Uncertainty-Aware Zero-Touch Drift Management for Trustworthy DRL in 5G/6G Networks

Mazene Ameur*, *Member, IEEE*, Bouziane Brik‡, *Senior Member, IEEE,* and Adlen Ksentini*, *Senior Member, IEEE*
*Communication Systems Department, EURECOM, Sophia Antipolis, France.
‡Computer Science Department, University of Sharjah, Sharjah, UAE.
Emails: {firstname.lastname}@eurecom.fr, bbrik@sharjah.ac.ae

***Abstract*—As the push toward fully autonomous 5G and future 6G networks accelerates, Deep Reinforcement Learning (DRL) has emerged as a cornerstone of intelligent decision-making, enabling real-time adaptability and self-optimization. However, this promise is increasingly overshadowed by a critical and underexamined risk: DRL Drift, which refers to sudden and often opaque degradation in agent performance after deployment. This phenomenon jeopardizes the reliability and trustworthiness of DRL systems operating in dynamic, real-world telecom environments. Despite growing adoption of DRL in both research and industry, the issue of DRL Drift remains largely overlooked in major telecommunications standards such as the 3rd Generation Partnership Project (3GPP) and the European Telecommunications Standards Institute (ETSI). To address this challenge, we propose a novel Zero-Touch Drift Management framework, developed in alignment with the ETSI closed-loop reference architecture. At its core lies the Composite Drift Index, a unified, domain-agnostic metric that combines key performance indicators, state-action transitions, and uncertainty estimation to enable proactive detection of degradation. Extensive evaluations in a representative network-slicing scenario demonstrate up to 23.4% higher detection accuracy than baseline methods, with strong generalization across diverse DRL approaches. This work offers the first standards-aligned solution to enhance DRL resilience against drift in next-generation networks.**

I. Introduction

The leap from 5G to 6G networks introduces unprecedented levels of complexity, heterogeneity, and dynamism into modern communication infrastructures. These rapidly shifting environments demand intelligent, self-optimizing control mechanisms capable of operating with minimal human intervention [1]. In this context, Deep Reinforcement Learning (DRL) has rapidly gained traction as a key enabler in this landscape, offering autonomous decision-making capabilities across a wide spectrum of network management tasks [2].

Despite its potential, the adoption of DRL in production-grade telecom environments is significantly hindered by trustworthiness concerns. One of the most critical and underexplored challenges is "DRL Drift", defined as an unintended shift in a DRL agent's learned policy or value function, leading to degraded performance after deployment [4]. Unlike traditional Machine Learning (ML) systems, DRL operates in dynamic environments and relies on continuous interactions, making it especially susceptible to distributional shifts and environmental changes [4]. The absence of robust drift detection mechanisms in DRL deployments can precipitate catastrophic consequences for telecommunications operators, including Service Level Agreement (SLA) violations, Quality of Service (QoS) degradation, and compromised Quality of Experience (QoE) for end-users [5].

While substantial research efforts have been devoted to drift detection and mitigation in supervised and unsupervised learning paradigms [3], [6], [7], [18], the unique challenges posed by DRL systems, including sequential decision-making, delayed rewards, and non-stationary policies, have received inadequate attention. This research gap is particularly pronounced when considering the current leading standardization landscape. Both the 3rd Generation Partnership Project (3GPP) and the European Telecommunications Standards Institute (ETSI) have extensively incorporated AI/ML methodologies, with particular emphasis on DRL approaches, into their architectural specifications [8], [9]. However, the critical aspect of post-deployment DRL model reliability and drift management remains largely underexplored in these standardization efforts, creating a substantial disconnect between theoretical frameworks and practical deployment requirements.

To bridge this gap, we propose a novel end-to-end Zero-Touch Drift Management (ZDM) framework, fully aligned with the ETSI Zero-touch network and Service Management (ZSM) closed-loop specification [10] and purpose-built for DRL-based systems in telecommunications networks. Central to our approach is the development of a robust, uncertainty-aware drift detection mechanism built around a Composite Drift Index (CDI). This unified metric continuously assesses DRL agent performance degradation by integrating diverse and complementary signals, including environment-level Key Performance Indicators (KPIs), state-action dynamics, and epistemic uncertainty estimation, offering a comprehensive and interpretable view of system behavior. A key innovation of our methodology is the use of evidential learning to enable periodic monitoring of epistemic uncertainty in DRL environments [11]. Epistemic uncertainty reflects the agent's lack of knowledge and serves to quantify its confidence in policy decisions, particularly when encountering unfamiliar or ambiguous state-action pairs. By continuously assessing this confidence, our framework can proactively detect drift before explicit performance degradation manifests.

The primary contributions ("C") and key findings ("F") of this paper are summarized as follows:

- **C1.** We introduce a novel end-to-end Zero-Touch Drift Management architecture that aligns with the ETSI ZSM closed-loop automation standards. This architecture is specifically designed for managing drift in DRL-based systems operating in 5G/6G networks.
- **C2.** We propose a novel drift detection method that seamlessly integrates into the proposed ZDM architecture. At its core is the Composite Drift Index, a unified and adaptive metric that combines environment-level KPIs (e.g., throughput, delay), state-action transition distributions, and periodic epistemic uncertainty estimates to enable early, interpretable, and proactive detection of DRL drift.
- **C3.** To the best of our knowledge, this is the first application of evidential deep learning for monitoring DRL agent uncertainty in the context of drift. Our approach enables lightweight and interpretable drift detection by representing uncertainty through probabilistic distributions: Dirichlet for discrete action spaces and Normal-Inverse-Gamma (NIG) for continuous ones.
- **C4.** We empirically validate our drift detection framework using a 5G/6G network slicing use case, demonstrating its capability to detect performance drift while maintaining service quality under realistic telecom workloads.
- **F1.** Experimental evaluations reveal that our approach surpasses existing DRL drift detection methods, achieving up to 23.4% improvement in detection accuracy.
- **F2.** We show that the proposed method generalizes effectively across a diverse set of DRL algorithms, including value-based, policy-based, and actor-critic paradigms.

The remainder of this paper is organized as follows: Section II provides a comprehensive review of related work. Section III presents our ZDM closed-loop architecture. Section IV details the system modeling approach and the theoretical foundations of our CDI-based drift detection methodology. Section V describes the use case employed to validate the proposed solution. Section VI presents the experimental setup, results, and performance analysis. Finally, Section VII wraps up the paper.

II. Related Work

This section provides a comprehensive analysis of existing drift detection approaches, progressing from general ML contexts to specialized DRL scenarios, ultimately highlighting the critical gaps that motivate our proposed solution.

A. Model Drift in Supervised and Unsupervised Learning

Foundational work in supervised learning has established a robust theoretical basis for drift detection in data streams. Gama et al. [12] introduced drift taxonomies (sudden, gradual, incremental) with temporal monitoring frameworks, while Bifet and Gavaldà [13] proposed adaptive windowing using statistical bounds. More recent approaches, such as DSA-AE [3], leverage deep learning by combining autoencoders with dual self-attention for multi-type drift detection, while uncertainty-based methods [18] use prediction confidence, though limited to static classification. However, these supervised and unsupervised methods do not generalize to DRL settings due to temporal dependencies, delayed rewards, and evolving policies. Our work addresses these unique DRL challenges with both discrete and continuous action spaces.

B. Deep Reinforcement Learning Drift Detection

The emergence of DRL for sequential decision-making has spurred the development of drift detection techniques tailored to dynamic RL environments. Fang et al. [14] introduce GPAction, a two-stage Gaussian Process model that forecasts agent actions and flags environmental changes via prediction error monitoring. Alternative methods frame drift as an Out-Of-Distribution (OOD) issue: Haider et al. [15] employ prediction error spikes from learned dynamics models to signal drift, while Greenberg et al. [17] track episodic return declines, though this proves inadequate in non-episodic, real-world contexts. Wang et al. [16] propose a detector-agent strategy with uniform exploration to uncover drift, but its dependence on exhaustive exploration and finite state assumptions hampers real-world scalability. Current DRL drift detection frameworks exhibit three core limitations: (1) reliance on static, environment-specific thresholds results in elevated false positives in non-stationary contexts; (2) a focus on deterministic environments restricts adaptability to stochastic settings and diverse DRL algorithms; and (3) a notable lack of domain-aligned solutions for 5G/6G networks persists. To overcome these challenges, we propose a Composite Drift Index that fuses diverse detection signals and incorporates epistemic uncertainty via evidential learning, enabling proactive, reliable detection. To the best of our knowledge, this marks the first ETSI-compliant zero-touch architecture specifically designed for DRL drift management in telecommunications systems.

III. Zero-Touch DRL Drift Management Framework

The ETSI ZSM framework offers a standardized approach for autonomous network management via the Observe-Orient-Decide-Act (OODA) loop, enabling continuous monitoring, analysis, decision-making, and action without human oversight [10]. Building on this paradigm, our ZDM architecture adapts the OODA loop to DRL systems operating within 5G/6G infrastructures, where high environmental complexity demands robust drift handling As illustrated in Fig. 1, the architecture includes four primary modules mapped with the OODA phases: the Monitoring System (Observe), Uncertainty Analytics Engine (Orient), Drift Detection Analytics Engine (Decide), and Decision Engine (Act). These components are unified by a central DRL knowledge repository (Knowledge Data), which stores logs and reference data essential for drift assessment. The DRL agent interfaces with diverse network elements such as core network functions, RAN components, and edge/cloud resources in real time.

A. Monitoring System

The Monitoring System constitutes the **"Observe"** phase of the closed-loop architecture and serves as the primary data

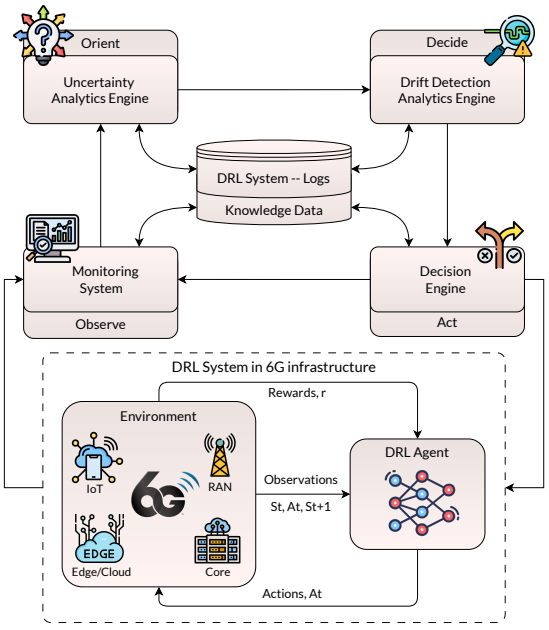

Fig. 1: The Proposed Zero-Touch DRL Drift Management Closed-Loop Architecture Compliant with ETSI ZSM Reference Model.

collection layer for the DRL system deployed in 5G/6G infrastructure. This module continuously monitors critical KPIs that are context-dependent based on the specific use case, including but not limited to throughput, packet loss ratio, end-to-end latency, etc. Beyond traditional network KPIs, the monitoring system implements deep observability by tracking DRL agent-level statistics, particularly the state-action pairs (s_t, a_t) that provide insights into the current environment state and agent behavior patterns. This dual-layer monitoring approach enables a comprehensive understanding of both network performance and agent decision-making processes.

B. DRL System Logs - Knowledge Data

As depicted in Figure 1, all collected data is systematically stored in the central DRL System Logs, which serves as a shared knowledge repository for the entire closed-loop system. This repository maintains not only real-time operational data but also reference datasets captured during the late (post-convergence) training phase, which serve as baseline distributions for subsequent drift detection algorithms. The temporal organization of this data enables historical trend analysis and facilitates the computation of statistical distances between current and reference distributions.

C. Uncertainty Analytics Engine

The Uncertainty Analytics Engine represents the **"Orient"** phase and quantifies the epistemic uncertainty of the DRL agent's neural network. Epistemic uncertainty captures the agent's confidence in its predictions and reflects the degree of knowledge about the environment state, indicating regions where the agent lacks sufficient training data or where the environment has shifted from the training distribution [19]. We employ evidential learning to compute epistemic uncertainty, treating neural network outputs as parameters of a higher-order distribution [20] (details in section IV). As the environment evolves and deviates from the training distribution, epistemic uncertainty increases, signaling potential performance degradation.

D. Drift Detection Analytics Engine

The Drift Detection Analytics Engine implements the **"Decide"** phase and houses the proposed CDI algorithm (details in section IV-D). This engine continuously analyzes data from two sources: (1) monitored KPIs and state-action pairs from the Knowledge Data repository, and (2) periodically computed epistemic uncertainty values from the Uncertainty Analytics Engine. The CDI algorithm employs statistical distance measures (Jensen-Shannon divergence [25] and Mahalanobis distance [24]) to compare current distributions with reference baselines. The algorithm operates on multiple timescales with high-frequency monitoring of state-action distributions and lower-frequency uncertainty analysis. When statistical distances exceed predefined thresholds or uncertainty levels show sustained increases, the system flags drift events and generates structured alerts forwarded to the Decision Engine.

E. Decision Engine

The Decision Engine constitutes the **"Act"** phase and implements the autonomous response mechanism for detected drift events. Upon receiving drift alerts, this component initiates a multi-stage mitigation strategy: (1) activation of a safe rule-based fallback policy that temporarily overrides the DRL agent's decisions to maintain system stability, and (2) triggering the retraining process using recent data from the DRL System Logs. The retraining procedure incorporates both historical reference data and newly collected samples that capture the shifted environment characteristics. The engine coordinates the training process while monitoring convergence metrics to ensure acceptable performance levels. As shown in Figure 1, the Decision Engine completes the closed-loop by feeding the retrained agent back into the 5G/6G infrastructure environment, ensuring continuous adaptation without human intervention.

IV. System Model

This section formally presents our proposed DRL drift detection framework. To establish the foundation, we first explore the foundational concept of epistemic uncertainty, which serves as the theoretical basis for our approach. Next, we define the drift detection problem within the context of DRL. Finally, we describe the design and modeling of our CDI, which integrates uncertainty estimation into proactive drift monitoring.

A. Epistemic Uncertainty

Uncertainty quantification in deep learning represents a fundamental challenge in distinguishing between aleatoric

uncertainty (inherent data noise) and epistemic uncertainty (model knowledge limitations) [11]. In DRL, epistemic uncertainty estimation becomes particularly critical as agents must make sequential decisions under partial observability while adapting to dynamic environments. The evolution of uncertainty estimation has progressed through several paradigmatic approaches [19]. Traditional methods such as Monte Carlo Dropout (MCD) [21] approximate epistemic uncertainty by treating dropout as a Bayesian approximation, sampling multiple forward passes during inference. However, MCD suffers from computational overhead and lacks theoretical grounding for uncertainty calibration. Ensemble methods [22] address some limitations by training multiple models with different initializations, yet they impose significant computational and memory requirements, making them impractical for real-time DRL applications. These limitations have catalyzed the development of the evidential learning paradigm [11], which provides a principled framework for uncertainty quantification without requiring multiple forward passes or ensemble training. In the context of DRL, evidential learning enables agents to explicitly model their confidence in value estimates and policy decisions, facilitating more robust decision-making in production environments.

B. Problem Formulation

Consider a Markov Decision Process (MDP) [5], [27] defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ represents the state space, $\mathcal{A}$ the action space, $\mathcal{P}$ the transition probability function, $\mathcal{R}$ the reward function, and γ the discount factor (all notations are summarized in Table I).

TABLE I: Notations Summary.

Symbol	Description
$\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma$	MDP: state, action, transition, reward, discount
$\pi_\theta(a\|s)$	Policy function parameterized by θ
$V_\phi(s)$	Value function parameterized by ϕ
$\boldsymbol{e}_i$	Evidence logits for action i
$\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_K]$	Dirichlet evidence parameters for all discrete action K
$\alpha_i = \exp(\boldsymbol{e}_i) + 1$	Evidence strength for action i
$S = \sum_j \alpha_j$	Total evidence strength
$p_i = \alpha_i / S$	Predictive probability for action i
$\mathcal{U}_{\text{epi}}(s)$	Epistemic uncertainty: K/S
$(\mu, \nu, \alpha, \beta)$	NIG parameters (mean, scaling, shape, scale)
$\mathcal{U}_{\text{val}}(s)$	Value uncertainty: $\beta/(\alpha - 1)$
D_t	Composite Drift Index
$D_{\text{KPI}}(t)$	KPI-based drift score
$D_{\text{SA}}(t)$	State-action drift score
$D_{\text{U}}(t)$	Uncertainty drift score
φ_t, ψ_t, ξ_t	CDI component weights
$I_{\text{trigger}}(t)$	Trigger flag (1 if drift conditions met)
$\tau_{\text{KPI}}, \tau_{\text{SA}}, \tau_{\text{CDI}}$	Drift detection thresholds
$\boldsymbol{k}_t$	KPI vector at time t
$\boldsymbol{\mu}_{\text{ref}}, \boldsymbol{\Sigma}_{\text{ref}}$	Reference mean and covariance
$D^{\text{raw}}_{\text{KPI}}(t)$	Raw Mahalanobis distance
P_t, P_{ref}	Current and reference state-action distributions
$\text{JS}(P, Q)$	Jensen-Shannon divergence
$\text{KL}(P\|Q)$	Kullback-Leibler divergence
$\mathcal{W}_t$	States visited in current window
$\mathcal{U}_{\text{agg}}(t)$	Aggregated epistemic uncertainty
$\mathcal{U}_{\text{ref}}$	Reference uncertainty level
$\mu(\cdot), \sigma(\cdot)$	Mean and std. over sliding window
$k_{\text{KPI}}, k_{\text{SA}}$	Sensitivity parameters for drift

In evidential DRL, we model the agent's policy $\pi_\theta(a|s)$ and value function $V_\phi(s)$ using evidential neural networks. Unlike traditional networks that provide point estimates, evidential networks output the parameters of probability distributions, allowing them to represent both predictions and the associated epistemic uncertainty. Specifically, for discrete actions, the neural network outputs the parameters of a Dirichlet distribution [11], while for continuous value estimation, it outputs the parameters of a NIG distribution [23].

*1) **Policy Modeling with Dirichlet Distributions**:* For a given state-action pair (s, a), the evidential network produces a set of non-negative evidence logits $\boldsymbol{e} = [e_1, e_2, \ldots, e_K]$, where K is the number of possible actions. These logits are transformed into concentration parameters $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_K]$ of the Dirichlet distribution using the following transformation:

$$\alpha_i = \exp(\boldsymbol{e}_i) + 1 \tag{1}$$

This transformation ensures that each $\alpha_i > 1$, representing a positive amount of "evidence" supporting action a_i. A higher α_i implies the agent is more confident about the suitability of action a_i in the current state.

The predictive probability for choosing action a_i is derived from the mean of the Dirichlet distribution [11]:

$$p_i = \frac{\alpha_i}{\sum_{j=1}^{K} \alpha_j} = \frac{\alpha_i}{S} \tag{2}$$

where $S = \sum_{j=1}^{K} \alpha_j$ represents the total evidence accumulated across all actions. This formulation allows the agent to express not just a distribution over actions, but also how strongly it believes in that distribution. To quantify how uncertain the agent is about its action selection, we compute the epistemic uncertainty as:

$$\mathcal{U}_{\text{epi}}(s_t) = \frac{K}{\sum_{j=1}^{K} \alpha_j} = \frac{K}{S} \tag{3}$$

This quantity is inversely related to the total evidence S. When S is small (i.e., low confidence), the uncertainty is high; when S is large (i.e., the network has gathered strong evidence from data), the uncertainty is low. Thus, this metric serves as a proxy for how much the agent "knows" about its current policy decision in state s_t.

*2) **Value Function Modeling with Normal-Inverse-Gamma Distributions**:* For the value function $V(s)$, the network models its uncertainty using a NIG distribution parameterized by $(\mu, \nu, \alpha, \beta)$. The neural network outputs these four parameters, which describe a belief distribution over possible value estimates [23]:

$$\hat{V}(s) = \mu, \quad \sigma^2 = \frac{\beta(\nu + 1)}{\alpha\nu} \tag{4}$$

Here, μ represents the predicted mean value for state s, and σ^2 gives the predictive variance, capturing both epistemic and aleatoric uncertainty. To isolate the epistemic component (the part of uncertainty stemming from lack of knowledge or data), we compute:

$$\mathcal{U}_{\text{val}}(s_t) = \frac{\beta}{\alpha - 1} \quad \text{for } \alpha > 1 \tag{5}$$

This expression is the expected value of the variance under the Inverse-Gamma distribution and is only valid for $\alpha > 1$. A lower α implies that the model has seen fewer examples like the current input, and hence is less certain about its value prediction. Conversely, a higher α suggests strong confidence, leading to low epistemic uncertainty.

C. Composite Drift Index (CDI) Modeling

1) Multi-Modal Drift Detection: Our proposed system integrates multiple drift indicators through a novel CDI that monitors both system performance and agent behavior. The CDI at time t is defined as:

$$D_t = \underbrace{\varphi_t \cdot D_{\text{KPI}}(t)}_{\text{KPI-based drift}} + \underbrace{\psi_t \cdot D_{\text{SA}}(t)}_{\text{State-action drift}} + \underbrace{\xi_t \cdot I_{\text{trigger}}(t) \cdot D_{\text{U}}(t)}_{\text{Uncertainty-based drift}} \tag{6}$$

where the trigger indicator $I_{\text{trigger}}(t)$ is defined as:

$$I_{\text{trigger}}(t) = \begin{cases} 1, & \text{if } D_{\text{KPI}}(t) > \tau_{\text{KPI}} \\ & \quad \vee\ D_{\text{SA}}(t) > \tau_{\text{SA}} \\ & \quad \vee\ t \bmod T = 0 \\ 0, & \text{otherwise} \end{cases}$$

where τ_{KPI} and τ_{SA} denote the thresholds for the KPI and state-action metrics, respectively, (see section IV-C5). In our implementation, the weights (φ_t, ψ_t, ξ_t) are fixed to $\frac{1}{3}$ to ensure equal sensitivity across all three drift modalities. However, in settings where certain indicators (e.g., epistemic uncertainty) are more critical for decision-making, these weights may be treated as tunable or even learnable parameters. Additionally, the periodicity parameter T is selected based on application-specific monitoring needs and available computational resources.

2) KPI-Based Drift Detection: The KPI drift component $D_{\text{KPI}}(t)$ employs the Mahalanobis distance to detect shifts in system performance metrics. For a KPI vector $\boldsymbol{k}_t = [k_1, k_2, \ldots, k_n]$ at time t, the Mahalanobis distance from the reference distribution is [24]:

$$D_{\text{KPI}}^{\text{raw}}(t) = \sqrt{(\boldsymbol{k}t - \boldsymbol{\mu}\text{ref})^T \boldsymbol{\Sigma}_{\text{ref}}^{-1} (\boldsymbol{k}t - \boldsymbol{\mu}\text{ref})} \tag{7}$$

where $\boldsymbol{\mu}$ref and $\boldsymbol{\Sigma}$ref represent the mean vector and covariance matrix of the reference KPI distribution, respectively. The distance is normalized to $[0, 1]$ using [32]:

$$D_{\text{KPI}}(t) = \frac{D_{\text{KPI}}^{\text{raw}}(t) - D_{\min}}{D_{\max} - D_{\min}} \tag{8}$$

3) State-Action Drift Detection: The state-action drift component $D_{\text{SA}}(t)$ utilizes the Jensen-Shannon divergence to measure distributional shifts in the agent's state-action space. For probability distributions P and Q representing current and reference state-action distributions, the Jensen-Shannon divergence is [25]:

$$\text{JS}(P, Q) = \frac{1}{2}\text{KL}(P|M) + \frac{1}{2}\text{KL}(Q|M) \tag{9}$$

where $M = \frac{1}{2}(P + Q)$ and $\text{KL}(\cdot|\cdot)$ denotes the Kullback-Leibler divergence [25]:

$$\text{KL}(P|Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)} \tag{10}$$

The state-action drift is then computed as:

$$D_{\text{SA}}(t) = \sqrt{\text{JS}(P_t, P_{\text{ref}})} \tag{11}$$

where P_t represents the current state-action distribution and P_{ref} the reference distribution.

4) Epistemic Uncertainty Drift Detection: The uncertainty drift component $D_{\text{U}}(t)$ captures shifts in the agent's epistemic uncertainty patterns. We aggregate the epistemic uncertainty across all visited states in a monitoring window:

$$\mathcal{U}_{\text{agg}}(t) = \frac{1}{|\mathcal{W}_t|} \sum_{s \in \mathcal{W}_t} \mathcal{U}_{\text{epi}}(s) \tag{12}$$

where $\mathcal{W}_t$ represents the set of states visited in the monitoring window ending at time t. The uncertainty drift is computed using the absolute difference from the reference uncertainty level [32]:

$$D_{\text{U}}(t) = \frac{|\mathcal{U}\text{agg}(t) - \mathcal{U}\text{ref}|}{\mathcal{U}\max - \mathcal{U}\min} \tag{13}$$

5) Adaptive Threshold Mechanism: The system employs dynamic thresholds to adapt to evolving network conditions. The thresholds are computed as [31]:

$$\tau_{\text{KPI}} = \mu(D_{\text{KPI}}) + k_{\text{KPI}} \cdot \sigma(D_{\text{KPI}}) \tag{14}$$

$$\tau_{\text{SA}} = \mu(D_{\text{SA}}) + k_{\text{SA}} \cdot \sigma(D_{\text{SA}}) \tag{15}$$

where $\mu(\cdot)$ and $\sigma(\cdot)$ denote the mean and standard deviation computed over a sliding window, and $k_{\text{KPI}}, k_{\text{SA}}$ are sensitivity parameters that control the detection sensitivity.

6) Drift Decision Mechanism: The final drift decision is made by comparing the CDI against a composite threshold τ_{CDI}:

$$\text{Drift Detected} = \begin{cases} \text{True}, & D_t > \tau_{\text{CDI}} \\ \text{False}, & \text{otherwise} \end{cases} \tag{16}$$

D. DRL Drift Detection framework

The proposed DRL Drift Detection framework, summarized in Algorithm 1, enables early detection of performance degradation by leveraging the complementary nature of KPI monitoring, agent behavioral analysis, and epistemic uncertainty quantification, providing a robust solution for maintaining DRL system reliability in dynamic 5G/6G environments. The algorithm operates through a three-pronged approach where system-level KPIs are continuously monitored using Mahalanobis distance $D_{\text{KPI}}(t)$ to detect statistical deviations from reference distributions $P_{\text{ref}}^{\text{KPI}}$. Concurrently, state-action behavioral drift is quantified through Jensen-Shannon divergence $D_{\text{SA}}(t)$, capturing distributional shifts in the agent's policy behavior. The third component employs evidential DRL to estimate epistemic uncertainty $U_{\text{epi}}(s_t)$ using either Dirichlet distributions for discrete actions or NIG distributions

for continuous control, with uncertainty aggregation (eq. 12) computed over a sliding window of size W. The CDI integrates these three modalities through adaptive weighting D_t, where $I_{\text{trigger}}(t)$ activates uncertainty monitoring periodically with period T, and adaptive thresholds τ_{KPI} and τ_{SA} provide sensitivity control through hyperparameters k_{KPI} and k_{SA}. When $D_t > \tau_{\text{CDI}}$, the algorithm triggers drift detection and initiates retraining mechanisms, while maintaining computational efficiency through selective uncertainty evaluation and online reference distribution updates during stable periods.

Algorithm 1: DRL Drift Detection Algorithm

Input : DRL agent π_θ, window size W, period T, sensitivity parameters $k_{\text{KPI}}, k_{\text{SA}}$
Output: Drift detection signal $\mathcal{D}_t$
Initialize $P_{\text{ref}}^{\text{KPI}}, P_{\text{ref}}^{\text{SA}}, \mathcal{U}_{\text{ref}}$;
Initialize $\mathcal{W}_{\text{KPI}}, \mathcal{W}_{\text{SA}}, \mathcal{W}_{\text{U}}$;
Set weights: $\varphi_t = \psi_t = \xi_t = 1/3$;
for *each time step* t **do**
  Collect system KPIs: $\boldsymbol{k}_t = [k_1, \ldots, k_n]$;
  Observe (s_t, a_t);
  Compute $\mathcal{U}_{\text{epi}}(s_t)$ using Eq. (3);
  KPI Drift Detection: Compute $D_{\text{KPI}}^{\text{raw}}(t)$ using Eq. (7);
  Normalize: $D_{\text{KPI}}(t) = \frac{D_{\text{KPI}}^{\text{raw}}(t) - D_{\min}}{D_{\max} - D_{\min}}$;
  State-Action Drift Detection: Update P_t with (s_t, a_t);
  Compute $D_{\text{SA}}(t) = \sqrt{\text{JS}(P_t, P_{\text{ref}})}$ using Eq. (11);
  Uncertainty Drift Detection: Update window: $\mathcal{W}_t \leftarrow \mathcal{W}_t \cup \{s_t\}$;
  $\mathcal{U}_{\text{agg}}(t) = \frac{1}{|\mathcal{W}_t|} \sum_{s \in \mathcal{W}_t} \mathcal{U}_{\text{epi}}(s)$;
  $D_{\text{U}}(t) = \frac{|\mathcal{U}_{\text{agg}}(t) - \mathcal{U}_{\text{ref}}|}{\mathcal{U}_{\max} - \mathcal{U}_{\min}}$;
  Adaptive Threshold Computation: $\tau_{\text{KPI}} = \mu(D_{\text{KPI}}) + k_{\text{KPI}} \cdot \sigma(D_{\text{KPI}})$;
  $\tau_{\text{SA}} = \mu(D_{\text{SA}}) + k_{\text{SA}} \cdot \sigma(D_{\text{SA}})$;
  Trigger Evaluation: **if** $D_{\text{KPI}}(t) > \tau_{\text{KPI}}$ ***or*** $D_{\text{SA}}(t) > \tau_{\text{SA}}$ ***or*** $t \bmod T = 0$ **then**
    $I_{\text{trigger}}(t) = 1$;
  else
    $I_{\text{trigger}}(t) = 0$;
  Composite Drift Index: $D_t = \varphi_t \cdot D_{\text{KPI}}(t) + \psi_t \cdot D_{\text{SA}}(t) + \xi_t \cdot I_{\text{trigger}}(t) \cdot D_{\text{U}}(t)$;
  Drift Decision: **if** $D_t > \tau_{\text{CDI}}$ **then**
    $\mathcal{D}_t =$ True;
    Trigger retraining or adaptation mechanism;
  else
    $\mathcal{D}_t =$ False;
  Update References: **if** $\mathcal{D}_t =$ *False* **then**
    Update $P_{\text{ref}}^{\text{KPI}}, P_{\text{ref}}^{\text{SA}}, \mathcal{U}_{\text{ref}}$ with current data;

V. Use Case: DRL for Dynamic 5G/6G Network Slicing

To rigorously evaluate the ZDM framework, we adopt network slicing as a representative and 3GPP-compliant use case of high industrial relevance [5], [26]. We focus on assessing the efficacy of the proposed drift detection mechanism via the CDI. Network slicing allows for multiple logically isolated, service-specific networks to coexist over a shared physical infrastructure, with each slice designed to meet distinct QoS and SLA requirements [27]. Our evaluation centers on a dynamic slicing scenario, where a DRL agent optimally allocates physical resources across three standard slice types: Ultra-Reliable Low Latency Communication (URLLC), massive Machine-Type Communication (mMTC), and enhanced Mobile Broadband (eMBB). The DRL agent dynamically assigns Resource Blocks (RBs) to each slice based on real-time traffic demands, ensuring SLA compliance while optimizing overall network efficiency. The environment is modeled as a discrete-time MDP, wherein the agent allocates R_{total} RBs among the slices at each decision epoch to meet their respective QoS targets. The system model components are detailed in Table II. This MDP-based formulation allows the DRL agent to learn adaptive resource allocation policies under dynamic traffic conditions, offering a suitable platform to evaluate the proposed drift detection approach.

VI. Performance Evaluation

This section presents a comprehensive evaluation of our novel drift detection approach. We first describe the experimental setup and highlight key implementation details, followed by an in-depth analysis of the results. Finally, we conclude with a discussion that distills critical insights and summarizes the principal findings of the study.

A. Experimental Setup

To evaluate our approach, we implemented three representative DRL baselines: Deep Q-Network (DQN) [28] as a value-based method, Proximal Policy Optimization (PPO) [29] as a policy-gradient method, and Advantage Actor-Critic (A2C) [30] as an actor-critic method. These algorithms represent a broad spectrum of DRL techniques applied to dynamic radio resource management in network slicing. The DRL models were implemented using the Stable-Baselines3[1] and Gymnasium[2] (a successor to OpenAI Gym) to create the custom environments for 5G/6G network slicing. Hyperparameters were tuned based on empirical evaluation. The learning rate was set to 1×10^{-4}, with a discount factor γ of 0.99. DQN used a replay buffer size of 10^5 and target network updates every 1000 steps. PPO and A2C were configured with generalized advantage estimation using $\lambda = 0.95$. For PPO, a clipping parameter of 0.2 was used, while A2C employed shared networks for the actor and critic. All components of the proposed architecture are implemented in Python 3.12.8. The experiments were conducted on a workstation equipped with an AMD 8-Core 3.2 GHz CPU, 16 GB RAM, and an NVIDIA GeForce RTX 3050 Ti GPU.

B. Implementation Details

We modified the neural network architectures of the baseline DQN, PPO, and A2C algorithms to incorporate evidential output layers for uncertainty-aware resource allocation in network slicing. Based on the system modeling presented in Table II, our formulation defines a continuous action space

[1]https://stable-baselines3.readthedocs.io/en/master/
[2]https://gymnasium.farama.org/

TABLE II: DRL System Model Components for 5G/6G Network Slicing.

State Space	**Action Space**	**Reward Function**
Traffic Load: $\mathbf{L}(t)$	RB Allocation: $\mathbf{a}(t)$	$R(t) = \alpha \cdot R_{\text{QoS}}(t) + \beta \cdot R_{\text{efficiency}}(t) - \gamma \cdot R_{\text{penalty}}(t)$
Current RB Allocation: $\mathbf{R}(t)$	Constraint: $\sum_i a_i(t) \leq R_{\text{total}}$	QoS Reward: $R_{\text{QoS}}(t) = \sum_i w_i \cdot 1_{\text{SLA}_i}(t)$
Buffer States: $\mathbf{B}(t)$	Min. Guarantees: $a_i(t) \geq R_{\text{min}}^{(i)}$	Efficiency: $R_{\text{efficiency}}(t) = \frac{\sum_i T_i(t)}{\sum_i a_i(t)}$
Latency Metrics: $\boldsymbol{\tau}(t)$	Action Bounds: $a_i(t) \in [0, R_{\text{max}}^{(i)}]$	Penalty: $R_{\text{penalty}}(t) = \sum_i \lambda_i \cdot \max(0, \tau_i(t) - L_{\text{max}}^{(i)})$
Throughput: $\mathbf{T}(t)$	$\mathbf{a}(t) = [a_{\text{URLLC}}, a_{\text{mMTC}}, a_{\text{eMBB}}]$	Weights: $\alpha = 0.6$, $\beta = 0.3$, $\gamma = 0.1$
SLA Violations: $\mathbf{V}(t)$		Priorities: $w_{\text{URLLC}} = 0.5$, $w_{\text{mMTC}} = 0.2$, $w_{\text{eMBB}} = 0.3$

$\mathbf{a}(t) \in \mathbb{R}^3$ representing resource block allocations across URLLC, mMTC, and eMBB service types, subject to capacity constraints $\sum_i a_i(t) \leq R_{\text{total}}$ and minimum guarantee requirements $a_i(t) \geq R_{\text{min}}^{(i)}$. Given the continuous nature of the action space, we employ NIG distributions to model the evidential output layers, where each network produces four parameters $(\gamma, \nu, \alpha, \beta)$ per action dimension to characterize both the predictive mean and the associated epistemic uncertainty. For DQN, we replace the standard Q-value output with evidential Q-value distributions; for PPO and A2C, we modify both the policy and value function networks to output NIG parameters, allowing the agents to quantify uncertainty in both action selection and value estimation during the resource allocation process. It is worth emphasizing that the central aim of this evaluation is to validate the effectiveness of the drift detection mechanism, which serves as the pivotal and most technically demanding element of our ZDM closed-loop architecture. In the current design, drift adaptation involves retraining the DRL agent upon detection and using a rule-based fallback policy to ensure operational continuity. More advanced, context-aware drift-handling methods will be explored in future work.

C. Experiment Results & Findings

In this section, we present the empirical results evaluating overall drift detection performance, generalizability, and robustness. We also include an ablation study on the CDI method, followed by a comparative analysis with existing DRL-based drift detectors.

*1) **Overall Effectiveness and Generalizability**:* To study the effectiveness of the proposed DRL drift detection mechanism, we evaluate the CDI under simulated non-stationary conditions characterized by a sudden and persistent OOD drift. The primary goal is to assess the sensitivity and reliability of CDI in identifying environment-induced distributional changes, while also investigating its robustness and generalizability across diverse DRL architectures. Fig. 2 illustrates the temporal evolution of CDI values for three representative DRL algorithms: DQN, PPO, and A2C. Before the induced drift (red vertical line), CDI values remain low and stable across all methods, indicating stationary system dynamics. Upon drift introduction, CDI values sharply increase and remain elevated, reflecting sustained drift awareness. The shaded area marks the true drift period, while the orange dashed line shows the 95th percentile threshold used to signify drift detection. CDI trajectories exceed this threshold shortly after drift onset across all methods, confirming timely and consistent detection. The rate and magnitude of CDI increase differ among algorithms: A2C shows the steepest rise, followed by PPO and then DQN, indicating varying sensitivity to drift. These differences highlight CDI's robustness and generalizability across diverse algorithmic architectures and learning behaviors.

*2) **Epistemic Uncertainty for DRL Drift Detection Validation**:* In this experiment, we analyze the behavior of epistemic uncertainty as a drift-aware signal to evaluate its sensitivity and effectiveness in detecting environmental non-stationarities. Epistemic uncertainty, which quantifies model ignorance stemming from insufficient or unfamiliar data, is particularly informative when an agent is exposed to OOD scenarios. Fig. 3 presents uncertainty monitoring plots for DQN, PPO, and A2C agents under both stationary and drift conditions, with each subplot capturing temporal fluctuations in epistemic uncertainty values across timesteps. Across all three subplots, a sharp increase in epistemic uncertainty occurs immediately after the drift initiation point (red dashed line), aligning with the shaded drift period. Before this, uncertainty levels remained low and stable, indicating the agents' confidence in familiar environments. Post-drift, uncertainty rises abruptly and remains elevated, reflecting reduced confidence under OOD conditions. The moving average curves emphasize this trend, stabilizing at higher values. The drift threshold is consistently surpassed shortly after drift onset for all algorithms, supporting the use of epistemic uncertainty as an early-warning signal for performance degradation. Differences in response magnitude and smoothness are observed across algorithms, with A2C and PPO exhibiting more gradual and consistent increases, while DQN shows higher variability.

These results reinforce the premise that epistemic uncertainty is an effective drift indicator, capable of capturing early signs of knowledge mismatch and model miscalibration.

*3) **CDI Ablation Study**:* To further investigate the behavior and reliability of the proposed CDI, we conducted a fine-grained analysis of its constituent components, namely KPI Drift, State-Action Drift, and Epistemic Uncertainty. As illustrated in Fig. 4, each subplot presents the normalized temporal evolution of these individual signals alongside the unified CDI metric, offering insight into how each dimension of drift manifests over time. Notably, the KPI Drift (top-left) exhibits marked deviations during drift intervals, particularly in response to environmental perturbations, while the State-

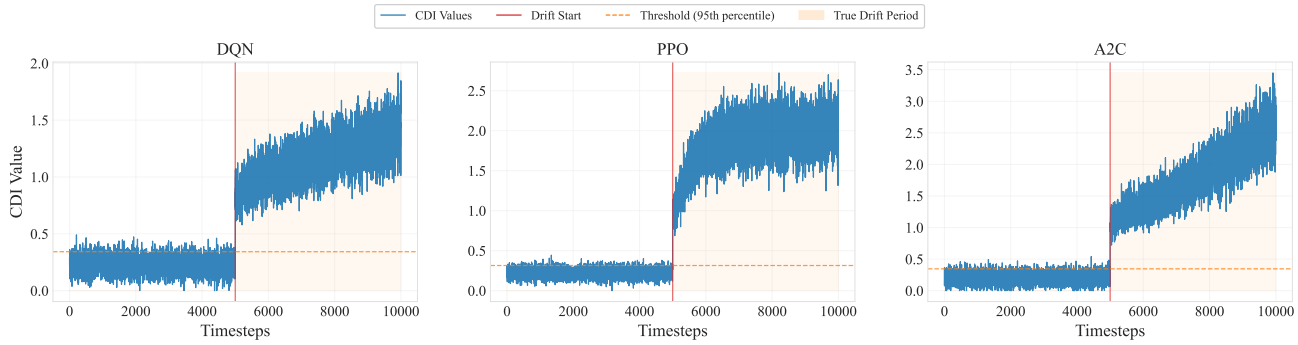

Fig. 2: Tracking drift using the proposed CDI across three DRL algorithms: DQN, PPO, and A2C.

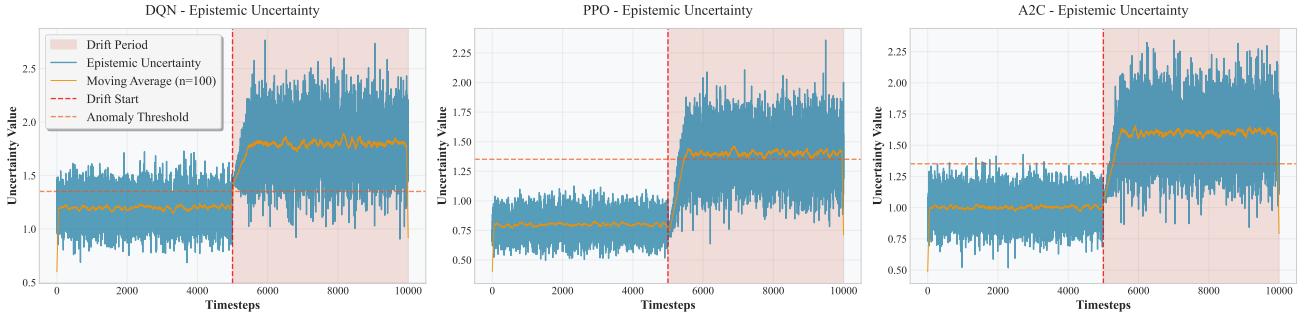

Fig. 3: Epistemic uncertainty monitoring across DQN, PPO, and A2C under normal and distributional drift.

Action Drift (top-right) captures shifts in policy behavior, such as those induced by changes in the state-action distribution. The Epistemic Uncertainty signal (bottom-left), reflecting model confidence, reacts prominently in regions indicative of unfamiliar or OOD states. Importantly, these diverse responses are not synchronous, underscoring that each signal is sensitive to distinct facets of the underlying distributional shifts. This complementary behavior is crucial, as it endows the CDI (bottom-right) with robustness and adaptability, allowing it to effectively aggregate heterogeneous sources of drift into a cohesive and reliable signal. Because CDI is composed of multiple signals rather than relying on a single indicator, it is also more robust against false positives, reducing the likelihood of misidentifying normal variability as drift. Collectively, the results validate the utility of the CDI as a generalizable and discriminative indicator for monitoring dynamic instabilities in DRL environments.

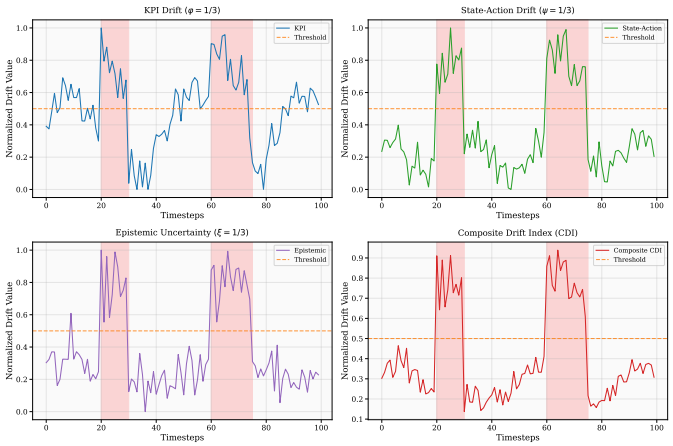

Fig. 4: Monitoring the individual components of the CDI: KPI Drift, State-Action Drift, and Epistemic Uncertainty.

4) ***CDI Correlation study****:* To further characterize the internal structure and dynamics of the CDI (see Eq. 6), we study the correlation among its constituent components. Fig. 5 presents the correlation matrix, which quantifies the degree of linear association between the CDI components: KPI Drift, State-Action Drift, and Epistemic Uncertainty. CDI exhibits a strong correlation with all components, confirming their meaningful contribution. State-Action Drift shows the highest correlation (0.81), highlighting its dominant role and supporting the observation that most DRL drift arises from state-action interactions. KPI Drift follows with a correlation of 0.72, reflecting sensitivity to environment-level changes. Epistemic Uncertainty presents a slightly lower correlation (0.69), which is expected since it is computed periodically rather than continuously. Overall, these results demonstrate the complementary nature of the components and validate the robustness of the composite CDI, which captures diverse drift signals while mitigating sensitivity to noise from any single source.

5) ***Comparative Study****:* In this experiment, we conduct a comparative analysis of the proposed CDI against state-of-the-

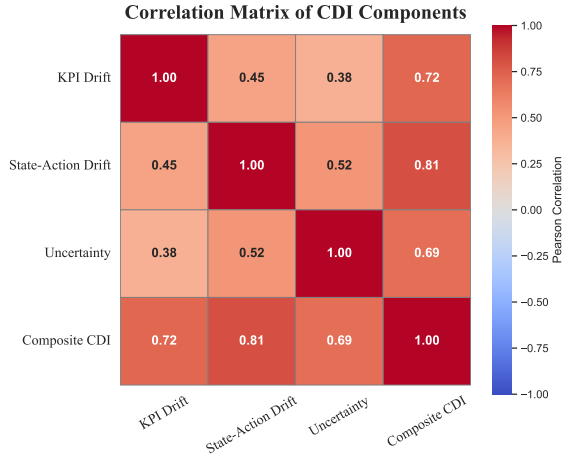

Fig. 5: Correlation heatmap between the components of the CDI.

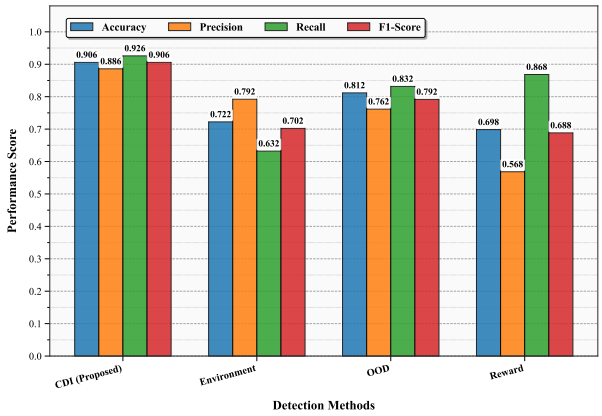

Fig. 6: Comparison of drift detection performance across different methods for drift detection in DRL.

art drift detection methods in DRL, which have been adapted from other domains to our telecom context. These include environment monitoring [14], reward-based monitoring [17], and OOD detection [15]. Fig. 6 summarizes the comparative performance across four key metrics: accuracy, precision, recall, and F1-score. The results clearly indicate that the CDI outperforms all baselines across every performance dimension. Specifically, the CDI achieves the highest accuracy (0.91), precision (0.89), recall (0.93), and F1-score (0.91), demonstrating not only its superior detection capability but also its effectiveness in minimizing false positives. This performance can be attributed to the CDI's robust integration of heterogeneous drift signals, including KPI drift, state-action drift, and epistemic uncertainty. The synergy among these components enables the CDI to maintain a holistic understanding of the agent's behavior and its operational context. Notably, the uncertainty signal plays a critical role in enhancing sensitivity to latent or subtle drifts by capturing the agent's epistemic confidence. This allows the CDI to identify periods where the agent is uncertain about its predictions, which is a strong indicator of distributional shifts. Consequently, the CDI benefits from both early drift detection and reduced susceptibility to noise, positioning it as a comprehensive and reliable solution for DRL drift monitoring.

D. Discussion & Learned Lessons

The analysis of our DRL drift detection system reveals critical insights. A central conclusion is the pivotal role of epistemic uncertainty, which emerges as a reliable and sensitive indicator of distributional shifts. Notably, sharp increases in this uncertainty consistently coincide with the onset of environmental drift, establishing its value for timely and sustained detection. Among various uncertainty estimation techniques, evidential learning proves particularly effective for DRL applications. It offers an advantageous trade-off between computational efficiency and estimation quality, making it well-suited for real-time or resource-limited settings when compared to ensemble or sampling-based methods. Furthermore, detection robustness is significantly enhanced by integrating multiple complementary indicators, including epistemic uncertainty, KPI changes, and state-action distribution shifts. This multimodal approach not only reduces false positives but also improves overall system stability by avoiding reliance on a single detection modality. Finally, the CDI demonstrates strong adaptability across diverse DRL algorithms, such as DQN, PPO, and A2C. Although detection sensitivity may vary slightly due to algorithm-specific learning dynamics, CDI consistently delivers reliable performance, confirming its generalizability and effectiveness in heterogeneous deployment scenarios.

VII. Conclusion

In this paper, we introduced a Zero-Touch Drift Management closed-loop architecture specifically tailored for DRL systems in 5G/6G networks. The proposed architecture is aligned with the ETSI ZSM closed-loop reference model and incorporates a novel drift detection mechanism centered around the CDI. The CDI integrates diverse and complementary drift signals, including environment KPI monitoring, state-action dynamics, and epistemic uncertainty estimation, providing a holistic view of system behavior. To the best of our knowledge, this is the first work to apply evidential learning for uncertainty estimation within the context of DRL drift detection. Extensive experiments demonstrate that our CDI method outperforms baselines across multiple metrics and generalizes well across different DRL algorithms. As part of our future work, we aim to extend this framework beyond detection to include intelligent and autonomous drift handling strategies.

Acknowledgment

This work was supported by the European Union's Horizon Program under 6G-DALI (Grant Agreement No. 101192750).

REFERENCES

[1] M. M. Azari et al., "Evolution of non-terrestrial networks from 5G to 6G: A survey," IEEE Communications Surveys & Tutorials, vol. 24, no. 4, pp. 2633–2672, 2022. doi:10.1109/comst.2022.3199901

[2] A. Abouaomar, A. Taik, A. Filali, and S. Cherkaoui, "Federated deep reinforcement learning for open RAN slicing in 6G networks," IEEE Commun. Mag., vol. 61, no. 2, pp. 126–132, 2023.

[3] M. Ameur, B. Brik and A. Ksentini, "Dual Self-Attention is What You Need for Model Drift Detection in 6G Networks," in IEEE Transactions on Machine Learning in Communications and Networking, vol. 3, pp. 690-709, 2025, doi: 10.1109/TMLCN.2025.3576727.

[4] Z. Liu, J. Lu, J. Xuan, and G. Zhang, "Deep reinforcement learning in nonstationary environments with unknown change points," IEEE Trans. Cybern., vol. 54, no. 9, pp. 5191–5204, 2024.

[5] M. Ameur, B. Brik and A. Ksentini, "Leveraging LLMs to eXplain DRL Decisions for Transparent 6G Network Slicing," 2024 IEEE 10th International Conference on Network Softwarization (NetSoft), Saint Louis, MO, USA, 2024, pp. 204-212, doi: 10.1109/NetSoft60951.2024.10588921.

[6] F. Bayram, B. S. Ahmed, and A. Kassler, "From concept drift to model degradation: An overview on performance-aware drift detectors," Knowl. Based Syst., vol. 245, no. 108632, p. 108632, 2022.

[7] D. M. Manias, A. Chouman, and A. Shami, "Model drift in dynamic networks," IEEE Commun. Mag., vol. 61, no. 10, pp. 78–84, 2023.

[8] 3GPP, "Management and orchestration; Artificial Intelligence/Machine Learning (AI/ML) management," 3GPP TS 28.105, ver. V19.2.0, Release 19, [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3970. [Accessed: Jul. 19, 2025].

[9] ETSI, "ETSI TS 104 224 V1.1.1: Securing Artificial Intelligence (SAI); Explicability and transparency of AI processing," Mar. 2025. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/104200_104299/104224/01.01.01_60/ts_104224v010101p.pdf. [Accessed: Jul. 19, 2025].

[10] ETSI, "ETSI GS ZSM 002 V1.1.1: ZSM Architecture," Aug. 2019. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/ZSM/001_099/002/01.01.01_60/gs_ZSM002v010101p.pdf. [Accessed: Jul 19, 2025].

[11] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential Deep Learning to Quantify Classification Uncertainty," in Advances in Neural Information Processing Systems 31 (NeurIPS 2018), 2018.

[12] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in Advances in Artificial Intelligence – SBIA 2004, Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 286–295.

[13] A. Bifet and R. Gavaldà, "Learning from time-changing data with adaptive windowing," in Proceedings of the 2007 SIAM International Conference on Data Mining, Philadelphia, PA: Society for Industrial and Applied Mathematics, 2007, pp. 443–448.

[14] Z. Fang and U. Zdun, "Detecting environment drift in reinforcement learning using a Gaussian process," Int Conf Tool Artif Intell, pp. 992–999, 2024.

[15] T. Haider, et al., "Out-of-distribution detection for reinforcement learning agents with probabilistic dynamics models," in Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, 2023, pp. 851–859.

[16] Z. Wang, C. Chen, H.-X. Li, D. Dong, and T.-J. Tarn, "Incremental reinforcement learning with prioritized sweeping for dynamic environments," IEEE/ASME Transactions on Mechatronics, vol. 24, no. 2, pp. 621– 632, 2019.

[17] I. Greenberg and S. Mannor, "Detecting rewards deterioration in episodic reinforcement learning," in International Conference on Machine Learning, PMLR, 2021, pp. 3842–3853.

[18] P. Lu, J. Lu, A. Liu, and G. Zhang, "Early concept drift detection via prediction uncertainty," Proc. Conf. AAAI Artif. Intell., vol. 39, no. 18, pp. 19124–19132, 2025, doi: 10.1609/aaai.v39i18.34105.

[19] H. Yang, C. Chen, Y. Chen, M. Scheppach, H.-C. Yip, and Q. Dou, "Uncertainty estimation for safety-critical scene segmentation via fine-grained reward maximization," Neural Inf Process Syst, vol. abs/2311.02719, 2023, doi: 10.48550/arXiv.2311.02719.

[20] S. Depeweg, J. M. Hernández-Lobato, F. Doshi-Velez, and S. Udluft, "Decomposition of Uncertainty in Bayesian Deep Learning for Efficient and Risk-Sensitive Learning," in Proc. 35th Int. Conf. Mach. Learn. (ICML), Stockholm, Sweden, Jul. 2018, pp. 1184–1193.

[21] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in Proc. 33rd Int. Conf. Mach. Learn. (ICML), New York, NY, USA, Jun. 2016, vol. 48, pp. 1050–1059.

[22] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 6402–6413.

[23] A. Amini, W. Schwarting, A. P. Soleimany, and D. Rus, "Deep Evidential Regression," Neural Inf Process Syst, vol. abs/1910.02600, 2019, Accessed: Jul. 21, 2025.

[24] L. Ming, H. Dezhi, and L. Dun, "A method combining improved Mahalanobis distance and adversarial autoencoder to detect abnormal network traffic," in Proceedings of the 27th International Database Engineered Applications Symposium, 2023.

[25] C. Chen, C. M. H. Unal, and A. C. P. O. Nijhuis, "Jensen–Shannon distance-based filter and unsupervised evaluation metrics for polarimetric weather radar processing," IEEE Trans. Geosci. Remote Sens., vol. 60, pp. 1–18, 2022, doi: 10.1109/tgrs.2022.3200731.

[26] 3GPP TS 28.530, "Management and orchestration; Concepts, use cases and requirements," Release 19, Mar. 2025. Accessed: Jul. 21, 2025. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3273

[27] W. F. Villota-Jacome, O. M. C. Rendon, and N. L. S. da Fonseca, "Admission control for 5G core network slicing based on deep reinforcement learning," IEEE Syst. J., vol. 16, no. 3, pp. 4686–4697, 2022, doi: 10.1109/jsyst.2022.3172658.

[28] T. Li, X. Zhu, and X. Liu, "An end-to-end network slicing algorithm based on deep Q-learning for 5G network," IEEE Access, vol. 8, pp. 122229–122240, 2020, doi: 10.1109/access.2020.3006502.

[29] J. Dai et al., "O-RAN-enabled intelligent network slicing to meet service-level agreement (SLA)," IEEE Trans. Mob. Comput., vol. 24, no. 2, pp. 890–906, 2025, doi: 10.1109/tmc.2024.3476338.

[30] R. A. Addad, D. L. C. Dutra, T. Taleb, and H. Flinck, "Toward using reinforcement learning for trigger selection in network slice mobility," IEEE J. Sel. Areas Commun., vol. 39, no. 7, pp. 2241–2253, 2021, doi: 10.1109/jsac.2021.3078501.

[31] A. Komadina, M. Martinić, S. Groš and Ž. Mihajlović, "Comparing Threshold Selection Methods for Network Anomaly Detection," in IEEE Access, vol. 12, pp. 124943-124973, 2024, doi: 10.1109/ACCESS.2024.3452168.

[32] H. W. Herwanto, A. N. Handayani, A. P. Wibawa, K. L. Chandrika and K. Arai, "Comparison of Min-Max, Z-Score and Decimal Scaling Normalization for Zoning Feature Extraction on Javanese Character Recognition," 2021 7th International Conference on Electrical, Electronics and Information Engineering (ICEEIE), Malang, Indonesia, 2021, pp. 1-3, doi: 10.1109/ICEEIE52663.2021.9616665.