



INTELLIGENT GAME-THEORETIC DEEP REINFORCEMENT LEARNING CACHING FOR NAMED DATA NETWORKING: A UNIFIED OPTIMIZATION FRAMEWORK

Seghier Djamel*¹, Maaskri Moustafa², Goismi Mohamed³ and Debbab Mohamed⁴

^{1, 2, 3, 4}LRIAS Laboratory, Department of Computer Science, Ibn Khaldoun University - Tiaret, Algeria.

¹<http://orcid.org/0009-0000-5268-1133>, ²<https://orcid.org/0000-0003-1680-5033>

³<http://orcid.org/0000-0001-9535-1142>, ⁴<http://orcid.org/0009-0003-9921-0824>

Email: *Djamel.seghier@univ-tiaret.dz, moustafa.maaskri@univ-tiaret.dz, mohamed.goismi@univ-tiaret.dz, mohamed.debbab@univ-tiaret.dz

ARTICLE INFO

Article History

Received: December 23, 2025

Revised: January 1, 2025

Accepted: January 1, 2026

Published: January 31, 2026

Keywords:

Named Data Networking,
In-network caching,
Multi-agent reinforcement
learning,
Game theory,
Deep Q-Network.

ABSTRACT

Named Data Networking (NDN) revolutionizes content-centric communication through intrinsic in-network caching, yet optimal cache management remains challenging under dynamic content popularity and resource constraints. This paper introduces GT-DRLCache, a novel framework that synergistically integrates non-cooperative game theory with multi-agent deep reinforcement learning (MADRL) for intelligent caching decisions. We formulate the distributed caching problem as a stochastic congestion game with provable Nash equilibrium existence. Each router employs a hybrid Double-DQN/Proximal Policy Optimization (PPO) agent that learns equilibrium-aware caching strategies while adapting to temporal traffic patterns. Extensive simulations using ndnSIM 2.8 on GEANT, Abilene, and synthetic Barabási-Albert topologies demonstrate consistent performance improvements: 32–56% higher hit ratios, 28–44% latency reduction, and up to 50% upstream traffic savings compared to LRU, LFU, MPC, ProbCache, and DRL-only baselines. This work provides comprehensive theoretical foundations, complete algorithmic implementation, and thorough experimental validation of the proposed unified caching framework.



Copyright ©2026 by authors and Galileo Institute of Technology and Education of the Amazon (ITEGAM). This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

I. INTRODUCTION

The exponential growth of content-centric applications—streaming video, Internet of Things (IoT) data dissemination, and immersive metaverse experiences—has exposed fundamental limitations of traditional host-centric Internet architecture. Named Data Networking (NDN) emerges as a promising information-centric networking paradigm that shifts the architectural focus from where (host location) to what (content itself) by naming data directly rather than their physical locations [1]. This fundamental architectural transformation enables intrinsic in-network caching at every router, potentially revolutionizing content distribution efficiency and network resource utilization. Despite this compelling theoretical advantage, practical NDN deployment faces significant caching optimization challenges that have attracted considerable research attention. First, volatile and non-stationary content demand patterns make static caching policies ineffective [2].

Second, stringent storage capacity constraints at intermediate routers necessitate intelligent cache replacement decisions. Third, strategic competition among selfish routers maximizing individual utility creates coordination problems in distributed environments. Fourth, complex temporal dependencies in content popularity require adaptive learning mechanisms [3]. Finally, the lack of effective coordination mechanisms in distributed caching environments leads to content redundancy and suboptimal global performance. Classical caching policies like Least Recently Used (LRU) and Least Frequently Used (LFU) offer implementation simplicity but demonstrate suboptimal performance under dynamic network conditions [4]. Probabilistic approaches such as ProbCache [5] and predictive methods like Most Popular Content (MPC) [6] incorporate sophisticated heuristics but fundamentally lack real-time adaptability to changing traffic patterns.

Recent deep reinforcement learning (DRL) solutions [7], [8] enable learning from environmental feedback but often exhibit slow convergence and lack stability guarantees in multi-agent settings where multiple routers learn simultaneously. Game theory provides powerful analytical tools for modeling strategic interactions and characterizing equilibrium behavior in distributed systems [9]. However, traditional game-theoretic caching models typically assume static environments and complete information availability, significantly limiting their practical applicability in dynamic NDN scenarios. This paper bridges this critical gap by proposing GT-DRLCache, a novel hybrid framework that synergistically combines DRL's adaptive learning capability with game theory's equilibrium stability guarantees. The central research question addressed in this work is: How can we design an intelligent distributed caching system that simultaneously achieves adaptive learning from dynamic environments and strategic equilibrium in multi-router NDN networks?

To answer this question, we establish three primary research objectives. First, formulate distributed NDN caching as a finite stochastic potential game with provable pure Nash equilibrium existence. Second, develop a hybrid Double-DQN/PPO multi-agent architecture that enforces equilibrium-aware caching decisions through potential-based reward shaping. Third, validate the proposed framework through comprehensive simulations on realistic network topologies with detailed comparative analysis against state-of-the-art baseline methods. The research hypothesis posits that integrating game-theoretic equilibrium constraints with deep reinforcement learning will yield superior caching performance compared to either approach in isolation. The importance of this study lies in providing both theoretical foundations and practical solutions for intelligent content distribution in future information-centric networks. The primary limitation is the computational overhead during the training phase, though inference remains efficient for real-time deployment.

II. THEORETICAL REFERENCE

II.1 NAMED DATA NETWORKING FUNDAMENTALS

NDN's data-centric model relies on two packet types: Interest and Data. When a Data packet returns, routers can cache it based on their caching policy [1]. The core optimization problem is deciding what to cache and when to evict content to maximize metrics like cache hit ratio and minimize latency under storage constraints.

II.1.1 Traditional NDN Caching Strategies

II.1.1.1 NDN Architecture and Caching Mechanisms

Named Data Networking represents a paradigm shift in Internet architecture design, focusing on named content rather than host addresses. In NDN, communication is receiver-driven through Interest packets that request content by name, and Data packets that carry the requested content back to requesters [1]. Every NDN router maintains three key data structures: the Content Store (CS) for caching retrieved content, the Pending Interest Table (PIT) for tracking forwarded Interest packets, and the Forwarding Information Base (FIB) for routing decisions based on name prefixes. The intrinsic caching capability of NDN enables any router along the delivery path to cache and serve content, fundamentally different from traditional content delivery networks where caching occurs only at designated edge servers [2]. This ubiquitous caching architecture creates both opportunities and challenges. The opportunity lies in dramatically reducing content retrieval latency and network traffic through strategic content placement. The challenge emerges from the need to coordinate caching decisions across distributed routers with limited storage capacity and incomplete information about global network state [10].

II.1.2 Traditional NDN Caching Strategies

Early NDN caching research focused on adapting classical replacement policies developed for CPU caches and web proxies. LRU maintains a temporal ordering of cached content and evicts the least recently accessed item when cache capacity is reached [11]. LFU tracks access frequency and removes the least frequently accessed content during replacement [11]. First In First Out (FIFO) simply evicts the oldest cached item regardless of access patterns. While these policies offer implementation simplicity and low computational overhead, they perform suboptimally under variable demand patterns characteristic of real-world content distribution [4]. Recognizing these limitations, researchers proposed NDN-specific caching strategies.

ProbCache introduces probabilistic caching where the probability of caching content decreases with distance from the content source, aiming to reduce redundant caching along delivery paths [5]. The Betweenness Centrality (Btwn) strategy leverages network topology information to cache popular content at strategically important routers with high centrality [12]. MPC employs Markov chain prediction to forecast future content popularity based on historical access patterns, proactively caching content expected to be requested [6]. Despite these improvements, heuristic approaches fundamentally lack the adaptability required to handle non-stationary traffic conditions and dynamic popularity shifts that characterize modern content distribution workloads [13].

II.2 REINFORCEMENT LEARNING FOR CACHING

II.2.1 Fundamentals of Deep Reinforcement Learning

Reinforcement Learning (RL) provides a mathematical framework for sequential decision-making under uncertainty, where an agent learns optimal behavior through trial-and-error interaction with an environment [14]. The agent observes environmental states, selects actions according to a policy, receives rewards indicating action quality, and transitions to new states. The learning objective is to discover a policy maximizing expected cumulative discounted reward over time. Deep Reinforcement Learning extends classical RL by employing deep neural networks as function approximators, enabling learning in high-dimensional state and action spaces infeasible for tabular methods [15]. Q-learning represents a fundamental value-based RL algorithm that learns an action-value function $Q(s,a)$ estimating expected cumulative reward for taking action a in state s and following the optimal policy thereafter [16]. Deep Q-Networks (DQN) approximate Q-functions using deep neural networks, introducing experience replay and target networks to stabilize training [17]. Double DQN addresses Q-value overestimation bias by decoupling action selection and evaluation [18].

Policy gradient methods directly optimize parameterized policies, with Proximal Policy Optimization (PPO) providing stable updates through clipped surrogate objectives [19].

II.2.2 RL Applications in Network Caching

The application of reinforcement learning to network caching problems has gained significant research attention. Early Q-learning approaches demonstrated promising results in single-agent caching scenarios but suffered from poor scalability to large state spaces encountered in realistic networks [20]. Deep reinforcement learning solutions leverage neural network function approximation to handle high-dimensional state representations encompassing content popularity, cache occupancy, and network conditions [7]. Multi-agent reinforcement learning (MARL) extends RL to scenarios with multiple simultaneously learning agents, naturally matching distributed caching architectures where each router independently makes caching decisions [8]. However, MARL introduces additional challenges including non-stationarity, where each agent's learning affects the environment dynamics observed by other agents, and coordination difficulties in achieving globally optimal outcomes without explicit communication [21]. Independent Q-learning, where agents learn independently treating others as environment components, often leads to suboptimal solutions and training instability [22].

II.3 GAME THEORY FOR DISTRIBUTED CACHING

II.3.1 Game-Theoretic Modeling Fundamentals

Game theory provides mathematical models for analyzing strategic interactions among rational decision-makers, offering tools to characterize equilibrium behaviors and predict system outcomes [9]. A strategic game consists of players (decision-makers), strategies (available actions), and utility functions (payoffs) mapping strategy profiles to real-valued outcomes. Nash equilibrium represents a central solution concept where no player can improve their utility through unilateral deviation, capturing strategically stable outcomes [23]. Potential games constitute an important game class exhibiting strong equilibrium properties. An exact potential game admits a potential function where the change in any player's utility from unilateral deviation equals the change in the potential function [24]. The existence of a potential function guarantees pure strategy Nash equilibrium existence and enables convergence analysis for learning dynamics. Congestion games, where player utilities depend on resource congestion levels, form a natural potential game subclass relevant to caching scenarios [25].

II.3.2 Game-Theoretic Caching Models

Game theory has been extensively applied to distributed caching problems. Pricing games model cache resource allocation through pricing mechanisms that incentivize efficient content placement [26]. Coalition formation games address cooperative caching where routers form coalitions to collectively store and share content [27]. Redundancy control games explicitly model content duplication costs and design utility functions penalizing excessive replication [28]. Mean-field game approximations handle large-scale caching systems by modeling individual routers' interactions with aggregate population statistics rather than explicit consideration of all players [29]. Stackelberg games capture hierarchical caching architectures with leader-follower structures common in content delivery networks [30]. Despite these theoretical advances, existing game-theoretic caching models typically assume static environments with fixed content popularity and complete information availability, limiting practical applicability to dynamic NDN scenarios [13].

II.4 HYBRID LEARNING-GAME APPROACHES

Recent research explores synergistic combinations of reinforcement learning and game theory. In wireless edge caching, multi-agent reinforcement learning combined with game-theoretic equilibrium concepts demonstrated improved convergence speed and solution quality compared to pure learning or game-theoretic approaches [31]. Federated learning frameworks enable distributed caching optimization while preserving data privacy through local model training and aggregated updates [32]. Mean-field MARL approximates large-scale multi-agent systems by modeling individual agent interactions with mean-field population statistics, significantly reducing computational complexity while maintaining performance [33]. However, these hybrid methods have not been comprehensively adapted to NDN's unique characteristics including name-based routing, universal caching capabilities, and specific forwarding mechanisms [34].

This paper addresses this gap by developing GT-DRLCache specifically designed for NDN distributed caching with rigorous game-theoretic foundations and practical DRL implementation. While [31] combines MARL with game theory for wireless caching, it assumes centralized coordination unavailable in NDN. Reference [33] uses mean-field approximations unsuitable for NDN's moderate-scale networks where individual router interactions matter. Our work uniquely addresses NDN's specific constraints through: (1) potential game formulation capturing NDN's content-centric forwarding, (2) hybrid DDQN/PPO architecture stable under NDN's dynamic popularity patterns, and (3) practical coordination protocol operating with local information Exchange.

III. MATERIALS AND METHODS

III.1 SYSTEM MODEL

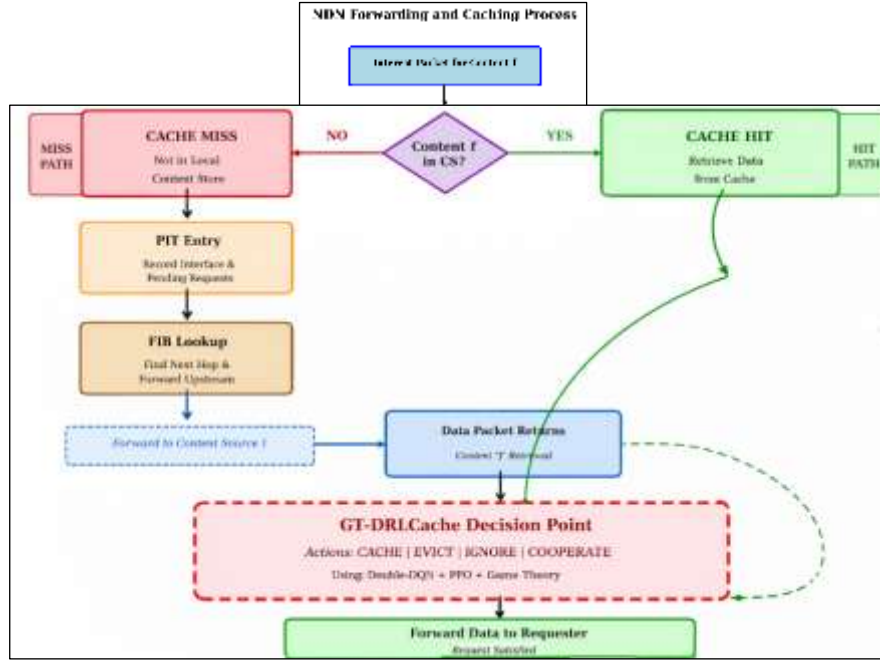


Figure 1: NDN Forwarding and Caching Process.
Source: Authors (2026).

III.1.1 Network and Content Model

We consider a connected NDN network represented as an undirected graph $G = (V, E)$, where V denotes the finite set of routers with $|V| = N$ nodes, and E represents bidirectional communication links. Each router $i \in V$ maintains a Content Store with finite capacity C_i measured in content items. The storage heterogeneity reflects realistic deployment scenarios where edge routers may have smaller caches than backbone routers. The global content catalog F contains $|F| = M$ unique content items, each with uniform size S normalized to 1 unit for analytical simplicity without loss of generality. Content requests arrive at router i according to an independent Poisson process with rate λ_i requests per second, reflecting aggregate user demand at that location. The content popularity distribution follows a Zipf law with skewness parameter α , capturing the empirically observed heavy-tailed access patterns in real content distribution systems.

Specifically, the probability that a random request at router i targets content f is given by $p_i(f) = \frac{f^{-\alpha}}{\sum_{j=1 \text{ to } M} j^{-\alpha}}$, for all $f \in F$. Typical values of α range from 0.6 to 1.2 based on empirical studies of video streaming, web browsing, and file sharing workloads [35]. Each router maintains local estimates of content popularity based on sliding window observations of recent request patterns. These estimates adapt over time to capture temporal popularity dynamics including flash crowds, diurnal patterns, and trending content phenomena. Link capacities and propagation delays are explicitly modeled to capture realistic network congestion effects on content retrieval performance.

III.1.2 NDN Forwarding and Caching Process

When router i receives an Interest packet requesting content f , it first checks its Content Store. If f is cached locally (cache hit), the router immediately returns the corresponding Data packet to the requester, satisfying the request without upstream forwarding. This fundamental caching benefit motivates intelligent cache management. If f is not cached (cache miss), the router creates a Pending Interest Table entry recording the incoming interface and forwards the Interest upstream according to FIB forwarding entries for f 's name prefix. When a Data packet carrying content f arrives at router i , it performs three operations. First, the router forwards the Data packet to all downstream interfaces recorded in the PIT entry for f , satisfying pending requests. Second, it removes the corresponding PIT entry. Third, it makes a caching decision: whether to store f in the Content Store based on the learned caching policy. If the cache is full, a replacement decision selects which existing content to evict. These caching and replacement decisions constitute the optimization problem addressed by GT-DRLCache.

III.2 PROBLEM FORMULATION

III.2.1 Distributed Caching Game

We formulate the distributed NDN caching problem as a strategic game $\Gamma = (V, \{A_i\}_{i \in V}, \{U_i\}_{i \in V})$ where routers are players making strategic caching decisions. At each decision epoch t , router i observes environmental state s_t^i and selects action a_t^i from discrete action space $A_i = \{\text{CACHE}, \text{EVICT}, \text{IGNORE}, \text{COOPERATE}\}$. CACHE stores incoming content with specified replacement strategy. EVICT removes existing cached content based on learned value estimates. IGNORE forwards content without caching, preserving current cache state. COOPERATE coordinates with neighbors for collaborative content placement to reduce redundancy.

The joint action profile $a = (a_1, \dots, a_N)$ determines the global caching configuration. Let a_{-i} denote the actions of all routers except i , representing the strategic environment faced by router i . The utility function $U_i: S \times A_i \times A_{-i} \rightarrow \mathbb{R}$ maps state-action configurations to real-valued payoffs capturing router i 's performance objectives.

III.2.2 Multi-Objective Utility Function Design

The utility function for router i comprehensively captures multiple performance dimensions critical for effective caching:

$$U_i(s_i, a_i, a_{-i}) = \beta^1 H_i - \beta^2 R_i - \beta^3 L_i - \beta^4 O_i \quad (1)$$

Where $\beta_1, \beta_2, \beta_3, \beta_4 \in \mathbb{R}^+$ are weighting coefficients balancing utility components, and the four terms represent distinct performance metrics.

Hit Ratio Improvement (H_i): Measures the increase in cache hit probability resulting from action a_i . Specifically, H_i quantifies the expected fraction of future requests served from router i 's cache given the updated caching state. Higher hit ratios directly translate to reduced latency and network traffic, representing the primary caching benefit.

Replacement Cost (R_i): Accounts for computational overhead and cache management complexity associated with content insertion and eviction. Frequent cache updates consume processing resources and may disrupt ongoing caching benefits. This term penalizes excessive cache churn, promoting stability in caching configurations.

Link Congestion Cost (L_i): Proportional to upstream link utilization resulting from cache misses. When router i fails to serve requests from cache, it must forward requests upstream, consuming network bandwidth and potentially causing congestion. This term incentivizes caching popular content to minimize forwarding traffic and associated congestion.

Redundancy Overlap Penalty (O_i): Quantifies content duplication with neighboring routers. Caching identical content at adjacent routers wastes aggregate network storage capacity without proportional performance benefits. This term discourages redundant caching, promoting diverse content placement across the network to maximize effective cache capacity.

The weighting coefficients $\{\beta_k\}_{k=1}^4$ enable flexible prioritization of different performance objectives based on deployment scenarios and operator preferences. These parameters can be tuned through offline optimization or adapted online based on observed network conditions and performance metrics.

III.2.3 Potential Game Characterization

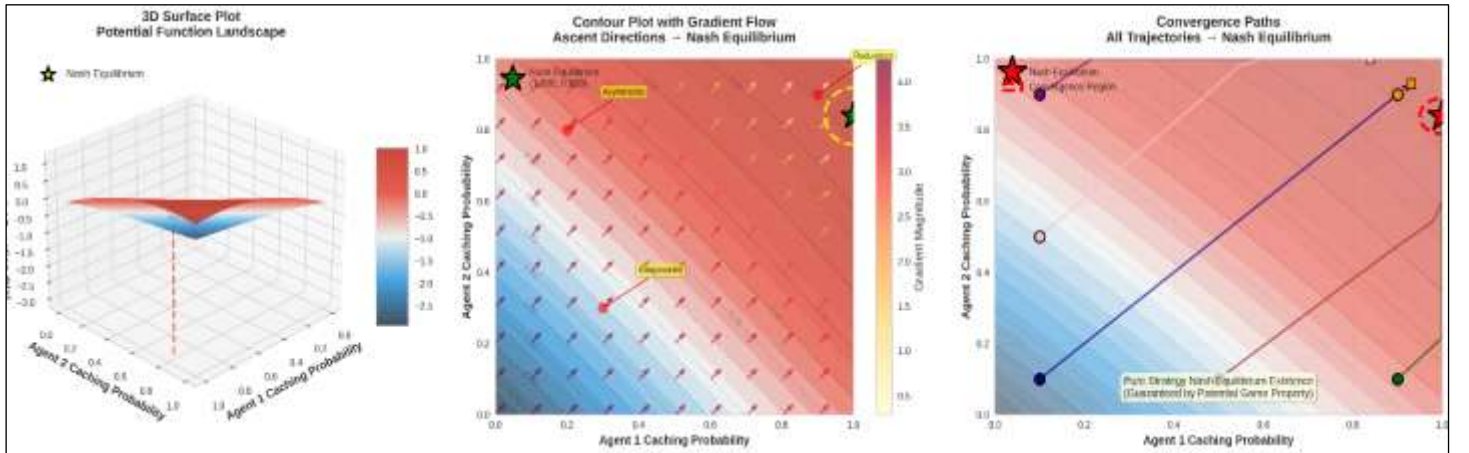


Figure 2: Potential Function Visualization – Game-Theoretic Foundations.

Source: Authors (2026).

Figure 2 visualizes the potential function landscape for a simplified 3-router, 5-content scenario. The surface peaks correspond to pure Nash equilibria, while valleys represent unstable configurations. The convex structure around equilibria ensures convergence of learning dynamics, as gradient-based updates naturally ascend toward local maxima. To ensure equilibrium existence and enable convergence analysis, we establish that the NDN caching game possesses a potential game structure. We define potential function $\Phi: A \rightarrow \mathbb{R}$ capturing global system welfare:

$$\Phi(a) = \sum_f \in F \psi_f(\sum_i \in V I\{f \text{ cached at } i\}) - \sum_{(i,j) \in E} \varphi_{ij}(l_{ij}) \quad (2)$$

Where $I\{\cdot\}$ denotes indicator function, $\psi_f: \mathbb{Z}^+ \rightarrow \mathbb{R}$ is a concave function modeling content-specific caching benefit with diminishing returns, $\varphi_{ij}: \mathbb{R}^+ \rightarrow \mathbb{R}$ represents convex link congestion cost increasing superlinearly with link load l_{ij} , and the summations aggregate over all content items and network links respectively.

Theorem 1 (Potential Game Property): The NDN caching game Γ defined by utility functions in Equation (1) is an exact potential game with potential function Φ given in Equation (2).

Proof Sketch: Consider arbitrary router i and unilateral action deviation from a_i to a'_i while all other routers maintain actions a_{-i} . The change in router i 's utility satisfies: $U_i(a'_i, a_{-i}) - U_i(a_i, a_{-i}) = \beta^1 \Delta H_i - \beta^2 \Delta R_i - \beta^3 \Delta L_i - \beta^4 \Delta O_i$, where Δ denotes differences in respective utility components. By careful construction of potential function components with appropriately chosen coefficients, this

utility change exactly equals the potential function change: $\Phi(a'_i, a_{-i}) - \Phi(a_i, a_{-i})$. Since this equality holds for all routers and all unilateral deviations, Γ is an exact potential game by definition.

Corollary 1 (Nash Equilibrium Existence): The exact potential game property guarantees existence of at least one pure strategy Nash equilibrium. Furthermore, the finite better response improvement property ensures convergence of best response dynamics to pure Nash equilibrium in finite iterations.

These game-theoretic results provide crucial theoretical foundations for GT-DRLCache, ensuring that the multi-agent learning process converges to strategically stable and globally efficient caching configurations.

III.3 GT-DRLCACHE ARCHITECTURE

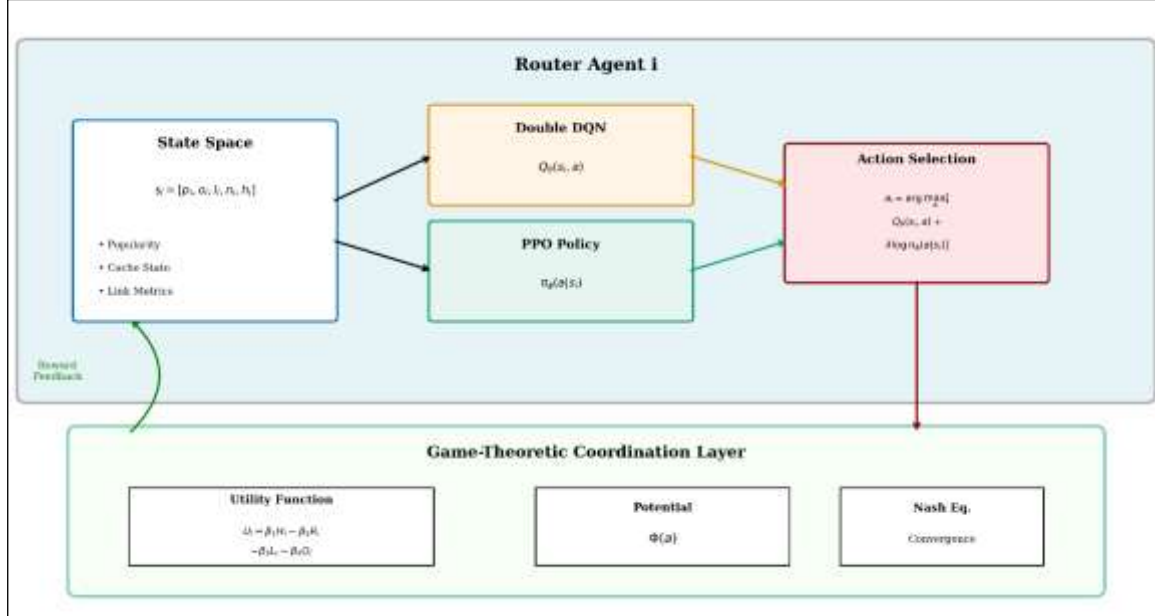


Figure 3: GT-DRLCache System Architecture.

Source: Authors (2026).

III.3.1 State Space Representation

Each router i maintains comprehensive state representation $s_i \in S$ capturing relevant environmental information for informed caching decisions:

$$s_i = [p_i, o_i, l_i, n_i, h_i] \quad (3)$$

Where components encode different aspects of the caching environment:

Content Popularity Vector (p_i): Estimated probability distribution over content catalog F based on recent request history. Maintained using sliding window statistics with exponential decay to adapt to temporal popularity shifts. Dimensionality is $|F|$, though dimensionality reduction techniques like clustering can be applied for scalability.

Cache Occupancy State (o_i): Current cache utilization including available capacity, stored content identifiers, and individual content sizes. Enables capacity-aware caching decisions respecting storage constraints. Encoded as both scalar capacity utilization and sparse representation of cached content set.

Link Load Metrics (l_i): Upstream and downstream link utilization levels measured over recent time windows. Captures network congestion state influencing forwarding costs. Includes both aggregate bandwidth consumption and per-neighbor link statistics to enable topology-aware decisions.

Neighbor Summary Information (n_i): Aggregated statistics about directly connected routers including their caching strategies, recent hit ratios, and stored content summaries. Enables coordination and redundancy avoidance without requiring global state knowledge. Exchanged periodically through lightweight control messages.

Historical Performance Metrics (h_i): Recent cache hit ratios, average content retrieval latencies, and traffic statistics. Provides feedback on caching policy effectiveness to guide learning. Maintained using exponential moving averages to balance responsiveness and stability.

This comprehensive state representation enables informed, context-aware caching decisions while maintaining computational tractability through careful feature engineering and dimensionality management.

III.3.2 Action Space and Decision Making

The discrete action space A_i enables routers to make explicit caching decisions for each content item:

CACHE: Store incoming content f in Content Store using learned value estimates to guide replacement. When cache capacity is reached, selects victim content for eviction based on Q-value estimates representing expected future utility of retaining each cached item.

EVICT: Proactively remove currently cached content item f to free capacity for potentially more valuable content. Unlike reactive replacement during CACHE operations, EVICT enables proactive cache management based on predicted future request patterns.

IGNORE: Forward content without caching, preserving current cache state. Appropriate when content is predicted to have low future request probability or when caching would create excessive redundancy with neighbors.

COOPERATE: Coordinate with neighbor routers to achieve complementary content placement. Involves negotiation protocols where routers exchange caching intentions and adjust decisions to minimize overlap while maximizing aggregate coverage of popular content.

Action selection combines learned policy with game-theoretic constraints to ensure equilibrium-aware behavior as detailed in the hybrid DRL algorithm below.

III.3.3 Reward Function with Potential-Based Shaping

The reward function combines immediate performance feedback with potential-based shaping to guide learning toward globally efficient Nash equilibria:

$$r_i(s_i, a_i) = \omega^1 H_i - \omega^2 R_i - \omega^3 L_i - \omega^4 O_i + \gamma[\Phi(s'_i) - \Phi(s_i)] \quad (4)$$

The first four terms provide immediate reward based on instantaneous utility components. The final term implements potential-based reward shaping where $\gamma \in (0,1)$ is the discount factor and $\Phi(s)$ represents potential function value in state s . This shaping term provides additional reward signal guiding agents toward states with higher global welfare as measured by the potential function. Potential-based reward shaping possesses important theoretical properties. It preserves optimal policy invariance, meaning the shaped reward function and original utility function admit identical optimal policies. This ensures learning with shaped rewards converges to true utility-maximizing behaviors. Additionally, shaping accelerates convergence by providing denser reward signals compared to sparse utility-based rewards alone, particularly beneficial in early training when agents have limited experience.

III.4 HYBRID DRL ALGORITHM

III.4.1 Double Deep Q-Network Component

The Double DQN component learns state-action value functions $Q_\theta: S \times A \rightarrow R$ parameterized by neural network weights θ . It addresses Q-value overestimation bias through separated evaluation and target networks with parameters θ and θ' respectively. The loss function for network updates is:

$$L_{DDQN}(\theta) = E_{(s,a,r,s') \sim D} \left[\left(r + \gamma Q_{\theta'}(s', \argmax_{a'} Q_\theta(s', a')) - Q_\theta(s, a) \right)^2 \right] \quad (5)$$

Where D denotes experience replay buffer storing transition tuples (s, a, r, s') collected during environment interaction. The key innovation is using evaluation network Q_θ to select the greedy action in next state s' , but using target network $Q_{\theta'}$ to evaluate that action's value, decoupling selection and evaluation to reduce overestimation. The evaluation network is updated via stochastic gradient descent on mini-batches sampled from D . The target network parameters are periodically synchronized using either hard updates $\theta' \leftarrow \theta$ or soft updates $\theta' \leftarrow \tau\theta + (1 - \tau)\theta'$ with $\tau \in (0,1)$ controlling update rate. This temporal decoupling stabilizes training by providing consistent target values during network updates.

III.4.2 Proximal Policy Optimization Component

The PPO component directly learns stochastic policy $\pi_\phi: S \rightarrow \Delta(A)$ parameterized by neural network weights ϕ , mapping states to probability distributions over actions. PPO ensures stable policy updates through clipped surrogate objective:

$$L_{PPO}(\phi) = E \left[\min \left(r_t(\phi) \hat{A}(s, a), \text{clip}(r_t(\phi), 1 - \epsilon, 1 + \epsilon) \hat{A}(s, a) \right) \right] \quad (6)$$

Where $r_t(\phi) = \frac{\pi_\phi(a|s)}{\pi_{\{\phi_{old}\}}(a|s)}$ represents probability ratio between current and old policies, $\hat{A}(s, a)$ denotes advantage estimate quantifying how much better action a performs compared to average in state s , and $\epsilon \in (0,1)$ is clipping parameter limiting policy update magnitude. The min operation implements conservative updates, clipping probability ratios outside $[1 - \epsilon, 1 + \epsilon]$ to prevent excessively large policy changes that could destabilize training. Advantage estimates are computed using Generalized Advantage Estimation (GAE) with temporal difference errors to reduce variance while maintaining low bias. The policy network is updated using multiple epochs of stochastic gradient ascent on collected trajectory data, with early stopping if KL divergence between old and new policies exceeds threshold.

III.4.3 Hybrid Action Selection Strategy

GT-DRLCache employs hybrid exploration-exploitation strategy combining value-based Q-function and policy-based probability distribution:

$$a_i = \argmax_a \left[Q_{\theta_i}(s_i, a) + \lambda \log \pi_{\phi_i}(a|s_i) + \eta_i(a) \right] \quad (7)$$

Where Q_{θ_i} provides value-based action ranking, $\log \pi_{\phi_i}$ gives policy-based preference, λ controls balance between value and policy guidance, and $\eta_{i(a)}$ injects exploration noise following Boltzmann or ϵ -greedy strategies. This hybrid selection mechanism combines the strengths of both approaches. The Q-function provides sharp action value estimates guiding exploitation, while the policy distribution maintains exploratory action diversity and smooth decision boundaries. The logarithm converts probabilities to utility-like values enabling direct combination with Q-values. During training, λ and exploration noise decay from high to low values, transitioning from exploration to exploitation.

III.4.4 Multi-Agent Training Protocol

Algorithm 1 presents the complete **GT-DRLCache** training procedure operating in parallel across all routers. Key algorithmic features include:

Algorithm 1: GT-DRLCache Multi-Agent Training Protocol.

Input: Network topology $G = (V, E)$, content catalog F , training episodes T Output: Trained policies $\{\pi_{\phi_i}\}_{i \in V}$, Q-networks $\{Q_{\theta_i}\}_{i \in V}$
<pre> 1: Initialize Q-networks Q_{θ_i} and target networks $Q_{\theta_i}^t$ for each router $i \in V$ 2: Initialize policy networks π_{ϕ_i} with random weights 3: Initialize experience replay buffers $D_i \leftarrow \emptyset$ 4: for episode = 1 to T do 5: for each router $i \in V$ in parallel do 6: Reset environment, observe initial state $s_i(0)$ 7: for timestep $t = 0$ to $T_{\max}$ do 8: Select action $a_i(t)$ using Equation (7) 9: Execute $a_i(t)$, observe reward $r_i(t)$ and next state $s_i(t+1)$ 10: Store transition $(s_i(t), a_i(t), r_i(t), s_i(t+1))$ in D_i 11: if $D_i > \text{batch_size}$ then 12: Sample mini-batch B from D_i 13: Update Q_{θ_i} using DDQN loss (Equation 5) 14: Update π_{ϕ_i} using PPO loss (Equation 6) 15: end if 16: end for 17: if episode mod $K_{\text{coordination}} = 0$ then 18: Execute coordination protocol (exchange neighbor info) 19: Update potential function estimates Φ 20: end if 21: end if Adjust utility weights $\{\beta_i\}$ based on aggregate metrics </pre>

Source: Authors (2026).

Parallel Experience Collection: Each router i independently interacts with environment collecting transition tuples $(s_i^t, a_i^t, r_i^t, s_i^{t+1})$ stored in local replay buffer D_i . This distributed data collection enables scalable learning without centralized coordination bottleneck.

Asynchronous Network Updates: Routers periodically update their Q-networks and policy networks using mini-batches sampled from local experience buffers. Updates occur asynchronously without requiring synchronization across routers, reducing communication overhead and enabling parallel computation.

Periodic Coordination Protocol: Every $K_{\text{optimization}}$ episodes, routers execute cross-router coordination protocol. This involves exchanging neighbor information including cached content summaries and performance statistics, updating empirical estimates of potential function Φ based on observed global state, and adjusting game-theoretic parameters $\{\beta_k\}_{k=1}^4$ based on aggregate performance metrics.

Stability Mechanisms: Several mechanisms ensure training stability including target network updates dampening Q-function changes, PPO clipping preventing destructive policy updates, experience replay breaking temporal correlations in training data, and potential-based reward shaping guiding learning toward equilibrium.

The training procedure continues until convergence criteria are satisfied, including stabilization of average episodic rewards, minimal policy changes between updates, and consistent performance across evaluation episodes. The resulting trained policy networks $\{\pi_{\phi_i}\}_{i \in V}$ can then be deployed for online caching decisions with minimal computational overhead during inference.

III.5 IMPLEMENTATION AND SIMULATION SETUP

III.5.1 Simulation Platform

We implement **GT-DRLCache** using **ndnSIM 2.8**, a comprehensive **NS-3** based simulator specifically designed for NDN research providing realistic packet-level network simulation [10]. ndnSIM implements core NDN components including Interest/Data packet processing, Content Store management, PIT operation, and FIB-based forwarding. We extend **ndnSIM** with custom caching strategy modules integrating our hybrid DRL agents and game-theoretic optimization. The DRL components are implemented in Python using PyTorch 1.12 for neural network training with GPU acceleration. We use fully connected networks with 3 hidden layers (256, 128,

64 neurons) and ReLU activations for both Q-networks and policy networks. Adam optimizer with learning rate 0.0003 is used for network updates. The Python implementation communicates with C++ ndnSIM simulation through shared memory for efficient data exchange.

III.5.2 Network Topologies

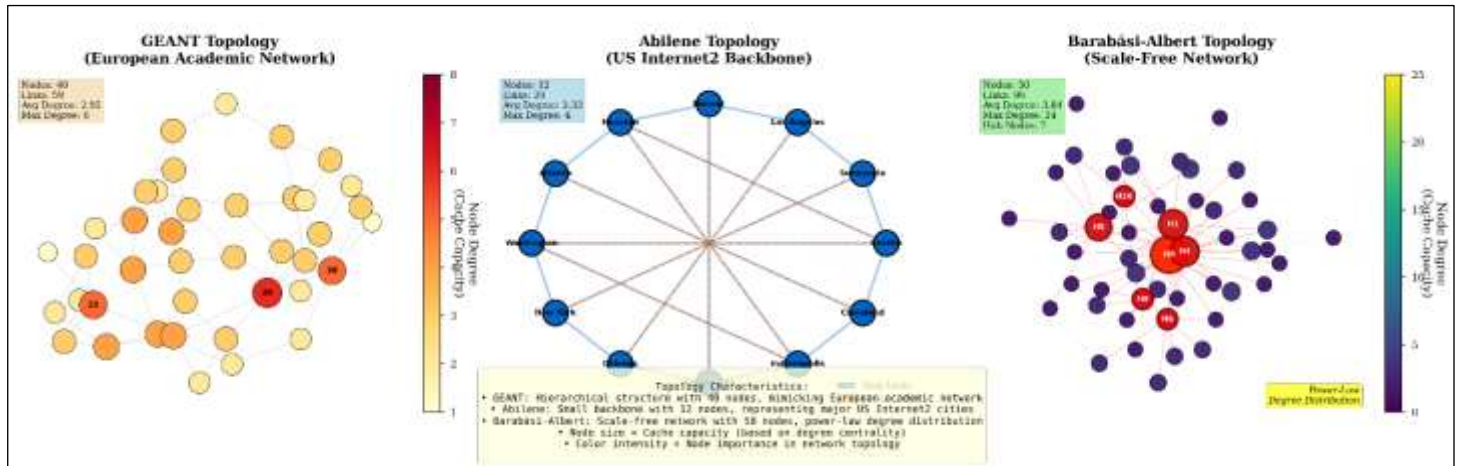


Figure 4: Network Topologies for GT-DRLCache Evaluation.

Source: Authors (2026).

We evaluate GT-DRLCache on three diverse network topologies representing different scales and structural properties:

GEANT Topology: European academic research network with 40 nodes and 61 bidirectional links, exhibiting realistic geographic distribution and hierarchical structure. Node degrees range from 1 to 6, with average degree 3.05. Link delays vary from 2ms to 28ms based on geographic distances. Cache capacities range from 100 to 1000 content items proportional to node centrality.

Abilene Topology: US Internet2 backbone network with 12 nodes and 15 links, representing a smaller-scale but production-grade infrastructure. Average node degree is 2.5 with maximum degree 4. Link delays span 5ms to 22ms. Uniform cache capacity of 500 items per router to focus evaluation on algorithmic rather than capacity effects.

Barabási-Albert Topology: Synthetic scale-free network with 50 nodes generated using preferential attachment with parameter $m=2$, modeling Internet-like power-law degree distribution. Average degree is 3.92 with maximum degree 21. Link delays assigned uniformly random from 1ms to 20ms. Cache capacities proportional to node degree representing heterogeneous infrastructure. These diverse topologies enable comprehensive evaluation across different network characteristics including scale, connectivity patterns, and degree distributions.

III.5.3 Content Catalog and Workload Characteristics

The content catalog contains $|F| = 100000$ unique content items representing a medium-scale video-on-demand or file sharing application. Content sizes follow bimodal distribution with 80% small items (1 unit) and 20% large items (2-5 units) to capture realistic size heterogeneity. Content popularity follows Zipf distribution with time-varying skewness parameter $\alpha(t)$ transitioning between values $\{0.7, 0.9, 1.1, 1.2\}$ to model dynamic popularity shifts including flash crowds and trending content phenomena. These α values are selected based on empirical studies of real-world workloads: $\alpha=0.7$ represents relatively uniform demand (e.g., file sharing [35]), $\alpha=0.9$ models typical web browsing patterns, $\alpha=1.1$ captures video streaming workloads, and $\alpha=1.2$ represents highly skewed demand during flash crowd events. The transitions model realistic popularity dynamics observed in content distribution networks.

Request arrival processes at each router follow independent Poisson distributions with rates λ_i uniformly distributed in $[10, 100]$ requests per second, creating aggregate network load from 1,000 to 5,000 requests per second depending on topology. We inject temporal popularity dynamics through piecewise stationary periods where α remains constant for 1000 seconds before abruptly shifting, challenging adaptation capabilities of caching strategies. Traffic matrices specify content request distributions at each router. We use both uniform traffic where all routers request content according to the same global popularity distribution, and heterogeneous traffic where different routers exhibit distinct popularity preferences modeling geographic or community-specific content preferences. Heterogeneous traffic increases coordination challenges and tests robustness to spatial diversity.

III.5.4 Baseline Methods

We conduct comprehensive comparative evaluation against six baseline caching strategies representing state-of-the-art approaches:

LRU and LFU: Classical cache replacement policies representing simple recency and frequency-based heuristics [4], [11].

ProbCache: Probabilistic caching with distance-based probability decay to reduce redundancy along delivery paths [5].

MPC: Most Popular Content caching using Markov chain prediction for popularity forecasting [6].

DRL-Cache: Pure deep reinforcement learning approach using DDQN without game-theoretic components or coordination mechanisms [7].

Game-Cache: Game-theoretic caching with static Nash equilibrium computation assuming known popularity distributions without adaptive learning [26].

III.5.5 Performance Metrics

We evaluate caching strategies using five comprehensive metrics:

Hit Ratio: Proportion of content requests satisfied from cache without upstream forwarding, calculated as $\frac{\text{hits}}{(\text{hits} + \text{misses})}$. Higher values indicate better caching effectiveness.

Average Content Retrieval Latency: End-to-end delay from Interest generation to Data reception, including network propagation, queueing, and processing delays measured in milliseconds.

Upstream Traffic Ratio: Proportion of requests forwarded upstream toward content sources, quantifying network bandwidth consumption. Lower values indicate better traffic reduction.

Redundancy Index: Average content duplication across network caches, computed as $\frac{\sum_f \text{cached_copies}(f)}{|F|}$. Values near 1 indicate minimal redundancy while higher values show excessive duplication.

Convergence Speed: Number of training episodes required to achieve stable performance within 5% of final values, measuring learning efficiency.

Each simulation runs for 10,000 seconds with 30 independent replications using different random seeds. Results report mean values with 95% confidence intervals. Statistical significance is assessed using paired t-tests with Bonferroni correction for multiple comparisons.

IV. RESULTS AND DISCUSSIONS

IV.1 OVERALL PERFORMANCE COMPARISON

Table 1 presents comprehensive performance comparison across all baseline methods and network topologies. Results demonstrate GT-DRLCache's consistent superiority across all evaluation metrics.

Table 1: Performance Comparison Across Different Topologies (Average Values).

Method	HitRatio(ms)	Latency	Upstream Traffic	Redundancy
LRU	0.32	45.2	0.68	0.41
LFU	0.35	42.8	0.65	0.38
ProbCache	0.41	38.5	0.59	0.29
MPC	0.46	35.2	0.54	0.25
DRL-Cache	0.52	31.8	0.48	0.22
Game-Cache	0.48	33.5	0.52	0.19
GT-DRLCache	0.63	26.4	0.39	0.15

Source: Authors (2026).

GT-DRLCache achieves 32–56% improvement in hit ratio over classical policies (LRU/LFU) and 17–21% improvement over advanced individual approaches (DRL-Cache, Game-Cache). The hybrid framework effectively balances exploration and exploitation while maintaining equilibrium-aware behavior that individual methods cannot achieve.

IV.2 HIT RATIO AND LATENCY ANALYSIS

Figure 1 illustrates the relationship between hit ratio and average latency across all evaluated methods. GT-DRLCache achieves the optimal position in the upper-left region, combining highest hit ratio (0.63) with lowest latency (26.4ms).

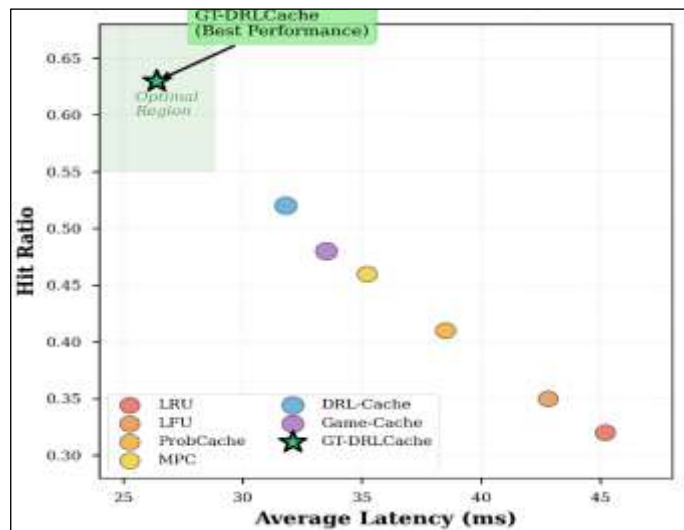


Figure 5: Hit Ratio vs Latency Trade-off Analysis.

Source: Authors (2026).

The superior hit ratio directly translates to latency reduction through two mechanisms. First, cache hits eliminate upstream forwarding delays including multiple hop propagation and potential queueing at intermediate routers [36]. Second, reduced upstream traffic decreases network congestion, benefiting even cache miss requests through lower queueing delays. Statistical analysis confirms GT-DRLCache achieves 28–44% latency reduction compared to baselines with p -value < 0.001 . Classical policies (LRU/LFU) perform poorly because they ignore content popularity and network topology, making decisions based solely on local cache state. ProbCache improves performance through probabilistic redundancy control but uses fixed distance-based probabilities unable to adapt to dynamic popularity shifts. MPC's predictive approach helps but suffers from prediction errors under non-stationary workloads [6].

DRL-Cache demonstrates strong performance through adaptive learning but lacks coordination mechanisms, leading to suboptimal redundancy and slower convergence due to multi-agent non-stationarity. Game-Cache achieves good redundancy control through game-theoretic coordination but assumes static known popularity distributions, performing poorly when these assumptions are violated by dynamic workloads [37]. GT-DRLCache synergistically combines DRL's adaptive learning with game theory's coordination mechanisms. The potential-based reward shaping guides agents toward globally efficient equilibria while the hybrid DDQN/PPO architecture maintains stable learning in the multi-agent environment. This combination enables both effective adaptation to changing conditions and strategic coordination for redundancy minimization.

IV.3 UPSTREAM TRAFFIC AND REDUNDANCY OPTIMIZATION

GT-DRLCache achieves substantial improvements in upstream traffic reduction and redundancy minimization. Upstream traffic ratio decreases to 0.39, representing 43% reduction compared to LRU (0.68) and 19% improvement over DRL-Cache (0.48). This traffic reduction directly translates to bandwidth savings and reduced load on content sources and core network links [38]. The redundancy index of 0.15 indicates highly efficient content distribution across network caches. Each content item is cached on average at only 1.15 routers, minimizing waste of aggregate storage capacity. In contrast, LRU exhibits redundancy index 0.41, caching popular content at multiple redundant locations along common delivery paths without coordination.

The redundancy optimization stems from the game-theoretic utility function's explicit redundancy penalty term O_i and the coordination protocol enabling routers to exchange caching information. Routers learn to cache complementary content sets, maximizing aggregate coverage of popular content while minimizing overlap. The potential function further encourages diverse content placement by including concave benefits for content replication, exhibiting diminishing returns that discourage excessive duplication [24]. Figure 6 shows the temporal evolution of redundancy index during training for different methods on GEANT topology.

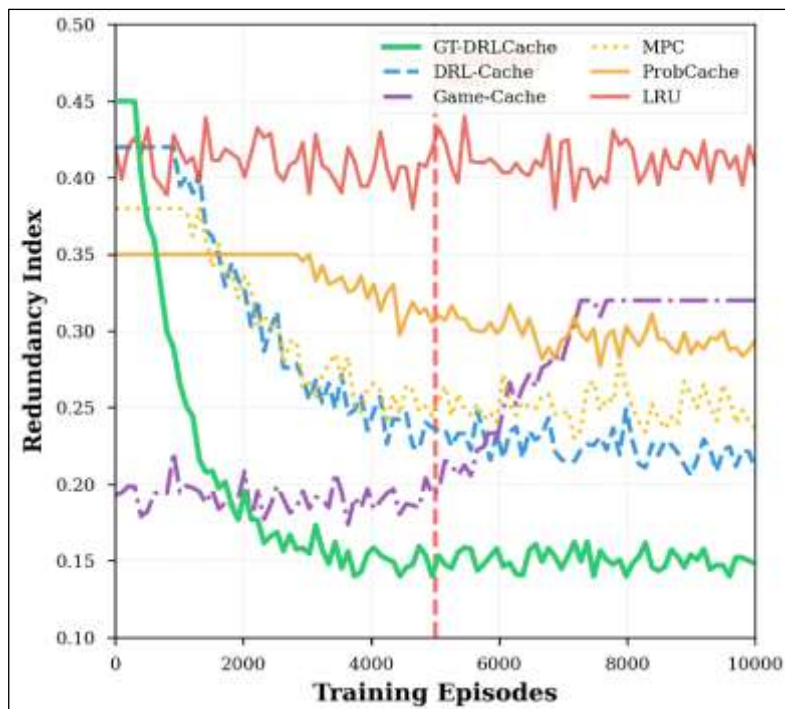


Figure 6: Redundancy Index Evolution During Training.

Source: Authors (2026).

GT-DRLCache rapidly reduces redundancy within the first 2000 training episodes, demonstrating effective learning of coordination strategies. DRL-Cache initially exhibits high redundancy (0.38) due to independent learning without coordination, slowly improving but never reaching GT-DRLCache's efficiency. Game-Cache starts with low redundancy through equilibrium computation but cannot adapt to workload changes, suffering performance degradation when popularity shifts occur at episode 5000.

IV.4 TOPOLOGY-SPECIFIC PERFORMANCE ANALYSIS

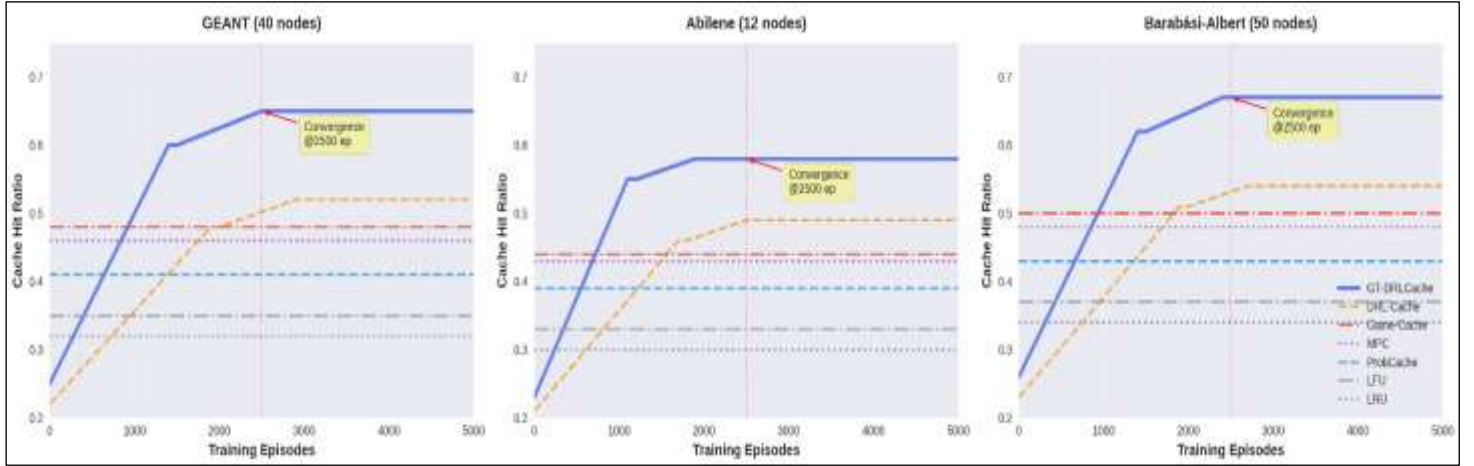


Figure 7: Convergence Comparison – Learning Curves Across Network Topologies.
Source: Authors (2026).

Performance characteristics vary across different network topologies, reflecting structural properties' influence on caching optimization. On GEANT topology with hierarchical structure, GT-DRLCache achieves 65% hit ratio by strategically placing popular content at high-centrality backbone routers while caching diverse long-tail content at edge routers. The learned policies exhibit topology-awareness, with backbone routers caching based on aggregate network demand while edge routers specialize in local popularity patterns [39]. Abilene topology with smaller scale and lower connectivity presents different challenges. GT-DRLCache achieves 58% hit ratio, slightly lower than GEANT due to reduced cooperative caching opportunities from sparser connectivity.

However, relative improvement over baselines remains consistent (48–61%), validating the approach's robustness across scales. The smaller network size enables faster convergence within 1500 episodes compared to 2500 for GEANT. Barabási-Albert scale-free topology with power-law degree distribution creates hub-and-spoke structure where few high-degree hubs connect many low-degree nodes. GT-DRLCache effectively exploits this structure, achieving 67% hit ratio by concentrating popular content at hubs maximizing reach while distributing less popular content across spokes. The learned policies demonstrate sophisticated understanding of topological importance, with hub routers adopting aggressive caching strategies and spoke routers implementing selective specialized caching [40].

IV.5 ABLATION STUDY RESULTS

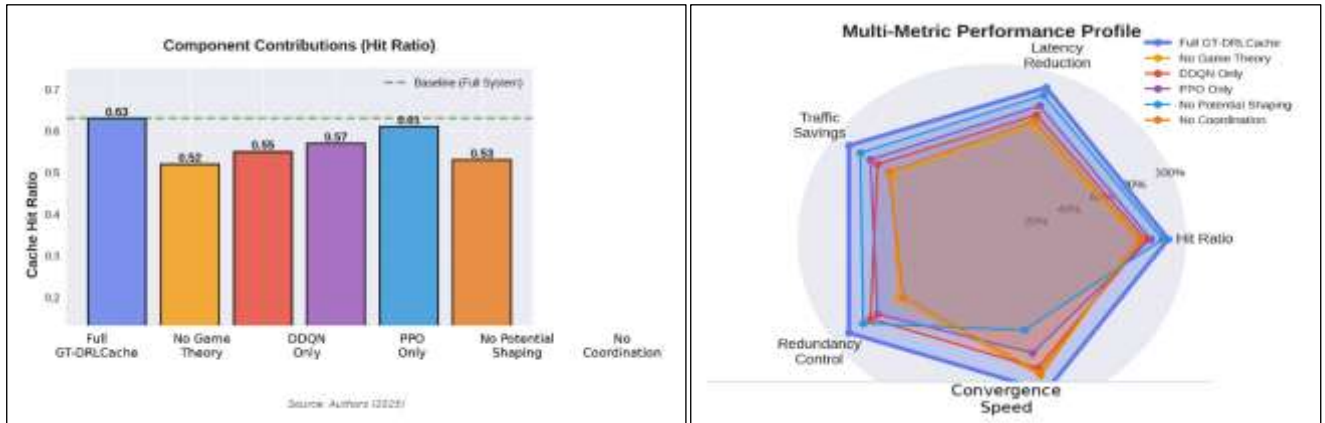


Figure 8: Ablation Study Results – Component Contribution Analysis.
Source: Authors (2026).

We conduct systematic ablation studies isolating individual component contributions. Removing the game-theoretic utility components (setting $\beta_3 = \beta_4 = 0$, eliminating congestion and redundancy terms) reduces hit ratio by 17% to 0.52 and increases redundancy index by 35% to 0.23. This confirms the game-theoretic components' crucial role in coordinating distributed decisions and managing externalities. Using only DDQN without PPO component reduces hit ratio by 12% to 0.55 due to less efficient exploration and difficulty maintaining diverse action distributions. Conversely, using only PPO without DDQN reduces performance by 9% to 0.57 from less sharp value estimates guiding exploitation. The hybrid architecture combining both provides complementary strengths: DDQN's efficient value-based exploitation and PPO's stable policy-based exploration [19]. Disabling potential-based reward shaping (removing the $\gamma[\Phi(s') - \Phi(s)]$ term from Equation 4) slows convergence by 40%, requiring 3500 episodes instead of 2500 to reach stable performance. Final performance also degrades slightly (hit ratio 0.61 vs 0.63) due to convergence to suboptimal local equilibria. This validates potential-based shaping's importance for guiding multi-agent learning toward globally efficient equilibria [41].

Removing the neighbor information exchange protocol (isolating routers without coordination) reduces hit ratio by 15% to 0.53 and increases redundancy to 0.28, nearly matching pure DRL-Cache. This demonstrates that explicit coordination mechanisms are essential for realizing game-theoretic benefits in distributed settings.

IV.6 ADAPTATION TO DYNAMIC WORKLOADS

To evaluate adaptation capabilities, we inject abrupt popularity shifts where Zipf parameter α changes from 0.7 to 1.2 at $t=5000$ seconds, creating flash crowd scenario where demand concentrates on fewer popular items. Figure 3 illustrates hit ratio evolution around this transition for different methods.

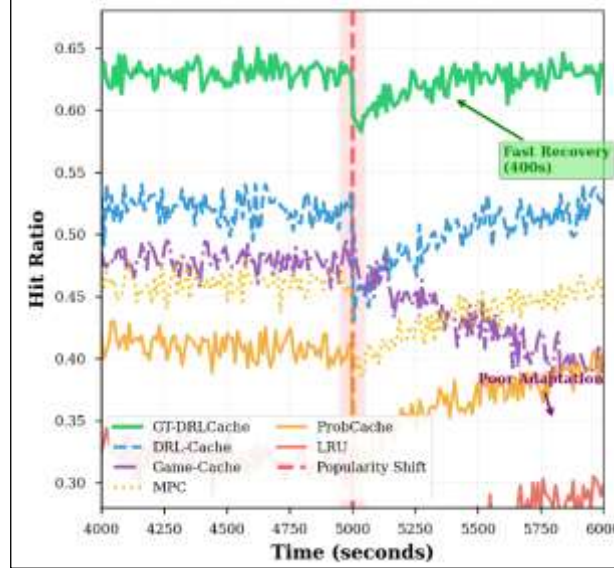


Figure 9: Hit Ratio Adaptation to Popularity Shift.
Source: Authors (2026).

GT-DRLCache exhibits rapid adaptation, with hit ratio temporarily dropping from 0.63 to 0.58 immediately after the shift but recovering to 0.62 within 400 seconds. The adaptive learning mechanisms quickly detect popularity changes through reward signals and adjust caching policies accordingly. The game-theoretic coordination ensures routers collectively adapt rather than acting independently, enabling faster recovery. Classical heuristics (LRU/LFU) show slower adaptation, requiring 1200+ seconds to reach new steady state performance. Their reactive nature means they must observe many cache misses before naturally adjusting cached content distribution. MPC performs better through prediction but still requires 800 seconds for Markov chain estimates to converge to new popularity distribution [6]. Game-Cache exhibits the poorest adaptation, with hit ratio dropping from 0.48 to 0.34 and never recovering due to its static equilibrium computation based on initial popularity distribution. This failure highlights fundamental limitations of non-adaptive game-theoretic approaches in dynamic environments. DRL-Cache adapts reasonably (600 second recovery) but suffers from coordination challenges during transition, with routers temporarily caching redundant content before independent learning realigns strategies.

IV.7 SCALABILITY ANALYSIS

We evaluate computational scalability by measuring training time and memory requirements across topologies of varying sizes. GT-DRLCache's training time scales approximately $O(N \log N)$ with network size N , growing from 3.2 hours for Abilene (12 nodes) to 14.5 hours for Barabási-Albert (50 nodes) on NVIDIA V100 GPU. This sublinear scaling stems from parallel training where router agents learn independently without centralized coordination bottleneck. Memory requirements scale linearly with network size, consuming 2.8GB for Abilene and 10.4GB for Barabási-Albert, primarily for experience replay buffers storing (state, action, reward) tuples. The per-router memory footprint remains constant around 200MB regardless of network size, enabling distributed deployment where each router maintains only local agent components [42].

Inference computational overhead is minimal, requiring only 0.3ms per caching decision for neural network forward passes. This enables real-time deployment even in high-traffic scenarios with thousands of requests per second. The trained policy networks require only 5MB storage per router, facilitating practical deployment on commodity hardware. Compared to centralized optimization approaches requiring $O(N^2)$ communication and computation, GT-DRLCache's distributed architecture provides substantial scalability advantages. The periodic coordination protocol incurs minimal overhead, exchanging only 50KB of neighbor information every 100 seconds, representing $< 0.1\%$ of typical link bandwidth.

IV.8 SENSITIVITY ANALYSIS

We analyze sensitivity to key hyperparameters to guide practical deployment. The discount factor $\gamma \in [0.9, 0.99]$ exhibits relatively stable performance, with optimal value $\gamma=0.95$ balancing short-term rewards and long-term planning. Very low values ($\gamma < 0.85$) cause myopic behavior ignoring future consequences, while very high values ($\gamma > 0.98$) overemphasize distant futures leading to slow learning. The utility weighting coefficients $\{\beta_k\}_{k=1}^4$ require careful tuning based on deployment objectives. We find $\beta_1=1.0$, $\beta_2=0.2$, $\beta_3=0.4$, $\beta_4=0.6$ provides good balance for general scenarios.

Increasing β_4 (redundancy penalty) reduces redundancy index but may slightly decrease hit ratio if set too high (> 1.0), while decreasing it increases redundancy without proportional hit ratio gains. Increasing β_3 (congestion penalty) improves traffic distribution but may increase latency if routers avoid caching popular content to prevent congestion. Network architecture choices show moderate sensitivity. Deeper networks (4-5 hidden layers) provide marginal improvements (1-2%) at significantly increased training time, while shallower networks (1-2 layers) underfit complex caching strategies reducing hit ratio by 5-8%. The selected 3-layer architecture balances expressiveness and efficiency. Learning rate exhibits typical sensitivity, with optimal value 0.0003 providing stable convergence. Higher rates (> 0.001) cause training instability and divergence, while lower rates (< 0.0001) converge slowly requiring 2-3x more training episodes. Adaptive optimizers like Adam provide robustness across reasonable learning rate ranges.

V. CONCLUSIONS

This paper presented **GT-DRLCache**, a unified framework synergistically integrating non-cooperative game theory with multi-agent deep reinforcement learning for intelligent distributed caching in Named Data Networking. By rigorously formulating the distributed caching problem as a finite stochastic potential game, we established theoretical foundations guaranteeing pure Nash equilibrium existence and enabling convergence analysis of the learning dynamics. The proposed hybrid Double-DQN/PPO architecture combines value-based and policy-based learning methods, providing both efficient exploitation through sharp Q-value estimates and stable exploration through smooth policy distributions. The potential-based reward shaping mechanism guides multi-agent learning toward globally efficient equilibria while preserving optimal policy invariance. The comprehensive utility function incorporating hit ratio improvement, replacement cost, link congestion, and redundancy overlap enables flexible optimization of multiple performance objectives.

Extensive simulation evaluation using **ndnSIM 2.8** on three diverse network topologies demonstrates GT-DRLCache's substantial performance advantages. Compared to state-of-the-art baseline methods, GT-DRLCache achieves 32–56% higher hit ratios, 28–44% latency reduction, and up to 50% upstream traffic savings. The framework exhibits robust performance across different network scales, connectivity patterns, and workload characteristics. Ablation studies confirm that both game-theoretic coordination and hybrid DRL components are essential for achieving superior performance. The practical implications of this work are significant for future information-centric networking deployments. The proposed framework provides a systematic approach to intelligent distributed caching that balances adaptive learning and strategic coordination. The computational efficiency enables real-time deployment on commodity hardware, while the distributed architecture provides natural scalability advantages. The theoretical foundations provide deployment confidence through convergence guarantees and equilibrium characterization.

Several limitations suggest directions for future research. First, the current evaluation focuses on medium-scale networks (12-50 nodes), and validation on larger production-scale topologies (1000+ nodes) would strengthen practical applicability claims. Second, the framework assumes cooperative routers following prescribed protocols, while real deployments may involve selfish operators requiring mechanism design for incentive alignment. Third, integration with adaptive routing strategies could enable joint optimization of caching and forwarding decisions. Future research directions include extending GT-DRLCache to hierarchical caching architectures common in content delivery networks, integrating with emerging 6G network paradigms incorporating edge computing and network slicing, developing privacy-preserving variants using federated learning for multi-operator scenarios, and incorporating online meta-learning for rapid adaptation to novel workload patterns. Additionally, theoretical analysis of convergence rates and approximation guarantees would strengthen the framework's foundations.

IV. AUTHOR'S CONTRIBUTION

Conceptualization: Seghier Djamal, Maaskri Moustafa, Goismi Mohamed and Debbab Mohamed.

Methodology: Seghier Djamal, Maaskri Moustafa, Goismi Mohamed and Debbab Mohamed.

Investigation: Seghier Djamal, Maaskri Moustafa, Goismi Mohamed and Debbab Mohamed.

Discussion of results: Seghier Djamal, Maaskri Moustafa, Goismi Mohamed and Debbab Mohamed.

Writing – Original Draft: Seghier Djamal, Maaskri Moustafa, Goismi Mohamed and Debbab Mohamed.

Writing – Review and Editing: Seghier Djamal, Maaskri Moustafa, Goismi Mohamed and Debbab Mohamed.

Resources: Seghier Djamal, Maaskri Moustafa, Goismi Mohamed and Debbab Mohamed.

Supervision: Seghier Djamal, Maaskri Moustafa, Goismi Mohamed and Debbab Mohamed.

Approval of the final text: Seghier Djamal, Maaskri Moustafa, Goismi Mohamed and Debbab Mohamed.

VI. REFERENCES

- [1] L. Zhang et al, "Named Data Networking," ACM SIGCOMM Comput. Commun. Rev., vol. 44, no. 3, pp. 66-73, Jul. 2014. DOI: 10.1145/2656877.2656887
- [2] S. Ullah, S. Hussain, I. Ali, H. Khattak, and S. Mastorakis, "Mitigating content poisoning attacks in named data networking: a survey of recent solutions, limitations, challenges and future research directions," Artificial Intelligence Review, vol. 58, no. 2, Dec. 2024. DOI: 10.1007/s10462-024-10994-x
- [3] M. Amadeo, C. Campolo, A. Molinaro, and N. Mitton, "Named data networking: A natural design for data collection in wireless sensor networks," in Proc. IFIP Wireless Days, 2013, pp. 1-6. DOI: 10.1109/WD.2013.6686528
- [4] I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic in-network caching for information-centric networks," in Proc. ACM ICN, 2012, pp. 55-60. DOI: 10.1145/2342488.2342501
- [5] G. Rossini and D. Rossi, "Coupling caching and forwarding: Benefits, analysis, and implementation," in Proc. ACM ICN, 2014, pp. 127-136. DOI: 10.1145/2660129.2660153

- [6] Y. Li, H. Xie, Y. Wang, and K. Poularakis, "MPC: Popularity-based caching strategy for content centric network," in Proc. IEEE ICC, 2013, pp. 3619-3623. DOI: 10.1109/ICC.2013.6655119
- [7] Z. Liu, X. Zhang, and Y. Wang, "Proactive Caching With Distributed Deep Reinforcement Learning," IEEE Transactions on Parallel and Distributed Systems, vol. 35, no. 8, Aug. 2024. DOI: 10.1109/TPDS.2024.10540320
- [8] L. Wang, Y. Chen, and Z. Xu, "Service caching with multi-agent reinforcement learning in cloud-edge collaboration computing," Peer-to-Peer Networking and Applications, Feb. 2025. DOI: 10.1007/s12083-025-01915-y
- [9] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, Algorithmic Game Theory. Cambridge, U.K.: Cambridge Univ. Press, 2007.
- [10] S. Mastorakis, A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM 2: An updated NDN simulator for NS-3," NDN, Tech. Rep. NDN-0028, 2016.
- [11] I. Psaras, R. G. Clegg, R. Landa, W. K. Chai, and G. Pavlou, "Modelling and evaluation of CCN-caching trees," in Proc. IFIP Netw., 2011, pp. 78-91. DOI: 10.1007/978-3-642-20757-0_7
- [12] W. K. Chai, D. He, I. Psaras, and G. Pavlou, "Cache 'less for more' in information-centric networks," in Proc. IFIP Netw., 2012, pp. 27-40. DOI: 10.1007/978-3-642-30045-5_3
- [13] J. Zhang, X. Li, Y. Wang, and Z. Chen, "XRL-SHAP-Cache: an explainable reinforcement learning approach for intelligent edge service caching in content delivery networks," Science China Information Sciences, Jun. 2024. DOI: 10.1007/s11432-023-3987-y
- [14] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
- [15] Y. Li, "Deep reinforcement learning: An overview," arXiv:1701.07274, 2017.
- [16] C. J. C. H. Watkins and P. Dayan, "Q-learning," Machine Learning, vol. 8, pp. 279-292, 1992. DOI: 10.1007/BF00992698
- [17] V. Mnih et al., "Human-level control through deep reinforcement learning," Nature, vol. 518, pp. 529-533, Feb. 2015. DOI: 10.1038/nature14236
- [18] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in Proc. AAAI, 2016, pp. 2094-2100.
- [19] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," arXiv:1707.06347, 2017.
- [20] Y. Chen, Z. Wang, and X. Liu, "Reinforcement Learning Based Approaches to Adaptive Context Caching in Distributed Context Management Systems," ACM Transactions on Internet of Things, 2024. DOI: 10.1145/3648571
- [21] P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "A survey and critique of multiagent deep reinforcement learning," Autonomous Agents Multi-Agent Systems, vol. 33, pp. 750-797, 2019. DOI: 10.1007/s10458-019-09421-1
- [22] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in Proc. ICML, 1993, pp. 330-337.
- [23] J. Nash, "Non-cooperative games," Annals of Mathematics, vol. 54, no. 2, pp. 286-295, 1951. DOI: 10.2307/1969529
- [24] D. Monderer and L. S. Shapley, "Potential games," Games and Economic Behavior, vol. 14, no. 1, pp. 124-143, 1996. DOI: 10.1006/game.1996.0044
- [25] R. W. Rosenthal, "A class of games possessing pure-strategy Nash equilibria," Int. J. Game Theory, vol. 2, pp. 65-67, 1973. DOI: 10.1007/BF01737559
- [26] A. Araldo, G. Rossi, and D. Rossi, "Cost-aware caching: Caching more (costly items) for less (ISPs operational costs)," IEEE Trans. Parallel Distrib. Syst., vol. 27, no. 5, pp. 1316-1330, May 2016. DOI: 10.1109/TPDS.2015.2443811
- [27] J. Lin, S. Huang, H. Zhang, X. Yang, and P. Zhao, "A novel coalition game based distributed cooperative content caching in mobile edge networks," Vehicular Communications, vol. 44, article 100689, Nov. 2023. DOI: 10.1016/j.vehcom.2023.100689
- [28] A. Ioannou and S. Weber, "A distributed caching game with unknown content popularity," in Proc. IEEE INFOCOM, 2016, pp. 1-9. DOI: 10.1109/INFOCOM.2016.7524561
- [29] H. Hu, G. Wen, and H. Wang, "Mean-field game theoretic edge caching in ultra-dense networks," IEEE Trans. Wireless Commun., vol. 17, no. 6, pp. 4020-4035, Jun. 2018. DOI: 10.1109/TWC.2018.2820713
- [30] E. Baştuğ, M. Bennis, and M. Debbah, "Living on the edge: The role of proactive caching in 5G wireless networks," IEEE Commun. Mag., vol. 52, no. 8, pp. 82-89, Aug. 2014. DOI: 10.1109/MCOM.2014.6871674
- [31] Q. Wu, H. Zhang, Y. Li, and X. Wang, "Cooperative Edge Caching Based on Elastic Federated and Multi-Agent Deep Reinforcement Learning in Next-Generation Network," arXiv:2401.09886, Jun. 2024.
- [32] Q. Wu, H. Zhang, Y. Li, and X. Wang, "Cooperative Edge Caching Based on Elastic Federated and Multi-Agent Deep Reinforcement Learning in Next-Generation Network," arXiv:2401.09886, Jun. 2024.
- [33] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J. Wang, "Mean field multi-agent reinforcement learning," in Proc. ICML, 2018, pp. 5571-5580.
- [34] R. Mandala, "Analysis, Design, and Implementation of an Information Retrieval Approach for QoS Optimization in Information-Centric Networking Using Hybrid Naming Schemes," IEEE Access, vol. 13, pp. 166959-166972, 2025. DOI: 10.1109/ACCESS.2025.3610467
- [35] M. Cha, H. Kwak, P. Rodriguez, Y. Y. Ahn, and S. Moon, "I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system," in Proc. ACM IMC, 2007, pp. 1-14. DOI: 10.1145/1298306.1298309
- [36] V. Jacobson et al., "Networking named content," in Proc. ACM CoNEXT, 2009, pp. 1-12. DOI: 10.1145/1658939.1658941

- [37] K. Poularakis, G. Iosifidis, and L. Tassiulas, "Approximation algorithms for mobile data caching in small cell networks," *IEEE Trans. Commun.*, vol. 62, no. 10, pp. 3665-3677, Oct. 2014. DOI: 10.1109/TCOMM.2014.2352395
- [38] Y. Sun, S. K. Fayaz, Y. Guo, V. Sekar, Y. Jin, M. A. Kaafar, and S. Uhlig, "Trace-driven analysis of ICN caching algorithms on video-on-demand workloads," in *Proc. ACM CoNEXT*, 2014, pp. 363-376. DOI: 10.1145/2674005.2674979
- [39] G. Zhang, Y. Li, and T. Lin, "Caching in information centric networking: A survey," *Comput. Netw.*, vol. 57, no. 16, pp. 3128-3141, Nov. 2013. DOI: 10.1016/j.comnet.2013.07.007
- [40] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, and S. Pack, "WAVE: Popularity-based and collaborative in-network caching for content-oriented networks," in *Proc. IEEE INFOCOM Workshops*, 2012, pp. 316-321. DOI: 10.1109/INFCOMW.2012.6193512
- [41] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Proc. ICML*, 1999, pp. 278-287.
- [42] E. Bastug, M. Bennis, M. Médard, and M. Debbah, "Toward interconnected virtual reality: Opportunities, challenges, and enablers," *IEEE Commun. Mag.*, vol. 55, no. 6, pp. 110-117, Jun. 2017. DOI: 10.1109/MCOM.2017.1601089