Eq. 3 and Eq. 5 for action selection and action-value updating in contextual MAB model become:

$$A_t = \begin{cases} \arg\max_a Q_t(a \mid s_j), & 1 - \epsilon \\ random, & \epsilon \end{cases} \tag{7}$$

$$Q(a \mid s_j) \leftarrow (1 - \alpha)Q(a \mid s_j) + \alpha r \tag{8}$$

where s_j is the detected context at *t*.

Algorithm 1: Non-contextual multi-armed bandit

```
Initialization (if not done): For RSU $m \in \mathcal{M}$ with the
  number of actions (RSU neighbors) $\mathcal{A}_m$, their
  Q-values are initialized to $Q(a) = 0$ for $a \in \mathcal{A}_m$ ;
while not the end of the test do
    if Content transmission is happening whilst in RSU
      $m$ then
        Predict the next RSU by:
        $a^* \leftarrow$ selection decision based on Eq. (3);
        Precaching content at $a^*$ if needed;
    end
    if Handover happens then
        $r^* \leftarrow$ observe the reward according to Eq. (6);
        Update $Q(a^*)$ with Eq. (5);
    end
end
```

As mentioned earlier, being able to distinguish the incoming directions could help resolve the dilemma in a non-contextual MAB problem. Prior RSUs can be straightforwardly accessed and used as a reference to such directions compared to other sorts of information (e.g., road information, vehicle angel, etc.), and this enables the agent RSU to solve separate bandit tasks associated with them, thereby guaranteeing a more effective policy learned than in a non-contextual MAB problem.

Algorithm 2: Contextual multi-armed bandit

```
Initialization (if not done): For RSU $m \in \mathcal{M}$ with the
  number of actions (RSU neighbors) $\mathcal{A}_m$, their
  Q-values are initialized to $Q(a) = 0$ for $a \in \mathcal{A}_m$ ;
while not the end of the test do
    if Content transmission is happening whilst in RSU
      $m$ then
        $s \leftarrow$ detect the previous RSU $s$ before $m$;
        if $s$ is a new detection then
            Create an entry of $s$ to its action values;
            Initialize $Q(a \mid s) = 0$ for $a \in \mathcal{A}_m$;
        end
        Predict the next RSU by:
        $(a^* \mid s) \leftarrow$ selection decision based on Eq. (7);
        Precaching content at $a^*$ if needed;
    end
    if Handover happens then
        $r^* \leftarrow$ observe the reward according to Eq. (6);
        Update $Q(a^* \mid s)$ with Eq. (8);
    end
end
```

We sum up the above in Algorithm 1 and Algorithm 2 for non-contextual and contextual bandit learning respectively, which have been applied to our proactive caching problem. Additionally, a general flowchart of MAB-based proactive caching is integrated and shown in Fig. 4, though contextual MAB may also involve identifying the context and updating its action values correspondingly.

V. Uncertainty analysis model

In decision-making problems, reducing uncertainty is deemed to be vital as less uncertainty means that an agent is likely to make more accurate decisions. Thus, it is meaningful to assess and quantify the uncertainty in a learning problem. In this work, we adopt *Subjective Logic* framework [17] and particularly adjust it to investigate uncertainty in bandit learning based proactive caching systems. The motivation behind this is to provide a more insightful analysis model for the performance of proactive caching systems and how uncertainty evolves during the learning process. We also aim to give a greater insight as to how MAB-based systems outperform the others and how the context introduced by the contextual MAB algorithm could benefit the whole system. This subsection will introduce some background and discuss how we achieve this.

A. Uncertainty

In the field of machine learning and statistics, a reliable estimation of uncertainty plays an important role in order to create reliable statistical models [18]. In [15], uncertainty in statistical models is classified as *aleatoric* and *epistemic*. Given a set of observed data samples $\mathcal{D} = \{d_1, d_2, ..., d_n\}$ that are generated by an unknown stochastic process P, if the task is to fit a model $p(\mathcal{D} \mid \Theta)$ that describes the observation $\mathcal{D}$, the set of parameters Θ needs to be learned from the collected observations. Apparently, the uncertainty that affects the accuracy of model $p(\mathcal{D} \mid \Theta)$ comes from both the internal randomness of process P and the limitation of the number of observations used to estimate the model. Therefore, these two types of uncertainty can be described as:

- *Aleatoric uncertainty* is inherent randomness in the data generation process P which can be reflected by the variability in the outcome of a trial. A typical example is coin flipping. For this type of uncertainty, however much data provided, the uncertainty of final fitted model $p(\mathcal{D} \mid \Theta)$ is unlikely to be less than the underlying model P [18].
- *Epistemic uncertainty* on the other hand, is due to the lack of knowledge about the best model such as finite sample size. Different from aleatoric uncertainty, epistemic uncertainty can be improved by having more samples or trials.

The present study concentrates on the overall uncertainty of bandit learning algorithms, accounting for both aleatoric and epistemic uncertainties, which can be computed as the entropy of the relevant distribution under the subjective logic framework.

B. Subjective Logic

Subjective logic [17] has been a promising approach to evaluate uncertainties in a statistical model. It is a compact formalism to represent specific forms of probability distributions (Dirichlet-multinomial and Dirichlet-categorical models) [18]. Specifically, given a discrete domain $\mathcal{X} = \{x_1, x_2, ..., x_k\}$ with *k* elements, there exists an *multinomial opinion* for the domain:

$$o = (\boldsymbol{b}, u, \boldsymbol{c}), \text{subject to } u + \sum_{i=1}^{k} b = 1$$

- $\boldsymbol{b} \in \mathbb{R}^k_{\geq 0}$: *belief vector* that represents the degree of certainty over the *k* elements
- $u \in \mathbb{R}_{\geq 0}$: *uncertainty scalar* that shows the degree of certainty on belief vector
- $\boldsymbol{c} \in \mathbb{R}^k_{\geq 0}$: *base rate vector* which often expresses the prior probability distribution of the *k* elements

According to [18], the belief vector acquires the *first-order* uncertainty of the distribution of beliefs over the domain mapping to the aleatoric uncertainty whereas *u* maps to epistemic uncertainty capturing the *second-order* uncertainty about the belief model. In such a model, the probability of an element x_i in the domain $\mathcal{X}$ with opinion o can be computed with:

$$p(x_i \mid o) = b_i + uc_i \tag{9}$$

An existing mapping between an opinion $o = (\boldsymbol{b}, u, \boldsymbol{c})$ and an evidential Dirichlet pdf $s = Dir_e(e)$ [18] [17]:

$$\begin{cases} e_i = \frac{Wb_i}{u} & \text{if } u \neq 0 \\ e_i = \infty & \text{otherwise} \end{cases} \tag{10}$$

whose reverse is:

$$\begin{cases} b_i = \frac{e_i}{W + \sum_{i=1}^{k} e_i} \\ u = \frac{W}{W + \sum_{i=1}^{k} e_i} \end{cases} \tag{11}$$

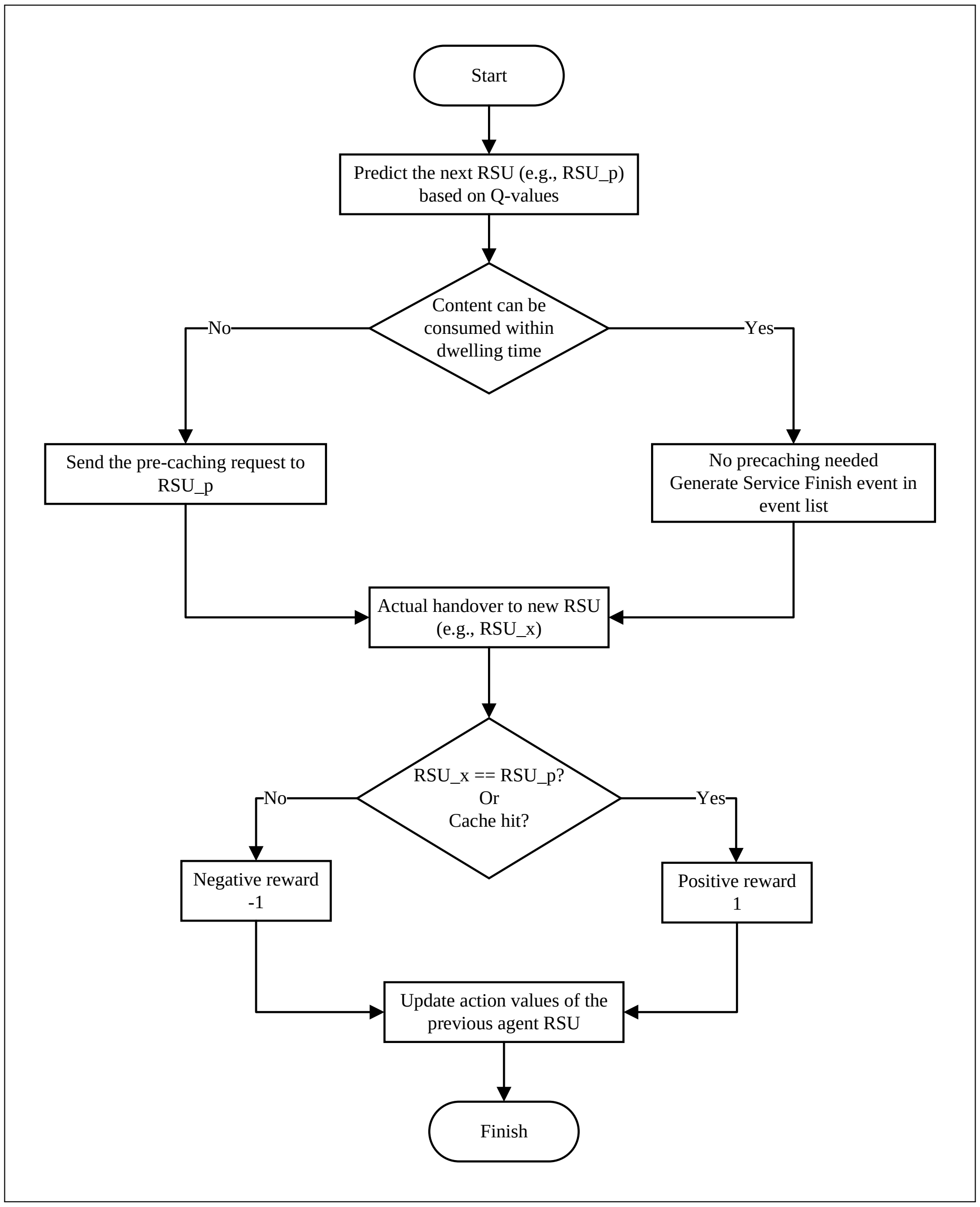

Start
Predict the next RSU (e.g., RSU_p) based on Q-values
Content can be consumed within dwelling time
No
Yes
Send the pre-caching request to RSU_p
No precaching needed Generate Service Finish event in event list
Actual handover to new RSU (e.g., RSU_x)
RSU_x == RSU_p? Or Cache hit?
No
Yes
Negative reward -1
Positive reward 1
Update action values of the previous agent RSU
Finish

Fig. 4: Flowchart of MAB-based proactive caching algorithm: this is a general cycle of an agent RSU serving a connecting vehicle, from *Start* when it receives content request from a connecting vehicle, to *Finish* when its action-value table is successfully updated with corresponding rewards.

where W is a non-informative prior weight normally specified equal to 2 for consistency.

Equation (10) and (11) form a theoretical foundation for the uncertainty analysis in the present work. Most importantly, Equation (11) allows to build the multinomial opinion o over actions of RSUs with experiment observations (i.e., evidence). Hence we are able to obtain the probabilities of actions and overall uncertainty in the form of entropy accordingly. How we define evidence and the overall uncertainty calculation will be discussed in the following.

C. Uncertainty evaluation of proactive caching systems

Similar to the uncertainty in decision-making theory, two sources of uncertainty exist in proactive caching systems, corresponding to aleatoric and epistemic uncertainties. On the one hand, for a RSU, the right decision depends on the proactive caching scheme as well as the randomness in the system. These are all inherent aleatoric uncertainty. On the other hand, epistemic uncertainty in such systems comes from the lack of visits of the RSU or the lack of chances for it to make decisions, which should be reduced as more observations are collected .

To form an opinion over an action set, the evidence of the set needs to be collected, with which the corresponding belief vector and uncertainty scalar of the opinion tuple can be obtained through Equation (11). The probability of an individual action can be achieved accordingly via Equation (9). For an arbitrary RSU with m actions, the subjective opinion

$o^t = (\boldsymbol{b}^t, u^t, \boldsymbol{c})$ at an arbitrary timestep t conveys:

- the belief of an agent on action a_i being the best action with b_i^t
- the global uncertainty over the beliefs with u
- the prior belief c which is constant

Therefore, at timestep 0 or in the beginning of the learning process, the initial values of the three elements are:

$$\begin{cases} b_i^0 = 0 & \forall i \in [1, m] \\ u^0 = 1 & \\ c_i = \frac{1}{m} & \forall i \in [1, m] \end{cases}$$

which means that the agent has no knowledge about which of its actions is likely to be the best and they have equal probabilities. The uncertainty at this point is the maximum, 1.

The rule we used to collect evidence that supports the belief that action a^t could be the best is straightforward:

$$e_i^{t+1} = e_i^t + \mathbb{1}\left[a^t = a_i\right]$$

where the evidence is updated by adding one piece if $[a^t = a_i]$ is true. Thus, the evidence at any timestep t forms the opinion o^t and with Equation (9) a categorical distribution of the action set can be induced: $p\left(\mathbf{a} \mid o^t\right) = \mathbf{Cat}(\mathbf{b}^t + u^t\mathbf{c})$. From this distribution, the overall uncertainty can be calculated as the entropy of the distribution:

$$H = -\sum_{i=1}^{m} p(a_i \mid o^t) \log_2 p(a_i \mid o^t) \tag{12}$$

For the non-contextual MAB algorithm, Equation (12) can be applied directly because of its single state feature. In contrast, for contextual MAB, the entropy computation needs to consider the number of contextual situations.

Given an agent that has n contextual situations denoted by $\mathcal{S} = \{s_1, s_2, ..., s_n\}$ with m actions, each of these situation is an independent bandit task as mentioned earlier. As a consequence, we can compute their entropy called *context entropy* as:

$$H(s_j) = -\sum_{i=1}^{m} p(a_i \mid o^t, s_j) \log_2 p(a_i \mid o^t, s_j) \tag{13}$$

For the agent, the global uncertainty in terms of entropy then becomes:

$$H = \frac{\sum_{j=1}^{n} H(s_j)}{n \log_2 m} \tag{14}$$

This draws on the *Exploration Entropy* in a full reinforcement learning problem [40] where multiple *states* are associated with an agent.

In the proactive caching system, the actions of an agent RSU have their own success probability, which is a source of the aleatoric uncertainty. As mentioned earlier, even the optimal model cannot have less uncertainty than the true process. The MAB-based algorithms cannot remove such intrinsic uncertainty but aim to form a belief vector $\boldsymbol{b}$ over the actions that best describe it. For non-contextual MAB, sufficient learning (trials) allows the agent RSU to have the best model for the aleatoric uncertainty, compared to other non-contextual baseline systems (which we shall see in the results section). In other words, enough evidence results in a small epistemic uncertainty u, and a smaller overall uncertainty means a better fitted model. Contextual MAB (cMAB), on the other hand, introduces a context (i.e., previous RSU) to further disaggregate the problem into context-related. The aleatoric uncertainty under each context s may be substantially reduced in contrast to non-contextual case. Therefore, after sufficient learning, the agent RSU will have the best model for the aleatoric uncertainty associated with each context s, thereby less overall uncertainty.

To sum up, Equation (12) will be applied to evaluate the overall uncertainty in the non-contextual MAB-based proactive caching algorithm, and Equation (13) and (14) will assess the contextual MAB-based algorithm.

VI. Simulation and Performance evaluation

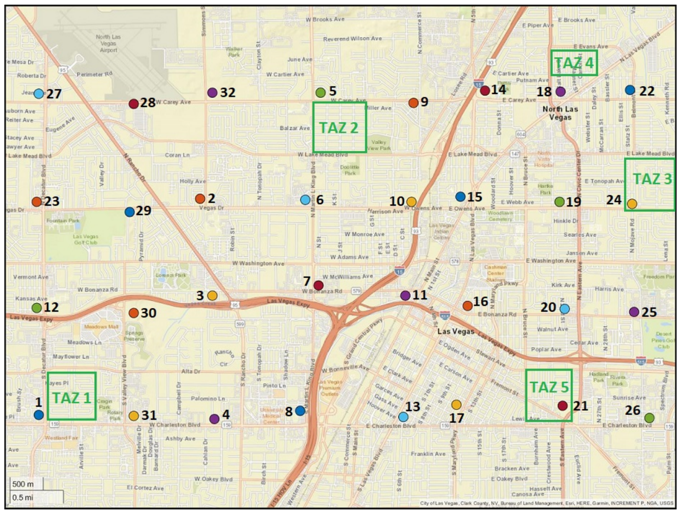
A. Simulation Setup

Simulation in this work includes two parts: traffic simulation and network simulation. Vehicle traffic traces are generated by Simulation of Urban MObility (SUMO) [41] and they are processed with event-driven network simulation program implemented in MATLAB [42].

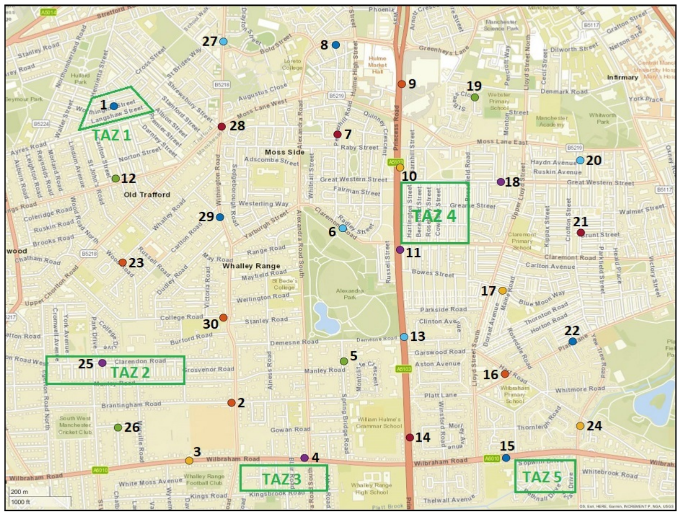
1) Traffic simulation: SUMO is used to simulate a real transportation network discussed in Section III. The scenario we are interested in is the daily commuting routine of people living in a particular urban area. We focus on an area in Las Vegas as our primary city and Manchester as a secondary city to generalize the application of MAB-based schemes to two cities with different road planning. For both areas shown in Fig. 5, five traffic zones (TAZs) are defined in SUMO and in total 174 vehicles travel from and to these zones as their origins and destinations. These TAZs are designed to simulate realistic residential and office areas. We assume that a TAZ contains both residential and office areas. In order to simulate vehicles with same daily routine, each vehicle has their own fixed departure and arrival zone. However, each vehicle may have different departure time and lanes (which may result in route difference) from test trace to test trace. Again, this is to imitate that people in reality may set off for work at various time slots, park at various places of an area, and take slightly different commuting routes, despite having the same workplace (TAZ).

200 files of test traces for each city have been generated to simulate 200 workdays and the simulation period in SUMO is between 8am to 9am. The vehicles' routes between two TAZs are defined by the tool *duarouter* and follow the Shortest or Optimal Path Routing rule. They depart at the *maxSpeed* and follow the default Car Following Model to keep the maximum speed which is safe in the sense of being able to stop in time to avoid a collision. Other road behaviors apply as well such as lane changing, accelerate/decelerate, intersections, etc. Technical details about these settings can be found in SUMO documentation[1].

[1]https://sumo.dlr.de/docs/



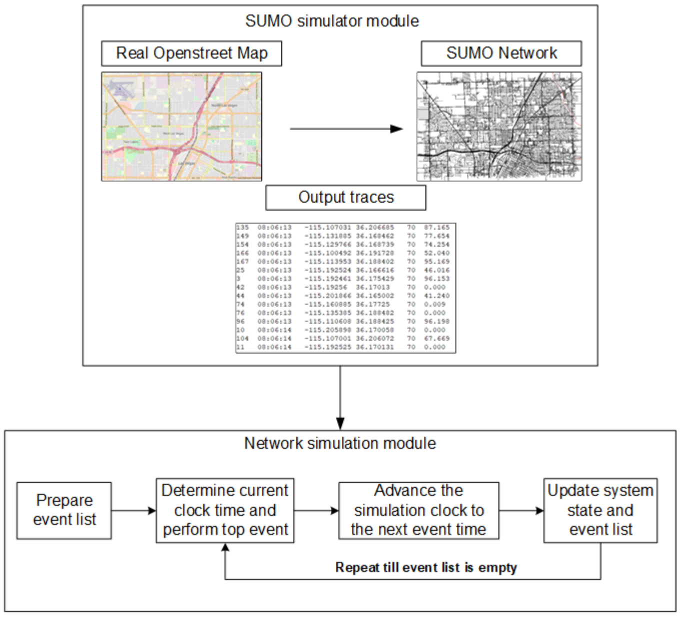
(a) RSU and TAZ distribution in Las Vegas



(b) RSU and TAZ distribution in Manchester

Fig. 5: Urban areas for simulation

2) Network simulation: Discrete event-driven system simulation [43] allows the vehicular network simulation to be performed through a series of events. Test traces are generated by SUMO and passed to the simulation system sequentially. The discrete event list corresponding to the test trace being tested is created at the beginning, which may include *departure* and *arrival* of vehicles, *content request*, *handover*, and *finishing of content consumption*. As the present work concentrates on online learning, a complete cycle of the simulation is testing 200 trace files and the learners (i.e., RSUs) make predictions as they learn throughout the simulation cycle and become increasingly knowledgeable as the simulation runs. Fig. 6 shows a structure of the modules mentioned and the relevant parameters are summarized in Table I. The closely related parameters to the MAB-based systems in Table I are learning rate α referenced in [44] and ϵ selected empirically. The network parameters such as transmission rate and backhaul link rate, are empirical values and they have no impact on the performance of proactive caching (i.e., prediction accuracy or cache hit ratio).


SUMO simulator module
Real Openstreet Map
SUMO Network
Output traces
Network simulation module
Prepare event list
Determine current clock time and perform top event
Advance the simulation clock to the next event time
Update system state and event list
Repeat till event list is empty

```
135  08:06:13  -115.107031 36.206685  70  87.165
149  08:06:13  -115.131885 36.168462  70  77.654
154  08:06:13  -115.129766 36.168739  70  74.254
166  08:06:13  -115.100492 36.191728  70  52.040
167  08:06:13  -115.113953 36.188402  70  95.169
25   08:06:13  -115.192524 36.166616  70  46.016
3    08:06:13  -115.192461 36.175429  70  96.153
42   08:06:13  -115.19256  36.17013   70  0.000
44   08:06:13  -115.201866 36.165002  70  41.240
74   08:06:13  -115.160885 36.17725   70  0.009
76   08:06:13  -115.135385 36.188482  70  0.000
96   08:06:13  -115.110608 36.188425  70  96.198
10   08:06:14  -115.205898 36.170058  70  0.000
104  08:06:14  -115.107001 36.206072  70  67.669
11   08:06:14  -115.192525 36.170131  70  0.000
```

Fig. 6: Simulation modules

TABLE I: Simulation Parameters

Parameter	Value
α for bandit learning	0.5
ϵ for bandit learning	0.05
No. of test traces	200
No. of Vehicles	174
SUMO Simulation Time	8:00 - 9:00
No. of RSUs	32 (Las Vegas) / 30 (Manchester)
Backhaul Link Rate	$\omega = 5Gbps$
Transmission rate	$e = 50Mbps$
Size of content database	$K = 30$
Fragment size	$F_c = 100MB$

B. Performance Evaluation

The performance of non-contextual and contextual MAB-based proactive caching systems is compared with three other proactive caching systems:

- *Equal Probability-based Proactive Caching System*: RSUs select a pre-caching node with equal weight from their neighbors. In other words, it is a random selection scheme.
- *Probability-based Proactive Caching System*: This allows RSUs to make the next pre-caching node decision based on their previous popularity using information from historical traces. This is an intuitive scheme where a RSU believes the neighbor with more frequent handovers deserves a higher weight to become the caching node.
- *CPT+ based Proactive Caching System*: This system is based on the sequence prediction algorithm CPT+. Different from the work [26], we have adjusted the algorithm to be used in an online mode. In brief, a RSU trains its prediction tree model with currently available vehicles' data and when predicting the next RSU for a vehicle, it matches all the past RSUs this vehicle has connected and gives out the most possible RSU (highest score).

Remark: the five systems are referred and denoted in the following as: *cMAB* and *MAB* for contextual and non-contextual bandit learning systems, respectively; *EQ*, *PB*, and *CPT+* represent for equal probability-based, probability-based, and CPT+ based systems, respectively.

1) Evaluation: We mainly focus on the evaluation of the proactive caching performance of the systems. An action selection is considered correct when the selected pre-caching neighboring RSU is the actual transited RSU. In the systems considered, it is identical to a cache hit. Additionally, the extended subjective logic framework discussed in Section V is applied to the systems to provide an analysis of uncertainty except for CPT+. This is because CPT+ is a fundamentally different algorithm compared to the other four, in terms of its model and algorithm design. The variability of its action set and the difficulty of accessibility to "contexts" have made the extended uncertainty model inapplicable. The entropy calculation for EQ and PB systems is also based on Equation (12) as the non-contextual MAB. Furthermore, how the proactive caching systems benefit the network is also considered.

The following aspects will be shown in the results:

- *Cumulative prediction accuracy*: Denoting the total number of predictions as $Q_{prediction}$ and correct ones as $Q_{correct}$ of test trace n, the cumulative prediction accuracy *PA* up till trace n is defined as:

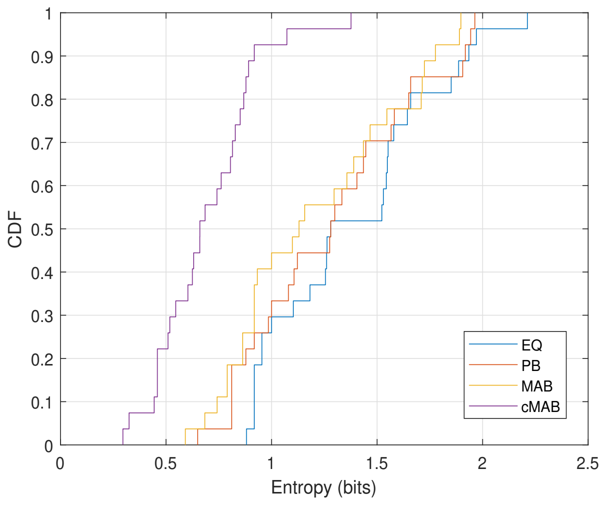
$$PA = \frac{\sum_{i=1}^{n} Q_{correct}}{\sum_{i=1}^{n} Q_{prediction}}$$

- *Cumulative distribution function (CDF) of uncertainty*: Aims to show uncertainty at the system level as well some particular RSUs.
- *Proportion of Proactive Caching Content Fragments*: the proportion of the number of content fragments that are proactively cached and transmitted to vehicular users. This reflects the effectiveness of a proactive caching system.

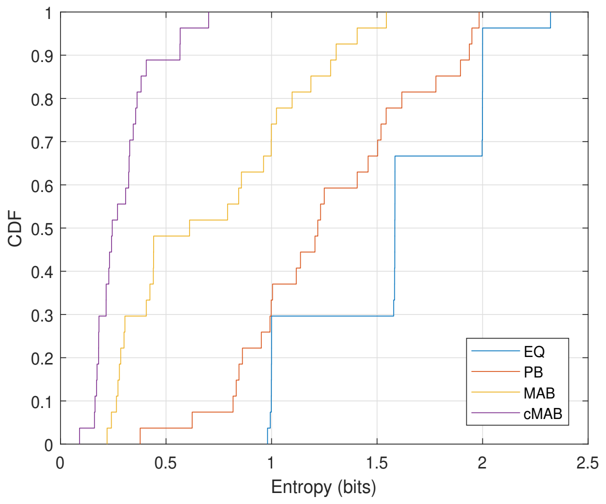
We evaluate the network performance of the systems using *Proportion of Proactive Caching Content Fragments* as a figure of merit, instead of network delay, because the communication model considered in the paper does not model underlying transmission layers and backhaul links, given the focus of the paper is to find where to cache accurately.

2) Experimental results: As Las Vegas is the primary city, we will first elaborate its results as well as detailed analysis. The secondary city Manchester will also be demonstrated but in a more general way.

Fig. 7 shows the uncertainty analysis of four proactive caching systems in Las Vegas at a system level. It is the cumulative distribution of the uncertainty (entropy) of 32 RSUs at the end of test trace 1 and 200, respectively. These results illustrate performance before and after learning. The two bandit learning schemes, non-contextual MAB-based (MAB) and contextual MAB-based (cMAB), outperform the other two baseline schemes in terms of the reduced amount of uncertainty in decision-making. Both MAB and cMAB have dramatically reduced the uncertainty level through sufficient learning after 200 traces. The proportion of RSUs with entropy less than 0.5 bits has increased from 0% to 49% and



(a) CDF of uncertainty at test trace 1



(b) CDF of uncertainty at test trace 200

Fig. 7: Overall Uncertainty of Las Vegas

20% to 90%, respectively. The superiority of the cMAB-based system over its counterpart benefit from introducing the context information. Uncertainty distributions of bandit learning schemes were close to PB and EQ systems in the initial stage of simulation, but this gap has been enlarged in the end. The percentages of RSUs with less than 1-bit entropy are 100%, 80%, 40%, and 0% for cMAB, MAB, PB and EQ respectively. The PB scheme has not experienced a significant change from this perspective because of its nature. Since the test traces simulate vehicles following their own daily commuting routines, the transition probability matrix or the weights used by PB scheme for decisions does not vary too much in the end of trace 1 and 200. By contrast, despite the fluctuations in the initial stage of simulation due to lack of samples, the EQ scheme is constantly the one with the highest overall uncertainty and converges to a stable state finally. This makes sense from the viewpoint of information theory [45] as the entropy of a RSU with *m* neighbors is maximized to

$\log_2 m$ with equal probability $\frac{1}{m}$ among the neighbors.

Fig. 8 shows the prediction accuracy (or hit ratio) in a cumulative way over the test traces. The accuracy superiority of bandit learning schemes over the PB and EQ is closely related to the uncertainty reduction. Another point to explain this is that in bandit learning based schemes, RSUs make their decision on Q-values and the goal is to maximize the rewards. Therefore, fewer attempts are wasted on those actions that are less likely to be successful, whereas PB and EQ schemes, especially the latter one, attempt "bad" decisions more frequently. We shall see this in individual examples later. In addition, CPT+ is also shown in the figure, whose prediction performance is in between cMAB and MAB. In contrast to MAB, this makes sense since CPT+ relies greatly on a vehicle's past RSUs as a kind of context and this reduce the prediction uncertainty. However, it is outperformed by the cMAB as a model-free scheme with only one context (i.e., previous RSU) required. The MAB scheme reaches its limitation of 53% at a much earlier stage compared to cMAB with an upper bound of nearly 80%. CPT+ seems to have an increasing trend after test trace 200 and we can infer that it would reach the performance of cMAB perhaps at test trace 500, because the performance of CPT+ depends on its model: the more data, the better model. However, this is also its limitation in terms of adaptability and flexibility. It is also observed that the introduction of contextual information helps RSUs make more accurate decisions throughout the simulation cycle and meanwhile, it takes relatively longer to fully train the model and converge due to this fact.

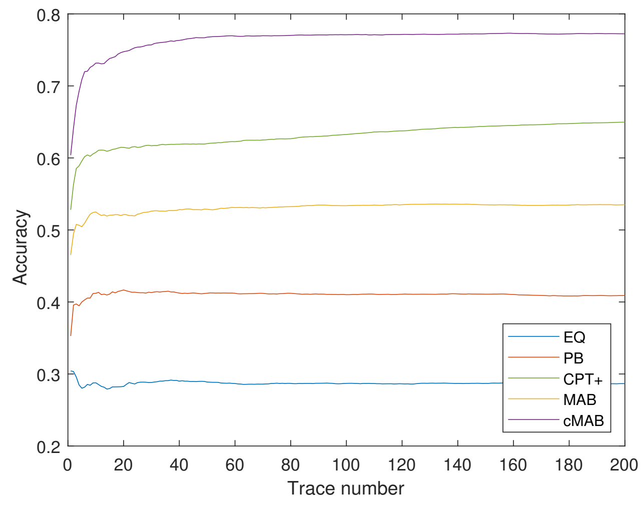


Fig. 8: Cumulative Prediction Accuracy of Las Vegas

Although Fig. 7 and Fig. 8 have demonstrated the potential interaction between prediction accuracy and uncertainty reduction, different RSUs may show significantly different variations on these two metrics. In Fig. 9 we have selected 4 types of RSUs according to the number of their actions/neighbors. From the top to bottom row, they are RSUs with 5 actions, 4 actions, 3 actions and 2 actions, respectively. The left column is the uncertainty CDF of relevant RSUs in an aggregated way. For example, there are 6 RSUs with 4 actions in our system. To achieve the plot on the left hand side, we have collected their uncertainty at the end of each test trace, resulting in 200 by 6 samples for the CDF plot. Note that there is only 1 RSU with 5 actions. Similarly, the right-hand column shows the cumulative prediction accuracy of the corresponding RSUs also in an aggregated way. The prediction accuracy of test trace 10 of 4-action RSUs is $\frac{\sum_1^6 \sum_1^{10} Q_{correct}}{\sum_1^6 \sum_1^{10} Q_{prediction}}$. Both columns share the same legend shown in the bottom left corner. In general, the distribution of uncertainty of test traces still supports the inner connection seen in Fig. 7 and 8. Although it may be difficult to quantify the benefits of the reduction in uncertainty to prediction accuracy at this point, it helps visualize such benefits.

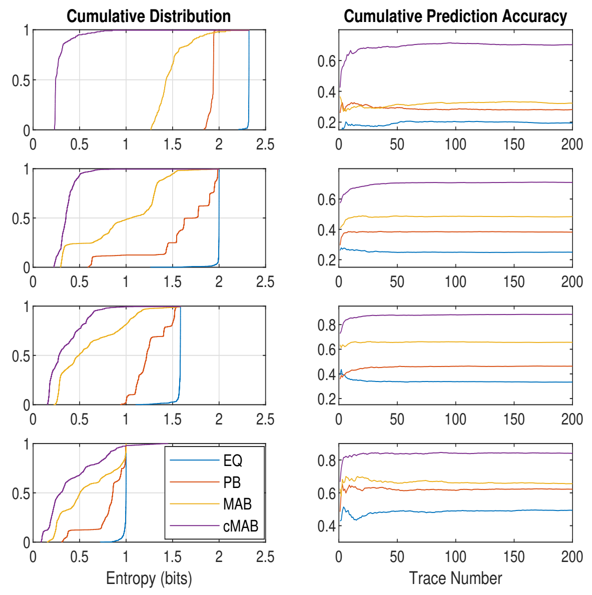
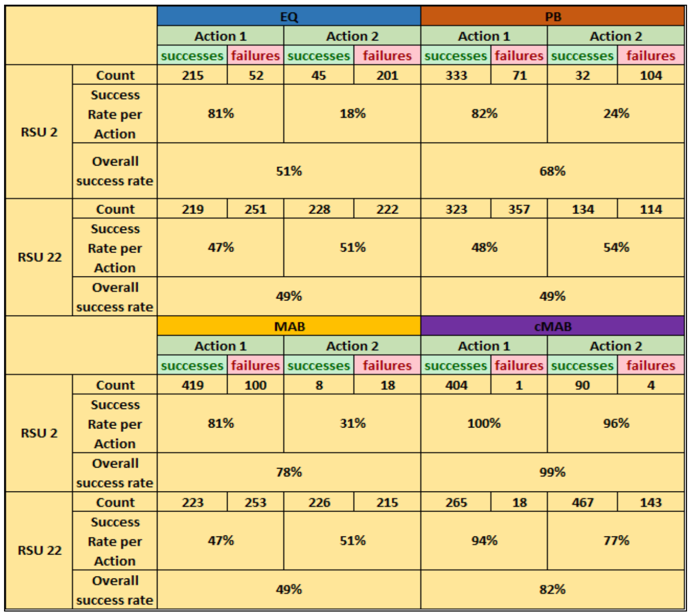


Fig. 9: Performance of RSUs with Different Actions in Las Vegas

Even with the same number of neighbors, RSUs may show completely different performance in terms of uncertainty and prediction accuracy, possibly depending on their geographical location, traffic patterns, connectivity patterns, etc. In Fig. 10, we have selected RSU 2 and RSU 22 from the map in Fig. 5a, both of which have two neighboring RSUs (actions to be more precise) with unbalanced traffic. Over the 200 test traces, there are 73% and 27% of the 1116 handovers from RSU 2 to its two neighbors respectively, and RSU 22 also has the same proportion based on 2872 handovers. Despite this, proactive caching schemes have shown significantly different performance on these two RSUs and we have summarized in the table of Fig. 10 some statistical data in the end of the simulation. Without additional context introduced, we believe there is an unknown inherent success rate of each action for non-contextual schemes (EQ, PB, and MAB), denoted as θ^*. For the action 1 of RSU 2, θ^* can be approximately 80% according to the table as the success rates of all the three schemes tend to converge to 80%. For action 2, however, there does not seem to have a clear converging success rate, but



		EQ				PB			
		Action 1		Action 2		Action 1		Action 2	
		successes	failures	successes	failures	successes	failures	successes	failures
RSU 2	Count	215	52	45	201	333	71	32	104
	Success Rate per Action	81%		18%		82%		24%	
	Overall success rate	51%				68%			
RSU 22	Count	219	251	228	222	323	357	134	114
	Success Rate per Action	47%		51%		48%		54%	
	Overall success rate	49%				49%			

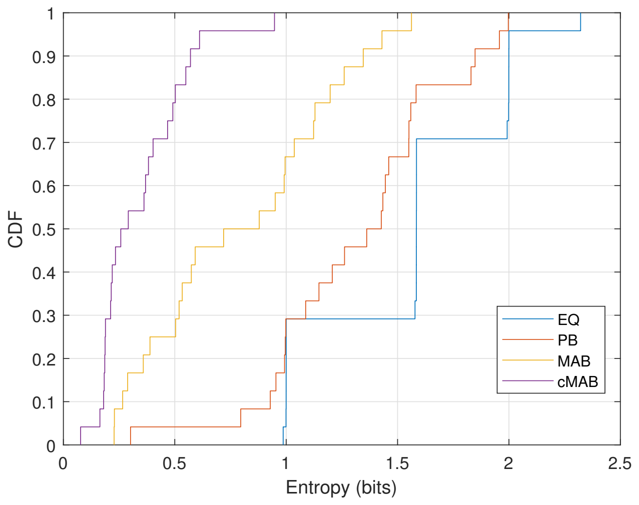
		MAB				cMAB			
		Action 1		Action 2		Action 1		Action 2	
		successes	failures	successes	failures	successes	failures	successes	failures
RSU 2	Count	419	100	8	18	404	1	90	4
	Success Rate per Action	81%		31%		100%		96%	
	Overall success rate	78%				99%			
RSU 22	Count	223	253	226	215	265	18	467	143
	Success Rate per Action	47%		51%		94%		77%	
	Overall success rate	49%				82%			

Fig. 10: Performance comparison of two RSUs in Las Vegas

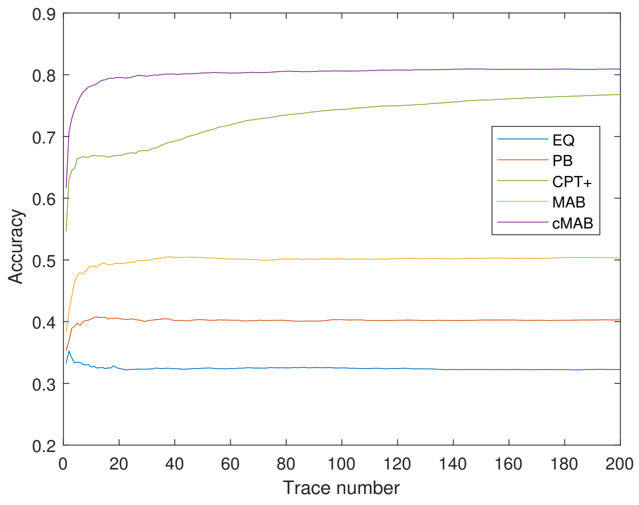
we can infer that it could be 18% as in EQ scheme. The reason that PB and MAB have higher performance for action 2 is because they have fewer selections on action 2 than EQ, referred to the "Count" row. Precisely because of this, during the learning process, MAB leans towards action 1 as it tends to have a better Q-value than action 2 and hence much fewer wrong decisions are made, resulting a 78% overall accuracy. On the other hand, θ^* for action 1 and action 2 of RSU 22 is tending to converge to somewhere around 50%. Consequently, the MAB scheme is unable to tell which action would be a better one as they both have similar Q-values and it shows basically the same prediction performance as EQ and PB.

It is obvious that the introduction of additional contextual information in cMAB has dramatically increased not only the success rate of each action of RSU 2 and RSU 22 but also their overall prediction accuracy to 99% and 82%, respectively. In particular, compared to its counterpart MAB, it has resolved the dilemma with RSU 22 where both actions have similar inherent θ^*. In stead of "hesitating" between the two actions, RSU 22 learns policy under different contexts in cMAB and becomes more certain about which action is likely to be correct. This is even more convincing for the case of RSU 2, where both actions have over 96% accuracy.

The system performance of Manchester is shown in Fig. 11, as a secondary city for generalizing the application. Similarly, we show the distribution of uncertainty among RSUs of four systems in the end of test trace 200 in Fig. 11a and cumulative prediction accuracy of all five systems in Fig. 11b. Bandit learning-based schemes still show comparable benefits to that in Las Vegas, especially cMAB whose prediction accuracy has reached 80%. The performance has successfully demonstrated the adaptability of the proposed bandit learning schemes in a relatively more complex transportation network. One of the reasons for this is that the proposed algorithms only rely on information from the vehicular network itself for proactive



(a) CDF of Uncertainty at trace 200



(b) Cumulative Prediction Accuracy

Fig. 11: Prediction Performance of Manchester

caching decisions instead of taking additional information from the road network. Despite the advantages over the other two non-contextual systems (EQ and PB) as before in Manchester, we clearly notice the performance limitation of non-contextual MAB in contrast to its counterpart MAB scheme. CPT+ still shows similar relative performance to cMAB and MAB but has a faster growth rate compared to Las Vegas. This might be because of the relative area size and traffic pattern difference between two cities (which will be explained in detail shortly).

One of the major goals of proactive caching in vehicular networks is providing vehicular users with seamless content delivery by bringing the content close to them accurately. We measure the amount of fragments transmitted directly from RSU caches to vehicular users and plot a bar chart of the *proportion of the average fragments served by proactive caching* for each of the proactive caching schemes of two cities in Fig. 12. Overall, the proportions of both cities are consistent with the cumulative prediction accuracy, and the cMAB scheme demonstrates remarkable superiority over the other four. On average, it has achieved 75% in Las Vegas and 81% in Manchester, nearly double that of EQ and PB systems. We can also conclude that our proposed proactive caching schemes perform similarly irrespective of the road topology. Note-worthily, the proportions in the two cities are based on different absolute total number of fragments transmitted to vehicles (around 1300 in Las Vegas and 750 in Manchester, varying trace by trace). This is because a) the Manchester area is relatively smaller than the Las Vegas area as a whole, b) the connectivity patterns of the two cities are distinct, and c) vehicles' content request pattern and frequency are different from test trace to trace of two cities. However, as the relative size of the center of two areas have been kept on a similar level, this is still an effective contrast.

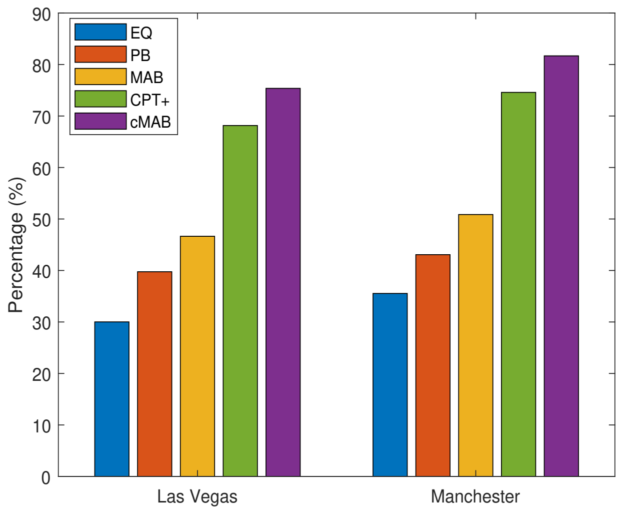


Fig. 12: Proportion of the Average Fragments Served by Proactive Caching

VII. Discussion

A. Theoretical analysis

Theoretically, the two proposed MAB-based algorithms have advantages in terms of computational complexity. For the three non-contextual algorithms i.e., MAB, Equal-probability and Probability-based, the Probability-based one has the highest computational complexity. This is because RSUs in this algorithm require some extra computational resources to store historical traffic information in order to establish a probability distribution over their actions. However, MAB algorithm is a localized algorithm where the RSUs' Q-tables get in-place update, and individual RSU has its own fixed probability distribution for prediction in Equal-probability algorithm. Although cMAB algorithm is also a localized algorithm as MAB, it does require RSUs to build context-related Q-tables and therefore, needs slightly more space than MAB. Nevertheless, this is worthwhile given the significantly reduced uncertainty and improved prediction accuracy by cMAB. CPT+ based algorithm, however, consumes the most resources because it requires building a large prediction tree model to achieve a certain prediction accuracy, which is still outperformed by cMAB. Such advantage also makes it practical for the implementation of MAB and cMAB algorithms.

In addition, we can also seek the theoretical accuracy of these algorithms. Assume a vehicle v connecting to a RSU m with N neighbors (actions). There exists an unknown probability distribution of v actually going to the N neighbors after m, denoted as $\mathcal{A} = [a_1, a_2, \ldots, a_n], n \in N$ and $\sum_{n\in N} a_n = 1$. If the RSU m makes prediction with $\mathcal{B} = [b_1, b_2, \ldots, b_n], n \in N$, then the chance that this is a correct prediction can be computed by $\mathcal{P} = \mathcal{A} \cdot \mathcal{B} = \sum_{n\in\mathcal{N}} a_n \times b_n$. Depending on which algorithm, $\mathcal{B}$ is different. In the most under-performed one i.e., Equal-probability algorithm, $\mathcal{B}$ is uniform distribution i.e., $b_1 = b_2 = \ldots = b_n = \frac{1}{N}$ and thus $\mathcal{P} = \sum_{n\in\mathcal{N}} a_n \times b_n = \frac{1}{N} \times \sum_{n\in\mathcal{N}} a_n = \frac{1}{N}$. In the Probability-based algorithm, $\mathcal{B}$ is the transition probabilities derived from previous traces, where $b_1 \neq b_2 \neq \ldots \neq b_n$, and therefore $\mathcal{P}$ remains to be $\mathcal{P} = \sum_{n\in\mathcal{N}} a_n \times b_n$. If the traffic pattern through RSU m does not change significantly over time, we can assume $a_n = b_n$, so $\mathcal{P} = \sum_{n\in\mathcal{N}} b_n^2$. In non-contextual MAB, $\mathcal{B}$ depends on Q-values and action selection algorithm (i.e., ϵ-greedy). Therefore, the probability b_n of its neighbor $n \in N$ to be predicted as the next RSU is: $b_n = \begin{cases} 1-\epsilon, & \text{if } n \text{ has the highest } Q\text{-value} \\ \epsilon \cdot \frac{1}{N}, & \text{Otherwise} \end{cases}$. Take an example of a RSU of two action choices (neighbors) with uneven traffic pattern (e.g., 80% vs 20%). Its theoretical accuracy with Equal-probability algorithm is 50% since it has 2 neighbors. Because it has an uneven traffic pattern where one of its neighbors has approximately 80% traffic, we can compute the theoretical accuracy with Probability-based algorithm $\mathcal{P} = 80\% \times 80\% + 20\% \times 20\% = 68\%$. It is because of this traffic pattern that MAB has a dominant action and therefore, the overall theoretical accuracy is $\mathcal{P} = 80\% \times (1-\epsilon) + 20\% \times \frac{\epsilon}{2} = 77\%$, where $\epsilon = 0.05$. The cMAB algorithm further expands the advantage of MAB and reduces uncertainty by breaking down into context level, hence resulting an even higher optimal boundary. The simulated result of RSU 2 in Fig. 9 is consistent with the theoretical values and this can be extended to other RSUs with different number of choices.

Furthermore, another notable advantage of the proposed cMAB and MAB algorithms is their natural capabilities of coping with sudden major changes in the topology or vehicular environment by rapidly adjusting Q-tables and policies, whereas Probability-based and CPT+ based algorithms become very clumsy in this regard due to high reliance on past data to establish their models.

B. Time complexity

The three main functionalities in the proposed MAB and cMAB algorithms are: A - *Next RSU selection* (including ϵ -greedy), B - *Pre-caching content* and C - *Q-table updating with rewards*. From the perspective of actual code implementation, for MAB algorithm, an agent RSU with k actions requires $O(k)$, $O(1)$, and $O(k)$ time complexity for function A, B and C respectively. This is because function A and C require action set traversal whereas B only needs insertion manipulation with a vector. In addition, as function A, B and C are executed sequentially, they account for a $O(k)$ complexity. The system may have multiple RSUs but due to the nature of event-driven simulation, only one of them is "working" at a time. Therefore, assuming the largest action set of these RSUs is K, then the overall performance of N-length test can be represented by $O(NK)$. The major difference between the two lies in the additional context s. Specifically, function A and C are executed based on s once it is detected. But this works in the same way as in a non-contextual MAB and therefore, their complexity is identical to that in MAB for an arbitrary RSU. Function B remains the same as well. Apart from this, cMAB algorithm also involves context detection and creation and these additional manipulations account for $O(1)$ complexity. Thus, cMAB has the same overall complexity, that is $O(NK)$ as above.

C. Convergence

The cumulative prediction accuracy in Fig. 8 and Fig. 11b demonstrates the convergence of the proposed MAB-based proactive algorithms. Although a cumulative way to show this may not be perfect, it is still sufficient to illustrate the performance boundary in the commuting traffic scenario we have considered. From the system level, theoretically the cMAB algorithm should converge slower than the non-contextual MAB because given a statistically fixed number of Q-table updates (identical test traces) for a RSU fewer updates are allocated to each individual context in cMAB in contrast to MAB where all the updates are used for only one Q-table. This difference in convergence can be found in the previously mentioned results.

During the learning process, Q-values or Q-tables of individual RSUs may converge to rather different values depending on the traffic pattern through it. For example, in the non-contextual MAB algorithm, we have noticed that a high-accuracy RSU (over 90%) with 4 actions have a converged Q-table with values: $\langle -0.9375, -1, -0.9961, 1\rangle$ at an early stage of the learning process. This demonstrates a convergence to the last action and that there may exist very deterministic routes for all the vehicles through this RSU. On the other hand, it has also been found that an average-accuracy RSU (approximately 50%) with same number of choices have a Q-table with values: $\langle -1, -1, -0.5643, -0.4379\rangle$. Throughout the learning process, the RSU tried to converge to the best action by trial and error but failed to do so because the last two actions are almost evenly good. This implies the dilemma in non-contextual MAB and should be resolved by contextual MAB exploiting the additional contexts available.

VIII. Conclusion

This paper studies how to cache the content at the next RSU in a proactive way. As a way of addressing this, the paper has proposed two bandit learning-based proactive caching algorithms: *non-contextual* MAB and *contextual* MAB and compared their performance with three other baseline schemes: *Equal Probability-based*, *Probability-based*, and *Compact Prediction Tree+ based* proactive caching strategy. In addition to this, the *subjective logic* framework has been extended to study the uncertainty associated with different proactive caching systems. With this model, we have analyzed in detail the overall entropy distribution of the systems as well as the distribution of representative RSUs. Furthermore, two urban areas of Las Vegas and Manchester with different road layouts have been tested to demonstrate the adaptability of the proposed schemes to a diverse set of road layouts.

Numerical results have shown the advantages of the proposed proactive caching algorithms over their counterparts. Contextual MAB-based scheme yields the highest benefit to the system thanks to the introduction of contextual information for uncertainty reduction. In both cities, the contextual MAB-based proactive caching scheme reaches a prediction accuracy of approximately 80% compared to roughly 50% of non-contextual MAB-based scheme. As a result of this, the network performance is dramatically improved with contextual MAB in terms of the number of fragments directly transmitted by caches. Performance of bandit learning-based systems is similar in both cities regardless of road topology. Particularly, 75% and 81% content fragments are proactively served with contextual MAB algorithm and over 53% and 50% with non-contextual MAB algorithm in Las Vegas and Manchester, respectively.

References

[1] H. Hartenstein and L. Laberteaux, "A tutorial survey on vehicular ad hoc networks," *IEEE Communications Magazine*, vol. 46, no. 6, pp. 164–171, 2008.

[2] S. Zhang, J. Chen, F. Lyu, N. Cheng, W. Shi, and X. Shen, "Vehicular communication networks in the automated driving era," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 26–32, 2018.

[3] K. Zheng, L. Hou, H. Meng, Q. Zheng, N. Lu, and L. Lei, "Soft-defined heterogeneous vehicular network: Architecture and challenges," *IEEE Network*, vol. 30, no. 4, pp. 72–80, 2016.

[4] L. Hou, L. Lei, K. Zheng, and X. Wang, "A Q-learning-based proactive caching strategy for non-safety related services in vehicular networks," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4512–4520, 2018.

[5] F. A. Silva, A. Boukerche, T. R. M. B. Silva, L. B. Ruiz, E. Cerqueira, and A. A. F. Loureiro, "Vehicular networks: A new challenge for content-delivery-based applications," *ACM Comput. Surv.*, vol. 49, no. 1, June 2016. [Online]. Available: https://doi.org/10.1145/2903745

[6] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[7] J. Zhang and K. B. Letaief, "Mobile edge intelligence and computing for the internet of vehicles," *Proceedings of the IEEE*, vol. 108, no. 2, pp. 246–261, 2019.

[8] D. Liu, B. Chen, C. Yang, and A. F. Molisch, "Caching at the wireless edge: design aspects, challenges, and future directions," *IEEE Communications Magazine*, vol. 54, no. 9, pp. 22–28, 2016.

[9] J. Ren, H. Guo, C. Xu, and Y. Zhang, "Serving at the edge: A scalable IoT architecture based on transparent computing," *IEEE Network*, vol. 31, no. 5, pp. 96–105, 2017.

[10] N. Nakamura, Y. Niimi, and S. Ishihara, "Live vanet cdn: Adaptive data dissemination scheme for location-dependent data in vanets," in *2013 IEEE Vehicular Networking Conference*. IEEE, 2013, pp. 95–102.

[11] N. Morozs, T. Clarke, and D. Grace, "Distributed heuristically accelerated q-learning for robust cognitive spectrum management in lte cellular systems," *IEEE Transactions on Mobile Computing*, vol. 15, no. 4, pp. 817–825, 2016.

[12] A. Mahajan and D. Teneketzis, "Multi-armed bandit problems," in *Foundations and applications of sensor management*. Springer, 2008, pp. 121–151. [Online]. Available: https://doi.org/10.1007/978-0-387-49819-5_6
[13] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
[14] D. Bouneffouf and I. Rish, "A survey on practical applications of multi-armed and contextual bandits," 2019.
[15] E. Hüllermeier and W. Waegeman, "Aleatoric and epistemic uncertainty in machine learning: A tutorial introduction," *CoRR*, vol. abs/1910.09457, 2019. [Online]. Available: http://arxiv.org/abs/1910.09457
[16] A. Jøsang, "Artificial reasoning with subjective logic," in *Proceedings of the second Australian workshop on commonsense reasoning*, vol. 48. Citeseer, 1997, p. 34.
[17] A. Josang, *Subjective Logic: A Formalism for Reasoning Under Uncertainty*, ser. Artificial Intelligence: Foundations, Theory, and Algorithms. Springer International Publishing, 2016. [Online]. Available: https://books.google.no/books?id=bJkJkAEACAAJ
[18] F. M. Zennaro and A. Jøsang, "Using subjective logic to estimate uncertainty in multi-armed bandit problems," *arXiv preprint arXiv:2008.07386*, 2020.
[19] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty," *arXiv preprint arXiv:1806.01768*, 2018.
[20] L. M. Kaplan, F. Cerutti, M. Sensoy, A. D. Preece, and P. Sullivan, "Uncertainty aware AI ML: why and how," *CoRR*, vol. abs/1809.07882, 2018. [Online]. Available: http://arxiv.org/abs/1809.07882
[21] N. B. Hassine, P. Minet, D. Marinca, and D. Barth, "Popularity prediction–based caching in content delivery networks," *Annals of Telecommunications*, vol. 74, no. 5, pp. 351–364, 2019.
[22] H. S. Goian, O. Y. Al-Jarrah, S. Muhaidat, Y. Al-Hammadi, P. Yoo, and M. Dianati, "Popularity-based video caching techniques for cache-enabled networks: A survey," *IEEE Access*, vol. 7, pp. 27 699–27 719, 2019.
[23] S. Yue and Q. Zhu, "A mobility prediction-based relay cluster strategy for content delivery in urban vehicular networks," *Applied Sciences*, vol. 11, no. 5, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/5/2157
[24] L. Yao, A. Chen, J. Deng, J. Wang, and G. Wu, "A cooperative caching scheme based on mobility prediction in vehicular content centric networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 6, pp. 5435–5444, June 2018.
[25] H. Khelifi, S. Luo, B. Nour, A. Sellami, H. Moungla, and F. Naït-Abdesselam, "An optimized proactive caching scheme based on mobility prediction for vehicular networks," in *Proc. IEEE Global Communications Conf. (GLOBECOM)*, Dec. 2018, pp. 1–6.
[26] Q. Wang and D. Grace, "Sequence prediction-based proactive caching in vehicular content networks," in *2020 IEEE 3rd Connected and Automated Vehicles Symposium (CAVS)*, 2020, pp. 1–6.
[27] T. Gueniche, P. Fournier-Viger, R. Raman, and V. S. Tseng, "Cpt+: Decreasing the time/space complexity of the compact prediction tree," in *Advances in Knowledge Discovery and Data Mining*, T. Cao, E.-P. Lim, Z.-H. Zhou, T.-B. Ho, D. Cheung, and H. Motoda, Eds. Cham: Springer International Publishing, 2015, pp. 625–636.
[28] C. J. C. H. Watkins, "Learning from delayed rewards," 1989.
[29] G. Rummery and M. Niranjan, "On-line q-learning using connectionist systems," *Technical Report CUED/F-INFENG/TR 166*, 11 1994.
[30] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[31] A. Lazaridis, A. Fachantidis, and I. P. Vlahavas, "Deep reinforcement learning: A state-of-the-art walkthrough," *J. Artif. Intell. Res.*, vol. 69, pp. 1421–1471, 2020.
[32] P. Dai, Z. Hang, K. Liu, X. Wu, H. Xing, Z. Yu, and V. C. S. Lee, "Multi-armed bandit learning for computation-intensive services in mec-empowered vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 7, pp. 7821–7834, 2020.
[33] Y. Miao, Y. Hao, M. Chen, H. Gharavi, and K. Hwang, "Intelligent task caching in edge cloud via bandit learning," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 1, pp. 625–637, 2020.
[34] S. Maghsudi and E. Hossain, "Multi-armed bandits with application to 5G small cells," *IEEE Wireless Communications*, vol. 23, no. 3, pp. 64–73, 2016.
[35] X. Xu, M. Tao, and C. Shen, "Collaborative multi-agent multi-armed bandit learning for small-cell caching," *IEEE Transactions on Wireless Communications*, vol. 19, no. 4, pp. 2570–2585, 2020.
[36] T. Muller, "An unforeseen equivalence between uncertainty and entropy," in *IFIP International Conference on Trust Management*. Springer, 2019, pp. 57–72.
[37] C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," *AAAI/IAAI*, vol. 1998, no. 746-752, p. 2, 1998.
[38] D. Russo, B. V. Roy, A. Kazerouni, and I. Osband, "A tutorial on thompson sampling," *CoRR*, vol. abs/1707.02038, 2017. [Online]. Available: http://arxiv.org/abs/1707.02038
[39] S. Bubeck and N. Cesa-Bianchi, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," *arXiv preprint arXiv:1204.5721*, 2012.
[40] B. Xin, H. Yu, Y. Qin, Q. Tang, and Z. Zhu, "Exploration entropy for reinforcement learning," *Mathematical Problems in Engineering*, vol. 2020, 2020.
[41] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using SUMO," in *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018. [Online]. Available: ⟨https://elib.dlr.de/124092/
[42] MATLAB, *version 9.4.0 (R2018a)*. Natick, Massachusetts: The MathWorks Inc., 2018.
[43] G. A. Wainer and P. J. Mosterman, *Discrete-event modeling and simulation: theory and applications*. CRC press, 2018.
[44] M. Bennis and D. Niyato, "A q-learning based approach to interference avoidance in self-organized femtocell networks," in *2010 IEEE Globecom Workshops*. IEEE, 2010, pp. 706–710.
[45] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

Qiao Wang (S'20) received his Master's degree from Tampere University of Technology (now Tampere University), Finland in October 2014. He worked in Ericsson China as a LTE System Developer since 2015. In September 2018, he joined the Communication Technologies Research Group at the Department of Electronic Engineering at University of York to pursue a PhD degree and has been supervised by Prof. David Grace since then. His current research interests include: proactive caching, vehicular networks, mobility prediction, reinforcement learning.

David Grace (S'95-A'99-M'00-SM'13) received his PhD from University of York in 1999, with the subject of his thesis being 'Distributed Dynamic Channel Assignment for the Wireless Environment'. Since 1994 he has been a member of the Department of Electronic Engineering at York, where he is now Professor (Research), Head of Communication Technologies Research Group, and Director of the Centre for High Altitude Platform Applications. Current research interests include aerial platform-based communications, application of artificial intelligence to wireless communications; 5G system architectures; dynamic spectrum access and interference management. He is currently a lead investigator on H2020 MCSA SPOTLIGHT, UK Government funded MANY, dealing with 5G trials in rural areas, and HiQ investigating Quantum Key Distribution from high altitude platforms.