Tutorial on Large Language Model-Enhanced Reinforcement Learning for Wireless Networks

Lingyi Cai, Wenjie Fu, Yuxi Huang, Ruichen Zhang, *Member, IEEE*, Yinqiu Liu, Jiawen Kang, *Senior Member, IEEE*, Zehui Xiong, *Senior Member, IEEE*, Tao Jiang, *Fellow, IEEE*, Xianbin Wang, *Fellow, IEEE*, Shiwen Mao, *Fellow, IEEE*, and Xuemin Shen, *Fellow, IEEE*

***Abstract*—Reinforcement Learning (RL) has shown remarkable success in enabling adaptive and data-driven optimization for various applications in wireless networks. However, classical RL suffers from limitations in generalization, learning feedback, interpretability, and sample efficiency in dynamic wireless environments. Large Language Models (LLMs) have emerged as a transformative Artificial Intelligence (AI) paradigm with exceptional capabilities in knowledge generalization, contextual reasoning, and interactive generation, which have demonstrated strong potential to enhance classical RL. This paper serves as a comprehensive tutorial on LLM-enhanced RL for wireless networks. We propose a taxonomy to categorize the roles of LLMs into four critical functions: state perceiver, reward designer, decision-maker, and generator. Then, we review existing studies exploring how each role of LLMs enhances different stages of the RL pipeline. Moreover, we provide a series of case studies to illustrate how to design and apply LLM-enhanced RL in low-altitude economy networking, vehicular networks, and space–air–ground integrated networks. Finally, we conclude with a discussion on potential future directions for LLM-enhanced RL and offer insights into its future development in wireless networks.**

***Index Terms*—Reinforcement learning, large language models, LLM-enhanced RL, decision optimization, wireless networks.**

Received 3 December 2025; revised 30 April 2026 and 1 June 2026; accepted 5 July 2026. Date of publication 13 July 2026; date of current version 22 July 2026. This work was supported by Mobile Information Networks–National Science and Technology Major Project of China under Grant 2025ZD1304800. *(Corresponding author: Tao Jiang.)*

Lingyi Cai, Wenjie Fu, Yuxi Huang, and Tao Jiang are with the Research Center of 6G Mobile Communications, School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: lingyicai@hust.edu.cn; wjfu99@outlook.com; huangyuxi@hust.edu.cn; tao.jiang@ieee.org).

Ruichen Zhang and Yinqiu Liu are with the College of Computing and Data Science, Nanyang Technological University, Singapore 639798 (e-mail: ruichen.zhang@ntu.edu.sg; yinqiu001@e.ntu.edu.sg).

Jiawen Kang is with the School of Automation, Guangdong University of Technology, Guangzhou 510006, China (e-mail: kavinkang@gdut.edu.cn).

Zehui Xiong is with the School of Electronics, Electrical Engineering and Computer Science (EEECS), Queen's University Belfast, BT7 1NN Belfast, U.K. (e-mail: z.xiong@qub.ac.uk).

Xianbin Wang is with the Department of Electrical and Computer Engineering, Western University, London, ON N6A 5B9, Canada (e-mail: xianbin.wang@uwo.ca).

Shiwen Mao is with the Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849 USA (e-mail: smao@ieee.org).

Xuemin Shen is with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: sshen@uwaterloo.ca).

Digital Object Identifier 10.1109/COMST.2026.3712862

I. Introduction

A. Background

LARGE Language Models (LLMs) have marked a pivotal breakthrough in Artificial Intelligence (AI), significantly reshaping the way machines understand and generate human language [1], [2], [3]. Different from earlier natural language processing models that relied on limited rule-based or shallow learning architectures, LLMs are built upon deep neural networks with hundreds of billions of parameters and trained on vast and diverse corpora of text data [4]. Thus, LLMs possess the ability to understand diverse inputs, follow complex instructions, and generate contextually appropriate responses across a wide range of tasks and modalities [1], [2].

The transformative potential of LLMs has extended to a wide range of applications from conversational agents and code generation tools to knowledge engines and automated reasoning systems. For example, OpenAI's GPT series,[1] Google's Gemini,[2] and Meta's LLaMA[3] models have demonstrated state-of-the-art performance across benchmarks such as question answering, summarization, and reasoning tasks. Moreover, GPT-4 can integrate multi-frame visual input with textual reasoning to enable interpretable and end-to-end driving systems [1]. Beyond language-centric tasks, LLMs have also begun to play a central role in embodied AI. DeepMind's Gemini Robotics[4] equips humanoid robots with LLM-guided control capabilities, enabling them to perform complex real-world tasks such as folding origami and organizing objects. These models differ in terms of scale, adaptation methods, and capabilities, and their evolution has fueled the rapid development of the LLM ecosystem. The work in [5] demonstrated that performance scales with model size, dataset size, and compute according to power-law relationships. By 2025, models like Grok-3 reflect a leap of over two orders of magnitude in compute compared to GPT-3, signaling the intensifying scale of LLM development.

Overall, the widespread adoption of LLMs is underpinned by several core advantages. They are able to recognize

[1]https://platform.openai.com/docs/models

[2]https://deepmind.google/models/

[3]https://www.llama.com/models/llama-4/

[4]https://deepmind.google/discover/blog/gemini-robotics-brings-ai-into-the-physical-world/?utm_source=chatgpt.com

structure across a wide range of inputs and adapt their responses to novel or evolving situations, where strength lies not only in recognizing patterns but in modeling intent, evolving with context, and supporting coherent decision-making [6]. Their ability to model context, incorporate prior knowledge, and refine behavior through iterative feedback makes them particularly well-suited for dynamic problem-solving [7]. Furthermore, their capacity to simulate, organize, and synthesize complex relationships allows them to accelerate learning and coordination across diverse tasks [8]. These capabilities collectively position LLMs as a foundation and engine in increasingly dynamic and intelligent systems.

B. Motivations

Reinforcement learning (RL) has demonstrated considerable promise in the operation of wireless networks by enabling data-driven and adaptive solutions to complex decision-making problems [9], [10], [11]. It has been successfully applied to a variety of core network optimization tasks, including dynamic spectrum access, power control, resource scheduling, and UAV trajectory planning [12], [13], [14]. These applications often involve high-dimensional state spaces, time-varying dynamics, and conflicting performance objectives, in which RL demonstrates its strength by learning optimal policies through interaction with the wireless environment [9], [15], [16]. However, as wireless systems evolve toward increasing complexity, scale, and heterogeneity, classical RL methods face four growing limitations: (i) generalization and multi-modal understanding, (ii) feedback bottlenecks in learning signals, (iii) decision instability and lack of interpretability, and (iv) sample inefficiency and inadaptability, which are discussed in detail later in Section II-B.

In the context of the recent rise of LLMs, new possibilities have emerged to address the abovementioned challenges of RL in wireless systems. Thanks to the remarkable abilities (as detailed in Section III-A) on knowledge generalization, contextual reasoning, and interactive generation [6], [8], [17], LLMs have evolved from text generators to universal cognitive engines that can model, simulate, explain, and guide complex decision processes [18], [19], [20]. The success in domains such as robotics[1] and human–AI interaction[5] suggests that LLMs could offer opportunities to complement the weaknesses of classical RL. Therefore, the integration of LLMs into RL presents a compelling paradigm shift. By leveraging the strengths of LLMs, this integration can offer a high-level semantic interface that enables the agent to incorporate task semantics, contextual understanding, and domain-specific priors into the learning process, thereby shaping learning beyond raw experience [21], [22], [23]. Thus, in addition to relying on learning through environment interaction, RL agents can be enhanced in terms of understanding task intent, simulating plausible outcomes, explaining actions, and accelerating learning [8], [24], [25]. The enhancement is particularly promising for wireless systems, where expert knowledge is hard to encode, real-world feedback is expensive, and decision-making is often difficult due to trade-offs in network optimization such as balancing latency, energy consumption, and throughput [9], [10], [26].

To this end, this tutorial adopts a structured viewpoint and proposes a taxonomy of LLM-enhanced RL frameworks tailored for wireless networks. We classify the roles of LLMs into four critical functions: state perceiver, reward designer, decision-maker, and generator, to analyze how each role enhances different stages of RL. This perspective provides a unified framework for understanding, designing, and deploying the LLM-enhanced RL framework, which integrates prior knowledge, contextual reasoning, and generative capabilities to support intelligent learning and decision-making to meet the complexity and diversity of future wireless networks.

C. Related Work

In recent years, many surveys and tutorials have emerged to capture the evolution of RL techniques in wireless networks from single-agent DRL to multi-agent DRL applied to communication systems and intelligent networks. By enabling autonomous agents to learn optimal control strategies through interaction with the environment, RL provides a powerful framework for addressing the complexity and heterogeneity of modern wireless networks.

The work in [26] provides a comprehensive survey of DRL applications in communications and networking. It outlines the MDP formulation underlying DRL and emphasizes its advantages in handling large-scale state–action spaces through adaptive decision-making and distributed learning. The paper further summarizes DRL's progress across key network functions such as dynamic spectrum access, rate control, caching, offloading, routing, and security, offering an integrated view of DRL's role in intelligent communication systems. The subsequent survey [27] adopted a layering view, which categorized RL applications across the physical, medium access control, network, and application layers. It shows how DRL models such as Q-learning and Actor–Critic architectures enable cross-layer control while emphasizing the need for unified decision-making frameworks. The authors in [28] provide a comprehensive taxonomy of major DRL algorithms and review their applications in Internet of Things (IoT) domains such as smart grid, intelligent transportation, and mobile crowd-sensing. Moreover, the study systematically reveals how DRL can address decision-making and performance optimization challenges in communication, computing, caching, and control without relying on prior system models. Subsequently, the study in [9] unified single-agent and multi-agent RL perspectives in the tutorial for AI-enabled wireless networks. They review both model-free and model-based DRL frameworks and further emphasize multi-agent reinforcement learning (MARL) as a key enabler for decentralized and cooperative control. This paper categorizes MARL approaches into learned cooperation, emergent communication, and networked agents, and reviews their applications in mobile edge computing, UAV networks, and cell-free massive MIMO. Furthermore, the work in [29] provides an in-depth investigation of MARL and its applications in the future Internet. It systematically classifies MARL algorithms into value decomposition, policy gradient and actor–critic categories, and examines their

[5]https://github.com/Significant-Gravitas/AutoGPT

mechanisms for cooperation, competition, and communication among agents. The paper highlights MARL's capability to address dynamic, decentralized, and large-scale decision-making problems, offering a comprehensive perspective on how MARL can enable autonomous and adaptive control in next-generation network systems. The authors in [10] explore RL from the more specific perspective of autonomous multi-UAV wireless networks. The paper provides a comprehensive overview of how RL techniques enable UAVs to make intelligent and adaptive decisions in dynamic wireless environments. It categorizes existing approaches into model-free, model-based, and multi-agent RL frameworks, and highlights RL's advantages in enhancing the autonomy, scalability, and energy efficiency of UAV networks.

The existing related works demonstrate that RL has achieved remarkable success in enabling intelligent and adaptive control for complex wireless networks. These efforts have also advanced the field from single-agent decision-making to cooperative multi-agent optimization, greatly improving resource utilization and network autonomy in wireless systems. However, in essence, the above research still focuses on the paradigm of classical RL, which relies on handcrafted state representations and handcrafted rewards with trial-and-error learning within task-specific environments [10], [30]. As the next generation of wireless networks evolves toward more dynamic, heterogeneous, and human-centric architectures, the classical RL approaches may face fundamental limitations in scalability, generalization, and adaptation [9], [31], [32].

Encouragingly, the recent emergence of LLMs offers new opportunities with capabilities of semantic perception, contextual reasoning, and generative world modeling [24], [33]. These abilities align naturally with the unmet needs of classical RL in wireless networks, such as understanding complex network semantics [6], designing adaptive states and rewards [34], and interpreting RL agents' behaviors and policies [35]. By bridging the powerful capabilities of LLMs with RL, our work advances beyond traditional RL surveys on wireless networks and offers a forward-looking tutorial on how LLMs can enhance classical RL for wireless networks. Specifically, this work differs from existing surveys and tutorials in several key aspects. First, while prior works primarily focus on classical RL paradigms such as DRL, MARL, and their applications in wireless systems, our tutorial explicitly investigates the integration of LLMs into RL, which introduces new capabilities beyond conventional learning paradigms. Second, instead of merely reviewing RL algorithms or applications, we propose a unified taxonomy that characterizes the functional roles of LLMs in enhancing RL (i.e., state perceiver, reward designer, decision-maker, and generator), thereby providing a systematic perspective and opening a new research direction at the intersection of RL and wireless network optimization. Finally, we provide representative case studies on low-altitude economy networking, vehicular networks, and space–air–ground integrated networks, demonstrating how to design LLM-enhanced RL frameworks and offering practical guidelines beyond the theoretical overviews in prior works.

D. Contributions

The main contributions of our tutorial are summarized as follows:

- We propose a unified framework that functions with LLMs as state perceiver, reward designer, decision-maker, and generator to enhance different stages of the RL pipeline and review existing efforts on LLM-enhanced RL in wireless networks.
- We provide several case studies to demonstrate the practical applications and effectiveness of LLM-enhanced RL in future wireless network scenarios, including low-altitude economy networking, vehicular networks, and space–air–ground integrated networks.
- We discuss potential future directions and open challenges for LLM-enhanced RL, which aim to guide the evolution of LLM-enhanced RL toward more adaptive, scalable, secure, interpretable, and intelligent wireless systems.

As shown in Fig. 1, the rest of this tutorial is organized as follows. In Section II, we present the fundamental background of RL and analyze its limitations in dynamic wireless environments. Section III introduces the motivation and key principles of integrating LLMs into RL. Then, a taxonomy categorizes the roles of LLMs into four critical roles and reviews existing studies on how to enhance RL via LLM in each role in applications of wireless networks. Sections IV to VI provide several case studies to illustrate how LLM-enhanced RL can be implemented in low-altitude economy networks, vehicular networks, and space–air–ground integrated networks. Finally, Section VII discusses open challenges and future research directions, and Section VIII concludes this tutorial.

II. Preliminaries

A. Background of RL in Wireless Networks

1) Fundamentals of RL: As shown in Fig. 2, RL is a type of machine learning where an agent learns to make decisions by interacting with an environment to maximize long-term rewards. In wireless networks, the RL agent typically represents a network controller (e.g., base station, UAV, or edge server), while the environment corresponds to the dynamic wireless system characterized by time-varying channels, traffic demands, and user mobility [9]. RL can be formalized using the framework of MDPs, which consist of the following components [36]:

(a) S: the set of all states. The state may include channel conidtions, queue lengths, interference levels, user locations, or spectrum occupancy.
(b) A: the set of all actions, which can be discrete or continuous. Typical actions include power control, channel allocation, routing decisions, or UAV trajectory control.
(c) P: the state transition probability function, where $P(s'|s,a)$ represents the probability of transitioning to state s' from state s after taking action a. In wireless environments, this transition is influenced by stochastic factors such as fading, mobility, and traffic dynamics.
(d) R: the reward function, where $R(s,a)$ gives the immediate reward received after taking action a in state s.

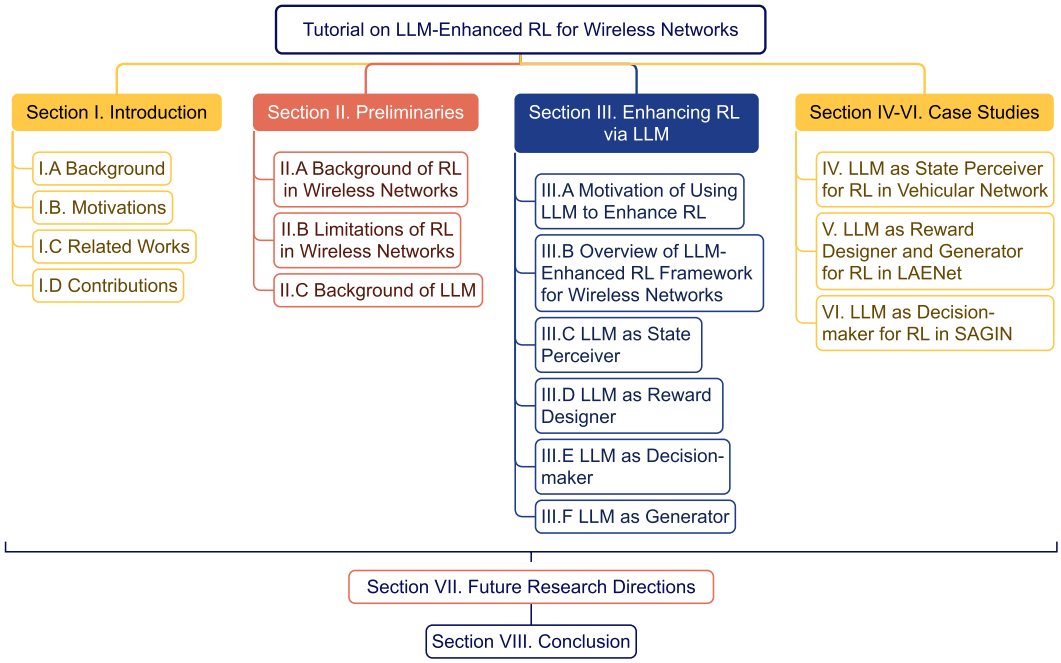

Fig. 1. Overall structure of the tutorial on LLM-enhanced RL for wireless networks. The tutorial is organized into eight sections, beginning with the introduction and preliminaries, followed by a taxonomy of LLM roles in enhancing RL, including the functions of state perceiver, reward designer, decision-maker, and generator. Subsequent sections present case studies in representative network scenarios, while the final sections discuss future research directions and conclude the tutorial.

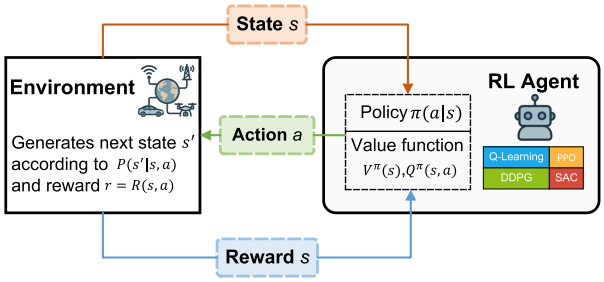

Fig. 2. The classical RL framework in wireless systems, where the RL agent interacts with the environment by observing the state s, selecting an action a according to the policy $\pi(a|s)$, and receiving a reward $r = R(s,a)$. The environment then transitions to the next state s' according to $P(s'|s,a)$, enabling the agent to optimize its value function $V^\pi(s)$ or $Q^\pi(s,a)$ through iterative learning.

Rewards are often designed to capture performance metrics in optimization problems such as throughput, latency, energy efficiency, or QoS.

(e) $\gamma \in [0,1]$: the discount factor, which determines the importance of future rewards.

The agent's goal is to learn a policy $\pi(a|s)$ that maps states to actions [37]. Such a policy can be deterministic (i.e., $\pi : S \to A$) or stochastic (i.e., $\pi : S \to p(A|S)$). The agent interacts with the environment in discrete time steps. At each time step t, the agent observes the current state s_t, selects an action a_t according to its policy $\pi(a_t|s_t)$, receives a reward $r_t = R(s_t, a_t)$, and transitions to the next state s_{t+1} based on the transition probability $P(s_{t+1}|s_t, a_t)$. The interaction corresponds to sequential decision-making under dynamic and uncertain conditions, where actions (e.g., resource allocation or scheduling) directly affect long-term network performance.

The objective of the agent is to maximize the expected cumulative discounted reward, also known as the return:

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}. \tag{1}$$

To measure the long-term value of states and actions, RL uses value functions [38]. The state-value function $V^\pi(s)$ represents the expected return when starting from state s and following policy π thereafter:

$$V^\pi(s) = \mathbb{E}_\pi[G_t | s_t = s]. \tag{2}$$

The action-value function $Q^\pi(s,a)$ represents the expected return when starting from state s, taking action a, and following policy π thereafter:

$$Q^\pi(s,a) = \mathbb{E}_\pi[G_t | s_t = s, a_t = a]. \tag{3}$$

2) Classic RL Algorithms: RL algorithms can be broadly classified into three main families based on what they learn and optimize:

Value-Based Methods: Value-Based Methods focus on learning an optimal value function, from which a policy is implicitly derived. A canonical example is Q-Learning, a model-free, off-policy algorithm that learns the optimal action-value function, $Q^*(s,a)$, by iteratively updating its estimates based on the Bellman optimality equation [39]. The policy is then typically formed by selecting the action with the highest Q-value in a given state. In wireless networks, such methods are commonly applied to discrete decision problems such as channel selection, spectrum access, and user association [27].

The advent of deep learning led to the Deep Q-Network (DQN), which employs a neural network to approximate the Q-function, making it applicable to high-dimensional state spaces [40]. DQN incorporates techniques like experience replay and target networks to stabilize training.

Policy-Based Methods: Policy-Based Methods directly learn a parameterized policy, $\pi_\theta(a|s)$, without needing an intermediate value function. These methods optimize the policy parameters θ by performing gradient ascent on the expected return. They are particularly suitable for continuous control problems in wireless systems, such as power allocation and UAV trajectory optimization [10]. However, policy gradient estimates can suffer from high variance, which can make training unstable.

Actor-Critic Methods: Actor-Critic Methods combine the strengths of the previous two approaches by maintaining two separate models: an actor and a critic. The actor, which represents the policy, selects actions, while the critic, which represents the value function, evaluates these actions. The critic's feedback provides a low-variance learning signal to guide the updates of the actor's policy, leading to more stable and efficient learning. This architecture has become the dominant paradigm for DRL-based optimization in wireless networks. This hybrid architecture forms the foundation for many state-of-the-art Deep Reinforcement Learning (DRL) algorithms, including:

(a) Deep Deterministic Policy Gradient (DDPG) [41]: An off-policy actor-critic algorithm designed for continuous action spaces. It is suitable for wireless network scenarios involving continuous control [14], [20], [42], such as transmit power allocation, beamforming optimization, and UAV trajectory control.

(b) Proximal Policy Optimization (PPO) [43]: An on-policy algorithm known for its stability and ease of implementation. It prevents large, destructive policy updates by using a clipped surrogate objective function, making it a robust and widely used choice [36]. In wireless networks, PPO has been widely adopted in tasks such as multi-user resource allocation and dynamic network slicing [25], where stable and reliable policy learning is essential under highly dynamic environments [42].

(c) Soft Actor-Critic (SAC) [44]: An off-policy actor-critic algorithm that incorporates an entropy maximization term into its objective. This encourages exploration and often leads to more robust and sample-efficient learning in continuous action spaces [12]. Such properties make SAC particularly effective for wireless optimization problems with stochastic dynamics [32], [45], such as joint power and spectrum allocation.

B. Limitations of RL in Wireless Networks

RL has shown great potential in enabling intelligent decision-making and optimization across various aspects of wireless network operations. However, the practical application of RL in dynamic wireless environments still faces several critical limitations as follows.

- **Generalization and Multimodal Understanding**: In wireless network scenarios, RL faces challenges related to limited generalization capability and difficulties in multimodal understanding. Traditional RL methods typically rely on numerical state representations derived from specific environments [9], making it difficult to transfer state observations and action policies to dynamic and heterogeneous network scenarios, such as Open Radio Access Network (O-RAN) slicing [46], UAV networks [47], and edge computing [28]. Moreover, classical RL agents are usually designed to process single-modality inputs from environments, such as state vectors [42]. Emerging network systems increasingly involve multimodal inputs (e.g., natural language instructions, image data, and log texts) [48], which further increases the cognitive burden on RL agents and limits their scalability in practical applications.
- **Feedback Bottlenecks in Learning Signals**: It is challenging to design effective reward functions (i.e., the most important learning signal feedback) for RL to optimize multiple objectives in wireless networks, such as throughput, energy consumption, latency, and quality of service [30]. Improper reward design without appropriate trade-offs among these objectives may introduce bias to the agent toward over-optimizing a single metric (e.g., maximizing throughput at the cost of excessive delay), leading to slow convergence and suboptimal policies. Moreover, the RL agents obtain reward feedback after long operation periods due to the highly dynamic nature of wireless environments and the wide state space, leading to sparse and delayed reward signals that further exacerbate the instability of the policy learning process [30].
- **Decision Instability and Lack of Interpretability**: On the one hand, model-based RL methods rely on accurate environment modeling. However, wireless environments are highly dynamic and uncertain due to factors such as unpredictable user mobility and time-varying channel conditions [10], [31]. In such cases, model errors can easily accumulate, leading to failures in policy learning such as resource allocation errors or service interruptions. On the other hand, the decision-making process of traditional DRL often operates as a closed-box mechanism, lacking transparency and interpretability to provide clear causal reasoning or decision rationales [49]. These limitations may prevent systems from passing regulatory audits or gaining user trust, thereby posing significant risks in high-stakes scenarios such as 6G network slicing, UAV scheduling, and Vehicle-to-Infrastructure (V2I) communications.
- **Sample Inefficiency and Inadaptability**: On the one hand, classical RL relies on interactions with the environment to collect extensive training data [32]. However, in real-world wireless scenarios, each interaction may incur significant costs (e.g., energy consumption and time delays) or pose potential risks such as communication failures, leading to inefficiency and high deployment costs in training processes. On the other hand, wireless networks may involve optimization requirements across

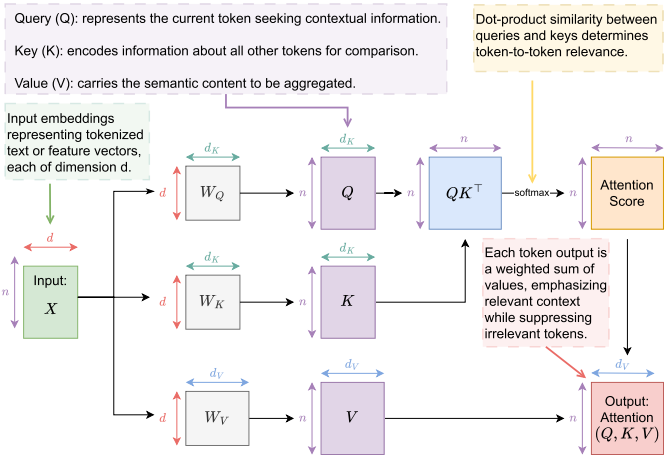

Fig. 3. Self-attention mechanism in Transformer architecture. The input sequence is first projected into query, key, and value representations through learned linear mappings. Dot-product similarity between queries and keys determines token-to-token relevance, which is normalized via softmax to produce attention weights. These weights form a weighted aggregation of value vectors, enabling each token to integrate contextual information from all others and produce a meaning-aware output representation.

different tasks, such as channel estimation and computing offloading [32], [42]. Traditional RL is usually task-specific and lacks abilities for knowledge transfer, making it difficult to share experience or formulate policy models across tasks [50]. As a result, even small changes in tasks (e.g., resource allocation or QoS assurance) often necessitate retraining from scratch, which limits their scalability and flexibility in complex and heterogeneous wireless systems.

The challenges discussed above highlight the limitations of classical RL when applied to wireless networks. Fortunately, recent advances in integrating LLMs with RL offer promising directions for addressing these challenges. In the following section, we will explore how the LLMs are used to enhance the RL for wireless networks.

C. Background of LLM

Generally, language models are trained to predict the next word in a given text sequence, which enables them to learn complex patterns and relationships in language. To further enhance their capabilities in understanding, generating, and reasoning over natural language, LLMs are developed by training advanced neural network architectures with billions of parameters on massive corpus datasets [51]. LLMs are typically based on the Transformer architecture, which utilizes self-attention mechanisms to capture long-range dependencies and contextual information in text data [52].

Mathematically, as shown in Fig. 3, given an input sequence represented by a matrix of word embeddings $X \in \mathbb{R}^{n\times d}$, where n is the sequence length and d is the dimensionality of the embeddings, the self-attention mechanism first projects X into three matrices: the query Q, key K, and value V:

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V, \tag{4}$$

where $W_Q, W_K \in \mathbb{R}^{d\times d_K}, W_V \in \mathbb{R}^{d\times d_V}$ are learnable weight matrices, d_K is the dimensionality of the queries and keys, and d_V is the dimensionality of the values.

The attention scores are then computed as the scaled dot-product between the queries and keys:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_K}}\right)V, \tag{5}$$

where the softmax function $\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$ normalizes the scores to obtain a probability distribution over the values. This formulation ensures that the model computes the relevance of each word in the sequence with respect to every other word, scaled by the factor $\sqrt{d_K}$ to prevent overly large dot product values that could lead to numerical instability. Stacked with multiple layers of self-attention and other components like feed-forward networks, normalization, and residual connections, the Transformer architecture allows LLMs to effectively capture complex relationships in language data.

GPT (Generative Pre-trained Transformer) models, which advance autoregressive pretraining, enable fluent text generation through next-token prediction [53]. Specifically, as shown in Fig. 4, GPT adopts a unidirectional attention mechanism, which makes it particularly suitable for general-purpose text generation and sequential decision-making tasks. BERT (Bidirectional Encoder Representations from Transformers) models, which introduce bidirectional attention layers, enable deep contextual understanding by predicting masked tokens in a sequence [54]. Later, models like RoBERTa [55] and T5 [56] refined training efficiency and task transferability. In contrast to GPT, LLaMA models focus on improving parameter efficiency and deployment flexibility. As shown in Fig. 4, LLaMA adopts an optimized decoder-only Transformer architecture with techniques such as RMSNorm and Rotary Position Embedding (RoPE), which enhance training stability and reduce computational overhead. LLaMA [57] demonstrated that carefully curated data and architectural optimizations can achieve strong performance with fewer parameters, where the first version of LLaMA (65B parameters) achieves most state-of-the-art results than contemporary GPT-3 (175B parameters) in multiple tasks including natural question answering, reading comprehension, and code generation. These advancements have led to the development of LLMs that can understand, generate, and reason over natural language with remarkable fluency and accuracy not only in research but also further in practical applications and cross domain tasks including reinforcement learning [24].

III. Enhancing RL via LLM

A. Motivation of Using LLM to Enhance RL

We outline four key capabilities of LLMs for supporting RL in complex wireless network environments evolution of LLMs has been driven by key innovations in model design, training objectives, and scaling strategies.

- **Generalization and Multimodal Comprehension**: LLMs are pretrained on vast and diverse datasets, enabling them to understand and process a wide range of input modalities, such as textual commands, spatial

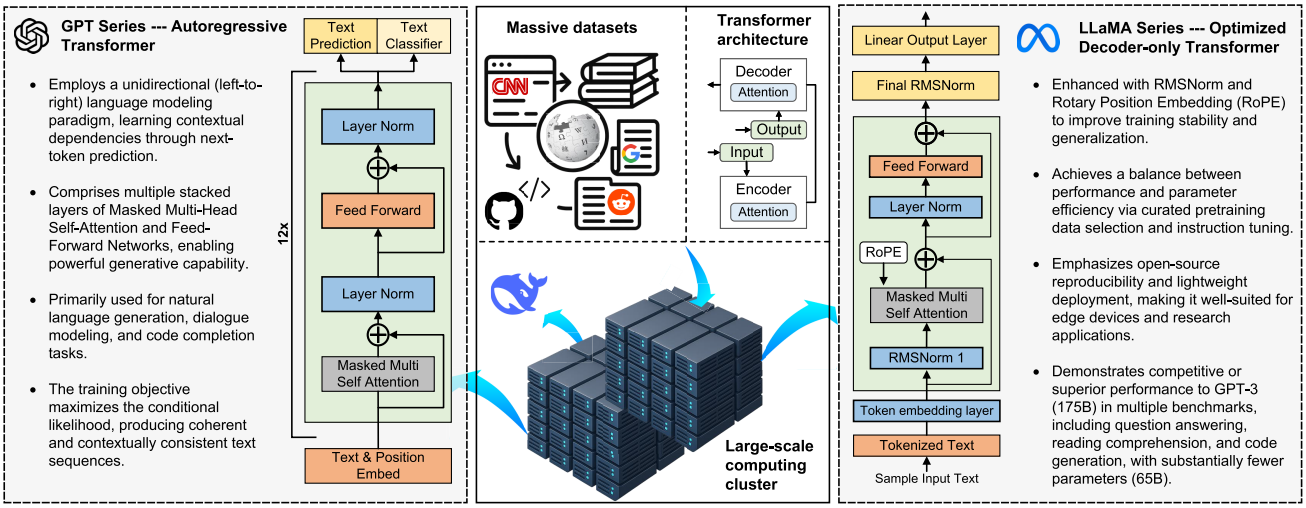

Fig. 4. Overview of representative LLM architectures and their training paradigms. The GPT series (left) adopts an autoregressive transformer with a unidirectional attention mechanism for next-token prediction, excelling in text generation and dialogue modeling. The central panel illustrates the large-scale pretraining pipeline, where massive multimodal datasets are processed through the transformer architecture. The LLaMA series (right) introduces an optimized decoder-only transformer enhanced with RMSNorm and Rotary Position Embedding (RoPE), achieving improved training stability, parameter efficiency, and adaptability for lightweight or edge-oriented deployment.

layouts, visual inputs, and structured data [33]. This capability of LLMs can assist RL agents to better interpret complex wireless environments and generalize across different tasks and domains. For example, LLMs can be used to parse high-level service requirements expressed in natural language [58], allowing the RL agent to better align decisions with user intent and wireless system constraints. Additionally, LLMs can assist agents in better interpreting various elements of wireless networks [59], such as traffic logs or sensor signals, thereby enhancing the agent's ability to make decisions in subsequent stages.

- **Context-Aware and Reward Shaping**: LLMs possess strong capabilities in semantic understanding, contextual modeling, and domain knowledge integration [6]. Thus, they can serve as intelligent reward designers in place of manually crafted or static reward functions traditionally used in RL based on expert knowledge. On the one hand, LLMs can generate reward functions based on optimization objectives [60], system constraints, and environmental states, thereby effectively balancing multiple goals in wireless optimization such as throughput, energy consumption, latency, and QoS. On the other hand, LLMs can evolve reward functions by quickly adapting semantic understanding and abstraction [21]. As a result, RL agents can leverage adaptive reward shaping in complex and dynamic wireless network environments to guide policy learning, enabling more flexible responses to changing objectives and tasks in wireless networks.
- **Transparent Reasoning and Stable Decision-Making**: By employing prompt strategies such as Chain-of-Thought (CoT), LLMs can guide RL agents to generate step-by-step reasoning chains to enable more interpretable, logically coherent, and causally grounded decisions [34]. This structured reasoning process improves the explainability of RL policies [61]. For example, translating complex decisions such as resource allocation into natural language explanations and supporting the generation of interpretable fault-recovery strategies that comply with domain constraints. Moreover, LLMs exhibit strong capabilities in cross-domain knowledge transfer and memory of past experiences, which helps reduce the possibility of random or suboptimal behaviors [62]. Such approach facilitates the sharing of key information across different tasks and domains, allowing RL agents to maintain policy stability even when faced with new environments or changing conditions in wireless networks.
- **Sample Generation and Cross-task Adaptation**: Empowered by extensive pre-trained knowledge and strong generalization capabilities, LLMs can generate high-quality synthetic samples through simulation, in-context learning, or prompt engineering [35], [63]. These generated samples (e.g., traffic patterns, network topologies, and signal types) help reduce dependence on costly real-world wireless environment interactions, thereby improving RL training efficiency and accelerating policy learning. Additionally, LLMs possess strong generalization and knowledge transfer capabilities, which can complement RL in handling cross-task scenarios. By leveraging their understanding of domain-specific concepts, LLMs can facilitate knowledge sharing across related tasks and assist RL agents in adapting to varying objectives [50], [64]. For example, an LLM can transfer knowledge of resource allocation gained from a channel estimation task to the optimization of computation offloading strategies by identifying commonalities between them (e.g., bandwidth usage and energy constraints), thereby supporting more efficient cross-task optimization of communication and computation resources.

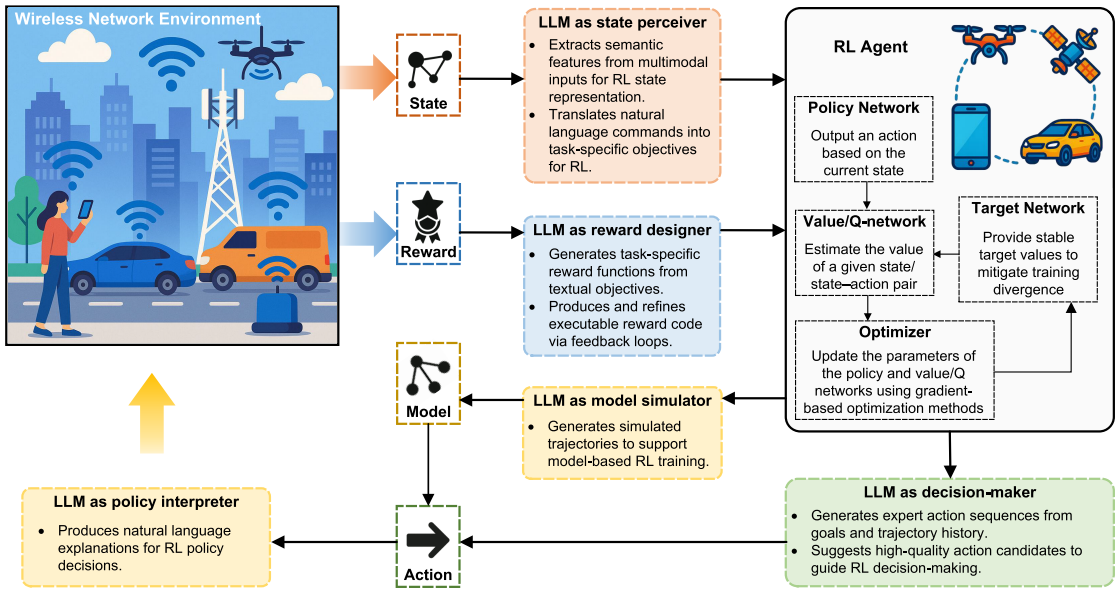

Fig. 5. The overview and architecture of LLMs in enhancing RL for wireless network environments. LLMs assist as state perceivers, reward designers, generators (including model simulators and policy interpreters), and decision-makers. Key capabilities and functions of each role are illustrated across various network scenarios including MEC, vehicular networks, SAGIN, and UAV networks.

B. Overview of LLM-Enhanced RL Framework for Wireless Networks

To integrate LLMs into RL for wireless network optimization, we present a unified LLM-enhanced RL framework, as illustrated in Fig. 5. At a high level, the framework follows the standard RL interaction loop, where an agent interacts with the wireless environment through state observation, action execution, and reward feedback. Different from conventional RL, LLMs are integrated as functional modules, namely state perceiver, reward designer, decision-maker, and generator (including model simulator and policy interpreter), that enhance different components of this loop.

1) LLM as State Perceiver: The LLM acts as a semantic interface between the wireless environment and the RL agent by transforming observations into structured representations. Specifically, the LLM extracts high-level features from multimodal inputs and maps them into task-relevant state representations. In addition, the LLM can translate natural language instructions into formal optimization objectives of wireless tasks.

2) LLM as Reward Designer: The LLM can automatically generate and refine task-specific reward formulations based on high-level objectives and system constraints. Moreover, the LLM can adjust reward structures through iterative feedback and refinement to enable more flexible objective optimization in wireless networks.

3) LLM as Decision-Maker: The LLM can formulate or suggest decision-making by generating high-quality action sequences for the RL agent based on optimization objectives, historical trajectories, and contextual knowledge, thereby guiding efficient exploration and convergence in the action space.

4) LLM as Generator (Model Simulator and Policy Interpreter): The generator role of the LLM enhances both the data and interpretability aspects of RL. As a model simulator, the LLM can generate synthetic trajectories or approximate environment dynamics, which can be used to support model-based RL or offline pre-training. As a policy interpreter, the LLM translates learned policies into natural language explanations to provide insights into decision rationales.

To further clarify the overall workflow of the LLM-enhanced RL framework, we summarize its unified pipeline in Algorithm 1. The algorithm explicitly illustrates the data flow across different stages, including offline LLM-assisted state and reward design, RL-based policy learning, and optional LLM modules for decision guidance, data generation, and policy interpretation. Guided by this unified framework, the subsequent subsections elaborate on each LLM role in detail, providing representative methodologies and practical design guidelines for LLM-enhanced RL[6] in wireless networks.

C. LLM as State Perceiver

LLMs trained on large-scale text corpora are able to internalize syntactic, semantic, and factual knowledge through context-aware representation learning, which enables them to process, interpret, and transform complex information structures effectively [2]. Specifically, the key role of LLMs as a state perceiver lies in task-relevant state abstraction. Specifically, LLMs can filter redundant information, extract meaningful features, and align observations with optimization objectives, thereby constructing more informative state representations for RL [61], [69]. In wireless networks, where observations are often heterogeneous, noisy, and partially observable, such enriched states are particularly important and directly involved in the RL learning process, thereby

[6]The summary of datasets and code repositories for LLM-enhanced RL for wireless networks and other domains is available at: https://github.com/ResearcherSG/Tutorial-on-Large-Language-Model-Enhanced- Reinforcement-Learning-for-Wireless-Networks

Algorithm 1 Unified Pipeline of LLM-Enhanced RL for Wireless Networks

Initialize RL agent parameters θ, replay buffer $\mathcal{D}$
// Stage 1: Offline LLM Initialization
Use LLM as state perceiver to design semantic state representation function $\phi(\cdot)$
Use LLM as reward designer to generate or refine reward function $R_{\text{LLM}}(\cdot)$
// Optional: Pre-training via LLM Generator
$\mathcal{D}_{\text{sim}} \leftarrow$ LLM_Simulator()
$\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_{\text{sim}}$
// Stage 2: RL Training Loop
for each training iteration **do**
  Observe raw state s_t from environment
  $s_t^{\text{LLM}} \leftarrow \phi(s_t)$
  // Optional: LLM-guided decision (intermittent)
  if guidance is required **then**
    $a_t^{\text{guide}} \leftarrow$ LLM_Decision(s_t^{LLM})
    $a_t \leftarrow$ RL_Policy($s_t^{\text{LLM}}, a_t^{\text{guide}}$)
  else
    $a_t \leftarrow$ RL_Policy(s_t^{LLM})
  end if
  Execute a_t, observe s_{t+1}
  Compute reward $r_t \leftarrow R_{\text{LLM}}(s_t, a_t, s_{t+1})$
  Store $(s_t^{\text{LLM}}, a_t, r_t, s_{t+1})$ in $\mathcal{D}$
  Sample mini-batch from $\mathcal{D}$ and update RL networks
end for
// Stage 3: Policy Interpretation (Optional)
Use LLM to generate explanations of learned policy π

stabilizing the policy updating and accelerating exploration efficiency. This grounding allows LLMs to be naturally extendable to RL contexts by serving as a bridge between high-level task descriptions and low-level policy learning.

Specifically, on the one hand, LLMs are capable of extracting meaningful and semantically rich features from raw observational data by leveraging their powerful pre-trained architectures [70]. By leveraging their extensive pre-training on diverse datasets, LLMs can automatically identify relevant patterns, abstract high-level semantics, and process unstructured or heterogeneous inputs (such as textual logs, sensor readings, or even symbolic network states), into structured and task-relevant features [2], [71]. This capability not only reduces the reliance on domain-specific feature engineering but also enhances the agent's ability to generalize across diverse environments and tasks. For instance, the authors in [66] propose an LLM-enhanced DRL framework for accurate bandwidth estimation (LLM4Band) in real-time communication systems. The LLM4Band leverages the pre-trained semantic representation capabilities of LLMs to extract high-dimensional network state features from raw network flow data (such as packet histories, latency trends and bandwidth). These features are converted into the continuous action for estimating bandwidth. As a result, the LLM-based policy extraction improves the performance of the RL policy network, which achieves a 12.35% improvement (compared to LSTM baseline [72]) in bandwidth estimation accuracy and a 21% enhancement (compared to Heuristic algorithm [73]) in communication quality.

Building on this foundation of feature extraction for RL policy networks, the challenge of sample inefficiency and poor generalization in dynamic O-RAN network slicing is addressed in [46] through a more sophisticated dual-LLM architecture. The dual-LLM architecture is employed to extract semantically enriched state representations from raw and unstructured network observations such as RF features, QoS targets, and traffic trends. Specifically, a domain-specific LLM pretrained on O-RAN control logs and slicing policies is dynamically used to generate context-aware prompts. These prompts are fused with trainable tokens and processed by a frozen general-purpose LLM (i.e., GPT-2) to output high-level embeddings. These embeddings replace raw observations as inputs for RL agents, enabling the agents to effectively interpret complex wireless environments and concentrate on policy learning. The experimental results show that the proposed scheme achieves approximately 12.5% higher cumulative reward compared to traditional MARL (i.e., SAC algorithm [44]).

Beyond structured network metrics, LLMs also excel at interpreting rich sensory inputs such as images or text in network services. The work in [65] addresses the challenge of maximizing subjective Quality of Experience (QoE) in AI-Generated Content. The LLM is used to analyze images and simulate human-like QoE feedback based on the Big Five personality model (i.e., Openness, Conscientiousness, Extraversion, Agreeableness, and Neuroticism) since user preferences for generated images vary significantly based on personality traits. Then, the high-dimensional visual inputs can be converted into scalar reward signals (aesthetic ratings from 0 to 1), which serve as rewards for the RL agent to learn on low-dimensional and semantically rich signals. Experimental results show that the LLM-based DRL policy can achieve up to approximately 133% higher QoE compared to the DRL with a random policy.

On the other hand, LLMs can significantly reduce the learning complexity for RL agents by translating informal natural language inputs in wireless networks, such as communication commands, user requests, or high-level operational guidelines, into structured and task-specific representations [2]. This semantic transformation enables RL agents to better interpret and align with high-level objectives that are often conveyed in ambiguous or unstructured human language, which improves the adaptability of RL systems in dynamic or context-sensitive wireless environments [74]. The authors in [58] employ the LLM as the state perceiver to bridge natural language intents with RL-based network optimization. Specifically, the paper leverages LLMs to parse unstructured natural language intents (e.g., "optimize throughput by 10%") and extract structured optimization targets (e.g., throughput, latency, or energy efficiency) and their desired thresholds. This decouples intent interpretation from the RL agent, where the processed intents are then fed into an attention-based hierarchical RL (HRL) framework, which orchestrates network applications to achieve the specified optimization objectives. Simulation results demonstrate that LLMs enable the RL agent

to focus on applications most relevant to the operator's intent, which reduces conflicting actions and outperforms traditional HRL (with improvements of 12.02% in throughput, 26.5% in delay, and 17.1% in energy efficiency).

Additionally, the LLM can bridge high-level language commands and semantic feedback in a wireless network. The study in [68] integrates LLMs to bridge language commands with RL for resource allocation in autonomous network orchestration. The LLM is employed to translate unstructured inputs (i.e., strategic textual commands and semantic QoE feedback such as minimizing cost or maximizing quality) into formalized task representations. Specifically, based on the chain-of-thought prompt of the LLaMA-3.1-8B model, it maps the command to system-level objectives via step-back questioning, and maps user feedback to quantifiable semantic requirements through similarity scoring to replace conventional bit-level metrics. These processed outputs (e.g., objectives, state histories, and reward-augmented exemplars) form structured inputs for the RL agent to solve the simpler and numeric sub-problem. As a result, the proposed scheme achieves near-optimal resource allocation costs (within 5% of optimal) in stable conditions.

Expanding further into more complex and real-time decision-making domains, the work in [67] showcases how LLMs can simplify joint optimization problems in vehicular networks by translating informal natural language task descriptions into executable actions, where the LLM is leveraged to translate informal natural language task descriptions (e.g., "maximize speed while minimizing collisions") into formal and executable AD actions (e.g., acceleration, lane changes). Specifically, by integrating in-context learning, the LLM processes structured prompts containing environmental states (e.g., vehicle positions, velocities), predefined action spaces, and curated examples of past experiences (e.g., good/bad decisions). These prompts enable LLMs to derive discrete AD actions for vehicles, which are embedded as part of the state of a Double Deep Q-Network (DDQN) that optimizes V2I base-station selection. The hybrid LLM-DDQN framework achieves 23% higher AD rewards and 18% higher V2I rewards compared to conventional DDQN baselines.

Lesson Learned. The reviewed studies on LLM as a state perceiver enhance the capability of reinforcement learning in state representation for wireless applications such as radio resource management and network orchestration. In radio resource management, the work in [46], [66], and [67] show that LLMs can transform raw and heterogeneous observations (e.g., traffic statistics, channel conditions, and mobility patterns) into semantically enriched state representations. However, misaligned semantic encoding may propagate errors into the RL policy. Introducing additional calibration mechanisms to mitigate the negative impact of semantic bias on the policy is a feasible solution [66]. For network orchestration, LLMs can translate unstructured textual inputs into structured state representations and utilize chain-of-thought reasoning to decompose orchestration tasks into structured sub-problems, enabling near-optimal resource allocation [46], [58], [68]. State perception in wireless environments is sensitive to the computational overhead and latency of CoT. If LLMs are continuously used for perception, it is necessary to consider lightweight processing of the reasoning process, such as compressing the reasoning chain [75]. In addition, asynchronous invocation of LLMs can be considered to intermittently update the state for the RL agent [76].

TABLE I
SUMMARY OF RELATED SURVEYS

Reference	Focus
[26]	A survey that systematically maps DRL algorithms to core wireless networking tasks, establishing an MDP-based framework for intelligent and adaptive network control
[27]	A survey that reviews RL applications from a layer-wise perspective of wireless networks, highlighting cross-layer optimization, algorithm–task alignment, and challenges in building unified and interpretable learning frameworks
[28]	A review that provides a comprehensive analysis of DRL in IoT systems, introducing a unified framework for communication, computing, caching, and control
[9]	A tutorial that unifies single-agent, multi-agent, and model-based DRL for AI-enabled wireless networks, emphasizing cooperative and distributed learning paradigms for scalable and autonomous network optimization
[29]	A survey that provides a comprehensive overview of multi-agent RL for future Internet architectures, highlighting Markov game formulations, cooperative and competitive learning algorithms, and applications in distributed and autonomous network management
[10]	A survey that reviews reinforcement learning-based approaches for autonomous multi-UAV wireless networks, which emphasize perception, decision, and action frameworks to enable key intelligent tasks such as efficient communication and adaptive trajectory optimization

D. LLM as Reward Designer

The efficacy of an RL agent is fundamentally determined by the design of its reward function, which is the very signal to defines its purpose and guides its learning process. Before the advent of LLMs, the design of reward functions in RL was often a manual and domain-specific process, requiring expert knowledge to define appropriate reward signals that align with the desired objectives and constraints of the task [9], [10], [45], [79], [80], [81]. However, LLMs have the potential to revolutionize this process by automating and generalizing reward function design through their advanced natural language understanding and reasoning capabilities.

LLM-based reward designs can be broadly categorized into two main approaches, as shown in Table III: LLMs as an implicit reward designer, where LLMs evaluate or judge the quality of agent actions or policies based on their understanding of the task and context; and LLMs as an explicit reward designer, where LLMs generate structured reward functions based on high-level task descriptions, system constraints, and optimization objectives [24].

1) Implicit Reward Designer: For example, the work in [77] proposes an LLM-based reward mechanism in multi-agent RL (MARL) for mobile access points organization using uncrewed aerial vehicles (UAVs). In this work, multiple agents, which represent UAVs relaying data to ground users or UAVs themselves, are trained to relocate their positions to optimize the average data rate of user devices and the number of connected devices. To achieve this, the authors

TABLE II

SUMMARY OF LLMS AS STATE PERCEIVERS FOR ENHANCING RL CIRCLES (•) DESCRIBE THE METHODS; CORRECT MARKERS (✓) AND CROSS MARKERS (✗) REPRESENT PROS AND CONS RESPECTIVELY

Taxonomies	Reference	LLM type	Pros & Cons
Extract meaningful features from raw observations	[66]	LLaVA[1]	• LLM-based agents simulate user feedback for personalized network service optimization ✓ LLM performs real-time multimodal evaluation to eliminate costly human feedback ✓ Adaptable to diverse user preferences and multimodal input scenarios ✓ Eliminates reliance on manual reward shaping via LLM-extracted features ✗ Multi-agent RL requires per-user LLM feedback, causing overhead as users grow
	[67]	GPT-2 T5[2] Qwen1.5[3]	• LLM extracts semantic features of bandwidth estimation for DRL's decision-making ✓ Leverages LLMs' generalization for unseen states to mitigate online RL interaction risks ✓ LLM enhances RL's cross-scenario generalization, improving bandwidth estimation accuracy ✗ Dependence on LLM feature quality; misalignment in state encoding can propagate errors
	[46]	Oransight -2.0[4] GPT-2	• Dual-LLM-augmented MARL for semantic state representation in O-RAN slicing ✓ LLM's domain knowledge enables robustness to network state changes for RL ✓ Integration of LLMs decreases state dimensionality and improves sample efficiency in RL ✗ Dual-LLM inference and RAG retrieval increase latency of RL in edge deployment
Reduce learning RL complexity by transferring informal natural language information into a formal task-specific language	[59]	ALBERT[5]	• LLM-driven intent parsing and validation combined with hierarchical RL ✓ Enhance HRL scalability by offloading high-level intent interpretation to LLMs ✓ Automates translation of natural language intents into structured optimization tasks ✗ Ambiguous intents of LLM's response may cause errors in intent extraction
	[68]	ChatGPT -3.5 LLaMA 3.1–8B 3.1–70B	•Prompt-driven LLMs generate AD actions as state features for the DDQN ✓ Achieve faster convergence and higher rewards in RL via LLM-guided exploration ✓ Explain decisions via natural language such as collision avoidance rationale ✓ Decouple perception and reasoning from RL exploration to reduce sample complexity ✗ Larger LLMs (e.g., 70B) incurring high latency are unsuitable for ultra-low-latency V2I ✗ Joint decision-making with multiple LLMs has limited scalability to many coupled tasks
	[69]	LLaMA 3.1–8B	• LLM intent parsing and CoT-driven task decomposition for continual RL agent ✓ Eliminate manual reward design via LLM-extracted structured tasks ✓ Enhance adaptability to dynamic environments with RAG-augmented context ✓ Reduce RL exploration complexity through CoT-guided sub-task sequencing ✗ Computational overhead from LLM inference and RAG retrieval limits edge deployment

[1]LLaVA is a multimodal LLM that integrates vision and language understanding, available at https://huggingface.co/docs/transformers/model_doc/llava.
[2]T5 is a text-to-text transformer model, available at https://huggingface.co/docs/transformers/model_doc/t5.
[3]Qwen1.5 is a family of efficient and capable LLMs developed by Alibaba, available at https://huggingface.co/collections/Qwen/qwen15.
[4]Oransight-2.0 is a Qwen-based LLM series for enhanced reasoning and domain-specific tasks, available at https://huggingface.co/collections/NextGLab/oransight-20-qwen.
[5]ALBERT is a parameter-efficient variant of BERT that reduces model size via factorized embeddings and cross-layer parameter sharing, available at https://huggingface.co/docs/transformers/model_doc/albert.

TABLE III

SUMMARY OF LLMS AS REWARD DESIGNERS FOR ENHANCING RL CIRCLES (•) DESCRIBE THE METHODS; CORRECT MARKERS (✓) AND CROSS MARKERS REPRESENT (✗) PROS AND CONS RESPECTIVELY

Taxonomies	Reference	LLM type	Pros & Cons
Implicit Reward Designer	[79]	Specific LLM	• LLM generates optimal positions for UAVs, used as a positioning reward signal ✓ Improves network metrics (e.g., +15% data rate, +20% connected devices) ✓ Guides agents based on high-level regional characteristics, not just local metrics ✗ Performance is dependent on the LLM's spatial reasoning capabilities ✗ Adds computational overhead from LLM inference during training
	[80]	PaLM 2 XS[1]	• LLM acts as an automated annotator to generate preference data for training a reward model ✓ Highly scalable alternative to expensive and slow human feedback (RLHF) ✓ Achieves performance comparable to RLHF on tasks like summarization ✓ Can improve model harmlessness more effectively than RLHF ✗ Risk of inheriting and amplifying biases from the AI labeler model ✗ Performance is bottlenecked by the quality and alignment of the labeler LLM
Explicit Reward Designer	[21]	GPT-4	• LLM generates executable, dense reward code from natural language goals ✓ Produces interpretable and debuggable reward functions (Python code) ✓ Computationally efficient for RL training (no repeated LLM calls) ✓ Performance matches or exceeds human experts on many robotics tasks ✓ Supports iterative human-in-the-loop refinement to fix failure modes ✗ The initial "first-shot" generated code may be suboptimal or buggy
	[22]	GPT-4	• Combines LLM code generation with an evolutionary search algorithm ✓ Autonomously discovers novel and high-performing reward functions ✓ Outperforms human-designed rewards on a majority of complex tasks ✓ Solves tasks previously intractable with manual reward design (e.g., pen-spinning) ✗ Extremely high computational cost due to the evolutionary outer-loop ✗ Requires access to the environment's source code to provide context
	[70]	Specific LLM	• LLM serves as reward designer to balance throughput, latency, and packet loss in routing ✓ Enable faster RL convergence in UAV-satellite routing tasks without manual weight tuning ✓ Prevents myopic routing by rewarding end-to-end transmission quality rather than hop count ✗ Correctness and hallucination of LLM-generated reward functions not verified ✗ LLM version and parameters are not specified, which makes reproduction difficult

[1]PaLM 2 XS is a lightweight variant of the PaLM 2, available at https://huggingface.co/papers/2305.10403.

propose a PPO-based MARL method, where the reward function consists of two components: the communication reward and the positioning reward. The communication reward is designed to encourage agents to maximize the

TABLE IV

SUMMARY OF LLMS AS DECISION MAKERS FOR ENHANCING RL CIRCLES (•) DESCRIBE THE METHODS; CORRECT MARKERS (✓) AND CROSS MARKERS (✗) REPRESENT PROS AND CONS RESPECTIVELY

Taxonomies	Reference	LLM type	Pros & Cons
LLM-based	[84]	GPT-4o-mini	• Multi-agent hierarchical framework (ACMA) for adaptive HAPS fleet positioning ✓ Decomposes complex optimization into interpretable subtasks for improved scalability ✓ Can interpret high-level natural language instructions into efficient, actionable strategies ✗ Increased system complexity due to multiple specialized agents and inter-agent coordination ✗ Potential response latency overhead from orchestrating multiple LLMs per decision cycle
	[85]	GPT-o3-mini GPT-4o-mini GPT-2.5-Trubo	• ICL-based Data Collection Scheduling (ICLDC) for UAV-assisted sensor networks ✓ Implements a feedback loop by incorporating real execution results into in-context prompts ✓ Adaptation occurs via prompt update rather than costly weight updates or retraining ✗ Vulnerable to prompt-based attacks (e.g., jailbreaking) which can degrade network performance ✗ Effectiveness depends on the feedback window, may be challenging with limited context length
	[86]	ViT[1]	• Multi-modal LLM-powered joint decision-making for ramp merging in vehicular networks ✓ Multi-modal encoder captures both textual and visual context, improving situational awareness ✓ Enforced reasoning and explicit meta-decision generation foster transparency and interpretability ✗ Fine-tuning and multi-agent rollout still require significant data engineering and simulation effort ✗ Real-time deployment may be challenged by large LLMs and vision processing on edge hardware
LLM-guiding	[69]	LLaMA 3.1–8B	• Hierarchical LLM-RL framework for resource allocation in SAGIN ✓ Combines high-level LLM strategy generation with efficient low-level RL execution ✓ Employs CoT reasoning and few-shot learning to improve reliability and mitigate hallucinations ✗ Added complexity from multi-layer (LLM+RL) architecture and prompt engineering requirements
	[87]	Not Specified	• LLM-guided knowledge distillation framework for efficient RL training in UAV networks ✓ Utilizes LLM's high-level decision knowledge and CoT reasoning to accelerate RL training ✓ Reduces inefficient RL exploration and risk of local optimum convergence ✗ Effectiveness depends on the quality and generalizability of LLM-generated teaching data ✗ Policy optimization from LLM outputs may suffer from biased distributions due to hallucinations
	[88]	LLaMA	• PPAE algorithm that integrates LLaMA into RL actor network training for AUV path planning ✓ Action probability distribution generated by LLaMA narrows the exploration space in training ✓ Effective distributional priors reduce policy variance in complex marine environments ✗ Policy optimization from LLM outputs may suffer from biased distributions due to hallucinations
	[89]	Gemma 2-2B	• Lightweight LLM-assisted DQN framework for task scheduling in multi-cloud environments ✓ Generate action candidates to narrow exploration space and accelerate convergence ✓ Lightweight LLM (2B parameters) enables deployment on resource-constrained devices ✗ Low-parameter LLMs may be limited in knowledge scope when processing complex state

[1]ViT is a transformer-based model that applies self-attention directly to image patches for vision tasks, available at https://huggingface.co/docs/transformers/model_doc/vit.

TABLE V

SUMMARY OF LLMS AS GENERATOR FOR ENHANCING RL CIRCLES (•) DESCRIBE THE METHODS; CORRECT MARKERS (✓) AND CROSS MARKERS (✗) REPRESENT PROS AND CONS RESPECTIVELY

Taxonomies	Reference	LLM type	Pros & Cons
World Model	[47]	OPT LLaMA-2 TimeNet Time-LLM	• Integrate multiple LLMs to enhance UAV mission security and operational efficiency ✓ Adopt diverse smaller mission-critical LLMs to reduce latency and improve real-time decisions ✓ Distributed architecture offloads computational tasks across on-board, edge, and cloud ✗ Each domain-specific LLM requires its own training and fine-tuning, increasing complexity
	[118]	GPT-4o GPT-4.0 Gemini- 1.5-flash	• A pre-processing pipeline for wireless environment perception through RAG-based LLM ✓ Utilizes RAG to enhance LLMs' understanding of wireless networks from multiple data sources ✗ Performance highly dependent on the performances of the RAG model and LLMs
	[119]	GPT-4	• An LLM-driven framework for generating cryptographic keys in IoV systems. ✓ Improve key generation efficiency while maintaining crucial channel information
	[120]	Claude-3 Opus GPT-4	• A framework for adapting and enhancing LLMs in the wireless communication networks ✓ Flexible enhancement approaches that only require closed-box access of LLMs ✗ The security and privacy are not guaranteed as the LLM may leak sensitive information ✗ The computational overhead raises concerns while deploying on diverse edge devices
Policy Interpreter	[121]	Falcon-7B	• Integrate LLMs into ZTNs to enhance the transparency of RL and improve user interactions ✓ Utilize LLMs to transform perplexing network actions into digestible insight ✗ The maintenance and retrieval of massive data reduce the practicality of real-time scenarios ✗ Using LLMs to process massive sensitive data will introduce privacy and ethical concerns

above-mentioned hyperparameter-weighted optimization objectives, and the relay intrinsic reward is designed to encourage agents to optimize the relay nodes that serve as bottlenecks in subsequent networks. The authors in [77] also introduce an LLM-guided reward shaping mechanism, where the LLM does not directly generate a reward function, but rather generates a reward signal to guide the agent's search. Specifically, the LLM receives the number of devices within meshed grid cells and the problem description, and generates a target positioning configuration (i.e., the positions where the agents should be located). A reward is then computed based on the similarity between the agents' achieved state and this target configuration. Combining the communication reward and the LLM-based positioning reward, the proposed

method achieves an up to 15% improvement in average data rate and an up to 20% improvement in number of connected devices and compared to the baseline method where the LLM is not used.

A prominent paradigm in this category is Reinforcement Learning from AI Feedback (RLAIF) [78]. RLAIF automates the costly process of Reinforcement Learning from Human Feedback (RLHF) by using a separate, often more powerful, LLM to generate preference labels for text-based tasks such as summarization and dialogue generation instead of human annotators. A reward model is then trained on this AI-generated data to provide scalable feedback for policy optimization.

This approach can achieve performance comparable to RLHF on tasks such as text summarization and dialogue generation. For example, RLAIF and RLHF are preferred over traditional supervised fine-tuning methods with 71% and 73% of the win rates for summarization, and 63% and 64% for dialogue generation, respectively. Moreover, RLAIF achieves a higher harmlessness rate of 88% compared to 76% for RLHF in dialogue generation tasks, thereby helping to address the scalability limitations of human annotation.

2) Explicit Reward Designer: In contrast to implicit methods, an explicit reward designer uses an LLM to generate an entire, executable reward function, typically as a Python program. This approach offers significant advantages in terms of interpretability, as the reward logic is transparent and can be inspected or debugged by a human. Furthermore, it is computationally efficient during RL training, as it eliminates the need for repeated, costly API calls to an LLM.

A foundational framework, Text2Reward [21], is proposed to automate the generation of dense, shaped reward functions from a natural language goal and a compact description of the environment. Specifically, given a natural language instruction such as "push the chair to the marked position" and a set of programmable state query functions (e.g., object poses, robot end-effector position, grasp status), Text2Reward generates a dense reward that combines sub-rewards for reaching, grasping, and placing. Analogously, such methodology can be transferred to wireless network optimization. For instance, an optimization objective such as "maximize throughput while keeping latency below 10 ms" can be paired with the channel state information interface that exposes key network metrics (e.g., channel quality indicators, transmit power levels, and packet delay). An LLM can then generate a corresponding dense reward function that includes a positive term for achieved throughput, a negative term for latency violations (e.g., a penalty when delay exceeds a threshold), and a regularization term for energy consumption.

Building on this, the EUREKA framework [22] introduces a higher level of autonomy by combining LLM code generation with an evolutionary search algorithm. EUREKA operates in a loop: it first uses an LLM to generate a population of candidate reward functions, often by providing the environment's source code as context. These functions are then evaluated by training RL agents in parallel. Finally, an automated "reward reflection" step summarizes the training outcomes and feeds this summary back to the LLM, which then generates an improved set of reward functions for the next iteration. This meta-learning approach has successfully designed rewards for highly complex dexterity tasks, such as simulated pen-spinning, that were previously intractable with manual reward engineering, where it outperformed human-designed rewards in 24 out of 29 tasks.

In the domain of routing in UAV–satellite networks, the study in [69] leverages an LLM as a reward designer, which interprets high-level objectives expressed in natural language (such as maximizing throughput, minimizing end-to-end migration delay, and penalizing packet loss) and automatically generates the corresponding reward function. This design embeds routing optimization objectives directly into the reward, balancing throughput and latency while penalizing packet loss. It reflects an end-to-end transmission view rather than traditional hop-count minimization, preventing the agent from taking myopic actions such as routing only to nearest neighbors at the expense of throughput. Compared with conventional DRL frameworks where reward functions are hand-crafted directly based on optimization goals, the LLM-generated reward function achieves approximately 12% higher throughput and 26.5% lower latency, and roughly a threefold improvement in convergence speed.

Lesson Learned. LLM as a reward designer can provide effective feedback for RL agents during policy training in wireless applications, which has attracted increasing attention from researchers. The authors in [77] show that LLMs can generate high-level spatial guidance as reward signals for MARL-based UAV deployment, which significantly improves performance in radio resource management. The study in [69] shows that LLMs can encode high-level objectives into reward functions to avoid myopic decisions for RL agents in routing planning. The work in [21] and [78] provide general insights into automating reward modeling for RL using LLMs. In general, using LLMs as reward designers does not require much consideration of latency, since the reward function of the RL agent is typically fixed once designed. However, a potential issue remains how to verify the feasibility of LLM-generated reward functions due to the randomness and hallucination in outputs. A feasible solution is provided in [22], which adopts a paradigm of multi-candidate generation, external evaluation, and reward reflection. More advanced validation mechanisms still require further investigation.

E. LLM as Decision-Maker

Benefit from the scaling law [5], LLMs demonstrate strong understanding capabilities on out-of-distribution (OOD) data, enabling them to make decisions in complex and rare environments [88]. Furthermore, with the emergence of several key capabilities, including multimodal understanding [89], task planning [90], and sequential reasoning [61], LLMs are expected to enhance or even substitute traditional RL as a better decision-maker [88] rather than as a standalone tools. In this section, the reviewed methods are categorized into two main types based on the role of LLMs in decision-making: 1) LLM-based methods that LLM directly generate actions and 2) LLM-guiding methods that LLM guides the RL agent to make decisions.

1) LLM-Based: Early studies have recognized the powerful reasoning and decision-making capabilities of LLMs, and have successfully embedded them into various types of network optimization and resource allocation tasks [68], [82], [83], [85]. Compared with traditional RL-based methods, these approaches have achieved significant improvements in key metrics such as QoS, coverage, and throughput.

Han et al. [82] leverages LLMs to operate a fleet of High Altitude Platform Stations (HAPS) [91] for dynamic user coverage in wireless networks. The proposed framework, called Action Coordination and Multi-Agent (ACMA), utilizes LLMs to optimize the positions of HAPS airships to provide a more efficient and reliable service. ACMA decomposes the complex optimization problem into a series of four subtasks, each handled by a specialized LLM-based agent [92]. The workflow involves a Data Analysis Agent that identifies high-demand areas, a Target Location Selection Agent that makes the primary positioning decisions, an Overlap Avoidance Agent that refines these positions to enhance resource efficiency, and a Special Event Handling Agent. Furthermore, the ACMA framework supports to plans and adjusts the stations like an autonomous agent, based on occasional high-level natural language instructions. This hierarchical approach allows for more nuanced decision-making, enabling the system to adapt to dynamic user demands and environmental changes effectively. ACMA demonstrates a 130% improvement in user coverage compared to RL-based method and random walk [93]. Similarly, Emami et al. [83] introduce an In-Context Learning-based Data Collection Scheduling (ICLDC) scheme for UAV-assisted sensor networks [94], offering an alternative to DRL methods in time-sensitive emergency scenarios. In this framework, the UAV collects real-time network data and relies on an edge-hosted LLM, which, guided by a dynamic context of task descriptions and in-context examples, directly generates scheduling decisions. ICLDC introduces a continuous feedback mechanism: after each scheduling decision, the outcome is incorporated back into the LLM's context, enabling the model to iteratively adapt its decisions based on recent performance—effectively achieving online adaptation without weight updates. This approach allows the LLM to adjust its actions responsively in non-stationary environments, bridging the gap between static pre-trained models and the demands of real-time edge intelligence. Compared with traditional methods that rely solely on channel conditions, ICLDC achieves a significant reduction (up to 56%) in packet loss. The study also highlights LLMs' potential vulnerability to adversarial prompt manipulation, suggesting that developing robust defense strategies is an important avenue for future research.

In addition to scheduling in aerial networks, LLMs have been explored for decision making in vehicular networks. For instance, Hu et al. [84] propose AgentsCoMerge, a framework for ramp merging tasks that leverages LLMs equipped with a multi-modal encoder. By incorporating visual as well as textual information, AgentsCoMerge enables the LLM to better interpret vehicle surroundings, enhancing its ability to make spatially and contextually informed decisions. The framework enforces explicit reasoning for decision transparency and reliability, requiring the LLM to justify high-level meta-decisions, which are then converted by each agent into executable trajectories. In cases of failure (e.g., collisions), the LLM receives feedback and reasoning prompts, iteratively refining its decisions based on detailed analysis of mistakes. To better align with domain-specific requirements, AgentsCoMerge also employs fine-tuning using LoRA [95], improving the LLM's ramp merging expertise with minimal computational overhead. This approach highlights the potential of LLMs as adaptive and explainable decision makers in complex, real-time vehicular network environments.

2) LLM-Guiding: Although these LLM-based methods demonstrate the potential of LLMs as decision-makers, they still face several challenges since almost all resource schedule and allocation are dictatorially determined by LLMs rather than RL agents. However, the computational-intensive nature of LLM makes it unable to respond promptly to the fast-changing wireless channel resource environment, resulting in its inability to handle most communication resource allocation tasks with low latency requirements [96]. Furthermore, with the issue of LLM hallucinations, the reliability of its responses is gradually being questioned [97], especially when applied to an OOD scenario [88]. Therefore, several recent studies have attempted to leverage LLMs to guide the decision-making process of RL agents, rather than replacing them entirely.

For example, Shokrnezhad and Taleb [68] introduces a framework called Autonomous Reinforcement Coordination (ARC) for orchestrating a Space-Air-Ground Integrated Networks (SAGIN). ARC decomposes the orchestration task into two layers: a high-level LLM-based layer and a low-level RL-based layer. The LLM layer generates high-level strategies and guidance as the prior information, while the RL layer executes actions based on these strategies. For instance, the LLM first generate a user sequence that includes user ID, service ID and its action profile, then the RL agent will execute these action selection for each user. This hierarchical approach allows for more efficient and adaptive resource allocation in dynamic environments. Besides, ARC employs Chain-of-Thought (CoT) reasoning [61] through few-shot learning to reduce the risk of hallucinations to enhance system robustness and reliability. Specifically, a few reasoning exemplars are demonstrated in LLM prompts to provide a more detailed instruction, which includes a chain of thoughts demonstrating the sequence of users and their corresponding actions selected for a specific history-objective pair. Compared with Non-Reinforcement ARC that disables the RL layer and allows the LLM to directly control the low-level allocations, the proposed ARC framework demonstrates a significant less fluctuations brought by LLM hallucinations. Therefore, ARC strikes a balance between the high-level decision-making capabilities of LLMs and the low-level execution efficiency of RL agents, achieving a more robust and reliable orchestration in SAGIN.

Compared to involving LLMs during the inference phase, Xu et al. [85] attempted to leverage LLMs to optimize the RL training process in an UAV-enabled multi-hop networking. The proposed LLM-guided knowledge distillation framework shares a similar intuition that guides the RL agent to learn the optimal UAV scheduling strategy based on the high-level

decision knowledge from LLM. However, the knowledge transfer process is conducted during the training phase of RL agents, where a knowledge distillation process is designed to prepare decisions that align with common sense for RL agents. Specifically, the LLM acts as a teacher model, reasoning out appropriate actions based on a carefully designed Chain-of-Thought (CoT) process and the current state. These state-action pairs are collected then used to teach the RL agent to perform decision. This approach effectively avoids the issue of the RL agent engaging in a large amount of inefficient exploration in the early stages of training under random initialization, while also reducing the risk of the RL agent prematurely converging to a local optimum. In a 3.5 Km $\times$3.5Km simulation environment with 18 UAVs and 150 UEs, the proposed framework achieves around an average of 23% higher UE coverage, 52% higher data rate, and a 19% higher UAV availability ratio compared to three multi-agent reinforcement learning (MARL) baselines, GVis [98], GA2C [99] and MAPPO [100], that without the LLM guidance.

In addition to serving as a teacher model to reason about the RL agent's actions [85], an LLM can also act as a soft expert policy to guide RL training. The authors in [86] integrate the LLaMA model into the RL pipeline to enhance decision robustness in autonomous underwater vehicle (AUV) path planning. Specifically, they design a proximal policy advantage estimation (PPAE) algorithm, where the LLaMA model is utilized to generate the action probability distribution based on environmental inputs. The action probability functions as the soft expert policy to narrow the exploration space of RL and guide the actor network training to benefit from the LLM's prior knowledge. Thus, the RL agent can optimize the actor's policy distribution based on both the guidance from the LLaMA model and the original reward signals. Compared with the traditional PPO algorithm, the proposed PPAE improves episode rewards by approximately 11.1% to 27.3% and achieves shorter path planning, with path lengths reduced by about 16.7% to 20%.

Apart from leveraging LLMs to generate the action probability distribution for RL agents, LLMs can also narrow down the exploration space by producing high-quality action candidates in complex system optimization. The work in [87] proposes a lightweight LLM-assisted DQN framework for task scheduling in multi-cloud environments. In this framework, the LLM acts as a decision guide, with its core function being the generation of an action candidate set to enhance the decision-making efficiency of the RL agent. Specifically, during the RL training process, the fine-tuned LLM receives the current system state—including the task queue, available resources, and optimization objectives—and generates a high-quality action candidate set (i.e., a group of recommended schedulable instances). The DQN agent then performs Q-value evaluation and makes final decisions within this refined, high-quality candidate set, thereby avoiding inefficient exploration in the vast original action space. This approach effectively combines the LLM's capabilities in semantic reasoning and domain knowledge with DQN's strengths in numerical optimization and long-term return trade-offs. Experimental results demonstrate that the proposed LLM-guided framework significantly outperforms baseline methods [101] in terms of cost, makespan, and energy consumption.

In addition, the recent study in [102] has shown that LLMs also exhibit great potential in solving multi-objective optimization problems, where their ability to act as planners can be leveraged to assist RL in policy learning and action selection. Specifically, the work in [102] formulates a multi-objective optimization problem (MOP) that jointly optimizes UAV deployment and transmission power control, which is decomposed into a series of optimization subproblems. To simultaneously work on these sub-problems, an LLM is integrated as a closed-box search operator and guided by MOP-specific prompts that consider the objective functions and constraint satisfaction to generate Pareto-optimal solutions. Experiments show that convergence speed improves by approximately 2.7% compared with the baseline reference vector guided evolutionary algorithm [103].

Lesson Learned. LLM as the decision maker of RL agents has been more widely explored in wireless networking applications. For resource management, the works in [82], [83], [86], [87], and [102] demonstrate that LLMs can directly generate or guide decisions for tasks including UAV deployment, data collection scheduling, task allocation, and multi-objective optimization. For routing-related decision making, the work in [84] illustrates that LLMs can implicitly capture routing and interaction dynamics through high-level reasoning rather than relying on predefined routing protocols. In network orchestration, the studies in [68] and [85] use LLMs to determine the high-level orchestration logic, such as user-service sequencing, action profiles, or scheduling knowledge, while RL agents perform low-level action optimization under such guidance. The main limitation of current designs is that replacing RL agents with LLMs for decision making may incur considerable latency and computational overhead. Furthermore, decisions made directly using LLMs require a high degree of accuracy and effectiveness. Therefore, using LLMs to guide or assist RL agents in decision making may be a more prudent design choice. Relevant measures can be taken to prevent LLMs from encountering out-of-distribution (OOD) states [104] and hallucinated outputs [105] due to the mismatch between their pretraining data and dynamic wireless environments. OOD detection [106] and hallucination detection mechanisms [107] can be incorporated, but the RL training loop may be impractical to afford additional latency and computational overhead from the detection modules. A more practical compromise is to improve the LLM's generalization ability to unseen data at the source. Domain adaptation and fine-tuning on wireless-specific datasets can significantly reduce the occurrence of OOD states and hallucinations [108], [109]. In addition, safety-oriented mechanisms can be incorporated into the decision pipeline. For example, safe RL techniques can achieve constraint-aware policy learning to prevent unsafe actions [110], while human-in-the-loop supervision can serve as a final safeguard to intercept critical decisions before execution [111].In this way, the RL agent can leverage the guidance provided by the LLM while still maintaining its own decision-making autonomy. In addition, collaborative decision making between small models (deployed at the edge to handle simple tasks)

and large models (deployed in the cloud to perform deep reasoning for complex tasks) can serve as feasible solutions to reduce the delay and computational burden [18], [112]. Finally, the latency of LLM inference must be reconciled with the ultra-low-latency demands of wireless networks, motivating hybrid designs where LLMs operate offline or asynchronously while real-time control relies on compact distilled policies. In the offline mode, knowledge distillation has the potential to transfer the offline reasoning capability of LLMs (as a teacher) into compact student policies to reduce the inference latency in decision making for wireless applications [113]. In the asynchronous mode, edge-cloud collaborative inference is a feasible paradigm that enables LLM reasoning to be executed at cloud servers to provide high-level policy guidance, while RL agents run at the network edge to perform efficient policy learning in dynamic wireless environments [114]. Overall, these considerations highlight the multiple trade-offs of LLM-enhanced RL for future wireless networks.

F. LLM as Generator

LLMs can serve as generators in various ways, such as generating synthetic training data, simulating environments, providing high-level abstractions, or formulating explanation that facilitate RL learning. In this section, we categorize the reviewed methods into two main types based on the function of LLMs: 1) LLM as a world model simulator, where LLMs simulate the environment dynamics or generate synthetic data; and 2) LLM as a policy interpreter, where LLMs interpret high-level policies or provide explanations to guide RL agents.

1) World Model Simulator: The application of RL to real-world wireless networks is fundamentally constrained by the practicalities of data collection. Training an RL agent on a live network is often prohibitively expensive, time-consuming, and carries significant operational risks [119], [120], such as service disruption or instability. Simulation offers a safe, scalable, and efficient alternative, allowing agents to undergo millions of training episodes in a controlled virtual environment. However, the efficacy of a simulation-trained agent is contingent upon the fidelity and diversity of the simulation. A significant gap between the simulated environment and real-world dynamics, often termed the "sim-to-real" gap, can lead to policies that perform poorly upon deployment.

Existing studies have attempted to adopt machine learning (ML) models to model and synthesize wireless data [121], [122], such as Orekondy et al. [121] adopt Generative Adversarial Networks (GANs) [123] to generate channel impulse response. However, prior ML models often lack a deeper contextual understanding, which can result in unrealistic or overly simplistic simulations that fail to capture the complexities of real-world wireless environments. In contrast, LLMs can utilize their extensive world knowledge to implicitly infer causal relationships between prompt concepts and expected data characteristics, thereby generating data from high-level semantic prompts. For example, Dharmalingam et al. [47] propose a distributed system engineered to improve the security and operational efficacy of UAV during mission-critical operations. This system adopts a team of diverse, smaller LLMs, each fine-tuned for a specific generative task, including real-time data inference, anomaly detection, and activities forecasting. Moreover, these "special-skilled" LLMs employ a combination of Supervised Fine-Tuning (SFT) [124] and Reinforcement Learning from Human Feedback (RLHF) [125]. This hybrid approach ensures that the LLMs are well-adapted to the specific data and operational context of UAV missions. The generated data can be used to train task-specific RL agents, enabling them to learn effective policies in a simulated environment that closely mirrors real-world dynamics.

Mohsin et al. [115] develop a wireless environment perception framework that translate raw, multi-modal sensor data into a structured, human-readable, and machine-actionable textual description. This perception is positioned as an essential input for higher-level RL tasks, such as global optimization, resource allocation, and intelligent handover decisions, which require a comprehensive understanding of the operational context. A Retrieval-Augmented Generation (RAG) architecture [126] is employed in this system, which ingests a diverse set of synchronized data streams from sensors commonly found in autonomous vehicle or smart city environments, including high-resolution cameras, GPS units, and LiDAR scanners. By processing and fusing this information, the LLM aims to generate a detailed narrative of the environment, identifying objects, their spatial relationships, and potential impacts on wireless signal propagation. Compared to a "Vanilla LLM" baseline that without the RAG-retrieved context, the proposed framework demonstrates an 8% improvement in relevancy, an 8% improvement in faithfulness (adherence to source facts), and a 12% improvement in overall accuracy.

Beyond applying LLMs for generating or integrating wireless channel data, emerging research has begun exploring the potential of leveraging LLMs for wireless key generation [116], which is essential for establishing secure communication links in Next-Gen Internet of Vehicles (IoV) systems [127]. Currently key generation methods typically require excessive and repeated channel probing, leading to significant overhead and latency [128]. LLMKey is designed to address this challenge by skipping channel probing and leveraging the LLM's few-shot ability to generate values for skipped channel probing. The experimental results demonstrate that even with a probing ratio as low as 0.5, the system still attains an average key agreement rate of 0.978. However, the LLM-driven key generation process is still limited by the high computational overhead and latency of LLMs, which may not be suitable for real-time applications. Therefore, adopting knowledge distillation techniques [129] to transfer the LLM's knowledge to a lightweight RL agent is a more promising solution to address this issue.

However, existing methods are still limited by the LLM's inherent capabilities, which may not be sufficient to fully understand the complex and dynamic nature of wireless environments. Therefore, Shao et al [117] release WirelessLLM, a comprehensive framework to further enhance the domain-specific knowledge and expertise of LLMs in wireless systems by incorporated several advanced techniques. WirelessLLM first involves carefully crafting CoT prompts and few-shot task demonstrations to guide the LLM's reasoning process.

Then, WirelessLLM allows LLMs to leverage external software tools, such as MATLAB or Python functions to perform complex calculations or simulations that are beyond the LLM's inherent capabilities. The generative capabilities of LLMs present a novel opportunity to create richer, more realistic, and more strategically complex simulations, thereby narrowing this gap and enhancing the robustness of RL agents.

2) Policy Interpreter: Despite the performance benefits of RL, its adoption in critical infrastructure like wireless networks is hindered by a significant trust deficit. RL agents often operate as "closed-boxes", with their decision-making rooted in complex, high-dimensional neural networks that are opaque to human operators [130]. This lack of transparency is a major barrier to deployment, as network engineers and operators are justifiably reluctant to cede control of mission-critical systems to AI whose reasoning cannot be understood, verified, debugged, or trusted [131]. Explainable Reinforcement Learning (XRL) has been introduced to address this challenge, and LLMs, with their innate mastery of human language, can be used for generating policy explanations [132]. For instance, Lin et al. [133] construct a decision tree based on the policy, and then use LLMs to derive explanation based on the decision tree. Das et al. [134] introduce State2Explanation, a framework that leverages a specific model to generate concept-based explanations based on the state-action pairs. Although several methods have been proposed to enhance the interpretability of RL policies, only limited research has deployed this paradigm in the scenario of wireless networks. A canonical example is presented by Ali et al. in their work on DRL-based anti-jamming strategies for Zero Touch Networks [118]. In their framework, a DRL agent learns a policy to switch communication channels to evade a jammer. To make the agent's decisions transparent, the system feeds the state (the vector of received jamming power across channels), the agent's chosen action (the selected channel), and the resulting reward to an LLM. The LLM is then prompted to generate a textual explanation for that specific decision.

Lesson Learned. From the perspective of wireless networking applications, the reviewed studies on LLM as a generator provide a complementary paradigm that enhances RL by enriching the environment, data samples, and interpretability for decision making. In radio resource management and handover-related tasks, the work in [115] demonstrates that LLMs can transform heterogeneous sensory inputs into structured descriptions of signal propagation conditions, thereby enabling RL agents to make more context-aware decisions in resource allocation and handover. For network orchestration, the study in [118] shows that LLMs can act as policy interpreters to translate RL actions into human-understandable explanations, thereby improving the transparency of orchestration decisions. More broadly, the work in [47] and [117] indicate that LLMs can construct richer simulation environments, which is essential for reducing the sim-to-real gap in RL training. Future research should further recognize the limitations in accurately modeling highly dynamic wireless environments, so that LLMs as generators can ensure precise physical-layer fidelity.

G. Lesson Learned Case Study Organization

The above studies indicate that the integration of LLMs into RL establishes a unified paradigm that enhances wireless network optimization by injecting semantic understanding, structured reasoning, and knowledge-guided decision support into the learning process. Through their roles as state perceiver, reward designer, decision-maker, and generator, LLMs improve different stages of the RL pipeline, leading to more efficient learning, better generalization, and enhanced adaptability. Each role contributes to performance improvement from a distinct perspective, such as improved state representation, more effective reward shaping, or guided decision-making.

Following this observation, this tutorial adopts a role-driven perspective to demonstrate how LLMs enhance RL performance under different functional roles across representative wireless scenarios. Each case study illustrates how incorporating LLMs into a specific component of the RL pipeline leads to measurable performance gains, thereby highlighting the functional advantages of each role and how they can jointly form a coherent and complete LLM-enhanced RL framework. Specifically:

- Section IV demonstrates how LLMs act as state perceivers in vehicular networks, improving state abstraction and enabling more efficient policy learning under dynamic and heterogeneous observations.
- Section V shows how LLMs function as reward designers and generators in low-altitude economy networking, leading to more structured and adaptive reward formulations for complex optimization objectives.
- Section VI illustrates how LLMs serve as decision-makers in space–air–ground integrated networks, providing high-level guidance to improve decision efficiency in large-scale systems.

IV. LLM as State Perceiver for RL in Vehicular Networks

A. Background and Motivation

With the advent of the 6G era, the Internet of Vehicles (IoV) is expected to deliver orders-of-magnitude increases in traffic and connectivity, enabling real-time navigation, perception, and autonomous decision-making at scale [135]. Vehicular networks sit at the core of this vision by linking vehicles and roadside infrastructure, where the proliferation of onboard and roadside sensors creates a continuous stream of multimodal data that must be processed and exchanged under tight latency and bandwidth budgets [96]. DRL can serve the latter by learning policies that continuously reconfigure channel usage, power, and scheduling from feedback [26]. Although traditional DRL is effective for object detection and path planning in static settings, it struggles to sustain accuracy and responsiveness when exposed to fast topology changes, heterogeneous scenes, and concurrent tasks [48]. Thus, it is crucial to develop a scheme that can turn rich sensor feeds into compact and decision-useful states while adapting online to non-stationary wireless conditions.

LLMs can serve the former by extracting salient semantics from images and videos and converting them into structured, text-level representations that are cheaper to transmit and easier to fuse [89], [136]. In this case, LLMs can serve as state perceivers for vehicular networks. Specifically, the LLM operates as a multimodal front end that distills raw sensor streams into compact semantic descriptors for RL state representation. Coupled with DRL, which optimizes actions from these enriched states, this integration links high-level semantic understanding with low-level control, improving sample efficiency, convergence behavior, and end-to-end QoE in dynamic vehicular environments.

B. Case Study: LLM as a State Perceiver for DRL in Quality of Experience (QoE) Maximization

1) System Description: We consider a cellular-based vehicular network deployed in an urban area, where a base station (BS) serves a fleet of embodied-AI vehicles. Association follows the strongest RSSI at the BS [137]. Communication occurs over two link types: V2I links carry high-priority or aggregated reports to the BS, while V2V links enable nearby vehicles to exchange information locally, relieving backhaul load and latency [135]. Each vehicle runs an onboard multimodal front end powered by an LLM to turn camera feeds into structured semantic messages (e.g., objects, counts, relations, and traffic cues). To ensure practical deployment, the LLM inference is performed locally on vehicle hardware or via lightweight models, avoiding frequent remote API calls and reducing communication latency and cost. These messages are compacted by a semantic encoder and then prepared by a channel encoder for over-the-air delivery. Because V2V links may reuse V2I subbands, cochannel interference arises and the effective SINR on each link becomes time-varying. At the receiver, channel and semantic decoders reconstruct the message and its meaning. Reconstruction quality is tracked using cosine similarity between BERT embeddings, complemented by mutual information to gauge how well semantics survive the wireless hop [138].

The system objective is to maximize a QoE metric inspired by the Weber–Fechner law, jointly reflecting semantic fidelity and resource usage [139]. Specifically, the QoE of vehicle i is defined as

$$Q_i = \alpha \log\left(1 + \xi_i\right) - \beta C_i, \tag{6}$$

where ξ_i denotes the semantic reconstruction fidelity measured by the cosine similarity between the transmitted and reconstructed sentence embeddings, and C_i represents the communication cost associated with resource usage (e.g., transmit power and semantic symbol length). The parameters α and β control the relative importance of semantic fidelity and resource consumption. Operational constraints include binary V2V–V2I sharing choices, one-to-one subband assignments, admissible symbol lengths, SINR thresholds for reliable reception, and power bounds for each transmitter. These constraints determine the feasible communication configurations and directly influence both the achievable semantic fidelity ξ_i (as determined by the resulting SINR and symbol allocation) and the associated resource cost C_i (as determined by transmit power and communication overhead). Vehicles act as embodied AI agents that tune communication decisions online. A DRL controller adapts channel reuse, transmit power, and the number of semantic symbols carried per report. The result is a closed-loop stack where LLM-based perception supplies compact, task-aware state to the RL controller, which in turn steers spectrum and payload to sustain high QoE under mobility and interference.

2) Workflow of LLM as State Perceiver for DRL: To tighten the perception–decision loop in vehicular networks, we adopt a vision–language model (LLAVA) [140] that distills raw images into compact, task-aware semantics that feed the DRL controller. The workflow proceeds as follows.

Step 1: Capture and pre-process multimodal observations. Each embodied-AI vehicle collects camera frames $\mathbf{I}_i$ together with basic link/context metadata. Images are normalized and fed to the vision front end so that only the information useful for downstream communication and driving decisions is retained.

Step 2: Visual feature extraction with CLIP-style encoder. LLAVA employs a contrastive language–image pretraining visual encoder $g(\cdot)$ to obtain compact features $\mathbf{Z}_i = g(\mathbf{I}_i)$, which summarize salient scene entities (i.e., vehicles, pedestrians, lanes, and signals) and their coarse relations.

Step 3: Visual–language alignment for semantic generation. The visual features extracted by the encoder are subsequently aligned with the linguistic embedding space through a trainable linear projection, expressed as $\mathbf{E}_i = \mathbf{W}\mathbf{Z}_i$. This mapping establishes a shared representation domain that bridges visual and textual modalities. Building on this alignment, the LLAVA model generates the corresponding semantic representation $\mathbf{M}_i = \text{LLAVA}(\mathbf{I}_i; \theta_i)$, where θ_i denotes the model parameters. The resulting $\mathbf{M}_i$ encapsulates high-level contextual descriptions of the observed scene (such as object identities, spatial relations, and environmental attributes) that serve as compact, information-rich inputs for downstream transmission and decision-making processes.

Step 4: Scenario adaptation via lightweight fine-tuning. To adapt the pretrained LLAVA model to the specific characteristics of real-world driving environments, the model is fine-tuned on vehicular data to minimize semantic mismatch between ground-truth semantics and predictions. This step improves robustness to urban lighting, occlusions, and viewpoint changes while keeping the front end computationally tractable on vehicle/edge platforms.

Step 5: Sentence-level encoding and quality tracking. For stable ingestion by the RL agent, $\mathbf{M}_i$ is encoded by a BERT model $B(\cdot)$ into a sentence-level vector. Reconstruction fidelity along the semantic communication pipeline is monitored by cosine similarity, which later participates in QoE calculation and thus links perception quality to control objectives.

Step 6: State assembly for the DRL backbone. The RL state aggregates (i) semantic descriptors from $B(\mathbf{M}_i)$ (e.g., counts, occupancy relations, and traffic cues) and (ii) communication-side observables, such as subband reuse indicators and interference/SINR statistics. As a result, two components constitute a compact and task-relevant representation that replaces pixel-heavy inputs and exposes high-level

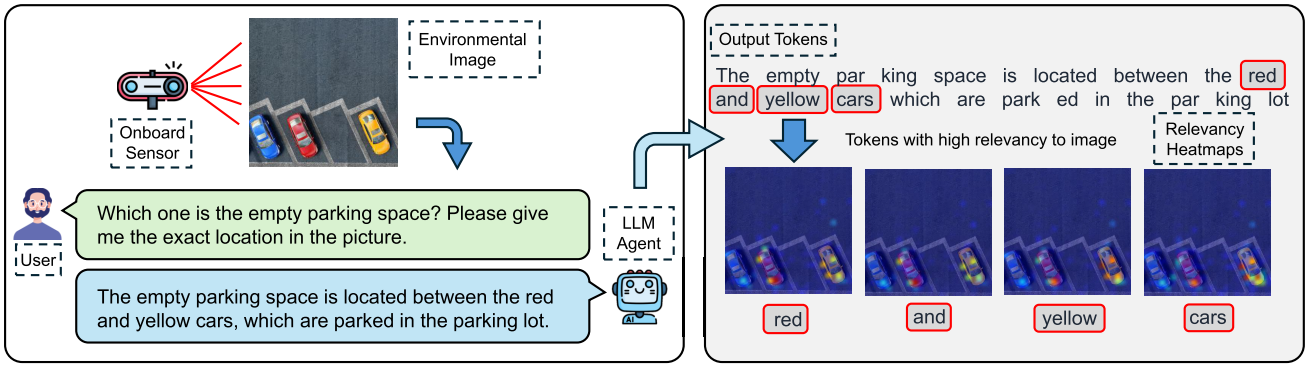

Fig. 6. Illustration of the LLM as state perceiver in an embodied-AI vehicle. LLAVA processes onboard camera images to identify semantic details, such as the empty parking space between the red and yellow cars. The relevancy heatmaps demonstrate accurate cross-modal grounding between visual regions and textual tokens, confirming the model's ability to extract and interpret environment-aware semantic information.

structure to the agent. In addition, these two components operate on different time scales. The semantic descriptors $B(\mathbf{M}_i)$ derived from the LLM-based perception vary relatively slowly, typically when the driving scene undergoes a significant change. Therefore, the LLM module updates the semantic description only at a low frequency. In contrast, the communication-side observables evolve frequently and are fed directly to the RL agent at each time slot. Meanwhile, it avoids requiring LLM inference latency to be smaller than the channel coherence time, since time-critical decisions rely on rapidly updated communication states rather than frequent LLM invocation. Furthermore, the low-frequency invocation of LLM inference significantly reduces computational overhead.

Step 7: DRL-driven communication control. Given the LLM-enhanced state, a PPO agent with generalized advantage estimation selects actions over channel reuse, transmit power, and the number of semantic symbols per report. Decisions are evaluated by a QoE objective inspired by the Weber–Fechner law, balancing semantic fidelity and resource usage under mobility and cochannel interference.

Step 8: Closed-loop update and continual refinement. Rewards derived from QoE and constraint satisfaction (e.g., SINR and similarity thresholds) drive policy updates. In parallel, monitoring signals (e.g., drops in ξ under specific scenes) can trigger periodic fine-tuning of LLAVA on recent samples, keeping the perceiver aligned with operation conditions and sustaining end-to-end convergence and stability.

Overall, the computational complexity of the proposed scheme is influenced by the LLAVA-based perception module and the DRL controller. The LLAVA module relies on a Transformer-based architecture to process multimodal inputs, where the dominant cost arises from self-attention operations over visual tokens and textual embeddings. Thus, its computational overhead increases with the input resolution and the length of semantic representations. The DRL controller is implemented using lightweight neural networks. Its complexity scales with the size of the state and action spaces. In our design, the LLAVA-based semantic perception is updated only when significant scene changes occur, and the DRL controller operates at every time slot for real-time decision making. Thus, the main complexity is governed by the DRL module, which

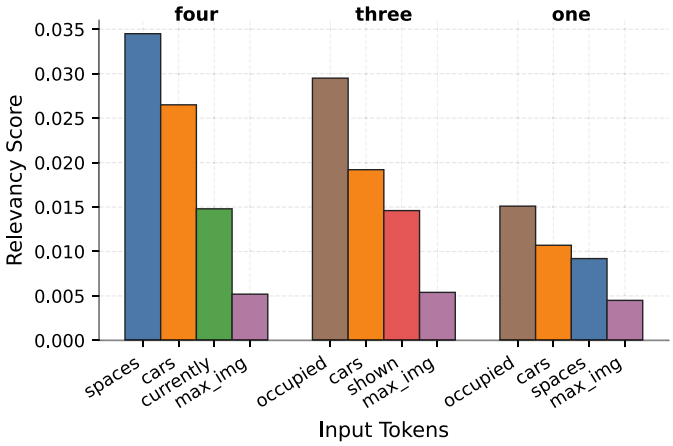

Fig. 7. Input–output text relevancy analysis of LLAVA in semantic perception. The bar charts illustrate the relevancy scores between input tokens and output tokens when LLAVA describes the parking scenario. Tokens such as spaces, occupied, and cars exhibit the highest contribution to generating the output tokens "four", "three", and "one", respectively.

can satisfy the latency and resource constraints of real-world vehicular networks.

3) Numerical Results: As shown in Fig. 7, LLAVA effectively identifies critical contextual cues such as the number of available parking spaces, their occupancy status, and the spatial relationship between vehicles. In this scenario, the system correctly infers that four parking spaces are present, three are occupied, and one remains vacant. Compared to transmitting the original image (approximately 614 KB), conveying the extracted textual description (around 12.1 KB) reduces the transmission bandwidth by nearly 98%, demonstrating the advantage of semantic-level communication in bandwidth-constrained vehicular networks.

As shown in Fig. 8, we consider three baseline schemes, including (i) a pure RL method based on PPO without LLM assistance, (ii) a DDPG-based RL method, and (iii) a random policy. The proposed scheme is based on PPO with LLM enhancement, which consistently achieves the highest QoE across all network scales. For all schemes, we adopt standard and identical hyperparameter settings without task-specific

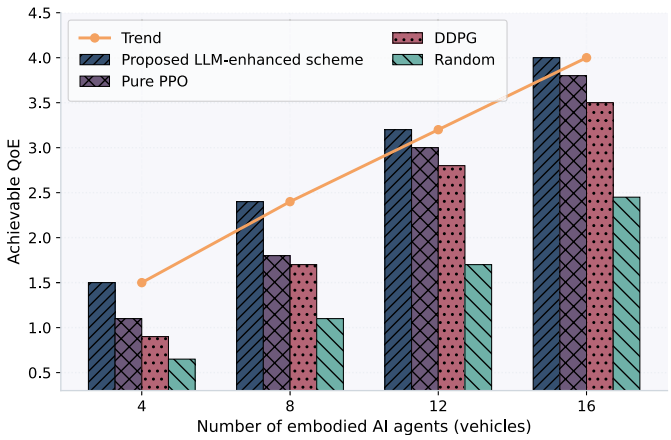

Fig. 8. Achievable QoE versus the number of LLM agents.

fine-tuning to ensure a fair comparison. The system exhibits a 36% QoE improvement over DDPG at eight vehicles and maintains steady scalability with 61.4%, 31.9%, and 25.2% incremental gains when the number of vehicles increases from 4 to 8, 8 to 12, and 12 to 16, respectively. These results highlight that the LLM-enhanced state representation contributes directly to the observed performance improvement, supporting faster learning and higher-quality decision making in large-scale vehicular networks.

Lesson Learned. Integrating LLM-enhanced RL into vehicular networks demonstrates that semantic understanding can fundamentally reshape communication and control in bandwidth-limited environments. By extracting and transmitting high-level scene semantics rather than raw sensory data, LLMs reduce communication overhead while preserving decision-critical information. This integration also illustrates the promise of hierarchical cooperation in LLM-enhanced RL for vehicular networks: LLMs provide semantic abstraction and intent reasoning, while RL agents formulate fine-grained decision optimization. Future work should prioritize lightweight multimodal compression, semantic calibration, and reliability assessment to ensure that LLM-enhanced RL can achieve scalable, robust, and trustworthy vehicular intelligence.

V. LLM as Reward Designer and Generator for RL in Low-Altitude Economy Networking

A. Background and Motivation

Low-altitude economy networking (LAENet) integrates communication and computation infrastructure to support the operation of manned and uncrewed aerial vehicles (UAVs) within airspace below 1,000 meters [141]. The primary vision of LAENet is to unlock economy and societal value through the flexible deployment of aerial platforms across various verticals, including intelligent transportation systems, emergency services, and ubiquitous wireless coverage [142]. Despite these advantages, the LAENet faces several complex challenges. First, the need for real-time decision-making in UAV control and multi-agent coordination requires high responsiveness under dynamic conditions [143]. Second, the wireless environment is subject to rapid changes due to user mobility and fluctuating channel conditions, making it difficult to maintain consistent connectivity [144]. In addition, the LAENet often operates under constraints related to energy, bandwidth, and computational power for resource-limited UAVs and edge devices [145].

To tackle these complex challenges, many existing studies have shown that RL offers a promising approach for LAENet, enabling autonomous, model-free, and adaptive control. By interacting with dynamic environments, RL can learn robust and time-sensitive decision policies to efficiently manage limited energy, bandwidth, and computational resources [10], [141]. Nevertheless, the reward function design of classical RL relies on manual specification, where some poorly designed signals can lead to inefficient behaviors and misguide learning for RL agent [141], [146]. LLMs possess strong capabilities in semantic understanding and domain knowledge integration, enabling them to design reward functions for RL agents in LAENet optimization. Meanwhile, LLMs can also be used to generate samples aligned with the LAENet scenario to pre-evaluate the quality of the reward design, thereby avoiding the application of suboptimal designs in RL. In the next part, we provide a case study to demonstrate how to use LLMs as both reward designers and generators for LAENet.

B. Case Study: Enhancing DRL for Energy Optimization in LAENet Through LLM as Reward Designer and Generator

1) System Description: The case study focuses on the LAENet scenario that consists of multiple IoT devices and a macro base station (MBS) for uplink communication. As illustrated in Fig. 9, a UAV serves as a mobile agent tasked with collecting data from distributed Internet of Things (IoT) terminals. These terminals include environmental sensors, health monitors, and surveillance cameras, all deployed across a target area requiring efficient aerial coverage. The UAV's role is to gather sensory data and relay it back to the MBS, while operating under constraints such as limited onboard energy and time-sensitive service requirements. The UAV energy consumption is modeled by jointly considering propulsion energy induced by trajectory movements and hovering, as well as communication-related energy for data collection and transmission. The DRL agent governs the UAV's behavior via a policy controller. At each decision time step, the UAV observes the system state (such as terminal locations, data freshness, and energy levels), selects an action accordingly (e.g., trajectory updates and hovering decisions), and receives feedback of reward from the environment. In the following, we propose utilizing the LLM to design a reward function to enhance the environment feedback for RL to address the energy consumption optimization problem.

2) Workflow of Using an LLM as a Reward Designer and Generator for DRL: The design of reward functions is a critical factor in determining the convergence speed and performance of DRL agents. To enhance flexibility and contextual alignment, we adopt an LLM-enhanced reward design workflow for energy optimization in LAENet. The workflow comprises several key steps as follows:

Step 1: Constructing Effective Prompts for the LLM. To enable the LLM to function as a reliable reward designer, the

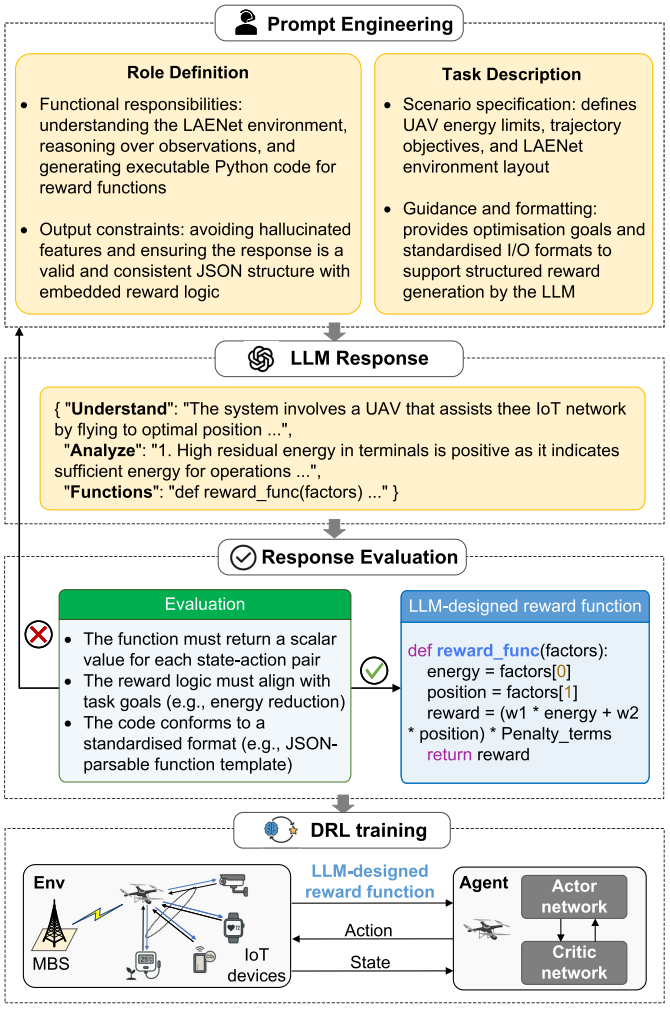

Fig. 9. LLM-assisted reward design framework for DRL in the LAENet. The framework guides LLMs to generate executable reward functions via prompt engineering. Validated outputs are used to train a DRL agent for UAV energy optimization in low-altitude IoT networks. The prompt templates and example generated code are available at: https://github.com/ResearcherSG/Tutorial-on-Large-Language-Model-Enhanced-Reinforcement-Learning-for- Wireless-Networks.

user needs to formulate a structured prompt, which comprises two essential components:

- Role Definition: The LLM is explicitly instructed to act as a reward function generator for the RL agent, including understanding the LAENet environment, reasoning based on system observations, and producing executable Python code. Constraints are included to prevent hallucination and to maintain consistency, such as the LLM is instructed to invent features and to output a valid JSON response containing the reward logic.
- Task Description: The prompt defines the UAV's operating scenario in LAENet, including energy constraints, trajectory objectives, and environmental layout. The description offers specific guidance on optimization goals (e.g., minimizing propulsion energy while maintaining coverage), allowing the LLM to identify and prioritize meaningful reward factors accordingly. Standardized input and output formats are included to guide structured generation.

Step 2: Generating Candidate Reward Functions via LLM. The LLM is invoked in an offline manner during the reward design stage, rather than being embedded into the online RL training loop. Therefore, LLM inference can be performed on servers prior to UAV deployment, avoiding any additional latency during real-time UAV operation. This design eliminates the need for frequent API calls, and the cost of calling the API is negligible even in offline designs. Once the prompt is issued, the LLM generates candidate reward functions by combining its logical reasoning abilities with code generation capabilities. Each output includes both a textual explanation and an executable function (e.g., Python code). Due to the stochastic nature of LLM outputs, responses may vary even under similar prompts. Thus, it is necessary to validate the generated reward functions before integrating them into the DRL. Additionally, structured evaluation is essential since LLMs may occasionally produce syntactically correct but semantically invalid outputs.

Step 3: Evaluating LLM Responses for Validity and Consistency. Rather than relying on a single output, the proposed scheme prompts the LLM multiple times to generate a diverse set of candidate reward functions. Then, these functions are evaluated against a set of predefined constraints:

- The function must return a scalar value for each state–action pair. If a hallucination occurs and the LLM outputs a list or dictionary (e.g., `return [energy_consumption, position_score]`), the training process will fail since DRL algorithms require a scalar reward at each step.
- The reward logic must align with task goals (e.g., energy reduction). For example, the LLM may produce a reward function that unintentionally encourages undesirable behavior, such as $reward = -energy\_consumption + w*(distance\_to\_centroid)$. Although minimizing energy consumption is the objective, this formulation may incentivize the UAV to move away from the centroid to maximize the positive term, thereby contradicting the original optimization goal. Such hallucinated logic needs to be filtered out.
- The code conforms to a standardised format (e.g., JSON-parsable function template). If the LLM output references undefined variables that are not present in the state space (e.g., battery level), or produces structures that cannot be serialized into JSON, the output will not pass validation.

Only candidate functions that pass all checks are retained for training the RL agent.

Step 4: Using the LLM as a Generator to Evaluate Reward Design. The LLM constructs a semantic proxy of the LAENet environment by generating synthetic samples (i.e., state s, action a and reward $R(\cdot)$) that approximate realistic state–action–reward relationships. This process allows *a-priori* evaluation of the reward functions without deploying an actual policy or running RL episodes, effectively reducing the sampling cost and avoiding unnecessary environment interactions. For each candidate reward function $R(\cdot)$, we

numerically compute the Lipschitz constant $\mathcal{L}$ over the state space $\mathcal{S}$ as

$$\mathcal{L} = \sup_{s_1, s_2 \in \mathcal{S}, s_1 \neq s_2} \frac{\|R(s_1) - R(s_2)\|_2}{\|s_1 - s_2\|_2}. \tag{7}$$

A smaller Lipschitz constant implies that the reward landscape varies more smoothly with respect to action perturbations [147], leading to better training stability and faster convergence of the DRL agent. To enable iterative refinement for the reward design, the evaluation results are fed back to the LLM as additional prompt inputs in subsequent rounds. Specifically, the updated prompt includes: (i) the estimated Lipschitz constant $\mathcal{L}$ as a quantitative indicator of reward smoothness, (ii) diagnostic feedback on problematic behaviors (e.g., which state dimensions contribute most to the Lipschitz constant), and (iii) suggestions for refinement such as reweighting terms or removing redundant components. This feedback allows the LLM to revise the reward structure in a guided manner, rather than regenerating functions from scratch. Following this principle, the LLM-generated reward functions are ranked according to their estimated Lipschitz constants, and the one with the lowest $\mathcal{L}$ is selected as the final reward design for integration into the RL training process. Since this evaluation is conducted offline, the computational energy cost of LLM inference is incurred only once during the design phase and does not affect the runtime energy efficiency of the UAV.

Step 5: Integrating Expanded Reward Factors for UAV Energy Optimization. One of the advantages of LLM-designed reward functions is the ability to integrate additional reward components beyond conventional formulations. In the implementation, the manually defined reward is usually designed in the form of reward = $w \times$ energy $\times$ Penalty [148]. After using the LLM to design the reward function, a richer reward structure can be inferred and generated by reward = $\left(w_1 \times \text{energy} + w_2 \times \text{position\_score}\right) \times$ Penalty, where position_score reflects how close the UAV is to the centroid of the IoT terminal distribution. This term encourages the UAV to maintain a central position, reducing both travel distance and hovering time. Such enhancements lead to more effective energy use by combining path planning and transmission efficiency via comprehensive and unified reward signals. Once the LLM-designed reward function is finalized in the offline design stage,[7] it is integrated into the UAV's DRL training process.

3) Numerical Results: Fig. 10 presents the training performance of two representative DRL algorithms (i.e., DDPG and TD3) under manually designed rewards and LLM-generated rewards. To ensure a fair comparison, all configurations (including training settings and hyperparameters) are kept consistent for the same type of RL algorithm (i.e., DDPG or TD3) across different reward schemes. For example, the baseline "DDPG, Manual Reward" and the proposed scheme "DDPG, LLM-Designed Reward" are identical except for the reward function. The evaluation metric is the total energy consumption of the UAV across training episodes. From the results, it is evident that incorporating LLM-designed reward functions leads to improvements in energy efficiency across two algorithms. The TD3 agent equipped with the LLM-derived reward achieves the lowest final energy consumption, showing a reduction of up to 6.8% compared to its manually designed counterpart. The performance gain stems from the LLM's ability to generate reward structures that better reflect the optimization objectives of LAENet. By reasoning over UAV dynamics, positional relevance, and energy constraints, the LLM formulates reward signals that implicitly guide the agent toward shorter flight paths, reduced hovering time, and more efficient communication scheduling. Thus, the simulation results demonstrate the effectiveness of LLM-assisted reward design in enhancing the decision-making capabilities of DRL agents for the LAENet.

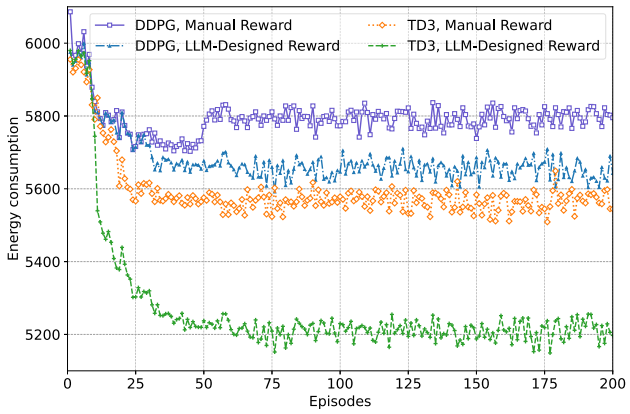

Fig. 10. Energy consumption across training episodes using manual and LLM-designed reward functions in the LAENet. The results compare DDPG and TD3 algorithms under two reward settings. LLM-designed reward functions lead to lower and more stable energy consumption.

Lesson Learned. The case study demonstrates that serving LLMs as the reward designer and generator can improve the efficiency and stability of UAV energy optimization. The effectiveness of this approach depends on grounding LLM reasoning in domain knowledge of low-altitude communication and mobility, ensuring that the generated reward function aligns learning objectives with mission-level constraints. Meanwhile, the LLM synthesizes state–reward samples to further accelerate and ensure the stability and feasibility of reward design. Future research can leverage LLMs' abilities to interpret heterogeneous data streams and reason over sequential context, allowing RL agents to translate multimodal data (such as raw imagery, sensory data, and mission updates) into comprehensive state representations for RL. Moreover, step-by-step reasoning mechanisms such as CoT prompting can be further explored to infer mission intent, operational constraints, and decision making for optimization in the LAENet.

VI. LLM as Decision-Maker for RL in Space-Air-Ground Integrated Network

A. Background and Motivation

Space–air–ground integrated network (SAGIN) is a next-generation architecture that couples spaceborne, airborne, and

[7]The LLM is only used prior to DRL training; therefore, its computational complexity is not considered. Moreover, the DRL component adopts standard DDPG and TD3 algorithms and its computational complexity is omitted for brevity.

terrestrial segments into a unified system for wide-area connectivity [19], [149]. In this paradigm, low Earth orbit (LEO) constellations furnish global coverage and low-latency backhaul, high-altitude platforms (HAPs) at stratospheric altitudes operate as quasi-stationary relays and traffic aggregators, while ground infrastructures provide access and edge computation [91]. A hybrid free-space optical (FSO)/radio frequency (RF) stack naturally aligns with this multi-tier design, where FSO links are suitable for satellite-to-HAP segments due to high bandwidth and license-free spectrum at high altitudes, whereas RF links offer robust HAP-to-ground access under cloud, fog, and precipitation [19], [150]. Despite these advantages, SAGIN presents tightly coupled design challenges. First, the fast orbital motion of LEO satellites continuously reshapes the topology, making handover planning and inter-segment continuity a critical issue. Second, heterogeneous FSO and RF links introduce cross-tier dependencies in resource allocation. Decisions on the satellite–HAP segment affect the feasible operating region of the HAP–ground segment. Third, although HAPs are quasi-stationary, slow drifts caused by changing environmental conditions perturb link geometry and channel statistics. These effects lead to non-convex objectives, time-coupled constraints, and a large combinatorial action space, where handover, user selection, and subcarrier/power scheduling must be coordinated across tiers [29].

To cope with these dynamics, many studies leverage RL as a model-free controller that adapts online to mobility and channel fluctuations [9], [42], [45]. Nevertheless, classical RL still faces practical limitations in SAGIN such as limited generalization to unseen traffic/topology regimes and brittle performance under non-stationary channels when training with fixed hyperparameters [151], [152]. These issues are amplified by the hybrid FSO/RF coupling and the visibility-constrained satellite selection space. Recent advances in LLMs provide a transformative opportunity to enhance DRL in SAGIN. An LLM can function as a decision guider that oversees and guides the learning dynamics of the DRL agent [59], [74]. Leveraging reasoning and contextual understanding capabilities, the LLM can interpret training feedback such as reward trajectories, convergence patterns, and exploration progress to infer how the learning process should adapt over time. This integration paradigm enables DRL to combine low-level environmental interaction with high-level adaptive oversight. Building on this perspective, the following subsection presents representative case studies of LLM as decision-maker for SAGIN.

B. Case Study: LLM as a Decision Guider for DRL in Downlink Throughput Maximization

1) System Description: As shown in Fig. 11, the considered system consists of a three-tier space–air–ground downlink architecture, where a constellation of LEO satellites provides high-capacity backhaul to a HAP hovering in the stratosphere, which in turn distributes data to multiple ground user clusters. The system adopts a hybrid FSO and RF communication framework: the satellite–to–HAP link uses FSO for its large bandwidth and interference-free propagation above the clouds,

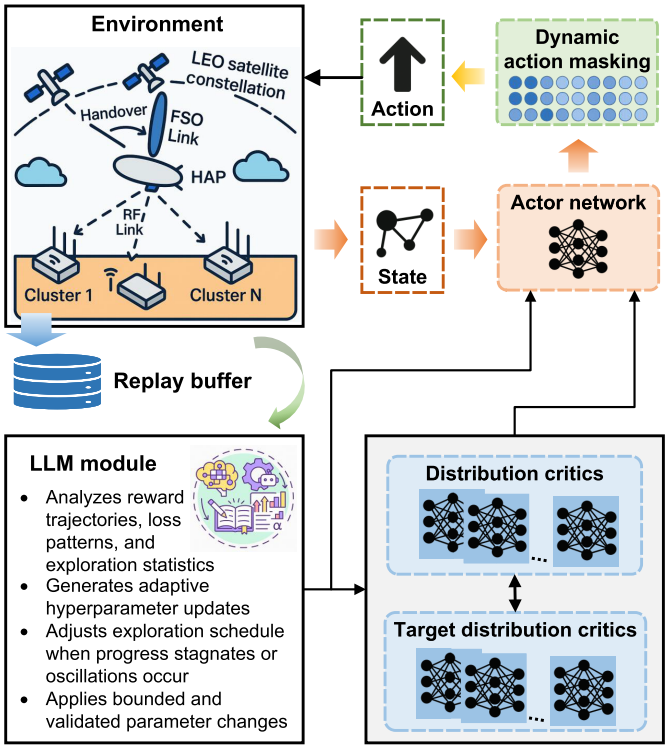

Fig. 11. Framework of the proposed LLM-guided TQC for hybrid satellite downlink optimization. The LLM module serves as a meta-controller that analyzes learning behavior and adaptively tunes hyperparameters to stabilize and accelerate training.

while the HAP–to–ground segment employs RF transmission to maintain reliable connections under variable weather conditions. Within each time slot, the network dynamically selects one visible LEO satellite to establish the FSO backhaul and simultaneously allocates limited RF subcarriers among ground clusters according to real-time channel quality. The optimization goal is to achieve joint coordination between satellite handover and network resource allocation, i.e., to maximize the overall downlink throughput from space to ground while minimizing frequent satellite handovers that can cause service interruptions and control overhead.

2) Workflow of Using LLM as a Decision Guider for DRL: To make RL robust under the non-stationary and hybrid FSO/RF conditions of a space–air–ground system, we employ the LLM *above* as the guider for the RL agent in policy learning. It periodically inspects training information and decides *how the guider should learn next*. The workflow is organized into the following steps.

Step 1: Define the LLM's role, guardrails and training information. Configure the LLM to act as a decision guider over the DRL backbone (TQC).[8] Its scope is restricted to learning dynamics (such as stability, progress, exploration), not task actions. We whitelist adjustable hyperparameters (e.g., learning rate, entropy/temperature α, truncation level k

[8]TQC [153] is an advanced variant of SAC based on distributional RL. Instead of estimating the expected Q-value, TQC models the full return distribution using multiple quantile critics and mitigates overestimation bias by truncating the highest quantiles during value updates, thereby achieving more stable and reliable policy learning in high-variance environments.

in TQC, batch size, target-network update rate, exploration decay) and set intervention cadence and safety constraints (i.e., min/max ranges, maximum step sizes). On a fixed schedule, the training loop assembles a structured information package including recent reward trajectory, actor/critic loss traces, entropy/exploration statistics, and a normalized training-progress indicator. The LLM is not invoked at every training step, but only at predefined intervals, which avoids introducing additional latency into the real-time decision loop.

Step 2: Prune the action space with dynamic action masking. A deterministic feasibility layer masks out invalid actions (e.g., satellites that are not currently visible), which ensures compute is focused on meaningful choices and that any exploration encouraged by the LLM happens within the feasible set.

Step 3: Calibrate exploration with an interpretable schedule. To avoid both aimless exploration and premature exploitation, the agent follows an episode-dependent exploration schedule:

$$\epsilon(e) = \max\left(\epsilon_0\left(1 - \frac{e}{e_{\text{decay}} \cdot E}\right), 0\right), \tag{8}$$

where ϵ_0 is the initial noise level, e is the current episode, E is the total episode budget, and e_{decay} sets how quickly exploration tapers off. The LLM may later suggest gentle, bounded adjustments to this schedule (e.g., slowing the decay when progress plateaus).

Step 4: Prompt the LLM for adjustments. At each intervention, the system presents a structured summary including current hyperparameters, recent rewards, and progress marker. The LLM outputs a set of small and coordinated hyperparameter adjustments that are consistent with the diagnosed learning phase:

$$\Theta_{e+\Delta e} = \mathcal{F}_{\text{LLM}}(\Theta_e,\ \mathcal{H}_e,\ \mathcal{P}_e), \tag{9}$$

where Θ_e is the current hyperparameter set enhanced by an LLM $\mathcal{F}_{\text{LLM}}$, $\mathcal{H}_e$ summarizes recent rewards, and $\mathcal{P}_e$ encodes training progress. For instance, when oscillations are detected in the reward curve, the LLM suggests reducing the learning rate to stabilize convergence; conversely, when training enters a plateau phase, it recommends increasing the entropy coefficient to enhance exploration. In practice, the LLM is invoked at a low frequency (e.g., once every tens of episodes). Each inference cost depends on the model size and deployment setting, and the corresponding API cost is negligible compared to the overall training cost of DRL.

Step 5: Maintain Effective Adaptation and Continuous Feedback. Before deployment, a local validator clamps every suggested change into certified ranges and rejects out-of-scope edits. Rate-limit rules cap per-interval movement to avoid instability (e.g., no large jumps in learning rate or α). Validated updates are applied for the next window. Fresh training information reflects the consequence of prior advice and becomes the next input to the LLM. This closes a feedback loop where the LLM continually refines the learning process in response to the non-stationary environment characterized by time-varying channel conditions, satellite mobility, and HAP dynamics.

Step 6: Log rationales and handle failure modes. Each LLM recommendation is stored with a short rationale (e.g., oscillatory rewards mean the lower step size, slightly increase entropy). If performance degrades beyond a tolerance, an automatic rollback restores the last stable configuration; the next prompt asks the LLM for a corrective countermeasure.

The computational complexity of the proposed framework is mainly determined by the DRL backbone and the LLM-based decision guider. The DRL component complexity scales with the structure of the actor–critic networks and the number of satellites, user clusters, and resource allocation variables in SAGIN. The LLM, serving as a decision guider, operates intermittently (e.g., every several episodes), and its inference cost depends on the size of the adopted model and the length of the input prompt. Such a split architecture preserves the benefit of adaptive hyperparameter tuning while meeting the latency constraints of SAGIN.

3) Numerical Results: For comparison, we consider TQC without LLM assistance as the primary baseline, where both methods share the same network architecture, training settings, and hyperparameter configurations, except for the LLM-based decision guidance. Numerical results show that, compared to the classical TQC, the proposed LLM-guided TQC achieves approximately 1.5% to 2% higher steady-state rewards. In addition, the proposed method converges within fewer than 400 episodes, corresponding to an improvement of approximately 13.73% in convergence speed, while the baseline TQC exhibits slower stabilization and larger reward fluctuations. These results verify that incorporating an LLM for decision guiding effectively enhances learning stability and adaptability in dynamic satellite communication environments.

Lesson Learned. The application of LLM-enhanced RL in SAGIN achieves improved coordination across heterogeneous links by utilizing the LLM as a decision guider to adapt hyperparameters and exploration schedules of RL. Such oversight is achieved through the LLM's capability for reasoning over sequential patterns and interpreting training feedback, enabling faster convergence for the RL agent in policy learning. However, it is important to note that when LLMs are used to enhance RL through parameter tuning, the embedding of expert-level domain prior knowledge is essential to ensure that the tuning decisions are meaningful and robust. Therefore, developing domain-specific LLMs trained on communication and network optimization corpora (beyond SAGIN) represents a promising research direction.

VII. Future Research Directions

Despite the promising progress of integrating LLMs into RL for wireless networks, this emerging paradigm remains in its infancy. The fusion of linguistic cognition and autonomous decision-making opens a wide spectrum of opportunities that demand deeper theoretical insights, efficient system design, and reliable deployment practices. In this section, we outline several promising future research directions on the LLM-enhanced RL.

A. Theoretical Foundations and Performance Guarantees

While LLM-enhanced RL has demonstrated remarkable adaptability and generalization, its theoretical underpinnings remain largely unexplored. This is primarily because the LLM is often treated as a closed-box component [136], making it unclear how LLMs theoretically influence the reinforcement learning process. Future research needs to bridge this gap by developing rigorous mathematical tools to analyze how semantic reasoning introduced by LLMs affects the convergence, stability, and optimality of RL [60]. Promising directions include (i) quantifying the sensitivity of LLM-generated states and rewards, and (ii) characterizing the uncertainty arising from LLM reasoning or hallucinations to ensure stable policy updates. Such theoretical progress is essential for transforming LLM-enhanced RL from an empirical closed-box integration into a principled and verifiable learning paradigm [146].

B. Lightweight and Edge-Deployable LLM-Enhanced RL Architectures

The computation-intensive nature of LLMs poses challenges for their real-time integration with RL in wireless systems [5]. Future research should pursue lightweight, modular, and domain-adaptive architectures that can retain the semantic reasoning capabilities of LLMs while satisfying strict energy, latency, and storage constraints [59]. Feasible research directions include knowledge distillation [129] and parameter-efficient fine-tuning [95] to compress foundation models for efficient inference. It is also possible to consider mechanisms for model invocation and control loops across timescales [154]. For instance, LLM reasoning can be triggered only when significant context changes occur (e.g., topology shifts and abnormal conditions), while fast-timescale decision loops are maintained using lightweight policies. In addition, edge–cloud collaborative inference frameworks can be developed to offload heavy semantic reasoning to nearby edge servers while retaining fast decision loops locally [71]. Thus, a key issue is how to optimize the trade-off among semantic gain, inference cost, and communication overhead in wireless environments.

C. Security and Trustworthy LLM-Enhanced RL Systems

Security assurance becomes paramount as LLMs participate in network control and resource allocation within wireless systems [2], [155]. Potential risks such as prompt injection, model hallucination, and adversarial manipulation may lead to catastrophic misallocations or service disruptions [59], [97], [118]. Future work should focus on establishing trustworthy LLM-enhanced RL pipelines. For example, designing robust prompt architectures and adversarial defense mechanisms can help resist malicious or misleading inputs [42]. Abnormal responses caused by hallucinations during the RL agent's policy execution can be avoided by incorporating consistency verification mechanisms [8]. In addition to external checks, intrinsic proof mechanisms can also be considered, where a verification signal is embedded directly into the model's parameters to validate the outputs [156].

D. Multi-Agent and Collaborative LLM-RL Coordination

Next-generation wireless systems involve multiple agents (such as UAVs, vehicles, base stations, and edge devices) to achieve collaborative intelligence [9], [157]. Extending LLM-enhanced RL to MARL or collaborative learning scenarios introduces new research challenges [74]. For example, using a single type of LLM to interpret, negotiate, and summarize the intents of distributed agents may face cognitive limitations [90]. For example, one LLM agent could handle knowledge retrieval from a network database, and another could manage constraint satisfaction (e.g., ensuring QoS requirements are met). How can communication efficiency and policy consistency be ensured when multiple LLMs operate concurrently under bandwidth constraints [66]? Therefore, integrating multiple types of LLMs into RL agents responsible for different tasks to construct a collaborative decision-making framework represents a novel research direction. Promising solutions include establishing standardized communication protocols among LLM agents to ensure interoperability and scalability [158].

E. Wireless Network-Specific LLM Pretraining for RL

Generic foundation models trained on web-scale corpora may lack awareness of the physical, statistical, and semantic characteristics of wireless environments [2]. Hence, a key frontier lies in developing domain-specialized LLMs for wireless networks [159]. These LLMs can be pretrained on multimodal datasets including communication logs, signal features, network management commands, and 3GPP specifications [46], [85], [116]. Significant progress has been made in research related to domain-specific LLMs for wireless networks. For example, TelecomGPT demonstrates that general-purpose LLMs require continual pretraining, instruction tuning, and alignment tuning on telecom-specific corpora to achieve reliable performance [108]. Moreover, recent work on 6G-native LLMs further highlights the design of domain-specific LLMs to capture fine-grained physical-layer behaviors and wireless system dynamics [109]. Thus, complex tasks such as massive MIMO precoding and channel-aware resource allocation may increasingly rely on vertical Telecom-LLMs. Such models are expected to achieve more precise reasoning over symbolic and numerical information in wireless environments, thereby enabling more accurate and reliable integration with RL for policy optimization.

F. Beyond LLMs: Generative and Agentic Intelligence for RL

Other generative paradigms, such as diffusion models [17], have recently been introduced into RL to model trajectory distributions and synthesize action sequences. They can generate feasible behaviors and handle multimodal policy distributions, particularly in offline RL and continuous control tasks [160]. Compared to diffusion models, LLMs provide cognitive capabilities such as intent understanding and semantic interpretation. For example, LLMs can be used to specify high-level network objectives (e.g., QoE targets), while diffusion models can be used to synthesize control actions such as resource allocation. Effectively exploiting these

complementary paradigms for wireless network optimization remains a challenging task. Beyond generative modeling, a more forward-looking direction is that LLMs evolve from passive perception modules into active decision-making agents, i.e., agentic AI [161]. In such architectures, LLMs can participate in planning, tool invocation, memory management, and high-level policy decomposition, while RL modules handle low-level control and adaptation [162]. Thus, achieving scalable coordination among multiple AI agents with limited communication bandwidth poses a critical challenge for wireless systems.

VIII. Conclusion

This tutorial has provided a comprehensive overview of the emerging paradigm of LLM-enhanced RL for wireless networks. By exploring how LLMs can be incorporated into the RL pipeline, we have shown LLM's superior abilities to overcome challenges of the classical RL. We propose the taxonomy, which covers the roles of LLMs as state perceiver, reward designer, decision-maker, and generator for RL agent, and offers the latest overview and unique insights into LLM-enhanced RL in wireless systems. Through multiple case studies in the LAENet, vehicular networks, and SAGIN, we demonstrated that LLM-enhanced RL can significantly improve adaptability, stability, and efficiency in dynamic wireless environments. Looking forward, several open challenges remain to be addressed, which will be key issues to realize the full potential of LLM-enhanced RL in wireless networks.

References

[1] J. Achiam et al., "GPT-4 technical report," 2023, *arXiv:2303.08774*.

[2] Y. Huang et al., "Large language models for networking: Applications, enabling techniques, and challenges," *IEEE Netw.*, vol. 39, no. 1, pp. 235–242, Jan. 2025.

[3] R. Zhang et al., "Generative AI agents with large language model for satellite networks via a mixture of experts transmission," *IEEE J. Sel. Areas Commun.*, vol. 42, no. 12, pp. 3581–3596, Dec. 2024.

[4] J. Shen, N. Cheng, W. Xu, H. Wang, Y. guo, and J. Xu, "LLM at network edge: A layer-wise efficient federated fine-tuning approach," in *Proc. 39th Annu. Conf. Neural Inf. Process. Syst.*, vol. 38, 2025, pp. 104922–104950. [Online]. Available: https://openreview.net/forum?id=DqRbfiTdKK

[5] J. Hoffmann et al., "Training compute-optimal large language models," in *Proc. Adv. Neural Inf. Process. Syst.* 35. Red Hook, NY, USA: Curran Associates Inc., 2022, pp. 30016–30030.

[6] G. Sun et al., "Large language model (LLM)-enabled graphs in dynamic networking," *IEEE Netw.*, vol. 39, no. 4, pp. 290–301, Jul. 2025.

[7] A. Madaan et al., "Self-refine: Iterative refinement with self-feedback," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., Red Hook, NY, USA: Curran Associates, 2023, pp. 46534–46594. [Online]. Available: https://proceedings.neurips.cc/paperfiles/paper/2023/file/91edff07232fb1b55a505a9e9f6c0ff3-Paper-Conference.pdf

[8] L. Cai et al., "Large language model-enhanced reinforcement learning for low-altitude economy networking," 2025, *arXiv:2505.21045*.

[9] A. Feriani and E. Hossain, "Single and multi-agent deep reinforcement learning for AI-enabled wireless networks: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 2, pp. 1226–1252, 2nd Quart., 2021.

[10] Y. Bai et al., "Toward autonomous multi-UAV wireless network: A survey of reinforcement learning-based approaches," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 4, pp. 3038–3067, 2nd Quart., 2023.

[11] R. Zhang, K. Xiong, W. Guo, X. Yang, P. Fan, and K. B. Letaief, "Q-learning-based adaptive power control in wireless RF energy harvesting heterogeneous networks," *IEEE Syst. J.*, vol. 15, no. 2, pp. 1861–1872, Jun. 2021.

[12] M. Ji et al., "Graph neural networks and deep reinforcement learning-based resource allocation for V2X communications," *IEEE Internet Things J.*, vol. 12, no. 4, pp. 3613–3628, Feb. 2025.

[13] N. Cheng, X. Wang, Z. Li, Z. Yin, T. H. Luan, and X. Shen, "Toward enhanced reinforcement learning-based resource management via digital twin: Opportunities, applications, and challenges," *IEEE Netw.*, vol. 39, no. 1, pp. 189–196, Jan. 2025.

[14] H. He, W. Yuan, S. Chen, X. Jiang, F. Yang, and J. Yang, "Deep reinforcement learning-based distributed 3D UAV trajectory design," *IEEE Trans. Commun.*, vol. 72, no. 6, pp. 3736–3751, Jun. 2024.

[15] R. Zhang, K. Xiong, Y. Lu, B. Gao, P. Fan, and K. B. Letaief, "Joint coordinated beamforming and power splitting ratio optimization in MU-MISO SWIPT-enabled HetNets: A multi-agent DDQN-based approach," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 2, pp. 677–693, Feb. 2022.

[16] W. Zhao et al., "A survey on DRL-based UAV communications and networking: DRL fundamentals, applications and implementations," *IEEE Commun. Surveys Tuts.*, vol. 28, pp. 3911–3941, 2026.

[17] H. Du et al., "Reinforcement learning with LLMs interaction for distributed diffusion model services," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 47, no. 10, pp. 8838–8855, Oct. 2025.

[18] H. Luo et al., "Toward edge general intelligence with multiple-large language model (multi-LLM): Architecture, trust, and orchestration," *IEEE Trans. Cognit. Commun. Netw.*, vol. 11, no. 6, pp. 3563–3585, Dec. 2025.

[19] R. Zhang et al., "Generative AI for space-air-ground integrated networks," *IEEE Wireless Commun.*, vol. 31, no. 6, pp. 10–20, Dec. 2024.

[20] J. Xue, K. Yu, T. Zhang, H. Zhou, L. Zhao, and X. Shen, "Cooperative deep reinforcement learning enabled power allocation for packet duplication URLLC in multi-connectivity vehicular networks," *IEEE Trans. Mobile Comput.*, vol. 23, no. 8, pp. 8143–8157, Aug. 2024.

[21] T. Xie et al., "Text2Reward: Reward shaping with language models for reinforcement learning," in *Proc. 12th Int. Conf. Learn. Represent.*, vol. 2024, 2024, pp. 35663–35699. [Online]. Available: https://openreview.net/forum?id=tUM39YTRxH

[22] Y. J. Ma et al., "Eureka: Human-level reward design via coding large language models," in *Proc. 12th Int. Conf. Learn. Represent.*, vol. 2024, 2024, pp. 26516–26560. [Online]. Available: https://openreview.net/forum?id=IEduRUO55F

[23] R. Zhang et al., "Interactive AI with retrieval-augmented generation for next generation networking," *IEEE Netw.*, vol. 38, no. 6, pp. 414–424, Nov. 2024.

[24] Y. Cao et al., "Survey on large language model-enhanced reinforcement learning: Concept, taxonomy, and methods," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 36, no. 6, pp. 9737–9757, Jun. 2025.

[25] Y. Guo et al., "Secrecy energy efficiency maximization in IRS-assisted VLC MISO networks with RSMA: A DS-PPO approach," *IEEE Trans. Wireless Commun.*, vol. 24, no. 8, pp. 6475–6489, Aug. 2025.

[26] N. C. Luong et al., "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3133–3174, 4th Quart., 2019.

[27] Y. Chen et al., "Reinforcement learning meets wireless networks: A layering perspective," *IEEE Internet Things J.*, vol. 8, no. 1, pp. 85–111, Jan. 2021.

[28] W. Chen, X. Qiu, T. Cai, H.-N. Dai, Z. Zheng, and Y. Zhang, "Deep reinforcement learning for Internet of Things: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1659–1692, 3rd Quart., 2021.

[29] T. Li et al., "Applications of multi-agent reinforcement learning in future internet: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 2, pp. 1240–1279, 2nd Quart., 2022.

[30] Z. Ji, A. K. Kiani, Z. Qin, and R. Ahmad, "Power optimization in device-to-device communications: A deep reinforcement learning approach with dynamic reward," *IEEE Wireless Commun. Lett.*, vol. 10, no. 3, pp. 508–511, Mar. 2021.

[31] J. Cui, Y. Liu, and A. Nallanathan, "Multi-agent reinforcement learning-based resource allocation for UAV networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 2, pp. 729–743, Feb. 2020.

[32] J. Chen, J. Zhang, N. Zhao, Y. Pei, Y.-C. Liang, and D. Niyato, "Joint device participation, dataset management, and resource allocation in wireless federated learning via deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 73, no. 3, pp. 4505–4510, Mar. 2024.

[33] Y. Chang et al., "A survey on evaluation of large language models," *ACM Trans. Intell. Syst. Technol.*, vol. 15, no. 3, pp. 1–45, Mar. 2024, doi: 10.1145/3641289.

[34] X. Wang et al., "Chain-of-thought for large language model-empowered wireless communications," 2025, *arXiv:2505.22320*.

[35] Y. Yu et al., "Large language model as attributed training data generator: A tale of diversity and bias," in *Advances in Neural Information Processing Systems*, vol. 36, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., Red Hook, NY, USA: Curran Associates, 2023, pp. 55734–55784. [Online]. Available: https://proceedings.neurips.cc/paperfiles/paper/2023/file/ae9500c4f5607caf2eff033c67daa9d7-Paper-DatasetsandBenchmarks.pdf

[36] R. Zhang, K. Xiong, Y. Lu, P. Fan, D. W. K. Ng, and K. B. Letaief, "Energy efficiency maximization in RIS-assisted SWIPT networks with RSMA: A PPO-based approach," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 5, pp. 1413–1430, May 2023.

[37] R. Zhang, K. Xiong, Y. Lu, D. W. K. Ng, P. Fan, and K. B. Letaief, "SWIPT-enabled cell-free massive MIMO-NOMA networks: A machine learning-based approach," *IEEE Trans. Wireless Commun.*, vol. 23, no. 7, pp. 6701–6718, Jul. 2024.

[38] R. Zhang et al., "Generative AI-enabled vehicular networks: Fundamentals, framework, and case study," *IEEE Netw.*, vol. 38, no. 4, pp. 259–267, Jul. 2024.

[39] C. J. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3, pp. 279–292, 1992.

[40] V. Mnih et al., "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.

[41] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[42] Y. Cao, S.-Y. Lien, Y.-C. Liang, D. Niyato, and X. Shen, "Collaborative computing in non-terrestrial networks: A multi-time-scale deep reinforcement learning approach," *IEEE Trans. Wireless Commun.*, vol. 23, no. 5, pp. 4932–4949, May 2024.

[43] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[44] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.

[45] N. Naderializadeh, J. J. Sydir, M. Simsek, and H. Nikopour, "Resource management in wireless networks via multi-agent deep reinforcement learning," *IEEE Trans. Wireless Commun.*, vol. 20, no. 6, pp. 3507–3523, Jun. 2021.

[46] F. Lotfi, H. Rajoli, and F. Afghah, "ORAN-GUIDE: RAG-driven prompt learning for LLM-augmented reinforcement learning in O-RAN network slicing," 2025, *arXiv:2506.00576*.

[47] B. Dharmalingam, R. Mukherjee, B. Piggott, G. Feng, and A. Liu, "Aero-LLM: A distributed framework for secure UAV communication and intelligent decision-making," 2025, *arXiv:2502.05220*.

[48] G. Gharsallah and G. Kaddoum, "MVX-ViT: Multimodal collaborative perception for 6G V2X network management decisions using vision transformer," *IEEE Open J. Commun. Soc.*, vol. 5, pp. 5619–5634, 2024.

[49] S. Milani, N. Topin, M. Veloso, and F. Fang, "Explainable reinforcement learning: A survey and comparative review," *ACM Comput. Surveys*, vol. 56, no. 7, pp. 1–36, Apr. 2024, doi: 10.1145/3616864.

[50] S. A. Serrano, J. Martinez-Carranza, and L. E. Sucar, "Knowledge transfer for cross-domain reinforcement learning: A systematic review," *IEEE Access*, vol. 12, pp. 114552–114572, 2024.

[51] R. Zhang et al., "Toward democratized generative AI in next-generation mobile edge networks," *IEEE Netw.*, vol. 39, no. 6, pp. 251–260, Nov. 2025.

[52] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, Red Hook, NY, USA: Curran Associates, vol. 30, pp. 1–12.

[53] T. Brown et al., "*Language models are few-shot learners*," in *Advances in Neural Information Processing Systems*, vol. 33. Red Hook, NY, USA: Curran Associates, 2020, pp. 1877–1901.

[54] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2019, pp. 4171–4186.

[55] Y. Liu et al., "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.

[56] C. Raffel et al., "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, no. 140, pp. 1–67, 2020.

[57] A. E. A. Grattafiori, "The LLAMA 3 herd of models," 2024, *arXiv:2407.21783*.

[58] M. A. Habib et al., "LLM-based intent processing and network optimization using attention-based hierarchical reinforcement learning," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Mar. 2025, pp. 1–6.

[59] M. Xu et al., "When large language model agents meet 6G networks: Perception, grounding, and alignment," *IEEE Wireless Commun.*, vol. 31, no. 6, pp. 63–71, Dec. 2024.

[60] J. Wang, A. Karatzoglou, I. Arapakis, and J. M. Jose, "Reinforcement learning-based recommender systems with large language models for state reward and action modeling," in *Proc. 47th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.* New York, NY, USA: Association for Computing Machinery, Jul. 2024, pp. 375–385, doi: 10.1145/3626772.3657767.

[61] J. Wei et al., "Chain-of-thought prompting elicits reasoning in large language models," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., Red Hook, NY, USA: Curran Associates, 2022, pp. 24824–24837. [Online]. Available: https://proceedings.neurips.cc/paperfiles/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf

[62] P. Wang et al., "WISE: Rethinking the knowledge memory for lifelong model editing of large language models," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 37, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, Eds., Red Hook, NY, USA: Curran Associates, 2024, pp. 53764–53797. [Online]. Available: https://proceedings.neurips.cc/paperfiles/paper/2024/file/60960ad78868fce5c165295fbd895060-Paper-Conference.pdf

[63] K. Nottingham et al., "Do embodied agents dream of pixelated sheep: Embodied decision making using language guided world modelling," in *Proc. 40th Int. Conf. Mach. Learn.*, vol. 202, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., Jul. 2023, pp. 26311–26325. [Online]. Available: https://proceedings.mlr.press/v202/nottingham23a.html

[64] B. Zhao and Z. Tang, "GIST: Guided interpretable large language model strategy transfer for multi-task reinforcement learning," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2025, pp. 1–5.

[65] H. Du, R. Zhang, D. Niyato, J. Kang, Z. Xiong, and D. I. Kim, "Reinforcement learning with large language models (LLMs) interaction for network services," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Feb. 2024, pp. 799–803.

[66] Z. Wang, R. Lu, Z. Zhang, C. Westphal, D. He, and J. Jiang, "LLM4Band: Enhancing reinforcement learning with large language models for accurate bandwidth estimation," in Proc. 35th, Ed., *Workshop Netw. Operating Syst. Support Digit. Audio Video*. New York, NY, USA: Association for Computing Machinery, Mar. 2025, pp. 43–49, doi: 10.1145/3712678.3721880.

[67] Z. Yan, H. Zhou, H. Tabassum, and X. Liu, "Hybrid LLM-DDQN-based joint optimization of V2I communication and autonomous driving," *IEEE Wireless Commun. Lett.*, vol. 14, no. 4, pp. 1214–1218, Apr. 2025.

[68] M. Shokrnezhad and T. Taleb, "An autonomous network orchestration framework integrating large language models with continual reinforcement learning," *IEEE Commun. Mag.*, vol. 63, no. 8, pp. 78–84, Aug. 2025.

[69] J. Zheng et al., "Large language model-enabled reinforcement learning for wireless network optimization," *IEEE Commun. Mag.*, vol. 64, no. 4, pp. 82–89, Apr. 2026.

[70] Y. Qu, M. Ding, N. Sun, K. Thilakarathna, T. Zhu, and D. Niyato, "The frontier of data erasure: A survey on machine unlearning for large language models," *Computer*, vol. 58, no. 1, pp. 45–57, Jan. 2025.

[71] X. Zhang et al., "Beyond the cloud: Edge inference for generative large language models in wireless networks," *IEEE Trans. Wireless Commun.*, vol. 24, no. 1, pp. 643–658, Jan. 2025.

[72] S. Khairy et al., "ACM MMSys 2024 bandwidth estimation in real time communications challenge," in *Proc. ACM Multimedia Syst. Conf. ZZZ*. New York, NY, USA: Association for Computing Machinery, Apr. 2024, pp. 339–345, doi: 10.1145/3625468.3653068.

[73] S. Holmer, H. Lundin, G. Carlucci, L. D. Cicco, and S. Mascolo. (2016). *A Google Congestion Control Algorithm for Real-Time Communication. Internet-Draft Draft-ietf-RMCAT-GCC-02*, [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-rmcat-gcc/02/

[74] F. Jiang et al., "Large language model enhanced multi-agent systems for 6G communications," *IEEE Wireless Commun.*, vol. 31, no. 6, pp. 48–55, Dec. 2024.

[75] X. Ma, G. Wan, R. Yu, G. Fang, and X. Wang, "CoT-valve: Length-compressible chain-of-thought tuning," in *Proc. 63rd Annu. Meeting Assoc. Comput. Linguistics*, 2025, pp. 6025–6035.

[76] X. Yin, F. You, H. Du, and K. Huang, "Ubiquitous intelligence via wireless network-driven LLMs evolution," *npj Wireless Technol.*, vol. 2, no. 1, p. 4, Feb. 2026.

[77] Y. Xu, Z. Jian, J. Zha, and X. Chen, "Poster abstract: Emergency networking using UAVs: A reinforcement learning approach with large language model," in *Proc. 23rd ACM/IEEE Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, May 2024, pp. 281–282.

[78] H. Lee et al., "Rlaif vs. RLHF: Scaling reinforcement learning from human feedback with ai feedback," in *Proc. 41st Int. Conf. Mach. Learn.*, 2024, pp. 1–28.

[79] K.-L.-A. Yau, P. Komisarczuk, and P. D. Teal, "Reinforcement learning for context awareness and intelligence in wireless networks: Review, new features and open issues," *J. Netw. Comput. Appl.*, vol. 35, no. 1, pp. 253–267, Jan. 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804511001755

[80] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access in wireless networks," *IEEE Trans. Cognit. Commun. Netw.*, vol. 4, no. 2, pp. 257–265, Jun. 2018.

[81] Y. Li, X. Hu, Y. Zhuang, Z. Gao, P. Zhang, and N. El-Sheimy, "Deep reinforcement learning (DRL): Another perspective for unsupervised wireless localization," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6279–6287, Jul. 2020.

[82] D. Han, A. Zhang, R. Chen, C. Feng, and S. Guo, "Agent in the sky: Intelligent multi-agent framework for autonomous haps coordination and real-world event adaptation," in *Proc. AAAI Workshop Artif. Intell. Wireless Commun. Netw. (AI4WCN)*, 2025, pp. 1–6.

[83] Y. Emami, H. Zhou, S. Nabavirazavi, and L. Almeida, "LLM-enabled in-context learning for data collection scheduling in UAV-assisted sensor networks," *IEEE Internet Things J.*, vol. 12, no. 23, pp. 51664–51676, Dec. 2025.

[84] S. Hu et al., "AgentsCoMerge: Large language model empowered collaborative decision making for ramp merging," *IEEE Trans. Mobile Comput.*, vol. 24, no. 10, pp. 1–15, Oct. 2025.

[85] Y. Xu et al., "Scalable UAV multi-hop networking via multi-agent reinforcement learning with large language models," *IEEE Trans. Mobile Comput.*, vol. 25, no. 7, pp. 11173–11190, Jul. 2026.

[86] J. Wen, Z. Li, M. Xi, and J. He, "An LLM-assisted AUV 3-D path planning scheme under ocean current interference via reinforcement learning," *IEEE Internet Things J.*, vol. 12, no. 19, pp. 39185–39196, Oct. 2025.

[87] X. Tang et al., "LLM-assisted reinforcement learning: Leveraging lightweight large language model capabilities for efficient task scheduling in multi-cloud environment," *IEEE Trans. Consum. Electron.*, vol. 71, no. 2, pp. 5631–5644, May 2025.

[88] S. Li et al., "Pre-trained language models for interactive decision-making," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 31199–31212.

[89] S. Wu, H. Fei, L. Qu, W. Ji, and T.-S. Chua, "Next-GPT: Any-to-any multimodal LLM," in *Proc. 41st Int. Conf. Mach. Learn.*, 2024, pp. 1–32.

[90] S. S. Kannan, V. L. N. Venkatesh, and B.-C. Min, "SMART-LLM: Smart multi-agent robot task planning using large language models," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2024, pp. 12140–12147.

[91] A. K. Widiawan and R. Tafazolli, "High altitude platform station (HAPS): A review of new infrastructure development for future wireless communications," *Wireless Pers. Commun.*, vol. 42, no. 3, pp. 387–404, Jun. 2007.

[92] A. Zhao, D. Huang, Q. Xu, M. Lin, Y.-J. Liu, and G. Huang, "Expel: LLM agents are experiential learners," in *Proc. AAAI Conf. Artif. Intell.*, 2024, vol. 38, no. 17, pp. 19632–19642.

[93] G. F. Lawler and V. Limic, *Random Walk: A Modern Introduction*, vol. 123. Cambridge, U.K.: Cambridge Univ. Press, 2010.

[94] X. Gao, X. Zhu, and L. Zhai, "AoI-sensitive data collection in multi-UAV-assisted wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 22, no. 8, pp. 5185–5197, Aug. 2023.

[95] E. J. Hu et al., "LoRA: Low-rank adaptation of large language models," in *Proc. ICLR*, 2022, vol. 1, no. 2, p. 3.

[96] X. Jiang et al., "Low-latency networking: Where latency lurks and how to tame it," *Proc. IEEE*, vol. 107, no. 2, pp. 280–306, Feb. 2019.

[97] F. Liu et al., "Beyond functional correctness: Exploring hallucinations in LLM-generated code," 2024, *arXiv:2404.00971*.

[98] X. Zhang, H. Zhao, J. Wei, C. Yan, J. Xiong, and X. Liu, "Cooperative trajectory design of multiple UAV base stations with heterogeneous graph neural networks," *IEEE Trans. Wireless Commun.*, vol. 22, no. 3, pp. 1495–1509, Mar. 2023.

[99] K. Li, W. Ni, X. Yuan, A. Noor, and A. Jamalipour, "Deep-graph-based reinforcement learning for joint cruise control and task offloading for aerial edge Internet of Things (EdgeIoT)," *IEEE Internet Things J.*, vol. 9, no. 21, pp. 21676–21686, Nov. 2022.

[100] C. Yu et al., "The surprising effectiveness of ppo in cooperative multi-agent games," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 24611–24624.

[101] X. Xia, H. Qiu, X. Xu, and Y. Zhang, "Multi-objective workflow scheduling based on genetic algorithm in cloud environment," *Inf. Sci.*, vol. 606, pp. 38–59, Sep. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025522004856

[102] H. Li, M. Xiao, K. Wang, D. I. Kim, and M. Debbah, "Large language model based multi-objective optimization for integrated sensing and communications in UAV networks," *IEEE Wireless Commun. Lett.*, vol. 14, no. 4, pp. 979–983, Apr. 2025.

[103] L. He, H. Ishibuchi, A. Trivedi, H. Wang, Y. Nan, and D. Srinivasan, "A survey of normalization methods in multiobjective evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 25, no. 6, pp. 1028–1048, Dec. 2021.

[104] S. Lu, Y. Wang, L. Sheng, L. He, A. Zheng, and J. Liang, "Out-of-distribution detection: A task-oriented survey of recent advances," *ACM Comput. Surveys*, vol. 58, no. 2, pp. 1–39, Jan. 2026.

[105] Y. Pan et al., "Towards reliable large language models: A survey on hallucination detection," in *Proc. Int. Conf. Intell. Comput. Springer*, 2025, pp. 438–451.

[106] L. Yang et al., "Out-of-distribution generalization in natural language processing: Past, present, and future," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2023, pp. 4533–4559.

[107] L. Huang et al., "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *ACM Trans. Inf. Syst.*, vol. 43, no. 2, pp. 1–55, Mar. 2025.

[108] H. Zou et al., "TelecomGPT: A framework to build telecom-specific large language models," *IEEE Trans. Mach. Learn. Commun. Netw.*, vol. 3, pp. 948–975, 2025.

[109] Z. Xiao et al., "Towards native intelligence: 6G-LLM trained with reinforcement learning from NDT feedback," 2026, *arXiv:2601.09992*.

[110] T. Su, T. Wu, J. Zhao, A. Scaglione, and L. Xie, "A review of safe reinforcement learning methods for modern power systems," *Proc. IEEE*, vol. 113, no. 3, pp. 213–255, Mar. 2025.

[111] T. V. Ngo, M. V. Ngo, B. Chen, T. Q. S. Quek, T. Kumari, and M. Nekovee, "LLM-based net analyzer rApp for explainable and safe automation in O-RAN non-RT RIC," 2026, *arXiv:2603.13775*.

[112] H. Jin and Y. Wu, "CE-CoLLM: Efficient and adaptive large language models through cloud-edge collaboration," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jul. 2025, pp. 316–323.

[113] T. Zeng et al., "RLink: Accelerate on-device deep reinforcement learning with inference knowledge at the edge," in *Proc. 19th Int. Conf. Mobility, Sens. Netw.* (MSN), 2023, pp. 628–635.

[114] Z. Lyu, M. Xiao, J. Xu, M. Skoglund, and M. D. Renzo, "The larger the merrier? Efficient large AI model inference in wireless edge networks," *IEEE J. Sel. Areas Commun.*, vol. 44, pp. 2839–2853, 2026.

[115] M. A. Mohsin, A. Bilal, S. Bhattacharya, and J. M. Cioffi, "Retrieval augmented generation with multi-modal LLM framework for wireless environments," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, Jun. 2025, pp. 184–189.

[116] H. Yang and W. Xu, "LLMKey: LLM-powered wireless key generation scheme for next-gen IoV systems," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, May 2025, pp. 1–6.

[117] J. Shao et al., "WirelessLLM: Empowering large language models towards wireless intelligence," 2024, *arXiv:2405.17053*.

[118] A. S. Ali, D. Michael Manias, A. Shami, and S. Muhaidat, "Leveraging large language models for DRL-based anti-jamming strategies in zero touch networks," 2023, *arXiv:2308.09376*.

[119] F. Deng, I. Jang, and S. Ahn, "DreamerPro: Reconstruction-free model-based reinforcement learning with prototypical representations," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 4956–4975.

[120] Y. Matsuo et al., "Deep learning, reinforcement learning, and world models," *Neural Netw.*, vol. 152, pp. 267–275, Aug. 2022.

[121] T. Orekondy, A. Behboodi, and J. B. Soriaga, "Mimo-GAN: Generative mimo channel modeling," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2022, pp. 5322–5328.

[122] C. Pandey, V. Tiwari, R. S. Rathore, R. H. Jhaveri, D. S. Roy, and S. Selvarajan, "Resource-efficient synthetic data generation for performance evaluation in mobile edge computing over 5G networks," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 1866–1878, 2023.

[123] I. J. Goodfellow et al., "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014.

[124] N. J. Prottasha et al., "Transfer learning for sentiment analysis using BERT based supervised fine-tuning," *Sensors*, vol. 22, no. 11, p. 4157, May 2022.
[125] L. Ouyang et al., "Training language models to follow instructions with human feedback," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, Dec. 2022, pp. 27730–27744.
[126] P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive nlp tasks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 9459–9474.
[127] J. Zhang, T. Q. Duong, A. Marshall, and R. Woods, "Key generation from wireless channels: A review," *IEEE Access*, vol. 4, pp. 614–626, 2016.
[128] Q. Wang, H. Su, K. Ren, and K. Kim, "Fast and scalable secret key generation exploiting channel phase randomness in wireless networks," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 1422–1430.
[129] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *Int. J. Comput. Vis.*, vol. 129, no. 6, pp. 1789–1819, 2021.
[130] R. Iyer, Y. Li, H. Li, M. Lewis, R. Sundar, and K. Sycara, "Transparency and explanation in deep reinforcement learning neural networks," in *Proc. AAAI/ACM Conf. AI, Ethics, Soc.*, Dec. 2018, pp. 144–150.
[131] F. Otto, O. Celik, H. Zhou, H. Ziesche, V. A. Ngo, and G. Neumann, "Deep black-box reinforcement learning with movement primitives," in *Proc. Conf. Robot Learn.*, 2023, pp. 1244–1265.
[132] E. Puiutta and E. M. S. P. Veith, "Explainable reinforcement learning: A survey," in *Proc. Int. Cross-Domain Conf. Mach. Learn. Knowl. Extraction*. Springer, 2020, pp. 77–95.
[133] J. Lin et al., "Learning to model the world with language," in *Proc. Int. Conf. Mach. Learn.*, 2024, pp. 29992–30017.
[134] D. Das, S. Chernova, and B. Kim, "State2Explanation: Concept-based explanations to benefit agent learning and user understanding," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36. Red Hook, NY, USA: Curran Associates, Inc., 2023, pp. 67156–67182.
[135] F. Liu, J. Chen, Q. Zhang, and B. Li, "Online MEC offloading for V2V networks," *IEEE Trans. Mobile Comput.*, vol. 22, no. 10, pp. 6097–6109, Oct. 2023.
[136] L. Wang et al., "A survey on large language model based autonomous agents," *Frontiers Comput. Sci.*, vol. 18, no. 6, Dec. 2024, Art. no. 186345, doi: 10.1007/s11704-024-40231-1.
[137] B. Shin, J. H. Lee, C. Kim, S. Jeon, and T. Lee, "LTE RSSI based vehicular localization system in long tunnel environment," *IEEE Trans. Ind. Informat.*, vol. 19, no. 11, pp. 11102–11114, Nov. 2023.
[138] R. Cheng, Y. Sun, D. Niyato, L. Zhang, L. Zhang, and M. A. Imran, "A wireless AI-generated content (AIGC) provisioning framework empowered by semantic communication," *IEEE Trans. Mobile Comput.*, vol. 24, no. 3, pp. 2137–2150, Mar. 2025, doi: 10.1109/TMC.2024.3493375.
[139] D. Sheng, Q. Qi, J. Wang, L. Li, W. Yu, and J. Liao, "PsyQoE: Improving quality-of-experience assessment with psychological effects in video streaming," *IEEE Trans. Services Comput.*, vol. 17, no. 6, pp. 4138–4150, Nov. 2024.
[140] C.-W. Xie, S. Sun, X. Xiong, Y. Zheng, D. Zhao, and J. Zhou, "RA-CLIP: Retrieval augmented contrastive language-image pre-training," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 19265–19274.
[141] L. Cai et al., "Secure physical layer communications for low-altitude economy networking: A survey," *IEEE Commun. Surveys Tuts.*, vol. 28, pp. 2497–2530, 2026.
[142] B. Hu et al., "MADQN-enhanced computation offloading and resource allocation for 6G low-altitude economy vehicular networks," *IEEE Trans. Cognit. Commun. Netw.*, vol. 12, pp. 2603–2617, 2026.
[143] Z. Jia et al., "Cooperative cognitive dynamic system in UAV swarms: Reconfigurable mechanism and framework," *IEEE Veh. Technol. Mag.*, vol. 19, no. 3, pp. 90–101, Sep. 2024.
[144] S. Ahmad et al., "Resource management for multi-drone communications in next-generation NOMA-enabled wireless networks," *Sci. Rep.*, vol. 15, no. 1, p. 23585, Jul. 2025, doi: 10.1038/s41598-025-09459-0.
[145] H. Jin et al., "A survey of energy efficient methods for UAV communication," *Veh. Commun.*, vol. 41, Jun. 2023, Art. no. 100594. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214209623000244
[146] R. Devidze, P. Kamalaruban, and A. Singla, "Exploration-guided reward shaping for reinforcement learning under sparse rewards," in *Proc. 36th Conf. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 35, 2022, pp. 5829–5842. [Online]. Available: https://proceedings.neurips.cc/paperfiles/paper/2022/file/266c0f191b04cbbbe529016d0edc847e-Paper-Conference.pdf
[147] G. Khromov and S. P. Singh, "Some fundamental aspects about Lipschitz continuity of neural networks," in *Proc. 12th Int. Conf. Learn. Represent.*, 2024. [Online]. Available: https://openreview.net/forum?id=5jWsW08zUh
[148] O. S. Oubbati, M. Atiquzzaman, H. Lim, A. Rachedi, and A. Lakas, "Synchronizing UAV teams for timely data collection and energy transfer by deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 71, no. 6, pp. 6682–6697, Jun. 2022.
[149] W. Li, F. Tang, M. Zhao, M. Omachi, and N. Kato, "Adaptive large language model for task orchestration in 6G space-air-ground integrated computing power networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 13, pp. 4847–4862, 2026.
[150] Y. Tan, T. Koketsu Rodrigues, N. Kato, M. Ariyoshi, and Y. Hasegawa, "Pivotal AI paradigms for satellite communication security: From machine learning to large language models," *IEEE Commun. Surveys Tuts.*, vol. 28, pp. 4654–4689, 2026.
[151] K. Wang, B. Kang, J. Shao, and J. Feng, "Improving generalization in reinforcement learning with mixture regularization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., Red Hook, NY, USA: Curran Associates, Dec. 2020, pp. 7968–7978. [Online]. Available: https://proceedings.neurips.cc/paperfiles/paper/2020/file/5a751d6a0b6ef05cfe51b86e5d1458e6-Paper.pdf
[152] T. M. Moerland, J. Broekens, A. Plaat, and C. M. Jonker, "Model-based reinforcement learning: A survey," *Found. Trends® Mach. Learn.*, vol. 16, no. 1, pp. 1–118, Jan. 2023.
[153] A. Kuznetsov, P. Shvechikov, A. Grishin, and D. Vetrov, "Controlling overestimation bias with truncated mixture of continuous distributional quantile critics," in *Proc. 37th Int. Conf. Mach. Learn.*, vol. 119, H. D. III and A. Singh, Eds., Jul. 2020, pp. 5556–5566. [Online]. Available: https://proceedings.mlr.press/v119/kuznetsov20a.html
[154] R. Zhang et al., "Embodied AI-enhanced vehicular networks: An integrated vision language models and reinforcement learning method," *IEEE Trans. Mobile Comput.*, vol. 24, no. 11, pp. 11494–11510, Nov. 2025.
[155] F. Tang et al., "MSADM: Large language model (LLM) assisted end-to-end network health management based on multi-scale semanticization," *IEEE Trans. Mobile Comput.*, vol. 25, no. 8, pp. 1–13, Aug. 2026.
[156] X. Qin, X. Yang, and X. Tang, "Repurposing backdoors for good: Ephemeral intrinsic proofs for verifiable aggregation in cross-silo federated learning," 2026, *arXiv:2603.10692*.
[157] T. Tan, M. Kato, F. Tang, M. Zhao, S. Omachi, and N. Kato, "Task-agentic resource scheduling in ISCC networks via multi-agent LLMS," *IEEE Commun. Mag.*, early access, Apr. 3, 2026, doi: 10.1109/MCOM.001.2500584.
[158] X. Li, M. Liu, and C. Yuen, "LLM agent communication protocol (LACP) requires urgent standardization: A telecom-inspired protocol is necessary," in *Proc. NeurIPS Workshop: AI ML Next-Generation Wireless Commun. Netw.*, 2025. [Online]. Available: https://openreview.net/forum?id=LpwE9cSLkS
[159] P. Gajjar and V. K. Shah, "ORANSight-2.0: Foundational LLMs for O-RAN," *IEEE Trans. Mach. Learn. Commun. Netw.*, vol. 3, pp. 903–920, 2025.
[160] H. Du et al., "Enhancing deep reinforcement learning: A tutorial on generative diffusion models in network optimization," *IEEE Commun. Surveys Tuts.*, vol. 26, no. 4, pp. 2611–2646, 2024.
[161] L. Cai et al., "Federated agentic AI for wireless networks: Fundamentals, approaches, and applications," 2026, *arXiv:2603.01755*.
[162] F. Jiang, C. Pan, K. Wang, P. Michiardi, O. A. Dobre, and M. Debbah, "From large AI models to agentic AI: A tutorial on future intelligent communications," *IEEE J. Sel. Areas Commun.*, vol. 44, pp. 3507–3540, Feb. 2026.