SAKARYA UNIVERSITY JOURNAL OF COMPUTER AND INFORMATION SCIENCES
http://saucis.sakarya.edu.tr/

e-ISSN: 2636-8129
Publisher: Sakarya University

Vol. 9, No.3, 846-867, 2026
DOI:10.35377/saucis...1778576

SAKARYA UNIVERSITY

RESEARCH ARTICLE

A Transformer-Assisted Cooperative Caching Framework with Q-Learning Optimization for Internet of Vehicles

Abdelkrim Lamraoui [1], **Lyamine Guezouli** [2*], **Kamel Barka** [3], **Mohamed Zohir Mabane** [2], **Sohaib Chine** [2]

[1]LAMIE Laboratory, University of Batna 2, Batna, Algeria
[2]LEREESI Laboratory, HNS-RE2SD, Batna, Algeria
[3]Lastic Laboratory, University of Batna 2, Batna, Algeria

*Corresponding author:
lyamine.guezouli@hns-re2sd.dz

Article History:
Received: 07 September 2025
Revised: 11 April 2026
Accepted: 13 April 2026
Published Online: 23 June 2026

Abstract
The Internet of Vehicles (IoV) is witnessing escalating data traffic that demands sophisticated caching to minimize retrieval latency and sustain Quality of Service. Recent cooperative approaches, notably the Cooperative Caching Strategy with Content Request Prediction (CCCRP) framework grounded in LSTM-based request prediction, struggle to capture the complex, long-range temporal dependencies inherent to dynamic vehicular environments. This paper proposes Transformer-based CCCRP (T-CCCRP), which replaces the LSTM predictor with a Transformer model leveraging self-attention to model long-range dependencies and enable efficient parallelism. The prediction module is integrated with a reinforcement learning controller to optimize cache placement jointly across vehicles and roadside units, thereby aligning predicted popularity with resource constraints. To ensure a realistic and comprehensive evaluation, the proposed framework is assessed under both small-scale and newly introduced large-scale IoV scenarios, involving up to 200 vehicles, multiple caching nodes, and expanded content libraries. The experimental setup further adapts the input sequence length of the prediction model to reflect increased temporal complexity in large-scale environments. Simulation results demonstrate that T-CCCRP consistently outperforms the original CCCRP and conventional caching strategies (LFU and LRU) across all evaluated scales.

Keywords: Edge caching, Cooperative caching, CCCRP, Transformer model, Reinforcement learning, Temporal dependency

1. Introduction

With the continuous growth of intelligent transportation systems (ITS), the Internet of Vehicles (IoV) has become increasingly essential for enabling advanced vehicular communication, efficient resource sharing, and improved user experiences. IoV networks, by nature, are highly dynamic and prone to intermittent connectivity, which presents significant challenges in ensuring timely and reliable content delivery [1]. As real-time demands for multimedia, navigation, and safety-critical data intensify, traditional centralized communication frameworks increasingly struggle with network congestion and fail to meet Quality of Service (QoS) expectations [2]. To address these limitations, edge caching has emerged as a promising solution. By storing frequently accessed content locally—either in vehicles or at roadside units (RSUs)—edge caching substantially reduces latency and eases the burden on core network infrastructure [3].

The paper reviews various prior caching strategies, including classical heuristics such as Least Recently Used (LRU) and Least Frequently Used (LFU), as well as statistical and machine learning-based approaches. While these methods have shown some success, particularly with Long Short-Term Memory (LSTM) models that capture temporal correlations in content demand [4, 5], they remain constrained in their ability to effectively model long-term dependencies and rapidly evolving request patterns [6]. In Internet of Vehicles (IoV) environments, content requests

Cite as: A. Lamraoui et al., "A transformer-assisted cooperative caching framework with q-learning optimization for internet of vehicles" , *Sakarya University Journal of Computer and Information Sciences* 9 (3) 2026, 846-867. doi: 10.35377/saucis...1778576

are influenced not only by recent interactions but also by historical behaviors spanning extended time periods. For instance, vehicles following recurring routes or operating under similar traffic conditions often exhibit repetitive demand patterns, where earlier requests provide important contextual signals for future content access. Such characteristics introduce long-range dependencies within request sequences, which are difficult to fully capture using conventional sequential models. This limitation underscores the need for more advanced prediction frameworks capable of learning complex temporal relationships and adapting to the highly dynamic nature of IoV systems.

Transformer models have recently demonstrated state-of-the-art performance in sequential prediction tasks across domains such as natural language processing, computer vision, and time-series forecasting [7, 8]. Leveraging self-attention mechanisms, they can model dependencies across all elements in a sequence regardless of temporal distance, thereby overcoming the inherent limitations of recurrent architectures. In the context of IoV, this capability enables the identification of latent correlations between temporally distant request events, such as periodic traffic information demands or recurring multimedia access patterns. Compared to LSTM, Transformers provide improved scalability and a more comprehensive representation of temporal dynamics, making them particularly suitable for capturing the complex and long-range dependencies present in vehicular content request data. Despite these advantages, their application in cooperative caching for IoV environments remains largely unexplored.

To address this gap, the study introduces a Transformer-based Cooperative Caching strategy with Content Request Prediction (T-CCCRP), an enhanced version of the Cooperative Caching strategy with Content Request Prediction (CCCRP) proposed by Wang et al., which replaces the LSTM predictor with a Transformer-based model. It explores whether this substitution, combined with reinforcement learning, improves content popularity prediction and caching decisions.

The study contributes to IoV research by integrating Transformer-based analytics into caching systems, optimizing decisions through reinforcement learning, and demonstrating superior performance over LSTM-based CCCRP via comparative analysis. The methodology involves vehicle clustering using K-means [9], prediction of future content requests via a Transformer model, and reinforcement learning-driven caching to maximize hit ratios and reduce latency.

The remainder of the paper is structured as follows: Section 2 provides a comprehensive literature review of relevant caching strategies and predictive methodologies. Section 3 describes the proposed T-CCCRP and its reinforcement learning integration in detail. In Section 4, the simulation setup and performance evaluation metrics are defined, followed by Section 5, which presents and discusses the experimental results. Finally, Section 6 concludes the paper by summarizing the main findings, implications for future research, and practical recommendations for implementing the proposed solution in real-world IoV systems.

2. Related Works

In this section, we provide a comprehensive analysis of existing literature pertinent to caching strategies within the context of the IoV, systematically tracing the evolution from basic caching approaches to advanced machine learning-driven methodologies, leading logically towards the transformer-based content request prediction approach proposed in this paper.

Early caching strategies primarily included simple heuristic techniques such as Least Recently Used (LRU), Least Frequently Used (LFU), and First-In-First-Out (FIFO). These conventional methods primarily relied on static measures such as content access frequency or recency to manage caching decisions [10]. Despite their simplicity and ease of implementation, these basic caching policies suffered significant limitations when applied to IoV scenarios. The highly dynamic network topology, mobility of vehicles, rapid fluctuation of user interests, and stringent latency requirements made traditional static caching insufficient for meeting the complex demands of vehicular communications [2].

To address the limitations of traditional caching, researchers developed intermediate cooperative caching strategies that involved collaboration among nodes to enhance content delivery efficiency and reduce delays. Cooperative caching approaches introduced clustering and hierarchical architectures in which vehicles and RSUs worked together to leverage collective caching resources and minimize redundant data storage, thereby significantly improving caching performance in dynamic scenarios [11–13]. Building upon the foundation of basic cooperative caching schemes, recent studies have introduced strategies that explicitly account for vehicle mobility and topology changes, demonstrating improved performance in highly dynamic vehicular scenarios. Fang and Fan (2017) developed a cooperative caching method tailored to vehicular clusters, optimizing decisions based on global caching states and storage constraints, thereby improving caching efficiency over isolated strategies [14]. To overcome challenges associated with distributed approaches, Jin et al. (2024) introduced a centralized edge cooperative caching (CECC) scheme, formulated as a multiple-choice knapsack problem. By leveraging global network insights, their method notably enhanced cache hit rates and reduced transmission latency compared to decentralized alternatives [15].

While these strategies improved cache hit rates and reduced latency significantly compared to isolated caching, they remained constrained by static popularity estimation methods. They did not effectively capture dynamic content access patterns. To address this, statistical models such as the Zipf distribution have been integrated into caching strategies [14]. Breslau et al. (1999) demonstrated Zipf-like patterns in web content accesses, inspiring subsequent caching methodologies [16]. Building upon this idea, Ling et al. (2024) predicted service popularity by analyzing vehicle trajectories and historical requests, achieving higher cache hits and lower latency [17]. Similarly, Lin et al. (2023) introduced a collaborative edge caching model using coalition game theory to predict popularity and cooperatively cache content, significantly enhancing caching performance relative to independent approaches [18].

To enhance caching efficiency further, researchers have developed strategies combining popularity estimation with other caching optimization techniques. Liang et al. (2023) proposed a multilayer coded caching strategy for lightweight Certificate Revocation List distribution in VANETs. By statistically modeling certificate request popularity across network layers, their method substantially decreased communication overhead and storage redundancy [19].

Despite considerable advancements in statistical models, further exploration remains necessary to refine prediction models, particularly in highly dynamic vehicular environments where traditional statistical models may struggle with rapid changes in content popularity and vehicular mobility.

This led to the adoption of machine learning approaches that adapt to real-time user behavior and better predict content popularity and vehicle mobility, enabling optimized caching strategies that significantly improve cache hit rates and reduce content delivery delays. Recent studies have integrated machine learning techniques, notably reinforcement learning, to adapt caching strategies dynamically. Zaman et al. (2023) employed Cuckoo Search optimization to enhance caching efficiency, demonstrating significant latency reductions compared to heuristic methods [20]. Meanwhile, Zheng Hui Ernest and Madhukumar (2023) utilized ensemble learning and meta-learning techniques to refine content popularity prediction, significantly improving caching performance under uncertain vehicular network conditions [21]. Expanding on contextual factors, Yao et al. (2021) combined social attributes with mobility prediction through Hidden Markov Models, maximizing cache hits by identifying socially similar vehicles likely to request similar content [22]. Likewise, Su et al. (2018) proposed a cross-entropy-based dynamic caching approach accounting for vehicle velocity and traffic density, significantly lowering content retrieval delays in highly dynamic scenarios [3].

While early machine learning methods enhanced caching decisions by using contextual features, they were limited in capturing complex temporal patterns. As vehicular data became more sequential and dynamic, researchers turned to deep learning models like Long Short-Term Memory (LSTM) and Echo State Networks (ESN), which better handled time-series dependencies and significantly improved predictive caching accuracy. Deep learning methods, particularly LSTM, have gained prominence in intelligent caching strategies for vehicular networks, mainly due to their ability to handle sequential data and capture long-term dependencies effectively. Kan et al. (2023) integrated LSTM for trajectory forecasting with deep reinforcement learning (DRL) for cache management, considering vehicle mobility and social interactions to achieve superior caching performance [23]. Similarly, Tian et al. (2023) coupled LSTM predictions with Double Deep Q-Networks, dynamically updating caches to markedly improve hit rates and content delivery times [24]. S. A. Elsayed et al. (2022) combined LSTM-based mobility prediction with Deep reinforcement learning-driven cache placement, effectively adapting to rapid vehicle movements, resulting in enhanced throughput and lower delays [25]. Hou et al. (2019) further exemplified LSTM's utility by incorporating Q-learning for proactive caching in infotainment scenarios, accurately predicting mobility patterns and optimizing RSU cache placements to significantly reduce latency compared to traditional approaches [26].

Notably, Wang et al. proposed an LSTM-based cooperative caching strategy (CCCRP), which combines K-means clustering and LSTM networks to proactively predict future content popularity. Their strategy employs reinforcement learning to solve optimization problems related to cache placement, effectively enhancing cache hit ratios and reducing content access delays by up to 7% compared to traditional LFU and LRU strategies. Nevertheless, critical limitations in their approach include suboptimal handling of dynamic network conditions, insufficient adaptation to abrupt changes in content popularity, and less efficient resource allocation under rapidly changing vehicular environments [27]. Specifically, the LSTM method, although effective for sequential prediction, encounters significant challenges, including difficulties in capturing long-range dependencies due to vanishing and exploding gradient problems, slower training times, and limited parallel processing capability [7, 28, 29]. Furthermore, LSTM architectures are predominantly reliant on recurrent structures, which diminishes their efficiency in scaling and in capturing complex temporal interrelations within large datasets.

To address these limitations, the Transformer model presents itself as a viable alternative, thanks to its ability to handle long-range dependencies more effectively via self-attention mechanisms, while avoiding the gradient issues inherent to LSTM. The implementation of Transformers allows for parallel processing, significantly diminishing both training and inference times, which in turn enhances scalability and efficiency. Their inherent design enables better adaptation to rapid changes in content popularity and dynamic vehicular network conditions, leading to more accurate predictions and improved caching decisions.

This work proposes a novel Transformer-based cooperative caching strategy aimed at improving cache hit ratios and reducing content acquisition delays in IoV environments. By replacing the LSTM model in Wang et al.'s approach with the Transformer model, the proposed method seeks to overcome the identified limitations. The Transformer model's parallel processing capabilities, self-attention mechanism, and ability to capture long-range dependencies are expected to provide a more efficient and accurate prediction mechanism, ultimately enhancing overall caching performance [7, 28, 29].

Table 1. Comparative Overview of Caching Strategies for IoV

Ref	Objective	Technical Approach	Input Data	Cons
[11]	• Reduces cache redundancy and latency	• Mobility-based clustering • Consistent hashing for content placement	• Vehicle mobility • Direction • Velocity	• Hashing imbalance risk • Relies on mobility prediction accuracy • Scales poorly in sparse networks
[12]	• Enhance cooperative caching in a hierarchical V2X network • Improves scalability and coordination	• Cluster-based cooperative caching using metadata-managed overlay	• Cache metadata • Network topology	• Metadata maintenance cost • Hierarchical rigidity • Cluster management overhead and dependency on stable hierarchy
[13]	• Maximize both individual and total profit	• Dynamic coalition formation based on profit optimization and strategic interaction • Profit-aware Hedonic Game Theory	• Cache capacity • Service demand • Utility values	• Requires truthful information • Risk of coalition instability • Profit-based model may not optimize for latency or QoS directly
[14]	• Reduce backhaul cost • Improve QoE	• Cluster-wide caching strategy with weighted cost evaluation	• Content requests • Cluster vehicle states	• Requires cluster status monitoring • Assumes uniform vehicle cache constraints • Lacks adaptivity
[15]	• Minimize latency	• Centralized optimization modeled as a multiple-choice knapsack problem • Greedy algorithm solving MCK for cache optimization	• Global network knowledge • Content popularity • RSU storage	• Centralized approach may introduce overhead and reduce scalability
[17]	• Optimize caching	• Gibbs sampling for cache placement • Vehicle trajectory analysis	• Vehicle trajectory • Service request history	• Computational complexity • High dependency on trajectory data accuracy
[18]	• Optimizing cache placement	• Coalition games to decide optimal cooperative caching	• Service provider resource • User demand	• Complex coalition formation process
[19]	• Efficiently distribute CRL data with minimal overhead	• Multilayer cache architecture for efficient CRL distribution	• CRL data • Vehicle request patterns	• Complex multilayer management
[20]	• Minimize content delivery delay	• Cooperative caching • Clustering-based vehicle prediction	• Vehicle mobility • Content popularity	• Computational complexity in large scenarios
[21]	• Enhance content popularity prediction for edge caching reliability	• Meta and Ensemble Learning • Outage probability analysis	• Historical content request	• Computational complexity of training multiple models
[22]	• Enhance cache performance	• Social attributes integration • Mobility pattern analysis	• Social interaction • Vehicle trajectories	• Dependency on accurate social and mobility data
[3]	• Optimize content distribution to improve retrieval times	• Edge-centric caching • Strategic content placement	• Content popularity • Vehicle positions	• Limited by accurate predictions

Table 1. Comparative Overview of Caching Strategies for IoV (continued)

Ref	Objective	Technical Approach	Input Data	Cons
[23]	• Cooperative caching by vehicle trajectory and social relationship prediction	• LSTM predicts trajectory • Social-aware DRL makes caching decisions	• Vehicle trajectory • Social interactions • Historical requests	• Complexity due to combined social and mobility models
[24]	• Dynamic caching	• LSTM predicts requests • DDQN optimizes caching	• Historical request • User preferences	• Complexity of integrating multiple deep learning models
[25]	• Mobility-aware caching strategy to optimize caching node throughput	• LSTM predicts vehicle positions • DDPG manages cache dynamically	• Historical trajectory • Content request	• Requires real-time prediction accuracy • Computational complexity
[26]	• Optimize caching policy to reduce latency	• RL with mobility prediction	• Vehicle trajectory • Caching demands	• Prediction accuracy dependence • Training complexity
[27]	• Increase cache hit ratio • Reduces average content acquisition delay • Provide more efficient and accurate predictions	• Cooperative RSU-vehicle caching • Clustering-based prediction • LSTM predicts requests	• Historical requests • Vehicle clustering	• Difficulties in capturing long-range dependencies • Slower training times • Limited parallel processing capability

3. Proposed Transformer-based CCCRP (T-CCCRP)

The proposed T-CCCRP framework constitutes a direct enhancement over the original CCCRP strategy by refining its predictive component. Specifically, it preserves the same systematic sequence of operational steps as the original framework, encompassing parameters initialization, vehicle clustering, and the reinforcement learning-driven caching decision process (Figure 1).

The principal modification introduced in this study involves replacing the original prediction mechanism, based on an LSTM neural network, with a Transformer-based prediction model. This substitution leverages the Transformer's advanced capacity for capturing long-range temporal dependencies and exploits its inherent parallel-processing capabilities, thereby addressing limitations associated with LSTM architectures, such as gradient vanishing and limited scalability. Consequently, the enhanced predictive accuracy achieved by the Transformer model is expected to significantly improve caching performance, particularly within dynamic and rapidly evolving IoV scenarios.

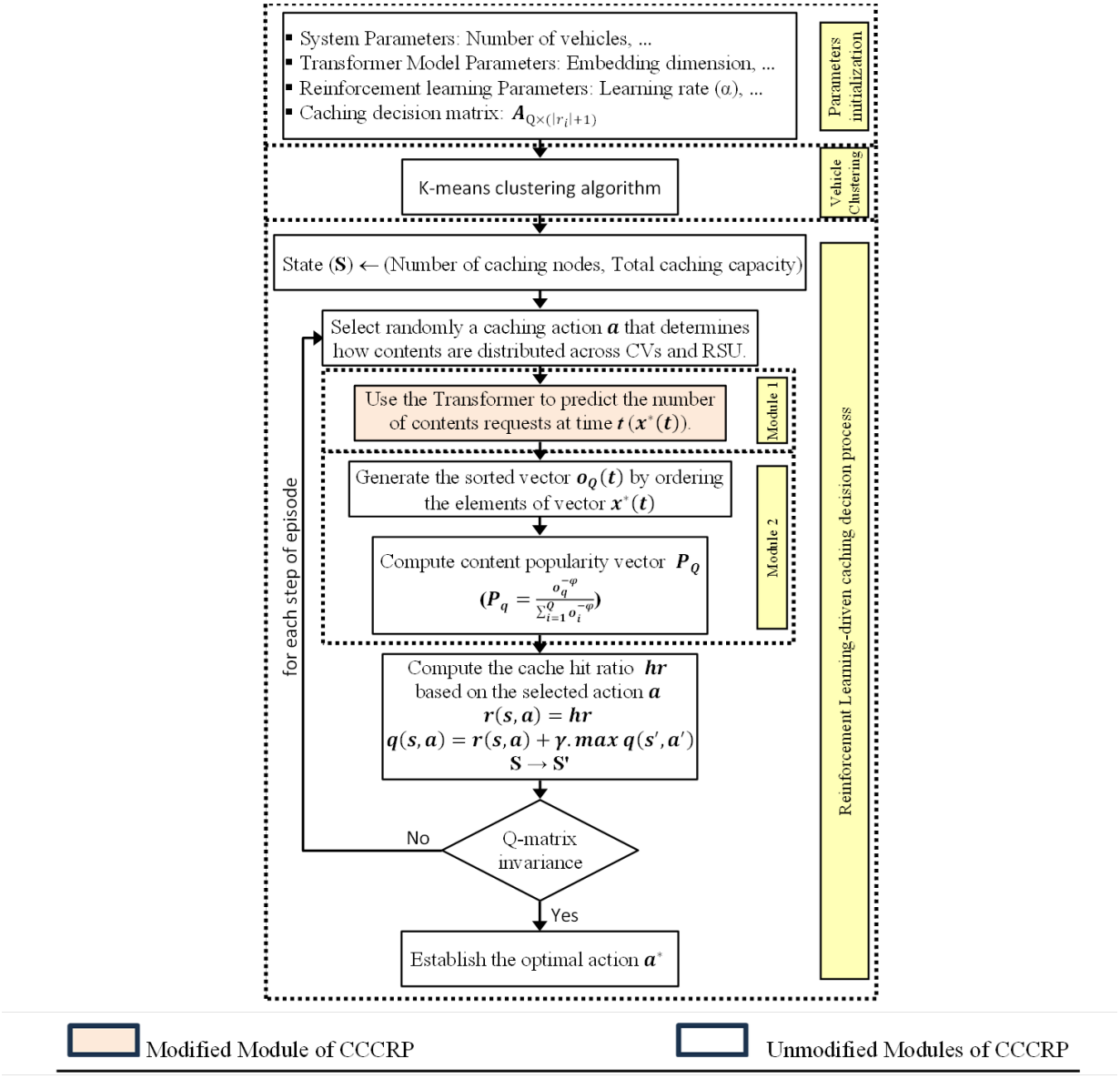

Figure 1. High-Level Architecture of the T-CCCRP Framework

3.1. Phase 1: Parameter initialization

The first phase establishes the operational environment by configuring essential system parameters, Transformer model settings, and reinforcement learning configurations. These include specifications for vehicular mobility patterns, the number and size of clusters, the available caching capacity at each node, and the setup of both the reinforcement learning algorithm and the Transformer model. This initialization process provides the structural and algorithmic foundation required for consistent execution across all subsequent phases. It ensures that vehicle mobility, communication capabilities, and learning mechanisms are coherently integrated into a unified framework.

3.2. Phase 2: Vehicle clustering

In the second phase, the T-CCCRP framework employs a K-means clustering algorithm to group vehicles based on their spatial and mobility characteristics, specifically their geographic positions and velocities. This clustering mechanism serves two principal objectives: to reduce the computational complexity associated with making individualized caching decisions and to localize content dissemination, thereby minimizing communication overhead and content acquisition delay. The clustering process begins with selecting initial cluster centers. Subsequently, each vehicle is assigned to the nearest cluster based on a predefined distance or similarity metric. After assignments are made, the algorithm recalculates the cluster centers to reflect the current composition of each group. This iterative process of reassignment and center recalculation continues until convergence is achieved—i.e., until the cluster centers stabilize and no longer change. The resulting clusters serve as the basic units for cooperative caching decisions in the following phase.

3.3. Phase 3: Reinforcement Learning-driven caching decision process

In the third phase of the proposed T-CCCRP framework, the system determines optimal caching decisions using a Q-learning reinforcement learning framework enhanced by Transformer-based content request prediction. This phase addresses the fundamental challenge of placing content within a limited caching infrastructure—comprised of caching vehicles (CVs) and RSUs—in such a way that maximizes the cache hit ratio, i.e., the proportion of content requests that are successfully served by local caches (CVs and RSUs) rather than via the remote content server. To formally represent this, a binary caching decision matrix $A \in \{0,1\}^{Q\times(|r_i|+1)}$ is defined, where Q is the total number of distinct content items and $|r_i|$ is the number of CVs under RSU r_i. The entry $a_{q,j} = 1$ indicates that content q is cached in node j, which can be the CV of the cluster j ($j = 1$ to $|r_i|$) or the RSU ($j = |r_i| + 1$).

The caching optimization problem is formulated as a binary integer program, aiming to maximize the overall expected cache hit ratio by determining an optimal caching matrix A:

$$\max_{\boldsymbol{A}_{Q\times(|r_i|+1)}} \left(hr_{RSU} + \sum_{j=1}^{|r_i|} hr_{CV} \right) \tag{1}$$

where:

- $hr_{CV} = P_Q \times a_{:,j}$ is the expected hit ratio at CV j.
- $hr_{RSU} = P_Q \times a_{:,|r_i|+1}$ is the expected hit ratio at the RSU r_i.
- $P_Q = [P_1, P_2, \ldots, P_Q]$ is the content popularity vector predicted using the Transformer model and Zipf distribution.

This optimization is subject to four constraints:

- **Binary constraint** – each cache decision must be binary:

$$a_{q,j},\ a_{q,|r_i|+1} \in \{0,1\} \tag{2}$$

- **Non-redundancy constraint** – each content item q may be stored in at most one node (CV or RSU) within the same RSU coverage zone:

$$a_{q,|r_i|+1} + \sum_{j=1}^{|r_i|} a_{q,j} \leq 1 \tag{3}$$

- **CV capacity constraint** – total size of cached content in any CV must not exceed its storage capacity C_{CV}:

$$S_Q \times a_{:,j} \leq C_{CV} \tag{4}$$

- **RSU capacity constraint** – similar constraint for the RSU:

$$S_Q \times a_{:,|r_i|+1} \leq C_{RSU} \tag{5}$$

Here, $S_Q = [S_1, S_2, \ldots, S_Q]$ denotes the size of each content item.

Given the combinatorial complexity of solving this non-convex, NP-hard problem directly, T-CCCRP employs Q-learning, a reinforcement learning approach that learns the best caching actions through interaction with the environment (Figure 2). Once content popularity has been forecasted, a reinforcement learning agent acts as a caching controller that observes the current network and caching state, selects a caching configuration (i.e., an action), and receives feedback in the form of the cache hit ratio—serving as the reward signal. Over time, through repeated interaction with the environment, the agent learns to select caching decisions that maximize cumulative rewards, thereby improving content availability and minimizing acquisition latency. For further clarification, the Q-learning implementation in T-CCCRP defines the caching decision process through three essential elements: state, action, and reward.

- **State (s):** The state space in T-CCCRP represents the total caching capacity and the number of cacheable nodes (RSUs and CVs) within a given region. After clustering vehicles, the reinforcement learning agent observes the available memory at each caching node and constructs the state $s = (|r_i| + 1, C_{total})$, where C_{total} is the cumulative available cache storage.

- **Action (a):** In the context of the T-CCCRP framework, accurately determining the cache hit ratio necessitates the implementation of a specific caching configuration. This configuration must be devised in compliance with the storage constraints inherent to the system, namely, the limited caching capacities of the involved nodes—both RSUs and CVs. Accordingly, any candidate caching strategy must ensure that the total volume of cached content does not exceed the available storage space at each node.
  Within the reinforcement learning paradigm, this process of configuring content placement is modeled as the execution of an action by the learning agent. Specifically, the agent observes the current system state, which encapsulates contextual information such as the number of caching nodes and their respective capacities, and subsequently selects a caching action that determines how content is distributed across the network.
  By mapping the action space of the reinforcement learning model to the set of feasible caching decisions encoded in this matrix, the agent is equipped to explore a range of content allocation strategies. The reward signal associated with each action—quantified by the resulting cache hit ratio—provides feedback on the effectiveness of the chosen configuration. Over successive iterations, the agent refines its action-selection policy to favor caching strategies that yield higher rewards, thereby improving the overall efficiency of content delivery in the IoV environment.
- **Reward (r):** In the T-CCCRP framework, the reward is defined as the cache hit ratio, guiding the reinforcement learning algorithm toward caching configurations that maximize local content availability. The overarching objective is to determine the optimal caching decision matrix that enhances the efficiency and responsiveness of IoV content delivery. When an action is executed based on the current system state, the agent receives an immediate reward, serving as a quantitative measure of the action's effectiveness. During training, this reward is used to iteratively update the corresponding Q-matrix entry, which represents the expected cumulative reward for each state–action pair, following the Bellman equation:

$$q(s,a) = (1-\alpha)q(s,a) + \alpha\left[r(s,a) + \gamma \max_{a'} q(s',a')\right] \tag{6}$$

  where:
  - $\alpha \in [0,1]$ is the learning rate, controlling the weight of new information;
  - $\gamma \in [0,1]$ is the discount factor, reflecting the importance of future rewards;
  - $r(s,a)$ is the immediate reward observed after executing the action a in state s;
  - s' and a' are the subsequent state and action, respectively.

  This process allows the agent to refine its policy over time by favoring actions that yield higher rewards. Once the Q-matrix stabilizes, the agent can reliably select the action with the highest Q-value for any given state, which is considered optimal within the learned environment dynamics. As an illustrative example, assume that a cluster head cache can store only 5 content items and is currently full. At a given decision epoch, the system state indicates that the content c_i is predicted to be requested frequently in the near future, while one of the cached items, c_j, has a low predicted future demand. The Q-learning agent then evaluates its action space, which may include maintaining the current cache content or replacing c_j with c_i. If the agent selects the replacement action and the subsequent requests confirm the predicted popularity of c_i, the cache hit ratio increases and the content delivery delay decreases, resulting in a positive reward. In contrast, if c_i is not requested again, the replacement yields little benefit and the received reward is lower. This iterative trial-and-feedback process enables the agent to learn caching decisions that are increasingly aligned with dynamic request patterns in the IoV environment.

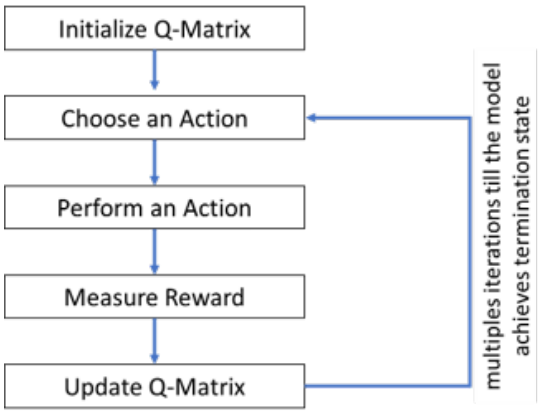

Figure 2. Q-learning algorithm process

This phase integrates two modules. The first is the **Transformer**, which is utilized to predict the content items most likely to be requested in the near future. By processing historical content request sequences, this module forecasts future demand patterns. These predictions subsequently inform the reinforcement learning model by assigning higher caching priorities to content with higher anticipated popularity, thereby improving the efficiency and responsiveness of the caching system. The second module is the content popularity calculator, which estimates the popularity of each content item based on the predicted request data.

3.3.1. Module 1: Transformer-based request prediction

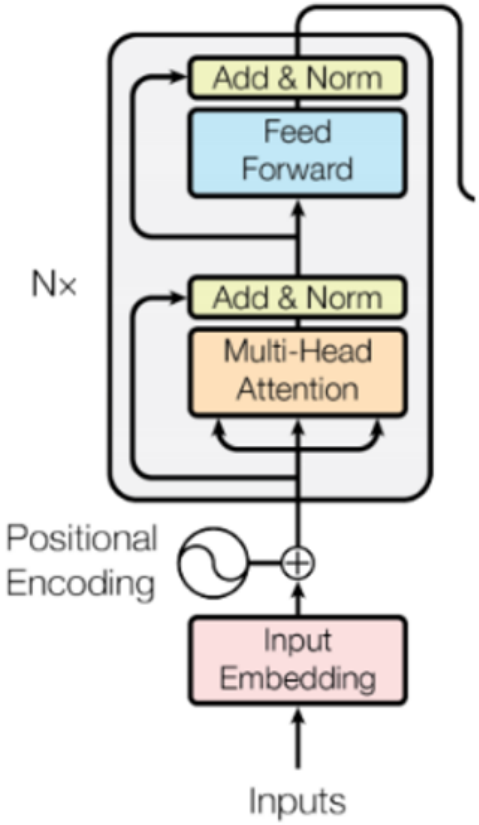

Figure 3. The Transformer (Encoder Block)

The cornerstone of the proposed T-CCCRP framework is the Transformer-based prediction module, meticulously tailored to forecast future content demands based on historical request data. Unlike conventional sequential models such as LSTM, which inherently rely on recurrent processing, Transformers utilize self-attention mechanisms, empowering them to effectively model both short- and long-range dependencies within sequential data. In particular, the Transformer architecture employed in T-CCCRP is strategically simplified by leveraging only its encoder component (Figure 3), thereby streamlining complexity and computational overhead, without compromising predictive accuracy essential for time-series forecasting tasks in IoV.

The data processing workflow begins by representing content requests observed at discrete intervals as vectors containing the frequency of requests for individual content items. Each input vector $x(t-1) = [x_1(t-1), x_2(t-1), \ldots, x_Q(t-1)]$ thus represents historical data at a given time step $(t-1)$, serving as the fundamental input to the Transformer encoder. The encoder systematically processes these sequences to yield a predictive output vector $x^*(t) = [x_1^*(t), x_2^*(t), \ldots, x_Q^*(t)]$ for the subsequent interval (t), encapsulating estimated request frequencies for each item. Through this systematic processing, the Transformer effectively identifies complex temporal patterns characteristic of vehicular content request behaviors, thereby providing a highly accurate predictive foundation for subsequent caching decisions.

At a structural level, the Transformer encoder consists of several critical components strategically arranged to maximize predictive efficacy. Initially, an input embedding layer translates content request tokens into high-dimensional vector representations, capturing intricate contextual patterns crucial for accurate sequence modeling. Recognizing that Transformers do not inherently encode positional information, positional encodings are explicitly integrated, assigning distinct positional context to each vector. This embedding of positional information is crucial in ensuring the Transformer maintains temporal coherence, accurately capturing sequential dynamics inherent in the data.

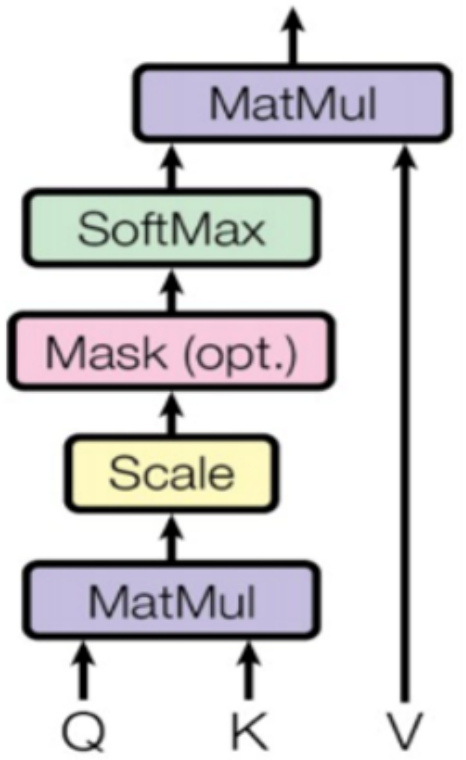

Figure 4. Single-Head Attention

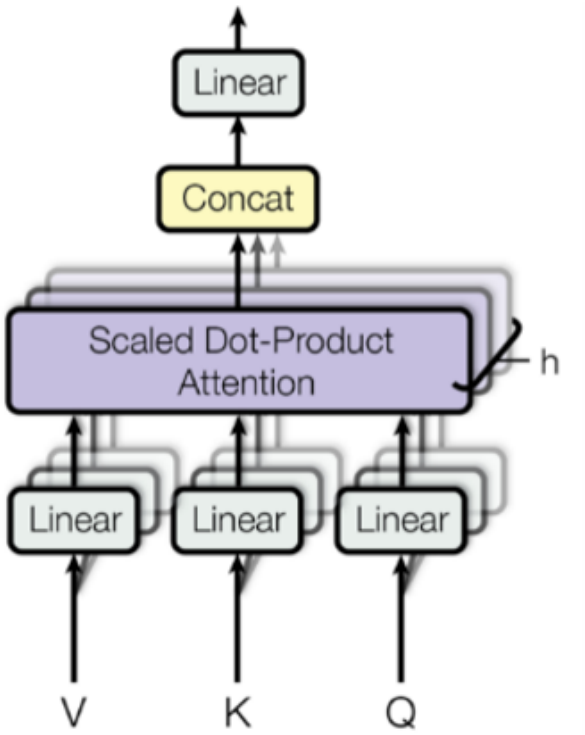

Figure 5. Multi-Head Attention

Central to the transformative capabilities of the Transformer architecture is the self-attention mechanism, implemented in both single-head (Figure 4) and multi-head (Figure 5) configurations. Single-head attention begins by projecting the input embedding matrix $\boldsymbol{X}$ into three distinct matrices—$\boldsymbol{Q}$ (query), $\boldsymbol{K}$ (key), and $\boldsymbol{V}$ (value)—using distinct linear transformations, through multiplication with learned projection matrices $\boldsymbol{W}^Q$, $\boldsymbol{W}^K$ and $\boldsymbol{W}^V$, respectively, resulting in:

$$\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W}^Q, \qquad \boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}^K, \qquad \boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}^V \tag{7}$$

Queries represent current tokens requiring contextual information, keys encode characteristics of tokens within the sequence, and values represent the contextual information aggregated by the model.

The attention mechanism computes normalized pairwise similarity scores between queries and keys by:

$$\text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right) \tag{8}$$

where d_k is the dimension of $\boldsymbol{K}$ and the factor $\sqrt{d_k}$ is used to solve the "vanishing gradient" problem caused by softmax. The resulting attention weights are then used to compute a weighted sum of the value vectors $\boldsymbol{V}$, producing output representations that incorporate relevant contextual information from the entire sequence. This process can be mathematically expressed as follows:

$$\text{Attention}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V} \tag{9}$$

To further enhance predictive performance, multi-head self-attention extends this process by simultaneously applying multiple attention operations across parallel subspaces. Each parallel head independently computes its own queries, keys, and values, generating distinct representations of sequence context. The outputs from each head (Head_i) are subsequently concatenated and linearly projected with a matrix $\boldsymbol{W}^O$ into a unified embedding. This parallel processing allows the Transformer to capture multiple nuanced dependencies simultaneously, markedly enhancing its representational power and predictive accuracy. The operation can be formally expressed as:

$$\text{MultiHead}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \text{Concat}(\text{Head}_1,\ldots,\text{Head}_h)\boldsymbol{W}^O \tag{10}$$

Following the multi-head attention mechanism, the Transformer incorporates a position-wise feed-forward network (FFN) applied independently at each token in the sequence. This network consists of two linear transformations interspersed with a nonlinear activation function, typically ReLU, facilitating deeper and more intricate feature extraction. The FFN is mathematically expressed as:

$$\text{FFN}(x) = \text{ReLU}(x\boldsymbol{W}_1 + b_1)\boldsymbol{W}_2 + b_2 \tag{11}$$

where $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are weight matrices, and b_1, b_2 are the corresponding bias vectors.

Additionally, residual connections coupled with layer normalization are employed throughout the encoder layers to ensure stable training dynamics and efficient gradient flow, enabling deeper network structures without loss of information. The operation is formally defined as:

$$\text{Output} = \text{LayerNorm}(x + \text{Sublayer}(x)) \tag{12}$$

where $\text{Sublayer}(x)$ is the function implemented by the sub-layer itself.

3.3.2. Module 2: Content popularity calculator

Upon completion of the Transformer-based prediction process, the predicted request vector $x^*(t)$ is sorted to generate a ranked content index vector $o_Q(t) = [o_1(t), o_2(t), \ldots, o_Q(t)]$—where $o_q(t)$ is the rank of content q—reflecting anticipated demand in the upcoming time interval (t). This ranked vector serves as the input into the Zipf model to compute content popularity $P_Q = [P_1, P_2, \ldots, P_Q]$ as follows:

$$P_q = \frac{o_q^{-\varphi}}{\sum_{i=1}^{Q} o_i^{-\varphi}} \tag{13}$$

where φ is the skewness parameter, which controls how steeply popularity declines with rank. This ensures that frequently predicted items are assigned higher caching priorities, aligning with realistic traffic distributions observed in vehicular environments.

4. Experimental Setup

To address the limitations identified in the initial evaluation and to ensure a more realistic and comprehensive assessment of the proposed framework, the experimental setup has been significantly extended to incorporate multiple large-scale IoV scenarios. These enhancements aim to evaluate the scalability, robustness, and generalization capability of the proposed T-CCCRP framework under varying network densities, content library sizes, and temporal dependencies. The experimental setup includes the dataset utilized, the environment configuration, detailed parameter configurations—including explicit justifications for key hyperparameters—and the selection of relevant performance evaluation metrics. Additionally, comparative methods are outlined to establish the benchmark against which the proposed T-CCCRP framework's effectiveness is measured.

4.1. Dataset

To ensure consistency and facilitate performance comparison with prior work, this study reuses the same dataset employed in the original CCCRP paper [27]. The dataset comprises time-series content request data generated as part of the Big Data Challenge [30]. Specifically, it includes the request records for ten distinct content items, collected over the period from January 11, 2013, to January 1, 2014, with a temporal resolution of 10 minutes per observation. In total, the dataset includes 1,000 samples, each representing a discrete time interval and capturing the aggregated number of requests for each content item.

4.2. Simulation environment

The simulation environment was redesigned to emulate realistic and scalable IoV deployments. In addition to the original small-scale configuration, three larger scenarios were introduced, characterized by increasing numbers of vehicles, clusters, caching nodes, and content items. These scenarios are summarized as follows:

- **Large scenario 1 (Medium-scale):** 50 vehicles organized into 5 clusters, with 6 caching nodes (5 CVs + 1 RSU), and a content library of 30 items.
- **Large scenario 2 (Large-scale):** 100 vehicles organized into 10 clusters, with 11 caching nodes, and a content library of 40 items.
- **Large scenario 3 (Very large-scale):** 200 vehicles organized into 20 clusters, with 21 caching nodes, and a content library of 60 items.

These configurations ensure that the evaluation captures varying levels of network density and caching complexity.

Vehicles are randomly distributed within the simulated environment and exhibit dynamic mobility patterns. Each cluster is managed by a cluster head vehicle (CV) and supported by an RSU, reflecting a hierarchical cooperative caching architecture commonly observed in IoV systems. The increase in vehicle density introduces higher variability in request patterns and intensifies competition for caching resources, thereby enabling a more realistic evaluation of caching strategies.

The content request process continues to follow a Poisson distribution; however, the arrival rate is adjusted proportionally to the number of vehicles in each scenario to reflect realistic traffic demand scaling. The simulation is conducted over 1,000 episodes, divided equally into training and testing phases, ensuring stable learning and reliable performance evaluation across all configurations.

4.3. Parameter configuration

To ensure the simulation's realism and reproducibility, meticulous consideration was given to the selection of all system, model, reinforcement learning, and communication parameters. Explicit justifications for critical hyperparameters are provided to reinforce methodological transparency and validity.

4.3.1. System parameters

The system parameters presented in Table 2, were extended to reflect realistic IoV conditions under varying scales. In particular, the number of vehicles, clusters, caching nodes, and content items increases progressively across scenarios. Content sizes remain uniformly distributed

between 1 MB and 3 MB, while caching capacities are scaled proportionally with the number of caching nodes to maintain comparable resource availability per node. The Zipf parameter governing content popularity is preserved across scenarios to ensure consistent evaluation of caching performance under varying demand distributions.

Table 2. System Parameters

Parameter	**Baseline scenario**	**Large-Scale Scenario 1**	**Large-Scale Scenario 2**	**Large-Scale Scenario 3**
Number of vehicles	15	50	100	200
Number of vehicle clusters	2	5	10	20
Number of caching nodes	3 (CVs + 1 RSU)	6 (CVs + 1 RSU)	11 (CVs + 1 RSU)	21 (CVs + 1 RSU)
Cache capacity per (CV, RSU)	(5, 10) MB	(5, 10) MB	(5, 10) MB	(5, 10) MB
Total content items	10	30	40	60
Content size	1–3 MB	1–3 MB	1–3 MB	1–3 MB
Zipf parameter	0.7	0.7	0.7	0.7

4.3.2. Transformer Model Parameters

The Transformer model parameters, summarized in Table 3, were selected based on a combination of preliminary empirical experimentation and established best practices within sequential data modelling literature. Specifically, an embedding dimension of 64 was adopted due to its proven effectiveness in capturing sequence complexities in similar studies [7, 8]. The number of attention heads was set to 4, achieving a balance between computational complexity and model performance, consistent with configurations demonstrated to yield optimal outcomes in prior Transformer-based research [9]. The feed-forward network dimension was set at 128, offering sufficient representational power without excessively increasing computational overhead. Furthermore, the dropout rate of 0.1 was chosen following standard practices aimed at preventing model overfitting while maintaining efficient learning dynamics. However, to align the prediction model with the increased system scale, the input sequence length of the Transformer was adjusted proportionally to the size of the content library and the number of vehicles. Specifically, sequence lengths of 30, 40, and 60 were adopted for the medium-, large-, and very large-scale scenarios, respectively. This adjustment ensures that the model captures sufficient temporal context as the complexity of content request patterns increases.

Table 3. Transformer Model Parameters

Parameter	**Value**			
	Baseline scenario	**Large-Scale Scenario 1**	**Large-Scale Scenario 2**	**Large-Scale Scenario 3**
Input sequence length	10	30	40	60
Embedding dimension (d_{model})	64			
Number of attention heads	4			
Feed-forward network dimension	128			
Number of Transformer blocks	2			
Dropout rate	0.1			
Optimizer	Adam			
Loss function	MSE			
Training epochs	1000			
Batch size	32			

4.3.3. Reinforcement Learning Parameters

The reinforcement learning parameters detailed in Table 4 were explicitly configured to ensure effective and stable training of the caching decision module. Specifically, the learning rate ($\alpha = 0.05$) was selected to provide a gradual yet stable convergence during training. The discount factor ($\gamma = 0.1$) was deliberately set to prioritize immediate rewards—appropriate in caching decisions where immediate caching hits significantly outweigh delayed benefits. The initial exploration rate ($\varepsilon = 1.0$) and its decay rate (0.995) were systematically chosen to balance exploration of various caching configurations early in training and exploitation of learned strategies as training progressed.

Table 4. Reinforcement Learning Parameters

Parameter	Value
Learning rate (α)	0.05
Discount factor (γ)	0.1
Initial exploration rate (ε)	1.0
Exploration decay rate	0.995
Episodes (train/test)	500 / 500

4.3.4. Communication Parameters

Communication parameters, summarized in Table 5, define realistic channel characteristics within the simulated IoV environment. These parameters include allocated bandwidth and Signal-to-Interference-plus-Noise Ratio (SINR) for both V2V and V2I communications, and remote server access delays. They were selected to realistically reflect operational network conditions that vehicles typically experience, thus directly influencing caching performance.

Table 5. Communication Parameters

Parameter	Value
Bandwidth	10 MHz
SINR (V2V)	10 dB
SINR (V2I)	20 dB
Remote server access delay	5 s

To comprehensively evaluate the effectiveness of the proposed T-CCCRP framework, a set of robust performance metrics was employed in conjunction with comparative benchmarking against established baseline methods. The primary evaluation metrics include the cache hit ratio, the average content acquisition delay, and predictive accuracy as measured through confusion matrix analysis. The cache hit ratio quantifies the proportion of content requests successfully served from local caches—either vehicle-based (CVs) or infrastructure-based (RSUs)—thus directly reflecting the strategy's efficiency in reducing reliance on remote servers. The average content acquisition delay captures the end-user latency associated with content retrieval and serves as a practical indicator of QoS, particularly in delay-sensitive IoV applications. Predictive accuracy, assessed using standard classification metrics such as precision, recall, and F1-score, evaluates the reliability of the predictor model's forecasting capability, which critically informs the caching decision process. In this context, precision measures the proportion of correctly predicted positive instances among all predicted positives, recall quantifies the proportion of correctly identified positive instances among all actual positives, and the F1-score represents the harmonic mean of precision and recall, providing a balanced evaluation of the model's performance.

To contextualize the performance of T-CCCRP, it was rigorously compared against three baseline caching strategies: LFU, which caches items based on access frequency without accounting for recency or future trends; LRU, which prioritizes recently accessed content but overlooks broader usage patterns; and the LSTM-based CCCRP, which leverages temporal sequence modelling for prediction but is constrained by limited scalability and long-term dependency challenges. These comparative methods, selected for their conceptual diversity and prevalence in IoV caching literature, enable a comprehensive assessment of T-CCCRP's advantages across multiple dimensions, including cache efficiency, latency reduction, and predictive robustness under varying caching capacities and content popularity distributions.

5. Results and Discussion

This section provides a comprehensive evaluation of the proposed T-CCCRP framework through a detailed comparison with baseline caching strategies, including LFU, LRU, and the CCCRP. The discussion focuses on three primary performance dimensions: cache hit ratio, average content acquisition delay, and prediction accuracy. These metrics are analyzed under varying simulation conditions, including changes in caching capacity and content popularity distribution, to demonstrate the robustness, adaptability, and efficiency of the T-CCCRP framework within the dynamics of IoV environments.

5.1. Cache Hit Ratio

The evaluation of the cache hit ratio, a key performance indicator reflecting the efficiency of caching strategies, is conducted across four distinct scenarios. The first scenario corresponds to the original configuration used to evaluate the CCCRP strategy, ensuring a direct and fair comparison, while the remaining three scenarios are specifically designed to assess the scalability and robustness of the proposed T-CCCRP approach, demonstrating its consistent superiority over baseline methods under varying system conditions.

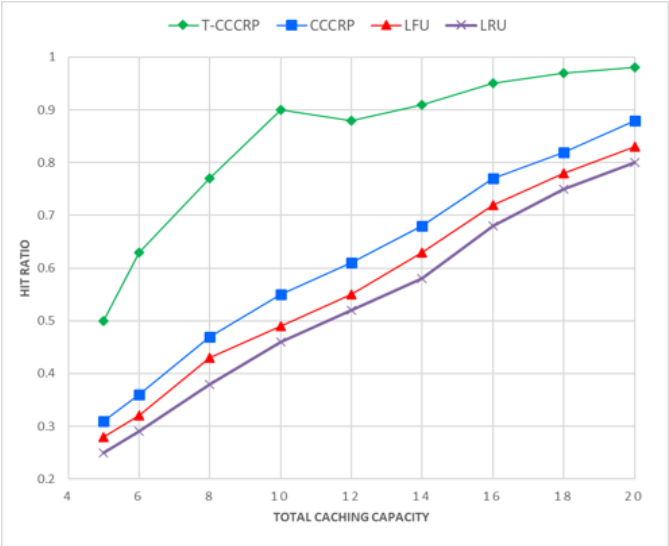

Figure 6. Cache Hit Ratio vs. Total Caching Capacity (Baseline scenario)

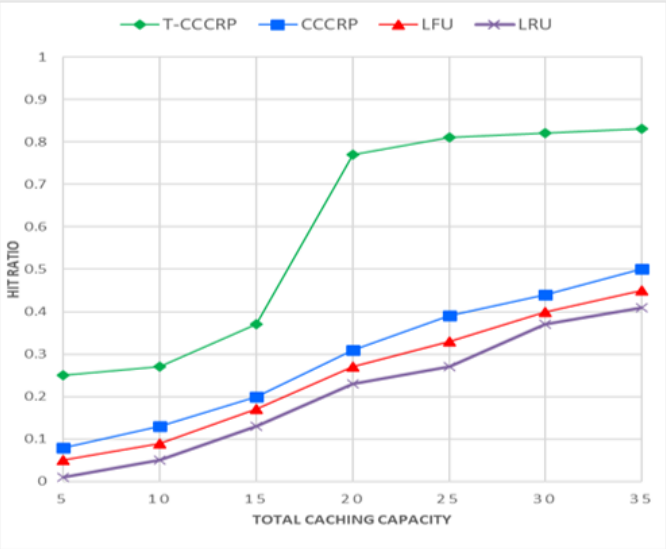

Figure 7. Cache Hit Ratio vs. Total Caching Capacity (Large-Scale scenario 1)

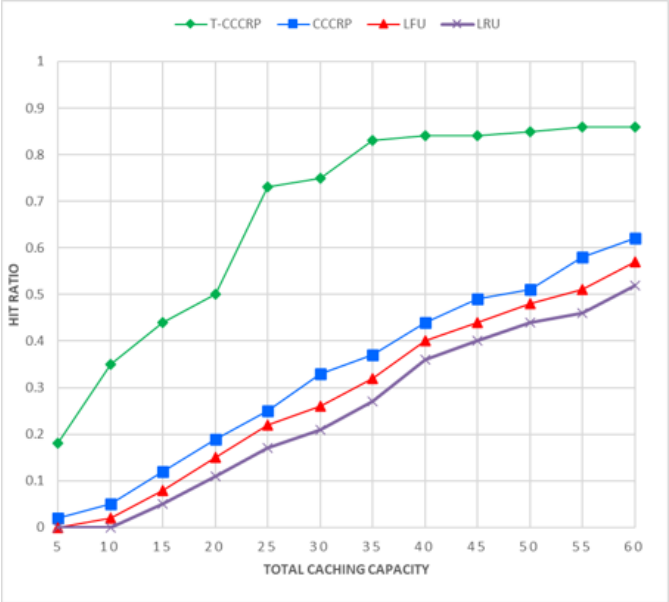

Figure 8. Cache Hit Ratio vs. Total Caching Capacity (Large-Scale scenario 2)

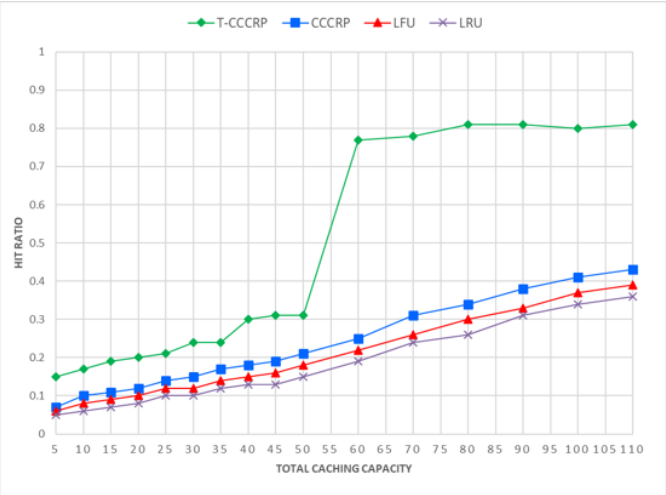

Figure 9. Cache Hit Ratio vs. Total Caching Capacity (Large-Scale scenario 3)

The discussion starts with the baseline scenario, used as the reference case for evaluating and comparing the performance of all caching strategies. The results in Figure 6 highlight how T-CCCRP adapts effectively to varying caching capacities, initially demonstrating remarkable efficiency even with minimal resources. Specifically, at the lowest tested capacity of 5 MB, T-CCCRP achieves a cache hit ratio near 50%, notably outperforming CCCRP at 35%, LFU at 30%, and LRU at 28%, underscoring its superior capability in accurately identifying and prioritizing critical content under tight resource constraints. As caching capacities increase from 10 MB to 20 MB, T-CCCRP consistently maintains its performance advantage, reaching approximately 90% at 10 MB, which sharply contrasts with CCCRP's 55%, LFU's 48%, and LRU's 45%. At intermediate capacities (15 MB), T-CCCRP achieves a near-saturation performance of 94%, whereas CCCRP improves modestly to 70%, with LFU and LRU reaching only 65% and 60%, respectively. Remarkably, at an optimal threshold of 20 MB, T-CCCRP attains a peak hit ratio of approximately 98%, significantly surpassing CCCRP (88%), LFU (82%), and LRU (80%), thus indicating the model's robust predictive accuracy and optimal cache management at moderate resource levels.

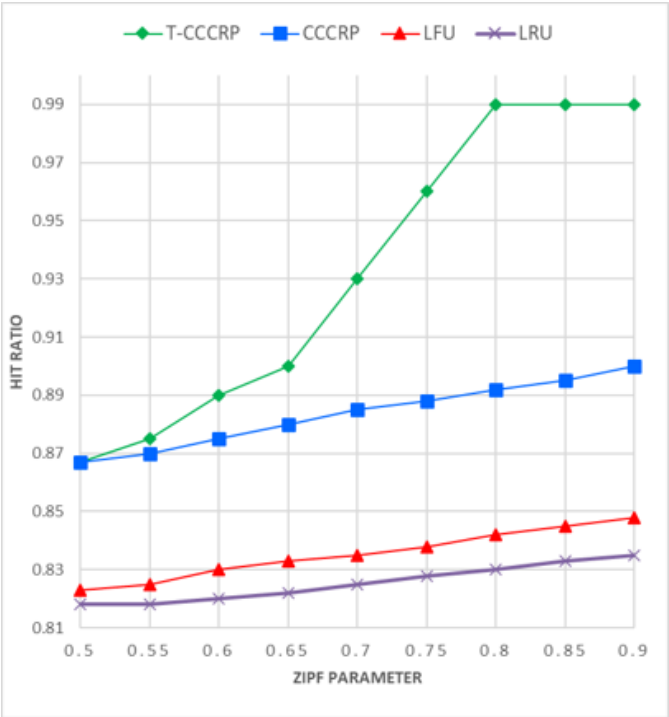

Figure 10. Cache Hit Ratio vs. Zipf Parameter (Baseline scenario)

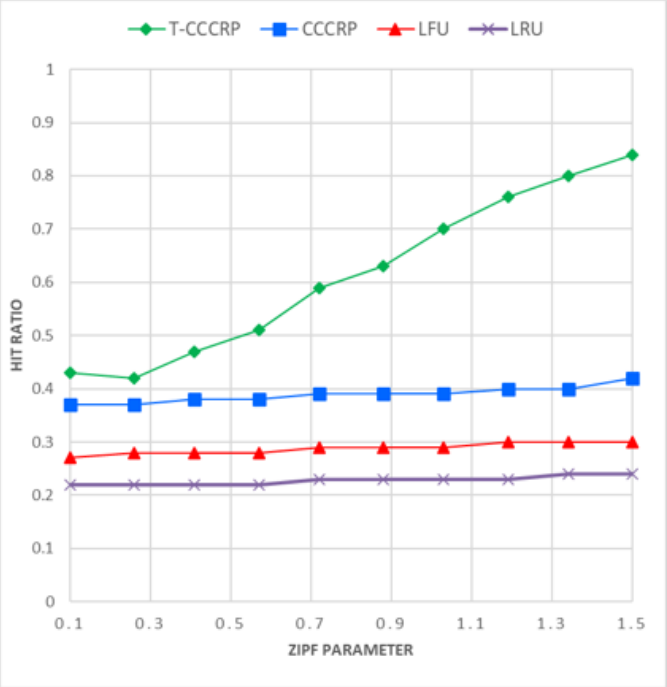

Figure 11. Cache Hit Ratio vs. Zipf Parameter (Large-Scale scenario 1)

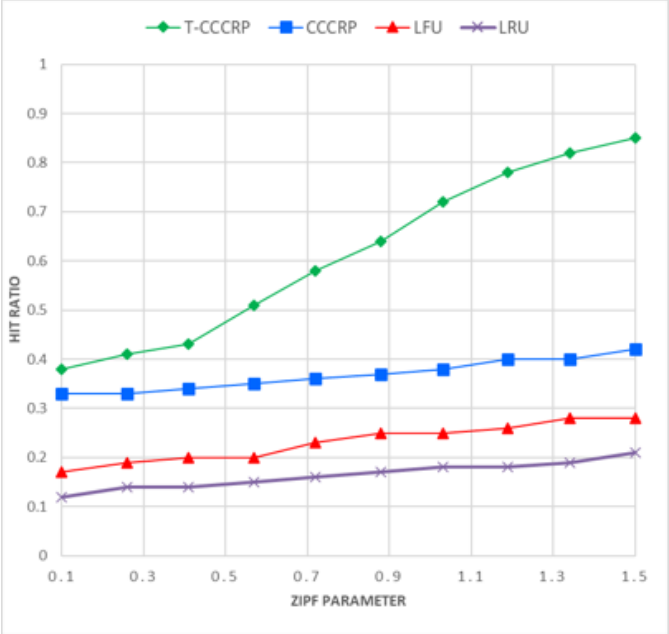

Figure 12. Cache Hit Ratio vs. Zipf Parameter (Large-Scale scenario 2)

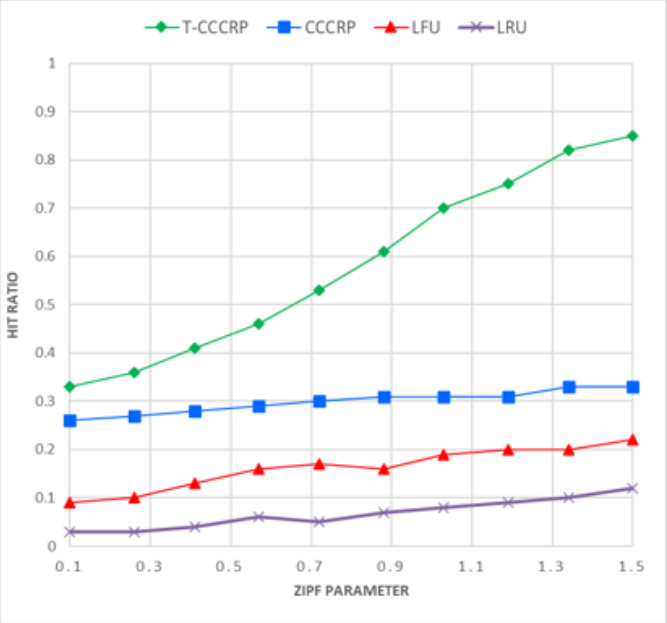

Figure 13. Cache Hit Ratio vs. Zipf Parameter (Large-Scale scenario 3)

Furthermore, as shown in Figure 10, we analyzed the sensitivity of the cache hit ratio to variations in content popularity distributions, represented by different Zipf parameters. For scenarios with lower Zipf parameters (≤ 0.65), indicating more uniform content request distributions, the T-CCCRP already demonstrates robust performance ($\geq$ 90% hit ratio). With increasingly skewed distributions (Zipf ≥ 0.75), indicative of real-world scenarios where a small fraction of content attracts most requests, T-CCCRP's performance escalates to near-perfect levels (˜99%), markedly outperforming the baseline methods.

In contrast, baseline caching strategies (CCCRP, LFU, and LRU) experience only modest incremental improvements and fail to approach the T-CCCRP's high adaptability and efficiency, remaining at least 5–10 percentage points lower. Specifically, CCCRP maintains a slight advantage over LFU and LRU but remains substantially below the T-CCCRP's adaptive capabilities.

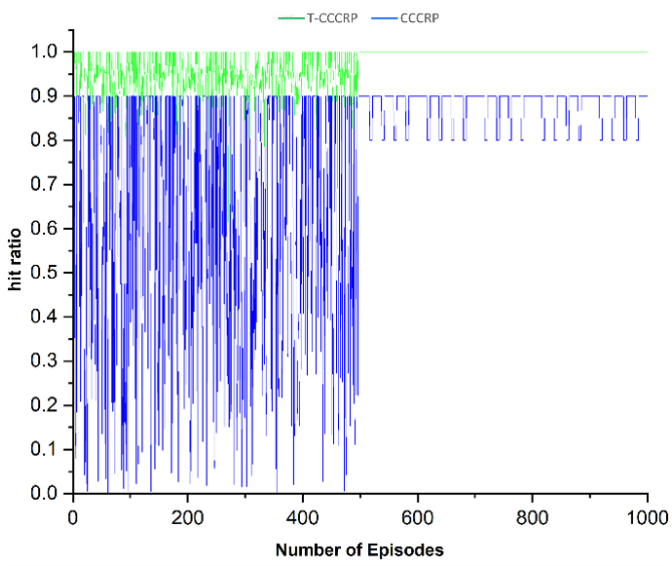

Figure 14. Comparison of cache hit ratios between strategies (Baseline scenario)

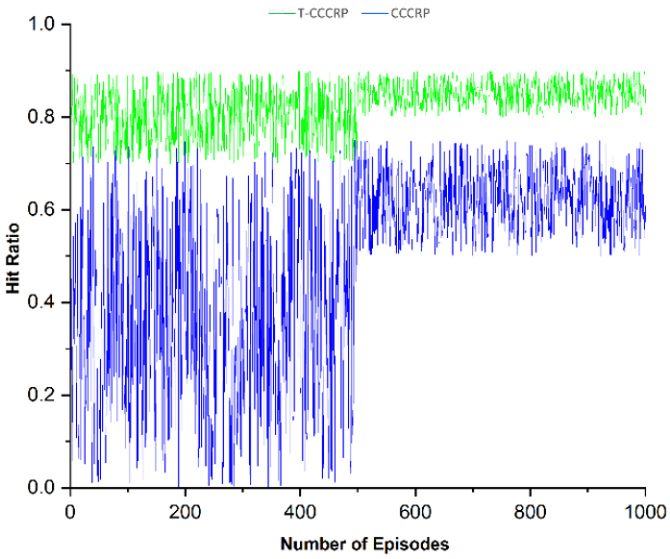

Figure 15. Comparison of cache hit ratios between strategies (Large-Scale scenario 1)

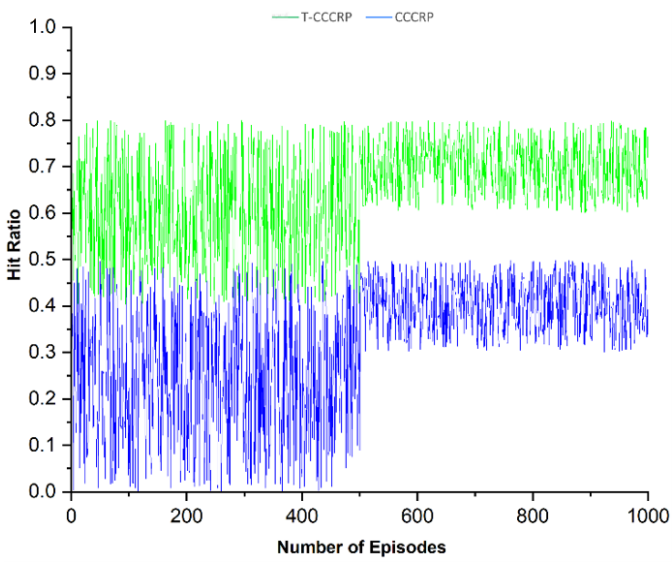

Figure 16. Comparison of cache hit ratios between strategies (Large-Scale scenario 2)

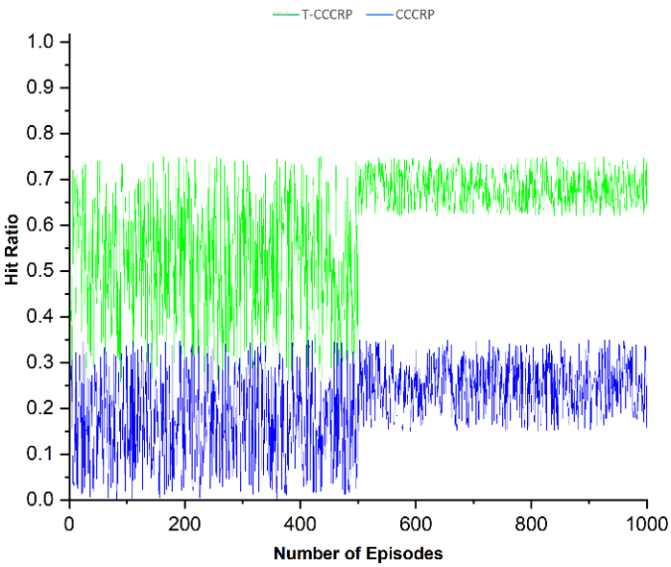

Figure 17. Comparison of cache hit ratios between strategies (Large-Scale scenario 3)

In Figure 14, the comparison between the T-CCCRP and CCCRP caching strategies over the course of 1000 simulation episodes reveals clear differences in performance stability, learning efficiency, and generalization capability. During the initial training phase, CCCRP exhibits significant variability in cache hit ratio, fluctuating between extreme values as low as 0.01 and occasionally peaking near 0.90. This erratic behavior reflects the LSTM's inherent sensitivity to the sequential ordering of input data and its tendency to overfit to short-term request patterns. The model's recurrent nature, while theoretically designed to capture temporal dependencies, appears to struggle with the complex and volatile nature of vehicular content demand.

In contrast, T-CCCRP demonstrates a markedly different learning profile. After a brief period of exploration in the early episodes, its cache hit ratio rapidly converges toward a stable value of approximately 0.98. Unlike CCCRP, T-CCCRP maintains this high level of performance with minimal oscillation, thanks to its ability to model long-range dependencies through multi-head self-attention mechanisms. This architectural advantage enables T-CCCRP to learn global request patterns efficiently, even under dynamic and noisy conditions typical of IoV scenarios.

The transition from training to testing further highlights the T-CCCRP's robustness. While CCCRP stabilizes at an average hit ratio of around 0.90 during the testing phase, it continues to show moderate fluctuations of about ± 0.05, indicating limited generalization to new, unseen data. These fluctuations suggest that CCCRP, though improved, remains somewhat brittle in adapting to shifting content request distributions.

T-CCCRP, however, maintains a near-perfect cache hit ratio of approximately 0.98 throughout the testing episodes. Its output remains smooth and consistent, demonstrating a strong capacity for generalization and a high degree of resilience to variations in request patterns. This performance stability across both training and testing phases underscores the Transformer's ability to internalize complex temporal structures and adapt to

evolving content demand with exceptional precision.

To address scalability and robustness, additional large-scale simulations were conducted by jointly increasing the number of vehicles, content items, and input sequence length, thereby emulating more realistic IoV conditions and evaluating the behavior of T-CCCRP under increasing system complexity.

In the first large-scale scenario, the results in Figure 7 indicate that T-CCCRP consistently outperforms all baseline methods across caching capacities, achieving a cache hit ratio of approximately 0.83 at higher capacities, compared to 0.50 for CCCRP, 0.45 for LFU, and 0.41 for LRU. From a Zipf-based perspective, Figure 11 shows that performance increases with content popularity skewness, rising from about 0.43 to 0.84, demonstrating the model's effectiveness in capturing concentrated demand patterns. In terms of convergence, Figure 15 demonstrates that T-CCCRP rapidly stabilizes after initial exploration, whereas CCCRP exhibits slower and more unstable learning dynamics.

In the second scenario, as system complexity increases, T-CCCRP further improves performance, reaching approximately 0.86 at high capacities, significantly surpassing baseline approaches (Figure 8). The Zipf analysis shows a similar trend, with performance increasing from 0.38 to 0.85, and the performance gap becoming more pronounced at moderate skew levels (Figure 12). Convergence behavior also improves, as shown in Figure 16, with faster stabilization and smoother learning curves, while CCCRP continues to display residual fluctuations.

In the largest scenario, despite increased request diversity and cache contention, as shown in Figure 9, T-CCCRP maintains superior performance, achieving around 0.80 at higher capacities compared to 0.41, 0.37, and 0.34 for CCCRP, LFU, and LRU, respectively. The Zipf-based results, as illustrated in Figure 13, confirm robustness under highly skewed demand, with performance increasing from approximately 0.33 to 0.85. As illustrated in Figure 17, the convergence analysis further emphasizes the stability and efficiency of the learning process, with rapid stabilization and minimal oscillations, unlike CCCRP, which remains less stable.

Across all scenarios, a clear relationship emerges between temporal context and caching performance. As sequence length increases, the Transformer benefits from richer temporal representations, leading to more accurate popularity estimation and improved cache allocation. This is particularly evident in faster convergence and enhanced stability.

Overall, the extended evaluation confirms that T-CCCRP achieves consistently higher cache hit ratios, faster convergence, and greater stability across varying conditions, demonstrating strong scalability and suitability for real-world IoV environments.

5.2. Average content acquisition delay

The evaluation of average content acquisition delay is conducted across four distinct simulation scenarios. The first scenario corresponds to the original experimental setting used for evaluating the CCCRP strategy, while the second, third, and fourth scenarios are newly introduced large-scale configurations designed to demonstrate that the proposed T-CCCRP framework consistently outperforms baseline approaches under varying network conditions and system complexities.

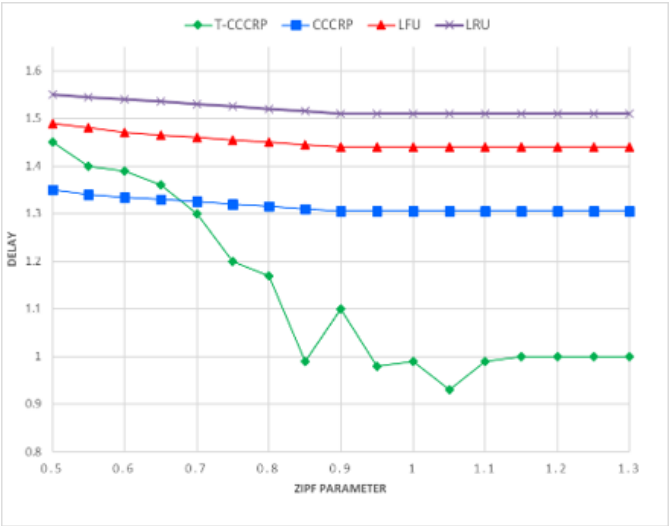

Figure 18. Delays vs. Zipf parameter (Baseline scenario)

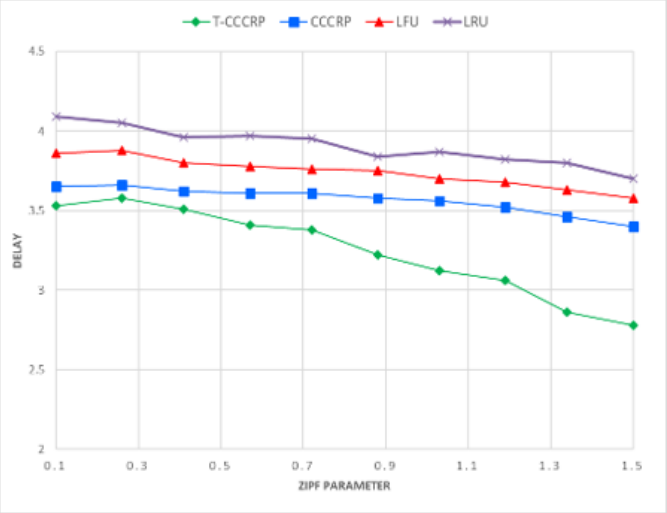

Figure 19. Delays vs. Zipf parameter (Large-Scale scenario 1)

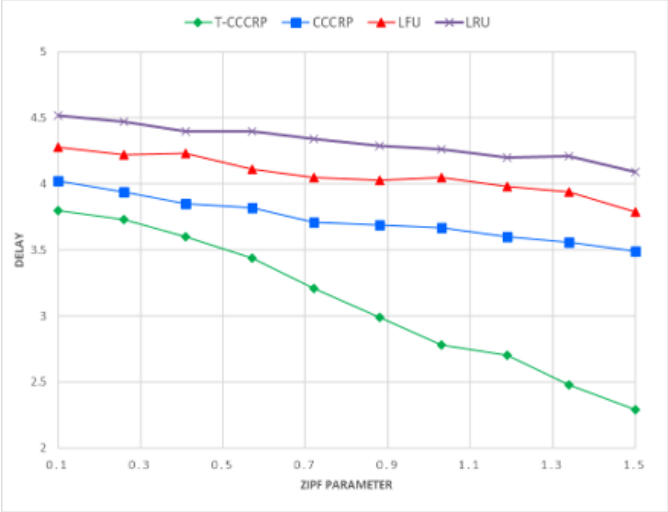

Figure 20. Delays vs. Zipf parameter (Large-Scale scenario 2)

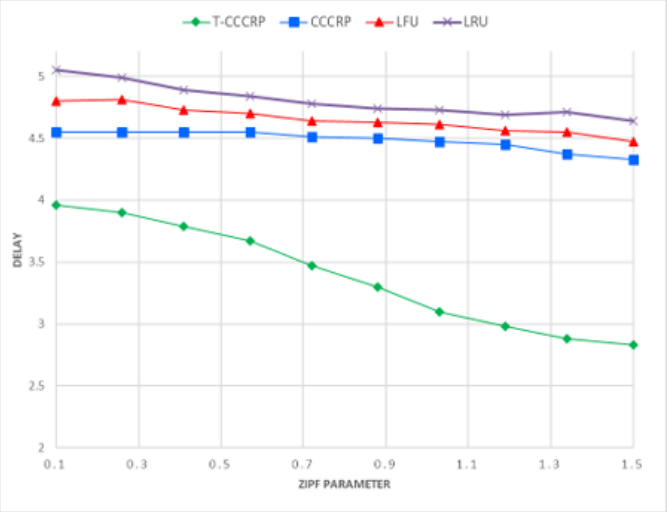

Figure 21. Delays vs. Zipf parameter (Large-Scale scenario 3)

The analysis begins with the baseline scenario. Simulation results in Figure 18 show that T-CCCRP consistently achieves the lowest acquisition delays across all tested Zipf parameters. For instance, at a Zipf parameter of 0.90, which reflects highly skewed content popularity, the T-CCCRP achieves an average delay of just 0.90 seconds. In comparison, CCCRP yields 1.27 seconds, while LFU and LRU reach 1.41 and 1.49 seconds, respectively. Even at lower Zipf values (e.g., 0.50), where content popularity is more evenly distributed, T-CCCRP remains competitive with a delay of 1.45 seconds, slightly outperforming LFU and LRU, which plateau near 1.50 seconds. These results underscore the Transformer's ability to adaptively prioritize popular content, reducing retrieval from remote sources and thereby lowering average delay.

The delay data further reinforces the observed trend of T-CCCRP's superior adaptability and efficiency. Baseline caching strategies exhibit gradual improvement with increasing Zipf skewness but fail to exploit high skew levels effectively, often plateauing in performance. In contrast, the Transformer's prediction mechanism dynamically adjusts to the distribution shift, quickly identifying high-demand items and ensuring their availability in local caches. This responsiveness is crucial in urban vehicular networks, where request patterns may vary sharply within short time spans due to traffic flow changes, localized events, or shifting user behavior.

The evaluation of average content acquisition delay was extended to large-scale simulation scenarios in order to assess the scalability and robustness of the proposed T-CCCRP framework under more realistic IoV conditions. These scenarios vary the number of vehicles, content items, and input sequence length, enabling a comprehensive analysis of how system scale affects delay performance.

In the first large-scale scenario, as shown in Figure 19, T-CCCRP consistently achieves lower delays across all Zipf parameter values compared to CCCRP, LFU, and LRU. As content popularity becomes more skewed, the delay decreases from approximately 3.53 s to 2.78 s, indicating improved caching efficiency. By contrast, baseline methods show only limited improvement, confirming that Transformer-based prediction enhances responsiveness even at a moderate scale.

In the second scenario, the delay reduction becomes more pronounced, as presented in Figure 20, T-CCCRP decreases from 3.8 s to 2.29 s, while CCCRP and traditional methods remain significantly higher. The performance gap widens with increased system scale, highlighting the growing effectiveness of the Transformer model as both request diversity and input sequence length expand. The longer temporal context enables more accurate prediction of content popularity, leading to improved caching decisions.

In the largest scenario, as depicted in Figure 21, T-CCCRP maintains strong scalability despite increased system complexity. Although absolute delays slightly increase due to higher contention and content diversity, T-CCCRP consistently outperforms baseline methods, achieving delays between 3.96 s and 2.83 s, while competing approaches remain above 4.3 s. This demonstrates the model's ability to mitigate the adverse effects of scale through more accurate prediction and adaptive caching.

A cross-scenario analysis yields three key insights. First, increasing the number of vehicles intensifies network load and raises baseline delays; however, T-CCCRP preserves a clear performance advantage, indicating robustness in dense environments. Second, expanding the content library increases request diversity, yet the Transformer-based predictor effectively adapts by accurately estimating content popularity. Third, increasing the input sequence length improves predictive accuracy by capturing richer temporal dependencies, directly contributing to delay reduction.

Overall, these findings confirm that T-CCCRP scales effectively with network size, content diversity, and temporal complexity. Its integration of Transformer-based prediction with reinforcement learning-driven caching enables sustained low latency and high responsiveness, supporting its applicability to large-scale IoV deployments.

5.3. Prediction capabilities (Confusion matrix analysis)

To ensure transparency and address the evaluation-scale concern, the prediction capabilities of the proposed T-CCCRP framework are assessed under two complementary scenarios: a Toy scenario and a Realistic scenario. The Toy scenario considers a content library of 10 content items, enabling an intuitive small-scale analysis of prediction behavior. In contrast, the Realistic scenario is based on 4,000 prediction events, providing a statistically more reliable evaluation of predictive performance under workload conditions that better approximate realistic IoV traffic dynamics. To guarantee a fair comparison, all other experimental parameters—including the number of vehicles, caching nodes, network configuration, and

model settings—are kept unchanged across both scenarios.

For methodological consistency with the CCCRP baseline, the Transformer predictor is trained exclusively on the original 1,000 time-series samples. During the testing phase, the evaluation workload is expanded through temporal resampling of the original request sequence, while preserving its statistical properties and Zipf-based popularity distribution. This process generates 4,000 aggregated prediction events across multiple testing episodes, which are used to construct the confusion matrices and compute the corresponding accuracy, precision, recall, and F1-score metrics. This protocol ensures both reproducibility and a more robust assessment of the predictive effectiveness of the proposed framework in realistic vehicular caching environments.

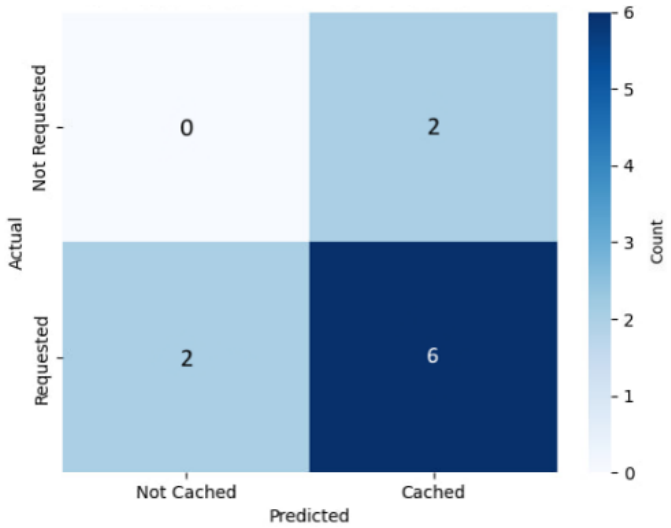

Figure 22. Confusion Matrix for content caching predictions (Toy Scenario)—CCCRP

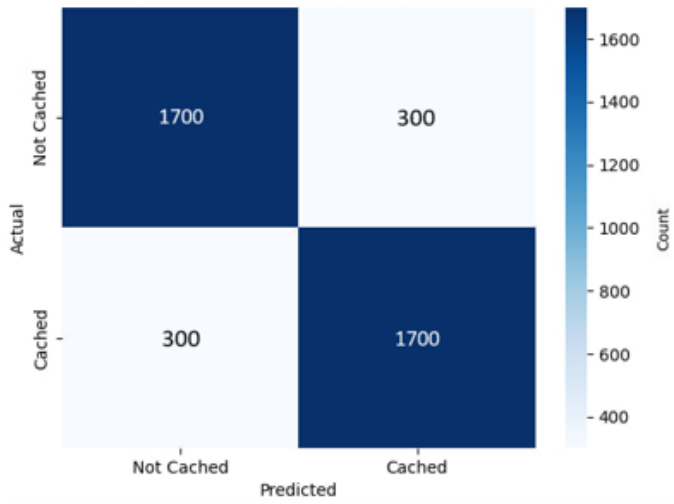

Figure 23. Confusion Matrix for content caching predictions (Realistic Scenario)-CCCRP

In the case of the original CCCRP strategy, an initial toy example involving 10 content access events provided a baseline assessment of predictive behavior (Figure 22). Out of the 10 events, the original CCCRP strategy correctly identified 6 actual requests (true positives) but mistakenly cached 2 items that were not ultimately requested (false positives) and failed to cache 2 items that were later requested (false negatives). The resulting performance metrics were as follows: an accuracy of 60.0%, precision and recall both at 75.0%, and an F1-score of 75.0%. These results indicate that, although the LSTM model can capture fundamental patterns in user behavior, it demonstrates a moderate degree of over-caching and missed caching opportunities in constrained environments. The symmetry between precision and recall further suggests that the original CCCRP strategy tends to err conservatively in both caching and non-caching decisions.

When scaled to a realistic IoV workload of 4,000 content access events (Figure 23), the original CCCRP strategy exhibited significant improvements across all performance metrics. It achieved 1,700 true positives and an equal number of true negatives, with 300 false positives and 300 false negatives. This configuration yielded an overall accuracy of 85.0%, with matching precision, recall, and F1-score values of 85.0%. These results highlight the LSTM model's ability to generalize effectively as data volume increases. The model maintains a balanced performance between capturing the majority of requested content and avoiding excessive caching of unnecessary items, making it reasonably suited for latency-sensitive cooperative caching.

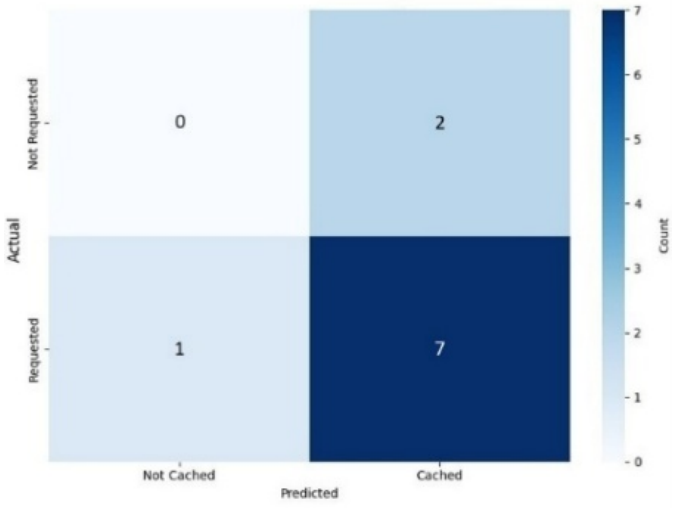

Figure 24. Confusion Matrix for content caching predictions (Toy Scenario)—T-CCCRP

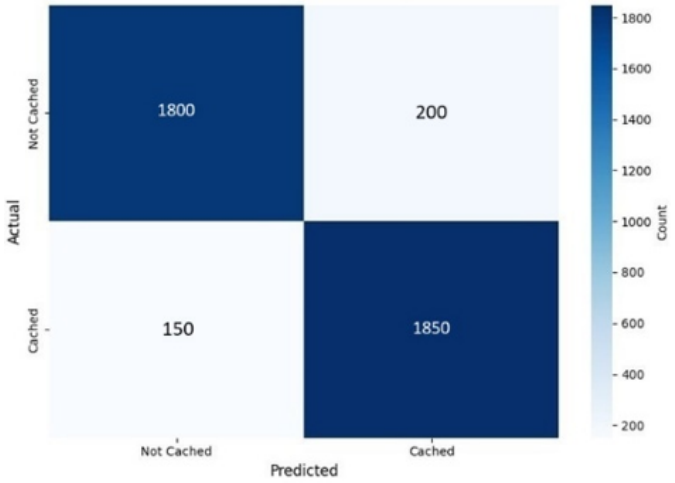

Figure 25. Confusion Matrix for content caching predictions (Realistic Scenario)—T-CCCRP

However, the predictive performance of the proposed T-CCCRP framework surpassed that of the original CCCRP strategy across all evaluated dimensions. In the same large-scale setting of 10 content access events (Figure 24), T-CCCRP correctly predicted 7 actual requests (true positives), incorrectly cached 2 items that were not requested (false positives), and missed 1 item that was requested but not cached (false negative). While the absence of correctly predicted non-requests (true negatives) slightly limited the interpretive depth of this small-scale case, it still demonstrated the model's preference toward over-caching as a protective measure. The computed precision and recall in this setting were approximately 77.8% and 87.5%, respectively, yielding an F1-score of 82.4%. These results, even at such a small scale, suggested a promising balance between aggressive caching and cache efficiency.

Extending the evaluation to a realistic workload of 4,000 content access events (Figure 25) offered a more comprehensive view of the T-CCCRP framework's effectiveness, consistently confirming its outperformance of the CCCRP strategy. In this large-scale scenario, T-CCCRP achieved an overall accuracy of 91.2%. Precision was 90.2%, indicating the model's strong capacity to avoid caching unnecessary content. Recall stood at an impressive 92.5%, reflecting minimal cache misses. These results combined to yield a balanced F1-score of 91.4%, underscoring the Transformer model's consistent effectiveness across prediction dimensions.

The confusion matrix revealed a slightly higher number of false positives than false negatives, suggesting a cautious but deliberate bias toward caching additional content when prediction certainty was marginal. This behavior is beneficial in the context of vehicular caching, where the cost of failing to cache needed content (a cache miss) is often higher than the marginal cost of caching an unrequested item. By favoring availability, the model effectively improves the Quality of Service (QoS) experienced by users.

What emerges from both the small-scale and large-scale analyses is a clear indication that the T-CCCRP framework is not only capable of making highly accurate caching decisions but also does so in a way that scales effectively with data volume. The increase in F1-score from 82.4% in the toy scenario to 91.4% in full-scale testing demonstrates the Transformer model's ability to generalize well and maintain predictive stability under real-world conditions. Its strong recall ensures that user requests are anticipated with minimal latency, while high precision conserves cache space and bandwidth by limiting redundant data retention.

The comparative results indicate that while the original CCCRP strategy is capable of achieving strong baseline performance, particularly in larger deployments, it is inherently limited by its sequential architecture and restricted memory scope. The proposed T-CCCRP framework, in contrast, benefits from its self-attention mechanism, which allows it to effectively model long-range dependencies in content request sequences and process input data in parallel. This architectural advantage not only contributes to superior accuracy but also ensures faster training and inference—key factors in real-time vehicular networks.

6. Conclusion and Future Work

This study proposed a novel Transformer-based Caching Strategy (T-CCCRP) aimed at enhancing content request prediction and caching efficiency within IoV environments. As an improvement over the existing CCCRP strategy, the proposed method replaces traditional LSTM neural networks with a Transformer architecture, thereby addressing key limitations associated with sequential processing, long-term dependency modeling, and computational complexity more effectively. The simulation results for the baseline scenario indicate significant improvements across key performance metrics, including a consistently superior cache hit ratio of around 98%, substantially outperforming conventional methods such as LFU, LRU, and CCCRP. Additionally, T-CCCRP significantly reduces average content acquisition delay by about 25% compared with the LSTM model, while achieving optimal performance at a caching capacity of approximately 20 MB, resulting in improved efficiency and reduced infrastructure costs. To provide a more realistic and comprehensive validation, the evaluation was extended beyond the initial small-scale setup to include multiple large-scale IoV scenarios with up to 200 vehicles, expanded content libraries, and increased temporal input lengths. The experimental results consistently demonstrate that T-CCCRP achieves superior performance across all scales. In particular, the framework maintains a high cache hit ratio—reaching approximately 0.80 in very large-scale environments—while significantly outperforming baseline strategies such as CCCRP, LFU, and LRU. Sensitivity analyses confirmed the T-CCCRP's robustness, maintaining a predictive accuracy

of approximately 91.2% across diverse content popularity scenarios.

Despite these promising results, several important research directions remain open and warrant further investigation. First, the current evaluation lacks extensive statistical validation across multiple runs, and future work should include rigorous robustness analysis using repeated experiments and variance-based metrics. Second, the individual contribution of each component (Transformer predictor and reinforcement learning module) has not been explicitly quantified; thus, ablation studies are needed to better understand their respective impact. Third, validating its scalability in ultra-large-scale environments with extensive content libraries and empirical testing in real-world vehicular environments remains essential for comprehensive assessment. Investigating adaptability to dynamically changing content catalogs, optimizing computational efficiency and energy usage for edge-device deployments, and exploring integration with emerging technologies—such as edge computing, millimeter-wave communication, blockchain-enabled decentralized caching, and 5G/6G networks—are promising future directions. Addressing these aspects will enhance the practical impact, scalability, and sustainability of T-CCCRP, contributing significantly toward the advancement of intelligent and responsive vehicular communication ecosystems.

Article Information Form

Author's Contributions: A. Lamraoui contributed to the drafting of the manuscript and to manuscript revision. L. Guezouli and K. Barka supervised the project, designed the system architecture, led the methodological development and to manuscript revision. M. Z. Mabane and S. Chine contributed to the training and validation of deep learning models, handled dataset creation, simulation testing and the system implementation and integration of the reinforcement learning agent.

All authors have read and approved the final manuscript.

Acknowledgments: The authors would like to thank all individuals who supported this work indirectly. No professional writing services or third-party materials were used.

Conflict of Interest: The author declares that there are no potential conflicts of interest.

Support Section: The author(s) received no financial support for the research, authorship, and/or publication of this article.

Ethical Approval: Not applicable. No human participants or animal subjects were involved in this study.

Artificial Intelligence Statement: No artificial intelligence tools were used while writing this article.

Plagiarism Statement: This article has been scanned by iThenticate™.

References

[1] H. Peng, L. Liang, X. Shen, and G. Y. Li, "Vehicular communications: A network layer perspective," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1064–1078, Feb. 2019.

[2] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, vol. 5, pp. 6757–6779, 2017.

[3] Z. Su, Y. Hui, Q. Xu, T. Yang, J. Liu, and Y. Jia, "An edge caching scheme to distribute content in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 5346–5356, 2018.

[4] L. Yao, A. Chen, J. Deng, J. Wang, and G. Wu, "A cooperative caching scheme based on mobility prediction in vehicular content centric networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 5435–5444, Jun. 2018.

[5] L. Li, Y. Xu, J. Yin, W. Liang, X. Li, W. Chen, and Z. Han, "Deep reinforcement learning approaches for content caching in cache-enabled D2D networks," *IEEE Internet Things J.*, vol. 7, no. 1, pp. 544–557, Jan. 2020.

[6] M. Chen, W. Saad, C. Yin, and M. Debbah, "Echo state networks for proactive caching in cloud-based radio access networks with mobile users," *IEEE Trans. Wireless Commun.*, vol. 16, no. 6, pp. 3520–3535, Jun. 2017.

[7] A. Vaswani et al., "Attention Is All You Need," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, pp. 5998–6008, Dec. 2017.

[8] Z. Zhou, D. Xu, Q. Zhang, and X. Zhang, "State-of-the-art deep learning for multi-source traffic prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 8, pp. 5197–5212, Aug. 2021.

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

[10] G. Ma, Z. Wang, M. Zhang, J. Ye, M. Chen, and W. Zhu, "Understanding performance of edge content caching for mobile video streaming," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 5, pp. 1076–1089, May 2017.

[11] L. Yao, X. Xu, J. Deng, G. Wu, and Z. Li, "A cooperative caching scheme for VCCN with mobility prediction and consistent hashing," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 11, pp. 20230–20242, 2022.

[12] Z. Ding, H. Li, H. Bai, Y. Chen, X. Wang, and P. Lu, "A cooperative caching scheme in MIN-V2X," in *2024 3rd International Joint Conference on Information and Communication Engineering (JCICE)*, 2024, pp. 13–19.

[13] R. Wu, G. Tang, T. Chen, D. Guo, L. Luo, and W. Kang, "A profit-aware coalition game for cooperative content caching at the network edge," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1361–1373, 2022.

[14] S. Fang and P. Fan, "A cooperative caching algorithm for cluster-based vehicular content networks with vehicular caches," in *2017 IEEE*

Globecom Workshops (GC Wkshps), 2017.
[15] Z. Jin, T. Song, W. Jiang, and J. Hu, "A centralized edge cooperative caching strategy for VANETs," in *2024 IEEE Wireless Communications and Networking Conference (WCNC)*, 2024.
[16] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence and implications," in *Proc. IEEE INFOCOM*, vol. 1, Mar. 1999, pp. 126–134.
[17] C. Ling, W. Zhang, Q. Fan, Z. Feng, J. Wang, R. Yadav, and D. Wang, "Cooperative service caching in vehicular edge computing networks based on transportation correlation analysis," *IEEE Internet Things J.*, vol. 11, no. 12, pp. 22754–22767, 2024.
[18] C.-C. Lin, Y. Chiang, and H.-Y. Wei, "Collaborative edge caching with multiple virtual reality service providers using coalition games," in *2023 IEEE WCNC*, 2023, pp. 1–6.
[19] J. Liang, M. Ma, and G. Yang, "Efficient-lightweight CRL distribution in VANETs: A multilayer coded caching methodology," in *2023 IEEE WCNC*, 2023, pp. 1–6.
[20] S. K. uz Zaman et al., "Cooperative content caching framework using cuckoo search optimization in vehicular edge networks," *Appl. Sci. (Basel)*, vol. 13, no. 2, p. 780, 2023.
[21] T. Z. H. Ernest and A. S. Madhukumar, "Ensemble learning-based edge caching strategies for internet of vehicles: Outage and finite SNR analysis," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 239–252, 2023.
[22] L. Yao, Y. Wang, X. Wang, and G. Wu, "Cooperative caching in vehicular content centric network based on social attributes and mobility," *IEEE Trans. Mob. Comput.*, vol. 20, no. 2, pp. 391–402, 2021.
[23] C. Kan, H. Wu, L. Xing, and H. Ma, "Cooperative caching strategy based mobile vehicle social-aware in internet of vehicles," *Trans. Emerg. Telecommun. Technol.*, vol. 34, no. 7, 2023.
[24] S. Tian, S. Zou, Y. Tan, D. Shen, and Y. Li, "Dynamic content cache strategy based on content prediction in the internet of vehicles," in *2023 19th International Conference on Mobility, Sensing and Networking (MSN)*, 2023.
[25] S. A. Elsayed, S. Abdelhamid, and H. S. Hassanein, "Predictive proactive caching in VANETs for social networking," *IEEE Trans. Veh. Technol.*, vol. 71, no. 5, pp. 5298–5313, 2022.
[26] L. Hou, L. Lei, K. Zheng, and X. Wang, "A Q-learning-based proactive caching strategy for non-safety related services in vehicular networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4512–4520, 2019.
[27] R. Wang, Z. Kan, Y. Cui, D. Wu, and Y. Zhen, "Cooperative caching strategy with content request prediction in internet of vehicles," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 8964–8975, 2021.
[28] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 9, pp. 3122–3131, Sep. 2019.
[29] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.
[30] G. Barlacchi et al., "A multi-source dataset of urban life in the city of Milan and the Province of Trentino," *Sci. Data*, vol. 2, Oct. 2015, Art. no. 150055.