Predictive Proactive Caching in VANETs for Social Networking

Sara A. Elsayed, *Member, IEEE*, Sherin Abdelhamid, *Member, IEEE*, and Hossam S. Hassanein, *Fellow, IEEE*

***Abstract*—Social media traffic constitutes the highest percentage of Internet traffic. Such traffic is largely facilitated by mobile devices, which imposes a huge traffic load on backhaul links in 5G networks, and can in turn affect the quality of service. This traffic load can be alleviated by using vehicular networks as a traffic offloading platform. In particular, vehicles can act as a resourceful asset for edge caching, thus enabling data acquisition from nearby caching nodes rather than the remote backhaul servers. In this paper, we propose the Predictive Proactive Caching Framework (PPCF), which exploits the daily driving routine and predictable behavior of users to pre-cache the data at parked vehicles for the requesters to proactively acquire as they pass by. PPCF is composed of a cache placement module and a prediction module. The former aims at maximizing cache hits by assigning replicas to caching spots that yield maximum certainty in their spatiotemporal availability for requesters. To estimate such availability, the prediction module aims at making accurate travel time predictions by proposing the use of a Long Short-Term Memory (LSTM) network trained using particle swarm optimization (PSO-LSTM). The predicted average travel time is then used to estimate a personalized travel time for users by considering their different driving behaviors. Extensive simulations demonstrate the ability of the proposed framework to achieve significant improvements in its targeted objectives compared to other prominent caching and prediction schemes in vehicular networks.**

***Index Terms*—Caching, prediction, social media, VANETs.**

I. Introduction

THE penetration rate of social media has been significantly increasing worldwide. In 2019, 79% of Internet users were social media users [1]. This substantial percentage is expected to escalate even further in the future, with an anticipated augmentation in the number of social media users reaching 4.41 billion in 2025 [2]. Mobile devices account for more than 60% of such excessive usage [3]. This causes backhaul links in Fifth Generation (5G) networks to incur a huge traffic influx [4], [5]. Note that 5G promises several attractive prospects, including significantly high bandwidth, fast connectivity, and low latency [5]. However, the limited transmission capacity affiliated with wireless backhaul links makes it difficult to cope with the explosively growing traffic [5]. Thus, it is imperative to reduce the load at backhaul links for 5G to keep its promises.

One promising solution is to use Vehicular Ad Hoc Networks (VANETs) as a traffic offloading platform. With their pervasive availability, mobility, and abundant storage resources, VANETs can play a vital role in edge caching. In edge caching, content access occurs via intermediate nodes at the edge of the network rather than the far-away server at backhaul links [6]–[8]. Note that caching social media traffic can be particularly beneficial since there is a high correlation in the data accessed by many users on social media. This is because the most followed accounts on social media platforms are those of public figures, such as politicians, actors, musicians, etc. Thus, increasing cache hits of such contents in VANETs can reduce the backhaul traffic load [7].

VANETs have emerged as a communication paradigm that fosters inter-vehicle communication on the road. They serve as an enabling technology for an extensive range of applications in Intelligent Transportation Systems (ITS), including Internet access [9]. Internet access in VANETs can take place via Vehicle-to-Vehicle (V2V) communication, as well as communication between vehicles and roadside access points, commonly referred to as RoadSide Units (RSUs) [9]. Along with their communication capabilities, vehicles and RSUs are also equipped with storage resources, which enable them to be used as caching units [10], [11].

According to an industry survey conducted by Michigan's Department of Transportation (DoT) and the Center for Automotive Research, one of the most challenging issues that hinder the wide adoption of vehicular technology is the required funding associated with roadside infrastructures [12], [13]. In fact, a prominent study sponsored by DoT indicated that the estimated average expenses per each deployed RSU is \$17,680 [14]. Such huge expenses prevent sheer deployments of RSUs. One way to mitigate this problem is to use parked vehicles instead of RSUs [13], [14]. Note that powerful On-Board Units (OBUs) in vehicles reinforce abundant storage capabilities [11]. In addition, in contrast to RSUs, roadside parked vehicles are natural infrastructures that exist in substantial numbers [10]. This further makes their combined storage resources drastically higher compared to RSUs. Hence, parked vehicles can serve as profuse and cost-effective caching resources. It has been substantiated in a study that examined roadside parking spaces in Ann Arbor city in the US that their occupancy ratio can mount

Manuscript received July 18, 2021; revised November 27, 2021; accepted January 5, 2022. Date of publication February 7, 2022; date of current version May 20, 2022. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under Grant ALLRP 549919-20. The review of this article was coordinated by Dr. Lin X Cai. *(Corresponding author: Sara A. Elsayed.)*

Sara A. Elsayed and Hossam S. Hassanein are with the School of Computing, Queen's University, Kingston, ON K7L 2N8, Canada (e-mail: selsayed@cs.queensu.ca; hossam@cs.queensu.ca).

Sherin Abdelhamid is with TELUS, Toronto, ON M5J 2V5, Canada (e-mail: sherin.abdelhamid@telus.com).

Digital Object Identifier 10.1109/TVT.2022.3148720

up to 93% and 80% during on and off-peaks, respectively [10]. Rechargeable batteries, especially in electric vehicles can highly facilitate the utilization of roadside parked vehicles for caching. It has been demonstrated that in such vehicles, the average time beyond which the battery gets drained is 160 hours [10].

In this paper, we use roadside parked vehicles for content caching, due to the aforementioned reasons which render parked vehicles promising candidates that can facilitate content sharing via caching. Note that one of the problems in content sharing in VANETs is the short contact time between vehicles [13]. This problem manifests whether the communication is between moving vehicles and parked vehicles, or between moving vehicles and RSUs [13]. However, the sequential nature of parked vehicles helps mitigate this problem. In addition, the short contact time issue is particularly concerning when the content size is large, thus triggering multiple transmissions. In this paper, we focus on social media access applications that require single transmissions.

Existing caching schemes are classified into proactive and reactive caching schemes. In the former, data is prefetched and stored, while in the latter, caching occurs as the data propagates back to the requester in response to a previous request [15]. Proactive caching can significantly improve the quality of service [16], [17]. Most existing proactive caching schemes employ a broadcast-based approach [16], or pre-cache the data while considering the spatiotemporal dynamics of the content popularity [18]–[23], or focus on large-sized content download where each content is too large to be acquired at once, thus triggering the need for multiple transmissions [24].

The broadcast-based approach is rendered unsuitable for social media access applications, due to the huge amount of overhead incurred [16]. Most proactive caching schemes that consider the dynamic popularity of contents, tend to overlook the effect of the dynamic mobility of requesters on the spatiotemporal availability of the cached replicas [18]–[23], particularly in highly dynamic environments, such as VANETs. Note that in the targeted social media access applications (e.g., Instagram), we focus on highly popular contents that are generated by public figures. In this case, the data center has prior knowledge of the contents that each requester is interested in. This knowledge is acquired as users provide the data center with access to their social media profiles. Thus, the data center knows the public figures that each user follows (i.e., the requested contents). Content popularity is associated with the popularity of the corresponding public figure. In the short run, the popularity of the latter is not subject to drastic changes. In contrast to content popularity, which is not drastically dynamic, the highly dynamic mobility of requesting vehicles is a major issue that can profoundly affect the spatiotemporal availability of replicas. Proactive caching schemes that focus on large-sized content download mostly require an initial request for the first chunk to be reactively acquired, before enabling the remaining chunks to be proactively procured [24]. This, along with focusing only on the next spatial location of vehicles rather than their future spatiotemporal availability with respect to the cached replicas, can affect the delay in these schemes [25]–[27]. Furthermore, such schemes are not suitable for the targeted social media platforms that require single transmissions, such as Instagram.

In this paper, we propose a Predictive Proactive Caching Framework (PPCF) for social media access applications that require single transmissions. PPCF reduces the load at backhaul links and provides a certain QoS to social media users that have a predictable behavior. PPCF takes into consideration the future spatiotemporal availability of requesters by predicting their period of encounter with each road segment along their trajectories. This is in order to ensure that each user proactively acquires (before a certain deadline) the requested content as it traverses the road segment at which the caching parked vehicle resides. PPCF relies on the fact that some users tend to maintain a daily routine. This is in terms of the route they follow, the time of navigating that route, and the time of requesting a certain content. PPCF consists of two basic modules executed by the data center (i.e, the original data provider). The first module is the prediction module that predicts the period of encounter of requesters with each road segment along their trajectory. The second module is the proactive cache placement module that enables the data center to select the appropriate road segments for caching.

Our contributions can be summarized as follows:

1) We propose a novel travel time prediction scheme that incorporates the use of a Long Short-Term Memory (LSTM) network, trained using particle swarm optimization (PSO-LSTM), to improve the training process. To the best of our knowledge, this is the first PSO-based LSTM model to be introduced in travel time prediction. In addition, in contrast to existing prediction models, the proposed LSTM model provides long-term predictions that enable multi-step ahead predictions. This is while taking the dynamic weather conditions into consideration. Furthermore, in contrast to existing models that focus on predicting the average travel time only, we estimate a personalized travel time for each user by considering their different driving behaviors.
2) We propose a prediction-based heuristic scheme, called the Predictive Proactive Cache Placement scheme (PPCP). PPCP aims at maximizing cache hits by assigning replicas to caching spots that yield maximum certainty in their spatiotemporal availability for requesters. This is while sustaining a cache capacity limit and a predetermined QoS. To the best of our knowledge, this is the first proactive caching scheme that performs prediction to pre-cache the data at parked vehicles to be procured by the users either before the time of their requests, or within a certain indicated deadline.
3) We introduce a benchmark that can quantify the potential gains of predictive proactive caching in improving the quality of Internet services in vehicular networks, namely the Vehicular Optimal Proactive Caching (VOPC) benchmark. The caching problem in VOPC is formulated as an Integer Linear Programming (ILP) problem.

We implement VOPC and PPCP using the NS-3 network simulator [28]. In order to generate the optimal solution in VOPC, we integrate NS-3 with the Gurobi optimizer [29]. We use VOPC to quantify the potential gains of PPCP. In addition, we assess the performance of PPCP compared to the baseline broadcast-based approach.

In order to assess the performance of the proposed prediction model PSO-LSTM, we compare it to the gradient descent-based LSTM model (GD-LSTM).

The remainder of the paper is organized as follows. Section II outlines some related work. Section III presents the predictive proactive caching framework, including the prediction module, and the cache placement module. Section IV illustrates the performance evaluation and simulation results. Section V highlights our conclusions and future work.

II. Related Work

A. Proactive Caching

In proactive caching, the data center caches the data at vehicles by prefetching the requester's contents of interest to be cached ahead of time [30]. This helps reduce latency and alleviate the traffic load at backhaul links [30]. For example, in [31], contents are periodically broadcast by the data center to all the vehicles within its communication range. The cached data can be later exchanged among vehicles upon encounter. The downfall of this scheme is that requesters can obtain the content of interest only if they opportunistically encounter a vehicle that happens to have it. In [16], data availability has been expanded by enabling contents to be periodically broadcast by the data center to all the vehicles in highly congested roads and at intersections. This scheme has been used as a baseline by several other caching schemes [32], with more emphasis on resolving the issues of broadcast storms and improving bandwidth efficiency. However, the broadcast-based approach is rendered unsuitable for user-specific applications, such as social media access applications [16].

Proactive caching has also been explored within the context of cellular and heterogeneous networks. In [18], the dynamic nature of content popularity is considered, by adopting local and global Markov processes to model user requests. In [19], a dynamic probabilistic caching scheme that considers time-varying content popularity is proposed. In [20], the authors propose a prediction scheme to predict future content popularity, regardless of any specific content and service characteristics. In [21], proactive caching is performed at base stations while considering the dynamic nature of both the number of contents, and their popularity. In [22], a general content replacement model is designed according to a unified model of probabilistic state transition for cache replacement schemes, where the dynamic change of the cache state distribution is considered. The authors in [23] propose a learning-based proactive caching scheme that strives to accurately predict content popularity, while preserving users privacy. Most of these works consider the spatiotemporal dynamics of the content popularity, while overlooking the impact of the requesters mobility on the spatiotemporal availability of replicas, particularly in highly dynamic environments, as the case in VANETs.

Proactive caching has been further investigated for downloading large contents in VANETs. These contents are not always feasible to be acquired all at once due to the short connectivity period between vehicles and RSUs [25]. Thus, such schemes divide the data into smaller chunks and aim at minimizing the download time of all the chunks constituting the entire content. Note that in large contents, multiple requests are typically issued by the requesters to fully retrieve the data [26]. In proactive caching schemes that accommodate large-sized contents, a vehicle typically sends a request for the first chunk, and then proactively acquires the remaining chunks [25]–[27]. In [25], a vehicle sends an initial request, associated with information about its location, speed, and frequency of interests to the data provider. The latter then predicts the spatial location of vehicles to pre-cache the remaining data chunks at nearby RSUs. However, the mobility prediction scheme assumes that vehicles are moving in one direction and at a constant velocity [25]. In [26] and [27], RSUs have been used to store content chunks while modeling the spatial mobility patterns of vehicles. However, the authors assume a constant velocity of vehicles and disregard the unstable conditions that could affect their mobility [26], [27]. In addition, RSUs require very large investments, so they might not be densely deployed [10]. Recently, some proactive caching schemes have proposed the utilization of roadside parked vehicles to download large-sized contents [24], [33].

Most of the schemes that accommodate the download process of large-sized contents, require multiple transmissions, where proactive caching in such schemes involve waiting for an initial request to be issued before proactively acquiring the remaining chunks, which triggers a waiting delay [25]–[27]. In addition, most of these schemes are not suitable for social media access applications with single transmissions. Such schemes also mostly focus on the spatial location of vehicles, poorly model the mobility dynamics of users, and fail to consider the spatiotemporal availability of replicas for requesters as they pass by the caching nodes [25]–[27].

In [34], we have proposed a proactive caching scheme for social media access applications with single transmissions. However, it has been assumed that all the necessary information required to make proper cache placement decisions are readily available, and a rather probabilistic approach is adopted to indicate varying percentages of vehicles with erroneous information.

In this paper, in contrast to existing works, we propose a predictive proactive caching framework that performs a prediction mechanism to predict the future spatiotemporal availability of vehicles. This, along with knowledge of the requested contents of users, are taken into consideration in order to enable caching to occur ahead of time. The proposed framework is suitable for social media access applications that require single transmissions. By accurately predicting the spatiotemporal availability of vehicles, the proposed framework strives to ensure that the cached replicas are available to each requester during its estimated period of encounter with the road segment at which the caching parked vehicle resides. It also strives to ensure a certain QoS by ensuring that the requested replicas are acquired before a certain deadline. As opposed to existing works, we predict the period of encounter of vehicles with each road segment along their trajectories. We then employ such predictions to maximize cache hits by prefetching and storing data at caching spots that yield maximum prediction certainty in their spatiotemporal availability for requesters, such that a certain QoS is maintained. In addition, we introduce a proactive caching benchmark to provide the upper bound on the reachable potential of proactive caching schemes. This is done by cultivating the optimal solution of the proactive cache placement problem under perfect knowledge of all spatiotemporal dynamics. This benchmark can help researchers quantify the potential gains of proactive caching schemes, evaluate their performance according to the gap to the optimal solution, and decide whether there is a room for improvement.

B. Travel Time Prediction in VANETs

The most prominently used travel time prediction models can be classified into four categories; historical average models, regression models, filtering models, and machine learning models [35], [36]. Historical average models predict future travel time based on historical data collected from previously observed trips. However, they work under the assumption that traffic conditions remain stationary [37].

In regression models, a multivariate statistical technique is applied to determine the correlation that exists between a set of independent variables and a given dependent variable [35]. In contrast to historical average models, linear regression models can perform adequately under unstable traffic conditions [35], [36]. However, the applicability of such models is typically limited due to the inter-correlation between variables in transportation systems [35]. The Autoregressive Integrated Moving Average (ARIMA) model [38] has also been explored.

Filtering models, such as the Kalman filter model [39], exhibit the dynamic capability of continuously updating the state variable (i.e., the travel time) as new observations manifest. However, data fluctuations might lead to difficulties in solving the time lag [36], [39].

Machine learning models, such as Artificial Neural Networks (ANNs), are highly popular in predicting travel time as a result of their ability to solve difficult non-linear relationships [40]. ANNs have shown significant superiority over other models in terms of prediction accuracy, including the historical average and regression models, as demonstrated in the studies conducted in [40], [41].

Recently, deep learning models have been explored in travel time prediction [42], [43]. It has been shown that deep learning models, such as Long Short-Term Memory (LSTM) neural networks, can render significant improvements in terms of accuracy in the travel time prediction problem, compared to historical average models, regression models, Kalman filter models [42], and ANNs [40]. Deep learning models that have been explored in travel time prediction include Deep Neural Networks (DNNs) [44], Gated Recurrent Unit (GRU) neural networks [45], Recurrent Neural Networks (RNNs) [46], the combination of DNN and RNN [47], and LSTMs [42]. Compared to other deep learning models, it has been shown that LSTMs have the ability to learn long-term dependencies, and are insensitive to gap length [43]. Thus, they are better suited for time series prediction than other deep learning models, particularly when targeting long-term travel time predictions [43].

As opposed to short-term predictions, long-term travel time predictions have been mostly overlooked in LSTMs [48]. The main goal of this type of models is to predict the travel time at a specific day and time at least one week ahead, thus enabling multi-step ahead predictions. Existing long-term prediction models do not consider the effect of dynamic weather conditions [48]. In addition, they are based on the Backpropagation algorithm (BP) that relies on Gradient Descent (GD), which can lead to local minima entrapment [40], [41]. Furthermore, existing models only focus on predicting the average travel time, which fails to account for the different driver behavior exhibited by each individual user. In contrast, in this work, we propose an LSTM prediction model that offers the following contributions: 1) enable long-term predictions, 2) handle dynamic weather conditions in such long-term predictions, 3) strive to improve the training process by proposing the use of Particle Swarm Optimization (PSO) rather than GD-based BP to train the LSTM model, and 4) provide personalized estimates of the travel time of each user by taking the varying driver behavior into consideration.

III. Predictive Proactive Caching Framework

In the Predictive Proactive Caching Framework (PPCF), we assume that requesters subscribing to the service grant the data center (i.e., the original data provider) access to their trajectories, as well as their social media profiles. This information is granted in exchange for the service and according to a privacy agreement. Each user indicates his/her maximum acceptable delay upon subscribing to the service. Thus, the deadline of each user's request is determined. We assume that the deadline is specific to each human user, and that it is acquired by enabling the service/application to monitor the duration for which each user endures delay before giving up and closing the application. Note that we are targeting recurrent users that keep using the service. Thus, the average historical information of each user (tolerable delay) is used to determine his/her desired QoS. The data center is responsible for choosing the optimal road segments for caching.

The data center is equipped with a prediction module that allows it to predict three pieces of information. First, it predicts the request time of each user using request patterns predictors, viz. the work in [49]. Second, the data center estimates the posting frequency of a given public figure. This information indicates the Time-to-Live (TTL) of the data. Note that the TTL of any content indicates the expected time before a public figure publishes a new post, rendering the previous data obsolete. The reason for this is that in most social media platforms, posts tend to be displayed in a chronological order to enable users to see the most recent posts first. This often makes older posts get so pushed down the users timeline that they might end up missing them. Thus, we impose this restriction in order to give more priority to the most recent posts, particularly with the huge amount of social media traffic that we might not have enough caching resources to fully accommodate. Third, the data center predicts the period of encounter of the requester with each road segment along its specified trajectory.

In the proposed prediction module, we focus on the prediction of the period of encounter only. We do so due to the fact that the prediction accuracy can be significantly affected by various factors, including dynamic weather conditions.

After a road segment is selected, the data center sends the replica to the parked vehicle that has the maximum available cache capacity at the selected road segment to be cached. Note that parked vehicles that are willing to dedicate part of their storage capacity to the caching service are solicited in exchange for some incentives, such as free parking, sponsored shopping discounts, free battery charging services at electrical vehicles charging stations, etc. In the following subsection, we present the system model.

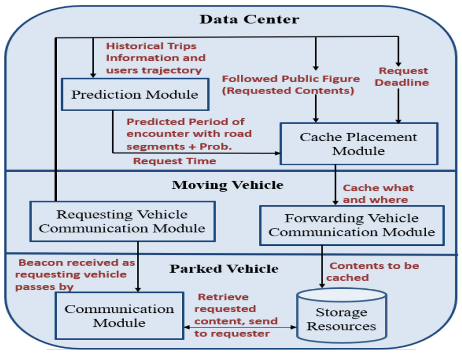

Data Center
Historical Trips Information and users trajectory
Prediction Module
Followed Public Figure (Requested Contents)
Request Deadline
Predicted Period of encounter with road segments + Prob.
Request Time
Cache Placement Module
Moving Vehicle
Cache what and where
Requesting Vehicle Communication Module
Forwarding Vehicle Communication Module
Beacon received as requesting vehicle passes by
Parked Vehicle
Contents to be cached
Communication Module
Retrieve requested content, send to requester
Storage Resources

Fig. 1. System architecture.

A. System Model

Let U be the set of users subscribing to the service. Every user $u \in U$ can request a certain data item $d \in D$, where D is the set of recent contents published by public figures followed by the users. Each data item d has a time-to-live TTL_d, after which a new post is generated by the corresponding public figure. Such a new post is considered a new version of d. Each version of a data item d, denoted l^d, is associated with a generation time g_l^d, and an expiry time t_{lexp}^d. The trajectory of every user u is denoted T_u. Note that $T_u = r_1, r_2, .. r_k$ represents the sequence of road segments along the vehicle's trajectory, where r_j denotes the road segment encountered by the vehicle. The set of all road segments is denoted R. A road segment $r_j \in R$ represents a directed edge e_{xy} between two different intersections I_x and I_y, where $e_{xy} \neq e_{yx}$. The time at which a given l^d is cached at a road segment r_j is denoted t_{lcach}^{jd}. The maximum cache capacity at r_j is denoted ζ_j.

Request generation is modeled using the Zipf-like distribution, which is controlled via a skewness factor, denoted α. The latter controls the skewness of the popularity distribution. Note that it is assumed that all requested contents are highly popular, since we focus on caching the contents generated by public figures. The time of request of user u for data d is denoted τ_d^u, and the deadline of the request is denoted η_d^u. A time period, denoted ϵ_j^u, is assigned to each road segment $r_j \in T_u$ along the trajectory of u. This period indicates the start time $t_{u,j,s}$ and the end time $t_{u,j,e}$ during which it is estimated that user u will encounter r_j (i.e., its time of arrival and departure to and from r_j). A probability of encounter, denoted P_j^u, is associated with each period of encounter. Such a probability represents the level of certainty of the estimated period.

The problem of estimating the period of encounter can be considered as a travel time prediction problem considering that the users can provide the time at which they begin their daily trips. This indicates the time of arrival to the first road segment along their trajectory. The time of departure from any road segment r_j along the user's trajectory can be calculated as given by (1), where $\gamma_j^{t_{u,j,s}}$ is the estimated travel time along r_j at time $t_{u,j,s}$. The time of arrival at any road segment r_j is equal to the departure time from the previous road segment along the user's trajectory (i.e., $t_{u,j,s} = t_{u,j-1,e}$). Thus, once the travel time is estimated, the period of encounter of each vehicle with each road segment along its trajectory can be determined. Fig. 1 depicts the system architecture, as well as the interaction between its key players; the data center, moving vehicles, and parked vehicles.

$$t_{u,j,e} = t_{u,j,s} + \gamma_j^{t_{u,j,s}} \tag{1}$$

It can be deduced from the way the period of encounter is estimated that the effect of the travel time prediction error tends to accumulate as users move further along their trajectory. This accumulation occurs since the arrival time of any vehicle at any road segment r_j along its trajectory $t_{u,j,s}$ relies on the departure time from its previous road segment r_{j-1}, which in turn depends on the estimated travel time at r_{j-1}. Thus, the first road segment along a user's trajectory tends to render the least error compared to subsequent road segments, whereas the last one renders the highest error. Accordingly, the closer a road segment is to the first road segment in the sequential order of a user's trajectory, the lower the prediction error, and thus the higher the certainty. Thus, the probability of encounter P_j^u is calculated as given by (2), where $\Lambda_{r_j}^u$ is the position of r_j in the sequential order of trajectory T_u of user u.

$$P_j^u = 1 - \frac{\Lambda_{r_j}^u - 1}{\sum_{i=1}^{|T_u|} \Lambda_{r_i}^u - 1} \tag{2}$$

Note that the data center can have access to a plethora of traffic monitoring services that provide the most recent traffic updates. This includes the actual average travel time at every road segment. Parked vehicles can also periodically report information pertaining to the actual average travel time at the road segments where they reside. Such updates can be used to adjust the cache placement decisions if the actual values of the average travel time differ from the predicted values.

In our system, the application implements a reliable data transfer protocol, with error recovery, in order to ensure the delivery of all packets. Upon data transmission, the sender waits for an acknowledgment from the receiver. If no acknowledgment is received, the packet is retransmitted. The rate at which the vehicles are moving compared to the rate of data propagation enables such a recovery protocol to ensure the robustness of data transmission under error-triggering channel conditions.

B. Prediction Module

In this section, we present the proposed PSO-LSTM prediction module, including the data preparation stage, the model prediction methodology, and the training procedure.

1) Data Preparation: The dataset used is generated using the Simulation of Urban MObility (SUMO) tool [50]. Using SUMO, we create a road topography that consists of 120 different edges. We emulate the traffic behavior of users that have a repetitive pattern in terms of the daily route they follow. This is done over a period of 6 months, where the traffic of one day is simulated over a period of 2000 seconds. Only two types of days, namely weekdays and weekends, are considered per week. Each type triggers a different traffic distribution. We divide the total period of 2000 seconds per day into a number of consecutive time intervals, each of which is composed of 30 seconds. Thus, for each edge, there is a total of 66 rows representing the travel time values per day at that edge, and 528 rows per month. The 6-month period involves 3,168 rows per edge, and a total of 380,160 for all edges.

Each of the aforementioned rows includes the following information: the weather condition, the day of the week, the time of the day, the road segment ID (i.e., edge ID), and the average travel time. We consider three types of weather conditions (sunny, moderately snowing, and heavily snowing), as well as two types of days (weekdays, and weekends). In [51], real data sets were used in order to facilitate the calibration of SUMO to different weather conditions, and it has been shown that the effect of weather conditions can be simulated by varying certain parameters in SUMO. Thus, we have varied such parameters according to the findings in [51]. Note that aside from the time of the day and the average travel time, the remaining aforementioned features are categorical features. Thus, we represent each of them as a vector of 1's and 0's using one-hot encoding [52].

In long-term travel time predictions, as the case in our proposed model, the prediction of future travel time along any given road segment at a certain time period on Saturday for example requires exploiting the data representing the travel time values at the same time period in the N previous Saturdays. Accordingly, the data preparation stage converts the aforementioned rows of the dataset into a sequential structure, where each sample is composed of N time steps, each of which consists of the previously discussed features.

2) Long Short-Term Memory Model (LSTM): An LSTM has a chain-like structure that is composed of a number of cells in its hidden layer (LSTM cells). An LSTM cell consists of three gates, referred to as input gate, forget gate, and output gate [42]. Such gates control the flow of information to the cell state [42], [48]. At time t, the input is denoted x_t, which is a multivariate input. It is a matrix of size $X \times B$, where X is the size of the input features, and B is the batch size. The cell input state is denoted $\tilde{C}_t$, the cell output state is denoted C_t, and its preceding state is denoted C_{t-1}. The hidden layer output and its previous output are denoted h_t and h_{t-1}, respectively. The size of $\tilde{C}_t$, C_t, h_t, and h_{t-1} is $H \times B$, where H is the number of hidden units [42]. The input, forget, and output gates are denoted i_t, f_t, and o_t, respectively, and are all of size $H \times B$.

In LSTM, both C_t and h_t are passed on to the next cell. In order to calculate C_t and h_t, the following procedure, referred to as the forward propagation procedure, is performed [42]:

1) The input, forget, and output gates are first calculated using (3), (4), and (5), respectively. The cell input state is then calculated as given by (6). The matrices W_x^i, W_x^f, W_x^o, and W_x^C are the weight matrices of size $H \times X$ that connect the input x_t to the input gate, forget gate, output gate, and the cell input state, respectively. In contrast, the matrices W_h^i, W_h^f, W_h^o, and W_h^C are the weight matrices of size $H \times H$ that connect h_{t-1} to the input gate, forget gate, output gate, and the cell input state, respectively. The matrices b_i, b_f, b_o, and b_C are the bias matrices of size $H \times B$ of the input gate, forget gate, output gate, and cell input state, respectively.

$$i_t = \sigma(W_x^i.x_t + W_h^i.h_{t-1} + b_i) \tag{3}$$

$$f_t = \sigma(W_x^f.x_t + W_h^f.h_{t-1} + b_f) \tag{4}$$

$$o_t = \sigma(W_x^o.x_t + W_h^o.h_{t-1} + b_o) \tag{5}$$

$$\tilde{C}_t = \tanh(W_x^C.x_t + W_h^C.h_{t-1} + b_C) \tag{6}$$

2) The cell output state C_t is calculated using (7).

$$C_t = i_t * \tilde{C}_t + f_t * C_{t-1} \tag{7}$$

3) The hidden layer output h_t is calculated using (8).

$$h_t = o_t * tanh(C_t) \tag{8}$$

The aforementioned steps require learning the weight and bias matrices that would render the minimum prediction error. Thus, the LSTM model needs to be trained in order for these matrices to be learned. We provide a detailed discussion of the proposed training procedure below.

3) PSO-LSTM Training: In most existing LSTM-based models, the training process is typically based on the gradient descent backpropagation algorithm (BP) [42], [48]. This algorithm can lead to local minima entrapment. Thus, we propose to exploit the use of PSO during the training process since it has the capability of significantly expanding the search space compared to the gradient descent algorithm, thus increasing the chance of finding the global minima [53]–[55].

In PSO, every solution of a given problem is represented by a particle that can iteratively navigate in the search landscape for the purpose of reaching the best solution. The total number of particles is denoted S, the total number of iterations is denoted K, and the total number of components is denoted J. The position of each component $j \in J$ of each particle $p_i \in L$ at iteration $k \in K$ is iteratively updated by considering two vectors, namely the position vector and the velocity vector, denoted Z_{ij}^{k+1} and V_{ij}^{k+1}, respectively. In the proposed PSO-LSTM, the position and velocity of each particle is expressed in the form of a matrix. In particular, the components of each particle are the aforementioned weight and bias matrices, whose optimal values (i.e., position) need to be learned. Note that the position of each particle is evaluated based on a certain fitness function. In the proposed model, this function is the Root Mean Square

Error (RMSE), which is given by (9), where Q is the set of observations/samples, and $|Q|$ is the number of observations, $\tilde{x}_q$ is the q^{th} predicted value of the travel time and x_q is its corresponding q^{th} observed value.

$$RMSE = \sqrt{\frac{\sum_{q=1}^{|Q|}(\tilde{x}_q - x_q)^2}{|Q|}} \tag{9}$$

In each iteration k, the matrices Z_{ij}^{k+1} and V_{ij}^{k+1} of the component j of each particle p_i are calculated using (10) and (11), respectively [55]. In (10), P_{ij}^k is the best local position of the j^{th} component of p_i, and G_j^k is the best global position of the j^{th} component among all particles [55].

$$V_{ij}^{k+1} = \omega V_{ij}^k + c_1 r_1 (P_{ij}^k - Z_{ij}^k) + c_2 r_2 (G_j^k - Z_{ij}^k) \tag{10}$$

$$Z_{ij}^{k+1} = Z_{ij}^k + V_{ij}^{k+1} \tag{11}$$

Note that the best position indicates the solution rendering the minimum RMSE. The parameter ω is the inertia weight that lies in the range [0,1] and controls how much the particle's movement is influenced by its preceding motion [55]. The parameters c_1 and c_2 are the learning coefficients of the local and global solution, respectively [55]. They help weigh the importance of the previous experiences of the particles [55]. The parameters r_1 and r_2 are two random numbers in the range [0,1] that are used to control the randomness of the search [54]. Such random parameters can help avoid premature convergence, thus increasing the chance of finding the global optima [55].

Based on the aforementioned discussion, the PSO-LSTM training procedure works as follows:

1) Set the number of components J to 13, where the components represent the weight and bias matrices. Initialize the the local minimum error of each particle, denoted RMSE$_i^{Lmin}$, as well as the global minimum error, denoted RMSE^{Gmin}. For each particle $p_i \in M$, and for each component $j \in J$, initialize the position and velocity matrices, Z_{ij}^k and V_{ij}^k, and set the best local position P_{ij}^k, as well as the weight and bias matrices to the corresponding Z_{ij}^k.
2) In each iteration $k \in K$, do the following for each particle $p_i \in M$:
   a) Perform the LSTM forward propagation procedure to calculate the predicted value of each sample $q \in Q$. This is done using (3)– (17).
   b) Based on the predicted travel time values acquired, calculate the error $RMSE_i^k$ using (9).
   c) If RMSE$_i^k$ is less than the particle's local minimum error, set the latter to RMSE$_i^k$, and its best local position to Z_{ij}^k. Otherwise, they remain the same.
   d) Determine the particle, denoted p_b, that renders the minimum error among all particles (i.e., the one that has the minimum local RMSE). If the particle's local minimum error is less than the global minimum error, set RMSE^{Gmin} to the particle's local minimum RMSE, and set p_b to p_i. Otherwise, RMSE^{Gmin} remains the same.
3) Once all the particles are processed, if RMSE^{Gmin} has changed, the best global position of each component G_j^k is set to that of p_b (i.e., $G_j^k{=}Z_{bj}^k$).
4) For each particle, update the position of the weight and bias matrices using (10) and (11).
5) After finishing the last iteration, the global best position G_j is acquired $\forall j \in J$. The final weight and bias matrices are set accordingly.

Note that the complexity of the aforementioned training procedure is O(SQNW) per iteration, where S is the number of particles, Q is the number of training samples, N is the number of time steps, and W is given by $4H^2 + 4HB + 3H + OH$ [56]. As previously mentioned, H, B, and O are the dimensions of the weight matrices. Despite the relatively long convergence time that this training procedure requires, it is worth mentioning that it is performed by the data center offline. Also, it is only repeated when large enough new datasets that would trigger updated parameters become available. We assume that such offline training is triggered once on a weekly or monthly basis. Accordingly, the practicality of the system is still preserved.

Once the training procedure is terminated, and all hyperparameters are determined, we perform the LSTM forward propagation procedure on the testing data to predict the average travel time on each road segment at each time interval. We then estimate the travel time of each individual user. In order for this to occur, we perform the following procedure after training the model.

4) Travel Time Estimation of Individual Users: In our proposed prediction scheme, we consider the fact that the personalized travel time of different drivers can deviate from the estimated average travel time based on their driving behavior. For instance, more cautious drivers tend to move relatively slower than most average drivers, while expert drivers can move a little faster. Thus, we classify the users in our dataset into three classes; class 1: cautious, class 2: average, and class 3: expert. This is done by calculating the degree of their membership to each class, denoted ϑ_{ij}. The degree of membership of user i to class 1, ϑ_{i1}, is calculated as the ratio of the number of times that the individual travel time of user i subceeds the average travel time by a certain threshold, denoted th_{dm}, to the total number of samples. In contrast, ϑ_{i3} is the ratio of the number of times that the individual travel time of user i exceeds the average travel time by th_{dm}, to the total number of samples, while ϑ_{i2} is the ratio reflecting the number of times that neither the first case nor the second one applies. The class rendering the highest membership is the one to which user i belongs. In order to estimate the individual travel time of each user belonging to each class, we do the following:

1) We cluster the training data into Y clusters, so that those belonging to users moving along the same road segment on the same type of day, as well as at the same time interval, and under the same weather conditions over the available months, are grouped together.
2) Each cluster $y \in Y$ is then divided into B_y sub-clusters enclosing users in y whose movement is associated with the same month and the same week.

3) For each cluster $b \in B_y$, $\forall y \in Y$, we calculate the standard deviation, $SD_{b,y}$ of the users' individual travel time. $SD_{b,y}$ is calculated based on (12), where $\acute{\gamma}_{b,y}$ is the average travel time of the users in cluster $b \in B_y$, $\gamma_{i,b,y}$ is the individual travel time of user $u_i \in b$, and $n_{b,y}$ is the number of users in $b \in B_y$.

$$SD_{b,y} = \sqrt{\frac{\sum_{i=1}^{n_{b,y}} (\gamma_{i,b,y} - \acute{\gamma}_{b,y})^2}{n_{b,y}}} \qquad (12)$$

4) We then calculate the pooled standard deviation for each cluster $y \in Y$, denoted SD_p^y to estimate the aggregated standard deviation pertaining to all the sub-clusters B_y belonging to y. Note that this will be used to estimate the final individual travel time of users. The value of SD_p^y is calculated using (13).

$$SD_p^y = \sqrt{\frac{\sum_{b=1}^{B_y} (n_{b,y} - 1) SD_{b,y}^2}{\sum_{b=1}^{B_y} (n_{b,y} - 1)}} \qquad (13)$$

5) Once the pooled standard deviation is determined, the estimated individual travel time of each user u in the testing set satisfying the same conditions in y can be calculated based on (14), where $\bar{\gamma}_y$ is the predicted average travel time.

$$\tilde{\gamma}_{u,y} = \begin{cases} \bar{\gamma}_y - SD_p^y & \text{if u is a cautious driver} \\ \bar{\gamma}_y & \text{if u is an average driver} \\ \bar{\gamma}_y + SD_p^y & \text{if u is an expert driver} \end{cases} \qquad (14)$$

C. Optimal Proactive Caching

In this subsection, we present the optimal solution of the proactive cache placement module, called the Vehicular Optimal Proactive Caching (VOPC). We provide a detailed discussion of the ILP cache placement problem formulation in VOPC. Our objective is to maximize cache hits that ensure the spatiotemporal availability of the cached replicas for the users to proactively acquire within a certain time frame. In order to ensure such an availability, the following six assignment restrictions need to be taken into consideration when making a caching decision at road segment r_j to satisfy the request of user u for data item d:

1) r_j must be part of the user's trajectory ($r_j \in T_u$).
2) The deadline of the request should be after the user u starts its period of encounter with r_j ($\eta_d^u > t_{jstart}^u$). This is to ensure that the user does not acquire the cached data after the deadline of the request.
3) The cached replica should still be valid by the time the user starts passing by r_j ($t_{lexp}^d > t_{jstart}^u$). This is to guarantee that the data is acquired before its expiry time.
4) The cached replica must still be valid by the time the user requests it ($t_{lexp}^d > \tau_d^u$).
5) The replica must be already cached at r_j before the user is done passing by the road segment ($t_{jend}^u > t_{lcach}^{jd}$).
6) The replica should already be cached and available at r_j before the deadline ($\eta_d^u > t_{lcach}^{jd}$).

The set of requests satisfying the aforementioned restrictions for each road segment $r_j \in R$ can be predetermined. To do so, for each $r_j \in R$, we specify the set of users F_j^d who request d and abide to restrictions (1) and (2). In order to check the remaining restrictions, potential updated versions of each data item $d \in D$, as well as their caching and expiry time need to be determined. Thus, we specify the potential versions of d that can satisfy the requests of the users in F_j^d. Such versions are the ones that will be generated during the period of encounter of the requesters in F_j^d with r_j. This period begins from the start time of their earliest period of encounter with r_j, denoted a, until the end time of the latest period of encounter, denoted b.

As depicted in Fig. 2(a), in order to specify the potential versions of a data item d, we use the version available at the current time, denoted l_c^d, whose generation time $g_{l_c}^d$ is known. Starting from l_c^d, we determine the subsequent versions. The first version that can satisfy the requests of F_j^d is the first one whose expiry time $> a$, whereas the last version is the one whose expiry time $> b$. For example, as shown in Fig. 2(a), the candidate versions of d_1 that can be assigned to requesters satisfying restrictions (1) and (2), are determined. The start and end time of the earliest and latest periods of encounter of such requesters with r_j are given by a and b, respectively. The generation time of the current version, l_{cur}, is known to be at 8:05. The TTL of d_1 is 30 min. Thus, l_{cur} expires at 8:35, marking the generation of the next updated version l_{next} which will expire at 9:05. This replica is considered the first candidate version of d_1 that can be cached at r_j, since its expiry time is greater than a. The version l_{next+1} is the last candidate since its expiry time exceeds b.

Once the potential versions are determined, the lifetime of each version at the corresponding road segment is specified. Thus, for each road segment, we define a set of intervals A_j^d for every data item. Each interval starts from the time at which the corresponding version is to be cached at r_j, denoted t_{lcach}^{jd}, until the time it expires, denoted t_{lexp}^d. Note that the data center sends each version for caching once it is generated. Thus, t_{lcach}^{jd} is equal to $g_l^d + \Delta_{jduration}$, where $\Delta_{jduration}$ is the amount of time required for the road segment (i.e., parked vehicle) to receive the replica from the data center. The value of $\Delta_{jduration}$ is calculated as the length of the shortest path to r_j divided by the average propagation speed. As depicted in Fig. 2(b), once the versions of d_1 at r_j are determined, the set of their intervals are defined. The defined intervals represent the spatiotemporal caching slots to which the requests can be assigned. Since d_1 has two versions, the two intervals I_{1d_1} and I_{2d_1} are defined. For each interval $k \in A_j^d$, and based on the aforementioned restrictions (3-6), we define the set of users belonging to F_j^d that consider k a feasible spatiotemporal caching slot. This set is denoted B_{jk}^d.

Another important restriction is ensuring that the cache capacity limit at each road segment is not exceeded. The cache capacity of a road segment is only affected by the replicas of different data items assigned to it simultaneously. For instance, in Fig. 2(b), intervals I_{1d_1}, I_{1d_2}, and I_{1d_3} overlap. Thus, if each of them has at least one request assigned to it, the consumed cache capacity would be 3. Hence, for each road segment $r_j \in R$, we define O_j as the set of sets ψ_j, where ψ_j is the set of intervals that overlap in r_j. In Fig. 2(b), this is given

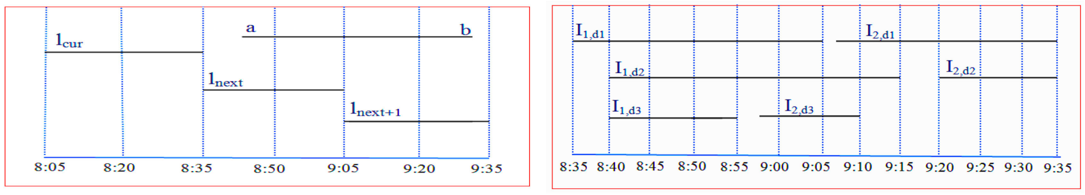


Fig. 2. An example illustrating the process of determining the spatiotemporal caching slots at road segment r_j. (a) Candidate versions of d_1 (b) Intervals for the updated versions of d_1, d_2, and d_3.

by $O_j=\{\{I_{1d_1}, I_{1d_2}, I_{1d_3}\}, \{I_{2d_1}, I_{1d_2}, I_{2d_3}\}, \{I_{1d_1}, I_{1d_2}, I_{2d_3}\}, \{I_{2d_1}, I_{2d_2}\}\}$.

The objective is to assign the requests to the proper spatiotemporal caching slots, such that the probability of cache hits that guarantee the acquisition of the data within the desired time frame is maximized. As previously mentioned, for each caching slot, the set of requests that consider it feasible for caching is denoted B^d_{jk}. B^d_{jk} is predetermined based on the aforementioned six assignment restrictions.

The problem is formulated as a 0-1 Integer Linear Program (0-1 ILP), where the decision variable x^d_{ijk} is set to 1 if the request of user i for data item d is assigned to interval k at road segment j, and 0 otherwise. In the objective function, the probability of cache hits is reflected using the probability of encounter of users with the road segments. The reason for this is that such a probability, P^u_j, determines the degree of accuracy of the estimated period of encounter, and thus plays a vital role in assuring the spatiotemporal availability of the assigned replicas. For example, if user u passes all the restrictions relative to a particular spatiotemporal caching slot at r_j, but its probability of encounter with the latter is small, then another caching slot might be a better alternative for u. In addition, the way the probability is calculated enables the delay to be minimized. This is since the earlier the road segment is encountered along the trajectory of users, the higher the probability.

The aforementioned objective is subject to the constraints C1-C4. Constraint C1 specifies that each request must be assigned to at most one interval. This is to make sure that each request is assigned only once. Constraint C2 indicates that for each $r_j \in R$, the total number of overlapping intervals at r_j that have at least one request assigned to them should not exceed the maximum cache capacity ζ_j. This constraint is checked for each set of overlapping intervals $\psi_j \in O_j$ at r_j. Constraints C3 and C4 are artificial constraints designed to verify C2 (i.e., the cache capacity constraint). In order for C2 to be verified, we need to determine the number of overlapping intervals that have at least one request assigned to them. For this purpose, we define an artificial variable, δ_{jk}, which is set to 1 if at least one request is assigned to interval k at road segment j, and 0 otherwise. Thus, if $\sum_{i\in B^d_{jk}} x^d_{ijk} \geq 1$, then $\delta_{jk}=1$. Otherwise, $\delta_{jk}=0$. This can also be expressed as follows: if $\delta_{jk}=1$, then $\sum_{i\in B^d_{jk}} x^d_{ijk} \geq 1$. Otherwise, if $\delta_{jk}=0$, then $\sum_{i\in B^d_{jk}} x^d_{ijk} < 1$. Since it is not possible to include a conditional statement in the formulation problem, constraints C3 and C4 are constructed to serve the same purpose. The variable M in C3 and C4 is set to a large positive value to ensure that it is larger than the maximum possible value of the summation $\sum_{i\in B^d_{jk}} x^d_{ijk}$.

In order to verify that the conditional statement is satisfied by C3 and C4, consider that the value of δ_{jk} is set to zero in both constraints. This results in the following inequality: $-M \leq \sum_{i\in B^d_{jk}} x^d_{ijk} < 1$. Similarly, if δ_{jk} is set to 1 in C3 and C4, the inequality, $1 \leq \sum_{i\in B^d_{jk}} x^d_{ijk} < M + 2$, is obtained. Thus, C3 and C4 serve the desired purpose.

$$\max_{x^d_{ijk}} \sum_{d\in D} \sum_{j\in R} \sum_{k\in A^d_j} \sum_{i\in B^d_{jk}} P^i_j x^d_{ijk}$$

s.t.

$$\text{C1:} \quad \sum_{j\in R} \sum_{k\in A^d_j} x^d_{ijk} \leq 1 \quad \forall i \in U, \forall d \in D$$

$$\text{C2:} \quad \sum_{k\in \psi_j} \delta_{jk} \leq \zeta_j \quad \forall j \in R, \forall \psi_j \in O_j$$

$$\text{C3:} \quad \sum_{i\in B^d_{jk}} x^d_{ijk} \geq \delta_{jk}(M+1) - M \quad \forall j \in R, \forall k \in A^d_j$$

$$\text{C4:} \quad \sum_{i\in B^d_{jk}} x^d_{ijk} < \delta_{jk}(M+1) + 1 \quad \forall j \in R, \forall k \in A^d_j$$

D. Heuristic-Based Proactive Cache Placement

In order to deal with practical real-time requirements, we propose a greedy heuristic algorithm called the Predictive Proactive Cache Placement scheme (PPCP) to solve the aforementioned cache placement problem.

In PPCP, the data center divides the users into different classes, each of which is composed of the users that request the same data item. Let C be the set of classes created by the data center. Each class $c \in C$ is assigned a rank based on the popularity of the content requested by the users in c. Such popularity is calculated based on the number of users requesting the content. Based on the calculated ranks, the classes are sorted in a descending order, thus specifying the sequential order of processing each class. The data center then applies the following two stages on each class $c \in C$ sequentially:

1) Specifying the eligible spatiotemporal caching slots for each user

The same aforementioned time-constraint restrictions (1-6) indicated in VOPC are evaluated to determine the eligibility of the spatiotemporal caching slots for each user. An eligibility matrix, denoted $E^c_{m\times n\times o}$ is generated accordingly. The eligibility matrix specifies the eligible spatiotemporal caching slots for each requester within the class c, where $m = |c|, n = |R|$, and $o = |A^{d,c}_j|$. Note that, as previously mentioned, A^d_j is the set of intervals at which the potential versions of d can be cached in r_j. In order to do that, for each road segment $r_j \in R$, we perform the same aforementioned steps indicated in VOPC until we determine the set of users in c, denoted $B^{d,c}_{jk}$, that consider the interval $k \in A^{d,c}_j$ an eligible spatiotemporal caching slot. The eligibility matrix is then created based on (15).

$$e_{ijk} = \begin{cases} 1 & u_i \in B^{d,c}_{jk} \\ 0 & Otherwise \end{cases} \tag{15}$$

2) Assigning requests to the proper spatiotemporal caching slots

As previously mentioned, the goal is to maximize cache hits by maximizing the probability of encounter of the requests assigned to the designated replicas. This is while maintaining the cache capacity limit and the QoS demanded by users. Note that this QoS is maintained through the aforementioned restrictions that were used to construct the eligibility matrix.

In order to solve the cache placement problem, an iterative procedure is executed. A matrix called the assignment matrix, denoted $G^c_{m\times n\times o}$ is generated and iteratively revised to assign spatiotemporal caching slots to all users/requests within class $c \in C$.

The assignment matrix is created based on (16), where w is the iteration number, P^i_j is the probability of encounter of user i with road segment r_j. As previously mentioned in VOPC, O_j is the set of sets ψ_j, where ψ_j is the set of intervals that overlap in r_j. Also, $\delta_{jk} = 1$ if at least one request is assigned to a replica at interval k in road segment r_j. Otherwise, $\delta_{jk} = 0$.

$$g^w_{ijk} = \begin{cases} P^i_j & \text{if} g^{w-1}_{ijk} > 0, \text{and} \\ & \sum_{k\in\psi_j} \delta_{jk} \le \zeta_j, \forall\psi_j \in O_j \\ 0 & \text{Otherwise} \end{cases} \tag{16}$$

Let U' be the set of users (i.e., requests) in $c \in C$ who have not been assigned a spatiotemporal caching slot yet and C' be the set of users in $c \in C$ who have already been assigned a spatiotemporal caching slot. Initially, the iteration number w is set to 1, and the assignment matrix is set to the eligibility matrix (i.e., $g^0_{ijk} = e_{ijk}$). Also, U' is set to $c \in C$, C' is empty, and the list of assigned replicas is empty. For each interval k at each road segment r_j, δ_{jk} is set to 0 if $c = 1$. Otherwise, δ_{jk} is equal to its latest value that was assigned during processing the previous class.

The following steps are iteratively applied as long as U' is not empty:

1) The assignment matrix is modified based on (16).
2) The level of cache hits over each interval in each road segment, denoted LH_{r_jk} is determined by calculating the sum of the assignment matrix values for all the users in U' over interval k in road segment r_j. The spatiotemporal caching slot yielding the maximum level of cache hits, denoted $r_{\hat{j}\hat{k}}$, is selected for caching.
The value of $\delta_{\hat{j}\hat{k}}$ is updated by setting it to one.
3) $r_{\hat{j}\hat{k}}$ represents the spatiotemporal caching slot assigned to users (i.e., corresponding requests) who consider it an eligible caching slot. Thus, the set of users in U' with non-zero entries in the assignment matrix over $r_{\hat{j}\hat{k}}$ are grouped together in $c' \in C'$ and eliminated from the set U'. The list of assigned replicas is updated by affiliating $c' \in C'$ to $r_{\hat{j}\hat{k}}$ and the data item to be sent d^c. A Least Frequently Used replacement policy is applied if U' is still not empty and all the road segments have exhausted their full cache capacity.
4) The iteration number is incremented by one (i.e., $w = w + 1$), and the steps 1-4 are repeated. This is done until U' becomes empty. Once this occurs, the list of assigned replicas can be obtained.

IV. Performance Evaluation

In this section, we use VOPC to quantify the potential gains of the heuristic-based predictive caching approach PPCP. Towards that end, we assume perfect knowledge in VOPC (i.e., 100% prediction accuracy). We also evaluate the performance of our predictive proactive caching approach compared to a representative of the Broadcast-based Proactive Caching (BPC) approach [30]. It is implemented while ignoring the effects of the bandwidth issues that render the broadcast-based approach unsuitable for social media access applications. In addition, we compare our proposed PSO-LSTM prediction technique to the Gradient Descent-based LSTM model (GD-LSTM). Thus, we implement two versions of PPCP; one using PSO-LSTM and another using GD-LSTM. These versions are referred to as PPCP-PSO and PPCP-GD. Furthermore, in order to assess the effect of taking varying weather conditions into consideration in the prediction model, we evaluate the prediction accuracy of both GD-LSTM and PSO-LSTM compared to the long-term GD-based LSTM model proposed in [48], which has not taken the weather into consideration. We refer to it as GD-LSTM-NoWeather (GD-LSTM-NW).

In order to predict the travel time, we create an LSTM neural network. At time t, the input x_t is the historical data that includes information about the weather, the day of the week, the time of the day, the road segment, as well as the historical average time. The final output of the network, denoted $\tilde{x}_{t+1}$, is the predicted future average time on the same day along that road segment. Using the calculated value of h_t, given by (8), we calculate the predicted value as given by (17), where W_2 is the weight matrix connecting the hidden layer and the output layer, and b is the bias term of the latter. Note that the series LSTM-based prediction is based on N historical data.

$$\tilde{x}_{t+1} = W_2.h_t + b \tag{17}$$

We use the following performance metrics: 1) the average delay experienced starting from the time a request is generated until the data packet is received, 2) the packet delivery ratio, which is the ratio of data packets successfully delivered to the total number of generated data packets, 3) the cache hit ratio, which is the ratio of requests satisfied by caching nodes to the total number of received data packets, and 4) the satisfaction ratio, which is the ratio of data packets received before the specified deadline to the total number of data received.

In addition to the aforementioned metrics, two more metrics are used in order to assess the prediction accuracy of the prediction model. The first metric is the RMSE, given by (9), and the second is the coefficient of determination, referred to as R^2, which is another prominent statistical measure for evaluating the performance of prediction models. R^2 is defined as the percentage of variance in a dependent variable that is caused by its relationship with an independent variable [42]. It is given by (18), where $SS_{Regression}$, and SS_{Total} are the squared sum of the regression error, and the squared sum of the total error, given by (19) and (20), respectively. Note that x_i, and $\hat{x}_i$ are the actual and predicted values of the average travel time, respectively, while $\bar{x}$ is the mean value of the data points.

$$R^2 = 1 - \frac{SS_{Regression}}{SS_{Total}} \tag{18}$$

$$SS_{Regression} = \sum_{i=1}^{Q} (x_i - \hat{x}_i)^2 \tag{19}$$

$$SS_{Total} = \sum_{i=1}^{Q} (x_i - \bar{x})^2 \tag{20}$$

A. Simulation Setup

Simulations are conducted using the NS-3 network simulator [28]. NS-3 is integrated with Gurobi [29] to generate the optimal solution in VOPC. Simulations are performed over a 6×6 road grid topography, comprised of 120 edges. Realistic vehicular mobility traces are created using the SUMO traffic simulator [50]. The network consists of $600/km^2$ moving vehicles, 200 of which are requesters. The simulation period is set to 2000 seconds. The transmission range is set to 200 m and the beacon interval is set to 1 s. We use the IEEE 802.11p WAVE standard. Traffic updates are reported to the data center every 3 minutes. In BPC, data is broadcasted every 8 seconds.

Based on the Zipf-like distribution with a skewness factor=1.2, the interest generation is distributed among 50 public figures, each of which publishes a new post every $5-7$ minutes, rendering the previous one obsolete. The number of parked vehicles is 240, uniformly distributed among the road segments. Each parked vehicle remains at its assigned parking space for the entire simulation period. We vary the dedicated cache capacity at roadside caching units at each intersection in BPC, as well as the total cache capacity of the parked vehicles at each road segment in VOPC and PPCP. This is expressed as the percentage, denoted θ, of the total number of contents that can be requested. The time of each request is assigned a random value that lies within the trip duration of the requester. The deadline of each request is set to the request time+β. Unless otherwise specified, θ and β are set to 60% and 10 seconds, respectively. The percentage of road segments that have parked vehicles residing at them is set to 100%. The threshold th_{dm} is set to 15 seconds.

The number of hidden units H in the three LSTM models is set to 12. The batch size is set to 1, the number of epochs in GD-LSTM and GD-LSTM-NW is set to 50, whereas that of PSO-LSTM is set to 30. The number of particles is set to 40, and the values of ω, c_1, and c_2 are set to 0.5, 1.5, and 1.5, respectively. The learning rate in gradient descent is 0.1, and the number of time steps in the three models is 6. We split the dataset into 80% training data and 20% testing data.

TABLE I
PREDICTION PERFORMANCE RESULTS

	R^2			RMSE		
Algorithm	$H=10$	$H=12$	$H=14$	$H=10$	$H=12$	$H=14$
GD-LSTM-NW	76	79	77	10.25	8.89	7.54
GD-LSTM	80	84	85	7.67	6.01	5.87
PSO-LSTM	85	93	93	4.74	3.18	3.18

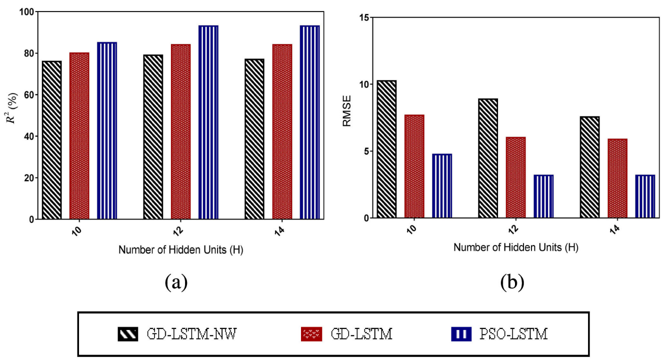


Fig. 3. RMSE and R^2 over varying hidden units in GD-LSTM-NW, GD-LSTM, and PSO-LSTM. (a) R^2 (b) RMSE.

B. Results and Discussion

In our experiments, we evaluate the performance of BPC, PPCP-GD, PPCP-PSO, and VOPC under varying cache capacity percentage θ, and varying percentage of road segments with parked vehicles κ. In addition, we evaluate the prediction accuracy of GD-LSTM-NW, GD-LSTM, and PSO-LSTM. The results acquired are discussed below. Simulation results are presented at a confidence level=90%.

1) Prediction Model Results: We first present the results of the three prediction models GD-LSTM-NW, GD-LSTM, and PSO-LSTM in terms of R^2 and RMSE in order to evaluate their performance in terms of prediction accuracy. Table I shows the numerical values of R^2 and RMSE of each of the three schemes over different number of hidden units H.

As shown in Fig. 3, PSO-LSTM yields the highest R^2 and lowest RMSE among all schemes at $H = 10$, with a difference gain of 9%, and 5% in R^2 (Fig. 3(a)) and a reduction of 53% and 38% in RMSE (Fig. 3(b)) compared to GD-LSTM-NW and GD-LSTM, respectively. GD-LSTM outperforms GD-LSTM-NW, where it yields a difference gain of 4% in terms of R^2, and a reduction of 25% in terms of RMSE.

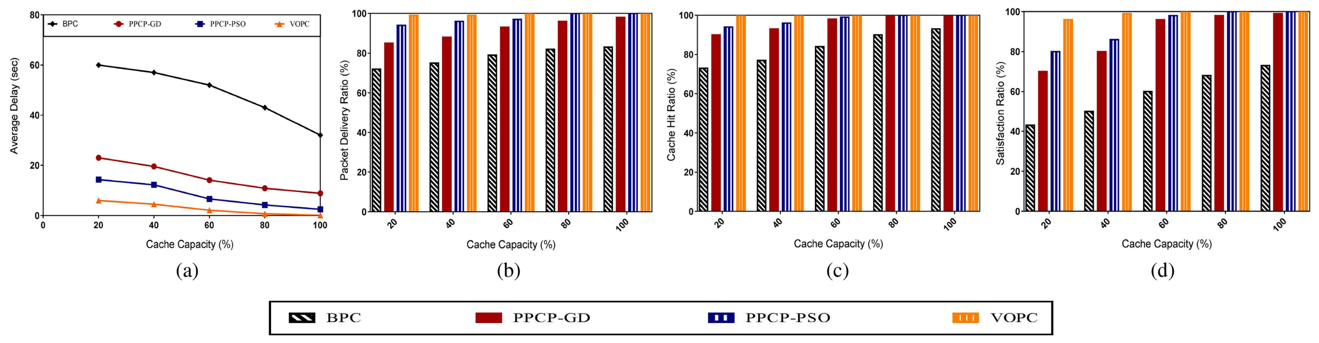


Fig. 4. Performance results of BPC, PPCP-GD, PPCP-PSO, and VOPC over varying cache capacity (θ). (a) Average delay (b) Packet delivery ratio (c) Cache hit ratio (d) Satisfaction ratio.

At $H = 12$, the leverage of PSO-LSTM further increases. It renders the highest R^2 and lowest RMSE, where it yields a difference gain of 14% and 9% in R^2, and a decrease of 64% and 47% in RMSE compared to GD-LSTM-NW and GD-LSTM, respectively. In addition, GD-LSTM improves both R^2 and RMSE compared to GD-LSTM-NW, where it renders a difference gain of 5% in R^2, and a decrease of 32% in RMSE. At $H = 14$, PSO-LSTM sustains the same performance in both R^2 and RMSE as those yielded at $H = 12$, while the corresponding values of GD-LSTM slightly improve, with a difference gain in R^2 of only 1%, and a reduction in RMSE of only 2% compared to its previous values at $H = 12$. In contrast, when H is equal to 14, the performance of GD-LSTM-NW slightly deteriorates compared to its previous values at $H = 12$. In particular, GD-LSTM-NW yields a 2% difference loss in its R^2, and a decrease of 15% in its RMSE. Since the performance of PSO-LSTM has not changed from $H = 12$ to $H = 14$, and that of GD-LSTM has only slightly improved, we set H to 12 in our experiments since the gain is not worth the time cost.

2) The Impact of Cache Capacity: In this experiment, the cache capacity percentage θ is varied from 20% to 100% to assess the performance of our proposed schemes under low, medium, and high cache capacity.

Fig. 4(a) shows the performance of BPC, PPCP-GD, PPCP-PSO, and VOPC in terms of average delay over varying percentage of cache capacity. As depicted in Fig. 4(a), the predictive approach demonstrates significant reduction in delay compared to BPC, where PPCP-GD, and PPCP-PSO improve the average delay by up to 72% and 81%, respectively, and the gap between BPC and the optimal solution VOPC reaches up to 99.7%. This is since requesters in BPC rely on opportunistic encounter with vehicles that happen to have the data. Such an encounter might take some time to occur, which could delay the process of data acquisition. The possibility of this encounter is further prolonged as the cache capacity θ decreases. This is due to the reduction in data availability. In contrast, the predictive approach enables the data to be proactively acquired within a certain time frame specified by users.

The optimal solution VOPC yields the upper bound on the potential improvement in delay. PPCP approaches VOPC as θ increases. In contrast, the gap between the two increases as θ decreases, where the gap between PPCP-GD and VOPC rises up to 64%, while that between PPCP-PSO and VOPC rises up to 48% when θ is equal to 20%. This indicates a room for improvement in the heuristic solution. The reason is that the lower the cache capacity, the higher the chance of assigning requests to road segments with which the users have more accumulated prediction error. This is since road segments yielding lower accumulated error might already be fully occupied. As a result of the higher prediction accuracy yielded by PSO-LSTM compared to GD-LSTM, the magnitude of the accumulated prediction error at road segments encountered further ahead along the users' trajectory is reduced in PPCP-PSO compared to PPCP-GD. This leads to a significant reduction in delay, reaching up to 38%, achieved by the former compared to the latter.

Fig. 4(b) demonstrates the performance of the schemes in terms of packet delivery ratio over varying θ. VOPC yields the upper bound on the reachable packet delivery ratio. The predictive approach exhibits a significant improvement over BPC, where PPCP-GD, and PPCP-PSO increase the packet delivery ratio by up to 18% and 24%, respectively, while the gap between BPC and the optimal solution VOPC reaches up to 36%. This is because the chance that the broadcasted data packets reach the requesters in BPC is largely dependent on the opportunistic encounter with the corresponding data holders. In contrast, in both VOPC and PPCP, the data is acquired by requesters as they deterministically pass by the caching nodes, thus significantly reducing the risk of packet loss.

PPCP approaches the optimal solution VOPC in terms of packet delivery ratio as θ increases, whereas the gap between PPCP and VOPC increases as θ decreases. In particular, the gap between PPCP-GD and VOPC can reach up to 15%, while that between PPCP-PSO and VOPC can reach up to 5% only. This is due to the increasing risk in PPCP that requesters pass by the designated caching nodes either before the data has been cached or after it has expired. Such a risk manifests as the cache capacity is reduced due to the higher risk that replicas are placed at road segments with higher accumulated prediction error. In contrast, as θ increases, this risk is significantly reduced. Note that PPCP-PSO outperforms its GD counterpart, with an increase of up to 11%. This is due to the higher prediction accuracy of PSO-LSTM compared to GD-LSTM, which in turn reduces the accumulated prediction error at road segments, thus reducing the risk of packet loss. As θ decreases, the risk of failing to find a caching parked vehicle at a road segment that the requester passes by along its trajectory, and that abides to all the time-sensitive constraints increases. This could force data acquisition to occur from the data center, which, as explained later, increases the risk of dropping the packets due to failure to reach the requesters.

We conduct the same experiment to evaluate the performance in terms of cache hit ratio over varying θ. As depicted in Fig. 4(c), the predictive proactive caching approach significantly outperforms BPC, where PPCP-GD, and PPCP-PSO improve the cache hit ratio by up to 23% and 29%, respectively, while the gap between BPC and the optimal solution VOPC can reach up to 37%. This is due to the fact that as θ decreases, data availability at caching nodes decreases in BPC, which makes it harder for requesters to opportunistically encounter caching nodes that have the contents they desire. In contrast, VOPC and PPCP are designed such that contents are mainly acquired from caching nodes as requesters pass by. Data acquisition from the data center could be triggered when none of the road segments that the requesters pass by, and which abide to the time-sensitive constraints, has enough cache capacity, or when the requesters pass by the designated parked vehicle that is responsible for acting as their data provider, but the data is not found. The latter could occur either because the requesters have departed the road segment before the data has been cached or they arrived after it has already expired. Note that since we are assuming perfect knowledge (i.e., 100% prediction accuracy) in VOPC, this possibility never occurs. In contrast, it could occur in PPCP. In which case, the data center can give the requesters another chance for data acquisition when it re-examines the cache placement procedure or it can directly send them a replica if there is no caching opportunity. The latter option could reduce the cache hit ratio. As previously mentioned, as θ increases, this risk is significantly reduced. Also, even if this risk occurs, there is a possibility that the data might be opportunistically acquired from nearby caching nodes, which could be parked or moving vehicles. Note that such a possibility increases as θ increases, due to the increased data availability at caching nodes.

VOPC provides the upper bound on the potential cache hit ratio increase under varying θ. The gap between PPCP and VOPC can reach up to 6% between PPCP-PSO and VOPC, and 11% between PPCP-GD and VOPC. This indicates that PPCP approaches the optimal solution even under low values of θ. This can be largely attributed to the aforementioned alternatives that enable the data to still be acquired from caching nodes even when requesters reach their designated caching slots too early or too late to be procured. Since the aforementioned risk is reduced in PPCP-PSO compared to its GD counterpart, it achieves a higher cache hit ratio compared to PPCP-GD, with an improvement of up to 5%.

Fig. 4(d) depicts the effect of varying θ on the satisfaction ratio of users. It shows that the predictive proactive caching approach yields a significant improvement in terms of satisfaction ratio compared to BPC, where PPCP-GD, and PPCP-PSO improve it by up to 63% and 81%, respectively, while the gap between BPC and the optimal solution VOPC reaches up to 116%. This can be attributed to the fact that in contrast to VOPC and PPCP, BPC does not attempt to abide by a specific QoS demanded by users. This makes users more susceptible to acquiring the data later than their desired deadline. The satisfaction ratio of the predictive approach significantly increases as θ increases. This can be attributed to the significant reduction in delay yielded by VOPC and PPCP, due to the aforementioned reasons. In addition, as θ decreases, the chance of finding caching nodes that the requesters pass by before their deadline decreases, and if found, they might have a high accumulated prediction error, thus increasing the risk of a low satisfaction ratio. VOPC provides the upper bound on the potential satisfaction ratio. As θ increases, PPCP gets closer to VOPC. In contrast, the gap between PPCP and VOPC increases as θ decreases, reaching up to 33% in PPCP-GD, and 16% in PPCP-PSO. This indicates a room for improvement in the heuristic solution.

3) The Impact of the Percentage of Road Segments with Parked Vehicles: In this experiment, the percentage of road segments that have parked vehicles residing at them, denoted κ, is varied from 20% to 100%.

Fig. 5(a) demonstrates the effect of varying κ on the average delay. As shown in the Figure, varying κ has no effect on the performance of BPC. This is since BPC does not use parked vehicles for caching. Due to the same aforementioned reasons, the predictive approach significantly improves the delay compared to BPC, where PPCP-GD, and PPCP-PSO improve it by up to 73% and 79%, respectively, while the gap between BPC and the optimal solution VOPC reaches up to 97%. Note that VOPC and PPCP yield higher delay as κ decreases. This can be attributed to the fact that decreasing κ limits the number of road segments available for cache selection, which increases the risk of caching the data at road segments with which the requesters encounter later on along their trajectory. In PPCP, the lower availability of road segments for caching can force the data center to place replicas at road segments with which the requesters have low probability of encounter (i.e., high accumulative prediction error). This poses the risk of having the data cached at road segments that requesters encounter after their specified deadline. This risk further increases as the prediction accuracy decreases. This explains the leverage gained by PPCP-PSO, which reaches up to 34%, compared to PPCP-GD. The gap between PPCP and VOPC increases as κ decreases, reaching up to 65% between PPCP-GD and VOPC, and 45% between PPCP-PSO and VOPC, thus indicating a room for improving the delay in the heuristic solution. This can be attributed to the fact that the aforementioned risk is handled by PPCP on a group-by-group basis, where it divides the users into groups and process them sequentially to allocate the requests to spatiotemporal caching slots. Thus, a replica might miss a better caching slot merely because a different group of requests have been handled first, and that slot has already been occupied. In contrast, the optimal solution can handle all requests at once and a 100% prediction accuracy is assumed. Accordingly, it can optimally allocate the requests to reach the global solution. Note that the gap between PPCP and VOPC significantly decreases as κ increases.

Fig. 5(b) shows the effect of varying κ on the packet delivery ratio, where BPC outperforms PPCP-GD, and PPCP-PSO by

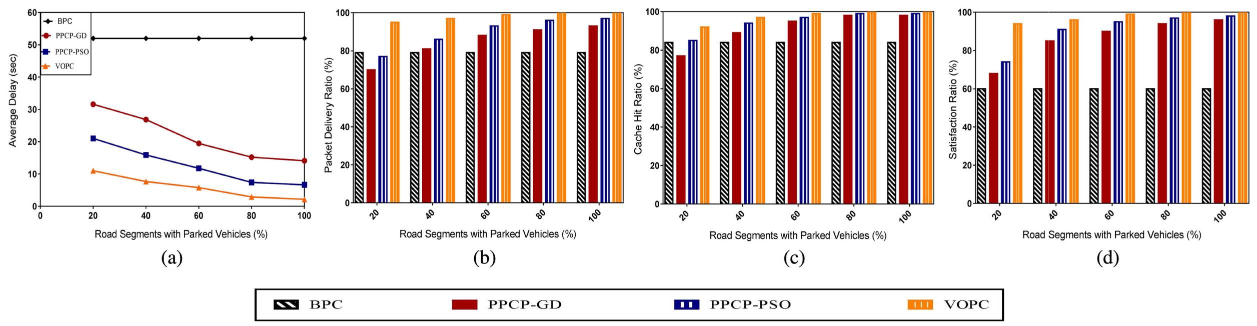


Fig. 5. Performance results of BPC, PPCP-GD, PPCP-PSO, and VOPC over varying percentage of road segments with parked vehicles (κ). (a) Average delay (b) Packet delivery ratio (c) Cache hit ratio (d) Satisfaction ratio.

11% and 2% when κ is equal to 20%. As κ increases, PPCP starts to improve the packet delivery ratio compared to BPC. In particular, PPCP-GD, and PPCP-PSO improve it by up to 18% and 22%, respectively, while the gap between BPC and VOPC reaches up to 27%. This is because as κ decreases in PPCP and VOPC, the risk of having requesters that do not pass by any road segment that has parked vehicles increases. This forces data acquisition to occur either from the far-away data center or from mobile caching nodes that are opportunistically encountered. Data acquisition from the data center suffers from the risk of dropping the packets due to failure to reach the requesters. This could occur due to the fast movement of the requesters combined with the slow propagation of packets, which could lead to the latter reaching the requester's position after it has already moved too far away. In PPCP, failure to reach the requester can further increase as a result of erroneous estimations of their location, as well as the lack of a stable residence for the replicas to be cached along the route of the requesters. Note that, as the number of road segments available for cache selection decreases in PPCP, the risk of allocating replicas to road segments that have high accumulative prediction error increases. This increases the risk of caching the data at road segments that requesters pass by either before the data has been cached or after it has already expired. Consequently, more reliance on opportunistic encounter with caching nodes, or resorting to the data center for data acquisition takes place. Since the broadcast-based approach increases data availability at caching nodes by broadcasting the data, which increases the possibility of opportunistic encounter, it renders a higher packet delivery ratio at $\kappa = 20\%$ compared to PPCP-GD and PPCP-PSO. However, as κ increases, the aforementioned risks decrease, which enables the predictive approach to render a much higher packet delivery ratio than BPC. PPCP-PSO outperforms PPCP-GD in terms of packet delivery ratio by up to 10%. This is due to the high prediction accuracy rendered by PSO-LSTM compared to GD-LSTM, which helps reduce the accumulated prediction error of the period of encounter at road segments. VOPC yields the highest upper bound on the potential packet delivery ratio. PPCP tends to approach VOPC as κ increases, while the gap between the two solutions increases as κ decreases, reaching up to 37% between PPCP-GD and VOPC, and up to 21% between PPCP-PSO and VOPC.

In Fig. 5(c), the impact of varying κ on the cache hit ratio is depicted. It is shown that as κ increases, the cache hit ratio increases. As κ decreases, the risk of resorting to the data center for data acquisition increases. This is due to the increased possibility of the aforementioned risks. When κ is equal to 20%, BPC slightly outperforms PPCP-GD, with an increase of 2%, while PPCP-PSO improves the cache hit ratio by only 2%. This is due to the fact that some requesters do not pass by any of the road segments that have parked vehicles for caching. As κ increases, all the predictive approach schemes start to significantly outperform BPC, reaching an improvement of up to 17% and 18% in PPCP-GD, and PPCP-PSO, respectively, whereas the gap between BPC and the optimal solution VOPC reaches up to 19%. Note that PPCP-PSO improves the cache hit ratio by up to 10% compared to PPCP-GD. This is due to the higher prediction accuracy rendered by LSTM-PSO compared to LSTM-GD, which reduces the risk of requesters passing by their corresponding caching road segments when the data is not there, thus reducing the risk of having to resort to the far-away data center. PPCP-PSO and PPCP-GD tend to approach VOPC as κ increases, while this gap increases as κ decreases, reaching up to 19% between PPCP-GD and VOPC, and 11% between PPCP-PSO and VOPC.

In Fig. 5(d), the effect of varying κ on the satisfaction ratio is demonstrated. As κ increases, the satisfaction ratio increases in the predictive approach. This can be attributed to two reasons. The first reason is the fact that as κ decreases, the number of available road segments in the cache selection process decreases, which forces the data center to assign replicas to road segments with which users have high accumulative prediction error. This means that more requests are assigned to road segments with which the corresponding users have inaccurate estimation of their period of encounter. This increases the risk of data acquisition after the specified deadline, which reduces the satisfaction ratio. This risk manifests in PPCP-GD and PPCP-PSO but not in VOPC, due to the perfect knowledge assumption in the latter. The second reason is the fact that lower values of κ increases the risk of the data center not being able to find caching slots for more users, since they do not pass by any of the road segments that have parked vehicles. This forces data acquisition to occur from the data center or a caching node encountered along the request forwarding path. The prolonged delay from the far-away data center further decreases the satisfaction ratio. Note that as κ increases, data availability increases, which means that even if the aforementioned risk occurs, the possibility of encountering a nearby caching node increases, which increases the satisfaction ratio. The predictive approach significantly outperforms BPC, where the satisfaction ratio increases by up to 60% and 63%, in PPCP-GD and PPCP-PSO, respectively, and the gap between BPC and VOPC reaches up to 67%. This is since, as opposed to BPC, the predictive approach enables data acquisition to occur in a deterministic way while abiding to a specific QoS. PPCP-PSO improves the satisfaction ratio by up to 9% compared to PPCP-GD. This can be attributed to the higher prediction accuracy rendered by PSO-LSTM compared to GD-LSTM, which reduces the aforementioned risk. In PPCP-GD and PPCP-PSO, the gap between PPCP and VOPC is reduced as κ increases, thus enabling it to get significantly closer to the optimal solution. However, as κ decreases, this gap increases, reaching up to 38% between PPCP-GD and VOPC, and 20% between PPCP-PSO and VOPC.

V. Conclusion

In this paper, we proposed a predictive proactive caching framework at parked vehicles, which includes a prediction module and a cache placement module. In the former, the travel time of users at each road segment along their trajectory is estimated in order to be fed into the latter. In the prediction module, we proposed a PSO-based LSTM model that takes the effect of weather conditions into consideration. The prediction module also enables the estimation of a personalized travel time for each user. In the cache placement module, we have introduced VOPC as a benchmark that allows researchers to quantify the potential gains of predictive proactive caching schemes and certify if there is a possibility for improvement. We also proposed a greedy heuristic approach that takes into consideration all the time-constraint issues required to select the most suitable road segments for caching. We evaluated the performance of the proposed prediction model in terms of prediction accuracy, and we compared it to two gradient descent-based LSTM models that are implemented with and without considering weather conditions. Simulation results have shown that PSO-LSTM outperforms both models in terms of $RMSE$ and R^2. We also implemented PPCP using both PSO-LSTM and GD-LSTM to evaluate the effect of the level of prediction accuracy on the cache placement process. We implemented the optimal solution VOPC assuming perfect knowledge and used it to quantify the potential gains of PPCP. In addition, we compared PPCP to the baseline broadcast-based approach. Simulations have shown that the former significantly outperforms the latter in terms of delay, packet delivery ratio, cache hit ratio, and satisfaction ratio. Also, PPCP approaches VOPC in various scenarios, whereas a room for improvement has been substantiated in other scenarios. In the future, we plan on reducing the gap to the optimal solution.

References

[1] Digital 2019: Global Digital Overview. Accessed: Feb. 12, 2022. [Online]. Available: https://datareportal.com/reports/digital-2019-global-digital-overview

[2] Number of social network users worldwide from 2017 to 2025 (*in billions*). Accessed: Feb. 12, 2022. [Online]. Available: https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/

[3] Mobile's Hierarchy of Needs Revealing Key Insights into Global Mobile Trends. Accessed: Feb. 12, 2022. [Online]. Available: https://www.comscore.com/Insights/Press-Releases/2017/3/comScore-Releases-New-Report-Mobiles-Hierarchy-of-Needs

[4] J. Nakazato, M. Nakamura, T. Yu, Z. Li, G. K. Tran, and K. Sakaguchi, "Design of MEC 5G cellular networks: Viewpoints from telecom operators and backhaul owners," in *Proc. IEEE Int. Conf. Commun. Workshops*, Dublin, Ireland, 2020, pp. 1–6.

[5] S. Safavat, N. N. Sapavath, and D. B. Rawat, "Recent advances in mobile edge computing and content caching," *Digit. Commun. Netw.*, vol. 6, no. 2, pp. 189–194, 2020.

[6] G. S. Paschos, G. Iosifidis, M. Tao, D. Towsley, and G. Caire, "The role of caching in future communication systems and networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1111–1125, Jun. 2018.

[7] W. Li, S. M. A. Oteafy, M. Fayed, and H. S. Hassanein, "Quality of experience in ICN: Keep your low-bitrate close and high-bitrate closer," *IEEE/ACM Trans. Netw.*, vol. 29, no. 2, pp. 557–570, Apr. 2021.

[8] S. Zhang, W. Sun, J. Liu, and N. Kato, "Physical layer security in large scale probabilistic caching: Analysis and optimization," *IEEE Commun. Lett.*, vol. 23, no. 9, pp. 1484–1487, Sep. 2019, doi: 10.1109/LCOMM.2019.2926967.

[9] S. Ilarri, T. Delot, and R. Trillo, "A data management perspective on vehicular networks," *IEEE Commun. Surv. Tut.*, vol. 17, no. 4, pp. 2420–1760, Oct./Dec. 2015.

[10] J. Balen, G. Martinovic, K. Paridel, and Y. Berbers, "PVCM: Assisting multi-hop communication in vehicular networks using parked vehicles," *Int. Congr. Ultra Modern Telecommun. Control Syst. Workshops*, 2012, pp. 119–122.

[11] S. Abdelhamid, H. S. Hassanein, and G. Takahara, "Vehicle as a resource (VaaR)," *IEEE Netw.*, vol. 29, no.1, pp. 12–17, Jan./Feb. 2015.

[12] The Center for Automotive Research, "Connected vehicle technology industry delphi study," *Center Automotive Res.*, Ann Arbor, USA, Tech. Rep., Sep. 2012. [Online]. Available: https://www.michigan.gov/documents/mdot/09-27-2012_Connected_Vehicle_Technology_-_Industry_Delphi_Study_401329_7.pdf

[13] A. Reis, S. Sargento, and O. K. Tonguz, "Parked cars are excellent roadside units," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 9, pp. 2490–2502, Sep. 2017.

[14] American Association of State Highway and Transportation Officials (AASHTO), "National connected vehicle field infrastructure footprint analysis," Tech. Rep., No FHWA-JPO-14-125, Jun. 2014.

[15] S. Abdelhamid, H. S. Hassanein, G. Takahara, and H. Farahat, "On-road caching assistance for ubiquitous vehicle-based information services," *IEEE Trans. Veh. Technol.*, vol. 64, no. 12, pp. 5477–5492, Dec. 2015

[16] J. Zhao, Y. Zhang, and G. Cao, "Data pouring and buffering on the road: A new data dissemination paradigm for vehicular ad hoc networks," *IEEE Trans. Veh. Technol.*, vol. 56, no. 6, pp. 3266–3277, Nov. 2007.

[17] H. Farahat and H. Hassanein, "Optimal caching for producer mobility support in named data networks," in *Proc. IEEE Int. Conf. Commun.*, 2016, pp. 1–6.

[18] A. Sadeghi, F. Sheikholeslami, and G. B. Giannakis, "Optimal and scalable caching for 5G using reinforcement learning of space-time popularities," *IEEE J. Sel. Top. Signal Process.*, vol. 12, no. 1, pp. 180–190, Feb. 2018.

[19] J. Gao, S. Zhang, L. Zhao, and X. S. Shen, "The design of dynamic probabilistic caching with time-varying content popularity," *IEEE Trans. Mobile Comput.*, vol. 20, no. 4, pp. 1672–1684, Apr. 2020.

[20] J. Famaey, F. Iterbeke, T. Wauters, and F. De Turck, "Towards a predictive cache replacement strategy for multimedia content," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 219–227, 2013.

[21] A. Krishna, R. Burra, and C. Singh, "Caching dynamic contents with varying popularity," in *Proc. 19th Int. Symp. Model. Optim. Mobile Ad Hoc Wireless Netw.*, 2021, pp. 1–8.

[22] J. Gao, L. Zhao, and X. Shen, "The study of dynamic caching via state transition fieldthe case of time-invariant popularity," *IEEE Trans. Wireless Commun.*, vol. 18, no. 12, pp. 5924–5937, Dec. 2019.

[23] Z. Yu, J. Hu, G. Min, H. Xu, and J. Mills, "Proactive content caching for internet-of-vehicles based on peer-to-peer federated learning," in *Proc. IEEE 26th Int. Conf. Parallel Distrib. Syst.*, 2020, pp. 601–608.

[24] H. Gong and L. Yu, "Content downloading with the assistance of roadside cars for vehicular ad hoc networks," *Mobile Inf. Syst.*, vol. 2017, 2017, Art. no. 4863167.

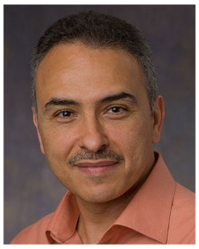
[25] D. Grewe, M. Wagner, and H. Frey, "PeRCeIVE: Proactive caching in ICN-based VANETs," in *Proc. IEEE Veh. Netw. Conf.*, 2016, pp. 1–8.

[26] Y. AlNagar, S. Hosny, and A. A. El-Sherif, "Towards mobility-aware proactive caching for vehicular ad hoc networks," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshop*, Marrakech, Morocco, 2019, pp. 1–6.

[27] Y. AlNagar, S. Hosny, and A. A. El-Sherif, "Proactive caching for vehicular ad hoc networks using the city model," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2019, pp. 1–7.
[28] "NS-3 network simulator." Accessed: Feb. 12, 2022. [Online]. Available: https://www.nsnam.org/
[29] Gurobi, "Gurobi optimizer reference manual." Accessed: Feb. 12, 2022. [Online]. Available: http://www.gurobi.com
[30] N. Abani, T. Braun, and M. Gerla, "Proactive caching with mobility prediction under uncertainty in information-centric networks," in *Proc. 4th ACM Conf. Inf.-Centric Netw.*, 2017, pp. 88–97.
[31] B. Xu, A. Ouksel, and O. Wolfson, "Opportunistic resource exchange in inter-vehicle ad hoc networks," in *Proc. IEEE Int. Conf. Mobile Data Manage.*, 2004, pp. 4–12.
[32] K. Liu *et al.*, "Network-coding-Assisted data dissemination via cooperative vehicle-to-vehicle-infrastructure communications," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 6, pp. 1–12, Jun. 2016.
[33] Z. Su, Q. Xu, Y. Hui, M. Wen, and S. Guo, "A game theoretic approach to parked vehicle assisted content delivery in vehicular ad hoc networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 7, pp. 6461–6474, Jul. 2017.
[34] S. A. Elsayed, S. Abdelhamid, and H. S. Hassanein, "Proactive caching at parked vehicles for social networking," in *Proc. IEEE Int. Conf. Commun.*, Kansas, MO, USA, 2018, pp. 1–6.
[35] W. Fan and Z. Gurmu, "Dynamic travel time prediction models for buses using only GPS data," *Int. J. Transp. Sci. Technol.*, vol. 4, no. 4, pp. 353–366, 2015.
[36] H. Xu and J. Ying, "Bus arrival time prediction with real-time and historic data," *Cluster Comput.*, vol. 20, no. 3, pp. 3099–3106, 2017, doi: 10.1007/s10586-017-1006-1.
[37] J. Myung, D.-K. Kim, S.-Y. Kho, and C.-H. Park, "Travel time prediction using K nearest neighbor method with combined data from vehicle detector system and automatic toll collection system," *Transp. Res. Record: J. Transp. Res. Board*, vol. 2256, pp. 51–59, 2011.
[38] D. Billings and J.-S. Yang, "Application of the ARIMA models to urban roadway travel time prediction: A case study," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, 2006, pp. 2529–2534.
[39] S. I. J. Chien and C. M. Kuchipudi, "Dynamic travel time prediction with real-time and historic data," *J. Transp. Eng.-Asce*, vol. 129, no. 6, pp. 608–616, 2003.
[40] W. D. Fan, "Artificial neural network travel time prediction model for buses using only global positioning system data," *J. Public Transp.*, vol. 17, no. 2, pp. 45–65, 2014.
[41] M. As and T. Mine, "Dynamic bus travel time prediction using an ANN-based model," in *Proc. 12th Int. Conf. Ubiquitous Inf. Manage. Commun.*, New York, USA, 2018, pp. 1–8.
[42] Y. Duan, Y. Lv, and F. Y. Wang, "Travel time prediction with LSTM neural network," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst.*, 2016, pp. 1053–1058.
[43] A. Abdi and C. Amrit, "A review of travel and arrival-time prediction methods on road networks: Classification, challenges and opportunities," *Peer J. Comput. Sci.*, vol. 7, pp. 689–736, 2021, doi: 10.7717/peerj-cs.689.
[44] W. Treethidtaphat, W. Pattara-Atikom, and S. Khaimook, "Bus arrival timevprediction at any distance of bus route using deep neural network model," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst.*, 2017, pp. 988–992.
[45] P. -Y. Ting *et al.*, "Freeway travel time prediction using deep hybrid model taking sun yat sen freeway as an example," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 8257–8266, Aug. 2020.
[46] X. Zeng and Y. Zhang, "Development of recurrent neural network considering temporal-spatial input dynamics for freeway travel time modeling," *Comput.-Aided Civil Infrastructure Eng.*, vol. 28, no. 5, pp. 359–371, 2013, doi 10.1111/mice.12000.
[47] Y. Yuan *et al.*, "Bus dynamic travel time prediction: Using a deep feature extraction framework based on RNN and DNN," *Electronics*, vol. 9, no. 11, 2020, Art. no. 1876, doi 10.3390/electronics9111876.
[48] I. Islek and S. G. Oguducu, "Use of LSTM for short-term and long-term travel time prediction," in *Proc. Conf. Informat. Knowl. Manage. Workshops*, 2018, pp. 2482–2485.
[49] R. Atawia, H. Abou-Zeid, H. Hassanein, and A. Noureldin, "Joint chance constrained predictive resource allocation for energy-efficient video streaming," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1389–1404, May 2016.
[50] "SUMO (Simulation of urban mobility)." Accessed: Sep. 12, 2022. [Online]. Available: http://sumo-sim.org/
[51] G. Potari, Z. Vincellera, and Z. Acs, "A method for real-time adaptation of weather conditions within a traffic simulation," in *Proc. Int. Conf. Appl. Informat. Eger Hung.*, 2014, pp. 41–47, doi: 10.14794/ICAI.9.2014.2.41.
[52] S. Elsworth and S. Guttel, "Time series forecasting using LSTM networks: A symbolic approach," 2020, *arXiv:2003.05672*.
[53] A. Singh, S. Dixit, and L. Srivastava, "Particle swarm optimization-artificial neural network for power system load flow," *Int. J. Power Syst. Oper. Energy Manage.*, vol. 1, no. 2, pp. 73–82, 2011.
[54] A. M. Ibrahim and N. H. El-Amary, "Particle swarm optimization trained recurrent neural network for voltage instability prediction," *J. Elect. Syst. Inf. Technol.*, vol. 4, no. 1, pp. 216–228, 2017, doi: 10.1016/j.jesit.2017.05.001.
[55] J. Kennedy and RC. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, 1995, pp. 1942–1948, doi: 10.1109/ICNN.1995.488968.
[56] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," 2014, *arXiv:1402.1128*.

Sara A. Elsayed (Member, IEEE) received the Ph.D. degree from the School of Computing, Queen's University, Kingston, ON, Canada, in 2020. She is currently a Postdoctoral Fellow and an adjunct Assistant Professor with the School of Computing, Queen's University. She coordinates an extensive research project for democratizing edge computing, involving an industrial collaboration with Queen's University. Prior to joining Queen's University, she was an Assistant Lecturer with the Faculty of Computers and Information, Cairo University, Giza, Egypt. She was the Co-founder and a Program Manager of the Teaching Assistant Qualifying Program, and an Assistant Focal Point with the Support Office for Research Cooperation & Mobility, Cairo University. She is a certified Cisco Networking Academy Instructor. She has several publications in top venues. Her research interests include edge computing, vehicular networks, caching in VANETs, information-centric networks, and Internet of Things.

Sherin Abdelhamid (Member, IEEE) received the Ph.D. degree from Queen's University, Kingston, ON, Canada, in 2014. She is currently a Senior Strategy Manager with TELUS, where she drives the strategy development & implementation of connected and autonomous vehicle services. She also manages partner relationships and large-scale projects that span across major TELUS frontline and internal facing business units. Prior to joining TELUS, she was a Technical Advisor and Manager with the Ontario Vehicle Innovation Network, where she owned and managed delivery of strategic projects in partnership with industry, government, and other ecosystem partners. She was also a Research Associate and an adjunct Assistant Professor with the School of Computing - Queen's University. She is a subject-matter expert in connected, autonomous, and electric vehicle technologies and mobility innovation, and has been named by Automotive News Canada as one of the 2020 Canadians to Watch in the auto industry. She is also experienced in data analytics, smart city innovations, and the Internet of Things. She is an award-winning Researcher with more than 40 publications in many top journals, magazines, and conferences.



Hossam S. Hassanein (Fellow, IEEE) is currently a leading authority in the areas of broadband, wireless and mobile networks architecture, protocols, control, and performance evaluation. He has authored or coauthored more than 600 publications in journals, conferences, and book chapters, in addition to numerous keynotes and plenary talks in flagship venues. He is the Founder and Director of the Telecommunications Research Lab, School of Computing, Queen's University, Kingston, ON, Canada, with extensive international academic and industrial collaborations. Dr. Hassanein was the recipient of several recognition and best paper awards at top international conferences, such as the 2016 IEEE Communications Society Communications Software Technical Achievement Award for outstanding contributions to routing and deployment planning algorithms in wireless sensor networks, and 2020 IEEE IoT, Ad Hoc and Sensor Networks Technical Achievement and Recognition Award for significant contributions to technological advancement of the Internet of Things, ad hoc networks, and sensing systems. He is a former Chair of the IEEE Communication Society Technical Committee on Ad hoc and Sensor Networks (TC AHSN). He is an IEEE Communications Society Distinguished Speaker and was a Distinguished Lecturer during 2008–2010.