International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 17 (2025) pp. 610-627

https://doi.org/10.70917/ijcisim-2025-0038
Article

Hybrid Intelligence for Industry 4.0: Integrating Large Language Models and Reinforcement Learning

Vagan Terziyan [1*], Oleksandra Vitko [2], Oleksandr Terziyan [2] and Artur Terziian [3]

[1] Faculty of Information Technology, University of Jyväskylä, Jyväskylä, FI-40014, Finland (V.T.)
[2] Department of Artificial Intelligence, Kharkiv National University of Radio Electronics, Kharkiv, UA-61166, Ukraine; oleksandra.vitko@nure.ua (O.V.); oleksandr.terziyan@nure.ua (O.T.)
[3] Faculty of Informatics and Statistics, Prague University of Economics and Business, Prague, CZ-13067, Czech Republic; tera03@vse.cz (A.T.)
* Correspondence author: vagan.terziyan@jyu.fi; ORCID: 0000-0001-7732-2962

Received date: 24 December 2024; Accepted date: 28 May 2025; Published online: 31 December 2025

Abstract: The future of industry is driven by intelligent systems capable of autonomous decision-making, dynamic adaptation, and integrating human knowledge. In this context, hybrid approaches emerge that combine data-driven methodologies with external knowledge sources. This paper introduces OPRA-RL, a hybrid framework that integrates Reinforcement Learning (RL) with the OPRA (Observation-Prompt-Response-Action) framework. OPRA-RL integrates the self-learning capabilities of RL with the contextual expertise and adaptive reasoning of Large Language Models, such as ChatGPT, to tackle challenges in complex, real-world environments. We present an analytical formulation of OPRA-RL, highlighting its complex reward structure designed to balance internal learning with external guidance through prompts. The proposed OPRA-Q-Learning variant is implemented and validated experimentally through a simulated decision-making game, illustrating how knowledge-informed autonomy can outperform traditional RL in scenarios characterized by sparse data, high complexity, or novel challenges. Our findings reveal how multifaceted reward systems, external knowledge integration, and dynamic decision-making enhance agent performance in unpredictable environments. By bridging the gap between knowledge-informed and data-driven AI, OPRA-RL contributes to smarter and resilient autonomous systems.

Keywords: Industry 4.0; artificial intelligence; autonomous agents; reinforcement learning; large language models

1. Introduction

The future of industry is increasingly shaped by the adoption of autonomous, intelligent systems designed to improve productivity, flexibility, sustainability, and resilience in dynamic environments [1]. Key components of this transformation include smart manufacturing, predictive maintenance, and human-machine collaboration [2, 3]. Within the broader evolution from Industry 4.0—characterized by the integration of IoT, big data, and automation—to what is increasingly discussed as Industry 5.0, emerging themes such as personalized production and enhanced human-in-the-loop collaboration are gaining traction [4].

These technological trends are underpinned by significant advances in Artificial Intelligence (AI) and Machine Learning (ML), which play central roles in data analysis, process optimization, and intelligent decision-making. AI is now fundamental to modern industrial systems, enabling machines to learn, reason, and adapt in ways that mirror human cognition. ML, particularly deep learning, explores large-scale data to improve performance, while the future of AI includes efforts toward autonomous AI and Artificial General Intelligence (AGI). In the context of Industry 4.0 and its extensions, AGI aims at generalizable reasoning across domains [5], and autonomous AI targets self-sufficient decision-making

in complex, evolving settings.

A critical enabler in this landscape is the advent of Large Language Models (LLMs), such as ChatGPT, which have emerged as powerful sources of information, knowledge, and intelligent services [6]. These models represent a paradigm shift in AI, providing unprecedented access to domain knowledge, contextual reasoning, and adaptive responses. LLMs have opened new avenues for combining knowledge-informed and data-driven modeling, enabling AI systems to transcend traditional limitations by integrating external expertise dynamically.

In our earlier work, we introduced the OPRA (Observation–Prompt–Response–Action) framework, along with its collaborative multi-agent variant, COPRA (Collaborative OPRA), as alternatives to classical agent models—particularly within intelligent manufacturing domains [7]. These frameworks are tailored for adaptive behavior in evolving environments, where agents can proactively seek external knowledge—such as responses from LLMs—to compensate for missing information and improve decision-making. OPRA is especially valuable in scenarios where agents encounter ambiguity, novelty, or lack of predefined operational rules, allowing them to query external systems for contextual guidance that conventional approaches may fail to provide.

Reinforcement Learning (RL) is a cornerstone of modern AI [8], focusing on agents that learn optimal actions through trial-and-error interactions with their environment. RL has proven critical in industrial contexts [9], enabling systems to adapt to dynamic scenarios, optimize resource allocation, and improve decision-making. From predictive maintenance to adaptive supply chains, RL drives innovation in Industry 4.0 and lays the foundation for future developments in Industry 5.0.

This article proposes a hybrid framework, OPRA-RL, which integrates the OPRA framework with RL. By combining the dynamic knowledge acquisition capabilities of OPRA with the self-learning mechanisms of RL, OPRA-RL aims to enhance the adaptability and performance of autonomous agents in complex, real-world environments.

The key expectation from OPRA-RL is its ability to balance internal learning (via RL) with external knowledge acquisition (via prompts to LLMs). This balance enables agents to navigate scenarios where either approach alone may fall short, such as when data is sparse, the environment is highly complex, or novel challenges arise. The hybrid framework introduces an additional reward component tied to the quality of prompts and the utility of external advice, incentivizing agents to effectively utilize external expertise while learning their internal policies.

While ChatGPT serves as a user-facing interface to underlying LLMs, it is treated in this work as a representative of dynamic external knowledge systems. We acknowledge that LLMs can exhibit limitations, such as hallucinations or imprecision in numerical reasoning. To address these challenges, the OPRA-RL framework includes mechanisms for assessing the usefulness of advice through outcome-based reward attribution. Furthermore, although human-like prompts are modeled in our experimental setting, the architecture is designed for autonomous agents to generate and evaluate prompts themselves, minimizing human intervention and supporting future automation of prompt engineering. In practical implementations, such LLMs are typically accessed through APIs (e.g., OpenAI's chat/completions endpoint). Various parameters and role definitions (e.g., system, user, assistant) play a critical role in shaping responses. In our simulations, we use default or commonly recommended values to balance creativity and consistency, and agents interact with LLMs via programmatically defined prompts that mimic API calls rather than requiring real-time human input.

Section 2 provides an analytical introduction to OPRA-RL, detailing the multifaceted external knowledge prompting reward. Section 3 outlines essential OPRA-RL variations, exploring how external information is prompted, used, and balanced with internal knowledge. Section 4 focuses on the OPRA-Q-Learning framework, while Section 5 presents its specific implementation and experimental evaluation through a simulated game that demonstrates its practical utility and potential for real-world applications. Here, rather than aiming for benchmark-level performance metrics, we focus on a conceptual demonstration of how OPRA-Q-Learning integrates observation, prompting, and reinforcement to improve agent behavior in partially observable environments. Section 6 summarizes related work, including popular RL variations, and specifies where OPRA-RL—being a kind of knowledge-informed RL—can perform beyond the current state-of-the-art. Finally, Section 7 concludes the study.

This study integrates RL with knowledge-informed decision-making, aiming to advance the capabilities of autonomous systems in complex industrial domains and beyond, thereby enabling smarter AI autonomy.

2. OPRA-RL's Analytical Framework with Complex Reward Function

Integrating the OPRA framework into the RL paradigm introduces a way for agents to interact with environments using dynamic knowledge acquisition and external prompting, while maintaining the

central concepts of RL: policies, rewards, and value functions. Observations inform agents of the state of the environment. However, instead of just relying on internal knowledge, OPRA agents query and explore external knowledge systems. The prompts and responses help guide action selection, but actions are still chosen according to a policy that aims to maximize cumulative reward. Rewards are given based on how effective the actions are in the context of the environment, allowing the agent to learn from its interactions and refine both its internal policy and the way it queries external systems. Let us formalize the OPRA-RL framework, which utilizes the key RL components: states, actions, rewards, policies, and value functions, while embedding the OPRA chain and, in addition, rewarding the prompt engineering skills of the agent.

In the traditional RL framework, the agent observes the environment and receives state s_t at time t. In OPRA-RL, this is modified to include both direct observations of the environment and knowledge acquired externally via prompts. The agent first makes an observation O_t, which may not fully capture the environment. Based on the incomplete observation, the agent generates a prompt P_t, querying an external knowledge source K_{ext}, such as ChatGPT or another system. The state s_t is an augmented state that includes both the observation O_t and the response R_t from the external system:

$$s_t = (O_t, R_t).$$

In RL, an agent selects an action a_t based on the current state s_t following a policy π. In OPRA-RL framework, the action comes after receiving and with respect to a response from the external source. The policy $\pi(a|s)$, therefore, maps from the augmented state to action:

$$\pi(a|s) = \pi(a_t|O_t, R_t).$$

The agent receives a reward r_t after taking an action a_t, which is based on how successful that action was in the environment. In OPRA-RL, a traditional reward r_t is given based on the outcome of the action in the environment r_t^{env}, but also takes into account the additional reward $r_t^{K_{ext}}$, which depends on quality of knowledge gained from the external system (in other words, rewarding the agent for the quality of generated prompts):

$$r_t = r_t^{env} + \lambda \cdot r_t^{K_{ext}}, \qquad (1)$$

where λ (normally < 1) is a weighting factor to balance the influence of environmental and knowledge rewards.

Such an approach encourages the agent not only to maximize environmental rewards but also to learn how to better generate prompts to obtain more useful knowledge. This balances the need to learn better environmental policies with the ability to improve external querying skills.

OPRA and RL integration opens the door for more sophisticated, context-aware RL agents that leverage external knowledge to solve real-world problems, particularly in complex industrial, social, or adversarial settings.

One may notice that the key added value of OPRA-RL comparably to traditional RL is a knowledge reward component $r_t^{K_{ext}}$ in the reward function (1). Let us provide a generic schema of how this additional reward can be computed. To assess the effectiveness of prompt generation in the OPRA-RL framework, we could consider multiple factors that align with the agent's ability to extract useful, actionable knowledge from external sources. These factors could directly influence the reward $r_t^{K_{ext}}$ associated with the knowledge obtained from the prompt. Let us design a comprehensive knowledge reward function for OPRA-RL by integrating the components provided below, which are prompt relevance (including prompt-goal and response-goal alignment), actionability, efficiency, and learning gain, into a unified formula.

2.1. Relevance

A well-formed prompt P_t should ask questions or seek information directly relevant to the agent's current objective G_t (goal or subgoal). The better the prompt aligns with the objective, the more likely it is to return useful knowledge as a response R_t that helps the agent make decisions or take actions that move it toward success. Therefore, rewarded component r_t^{prompt} ("primary relevance") of the prompt quality could be computed as cosine similarity between P_t and G_t represented within a shared embedding space. Although the primary focus is the prompt quality, the response quality matters too. Therefore, the computed cosine similarity between R_t and G_t provides an additional rewarded component $r_t^{responce}$ ("secondary relevance") of the prompt quality.

To convert prompt, response, and goal to embeddings one needs to tokenize these into smaller units (e.g., words or subwords) and use pre-trained language models like Hugging Face [10], or Sentence

Transformers [11] to convert the natural language goal G_t, prompt P_t, and the response R_t into vectors. These embeddings encode the semantic meaning of the text in a high-dimensional vector space.

Having vector representations for G_t, P_t, and R_t, we can compute:

$$r_t^{prompt} = \text{cosine\_similarity}(P_t, G_t) = \frac{P_t \cdot G_t}{\|P_t\|\|G_t\|}, \quad (2)$$

$$r_t^{responce} = \text{cosine\_similarity}(R_t, G_t) = \frac{R_t \cdot G_t}{\|R_t\|\|G_t\|}. \quad (3)$$

Both (2) and (3) ranges are within the [-1, 1] interval, where 0 means that vectors are orthogonal, 1 means perfect alignment, and -1 indicates opposite alignment (complete irrelevance).

2.2. Actionability

Another component to evaluate external knowledge is its actionability—the extent to which the external response assists the agent in decision-making. A response may offer actionable insights by providing specific steps or new options for the agent to consider. Actionability measures whether the response meaningfully guides the agent's actions, producing a score within the interval [-1, 1]:

- 1: Fully actionable information that leads to successful actions.
- 0: Neutral information that neither helps nor harms decision-making.
- -1: Harmful information that worsens decision-making or increases uncertainty.

The actionability score r_t^{act} evaluates how effectively the external response R_t reduces uncertainty in selecting action a_t at time t, based on the agent's action distribution $\pi(a_t|s_t, R_t)$. Here, $s_t = \{O_t, P_t, R_t, G_t\}$ includes observation, prompt, response, and goal of the agent at time t. A well-informed response narrows the agent's focus on a subset of viable actions, improving decision confidence.

The score r_t^{act} can be computed by comparing the following three entropies:

(1) $H_{\text{informed}} = H(\pi(a_t|s_t, R_t))$: Entropy after incorporating response R_t;
(2) $H_{\text{uninformed}} = H(\pi(a_t|s_t))$: Entropy before receiving R_t;
(3) $H_{\max} = \log_2 n$: Maximum entropy for n actions, representing uniform uncertainty.

The formula is:

$$r_t^{act} = \frac{H_{\text{uninformed}} - H_{\text{informed}}}{H_{\max}}. \quad (4)$$

Key interpretations of formula (4) are as follows:

- *Best case*: when $H_{\text{informed}} = 0$ and $H_{\text{uninformed}} = H_{\max}$, i.e., when the response removes completely the maximal uncertainty, we have maximal reward $r_t^{act} = 1$;
- *Neutral case*: when $H_{\text{uninformed}} = H_{\text{informed}}$, i.e., when the response does not change the uncertainty, we have a neutral reward $r_t^{act} = 0$;
- Worst case: when $H_{\text{uninformed}} = 0$ and $H_{\text{informed}} = H_{\max}$, i.e., when the response brings maximal uncertainty to a completely certain case, we have minimal reward (or maximal punishment) $r_t^{act} = -1$.

One may note that actionability is not an inherent property of the response itself but is assessed based on the response's influence on the agent's action selection. Specifically, it evaluates the reduction in entropy of the action probability distribution, reflecting the agent's improved confidence in choosing a specific action. This emphasizes that actionability is a measure of the practical utility of the response in guiding the agent's decisions effectively.

2.3. Accumulation (Long-Term Learning Gain)

In RL, the value function $V(s_t)$ estimates the expected cumulative reward starting from state s_t and following policy π. In OPRA-RL, value function is defined as:

$$V(s_t) = \mathbb{E}[\sum_{i=t}^{\infty} \gamma^{i-t} \cdot r_i \mid s_t = (O_t, P_t, R_t, G_t), \pi],$$

where γ is the discount factor and r_i is the reward at time step i.

The Q-function $Q(s_t, a_t)$ in RL estimates the expected cumulative reward of taking action a_t in state s_t and then following policy π. In the OPRA-RL framework, it becomes:

$$Q(s_t, a_t) = \mathbb{E}\left[r_t + \gamma \cdot \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) \mid s_t = (O_t, P_t, R_t, G_t), a_t\right].$$

To compute the long-term learning gain (or shorter – "accumulation") component of the reward (r_t^{acc}), we measure how much the external response R_t improves the agent's future decision-making ability.

This is achieved by comparing the Q-values with and without R_t, scaled to the interval [-1,1] as follows:

$$r_t^{acc} = \frac{Q(s_t, a_t|R_t) - Q(s_t, a_t|\emptyset)}{\Delta Q_{\max}}, \qquad (5)$$

where:
$Q(s_t, a_t|R_t)$ is Q-value with the external response R_t;
$Q(s_t, a_t|\emptyset)$ is Q-value without the external response ($\emptyset$ denotes no response);
$\Delta Q_{\max} = \max_{s_t, a_t} Q(s_t, a_t|R_t) - \min_{s_t, a_t} Q(s_t, a_t|\emptyset)$ is the maximum achievable range of improvement due to R_t.

Formula (5) explicitly compares the Q-values to assess the response's impact and aligns conceptually and numerically with other reward components like actionability and relevance. The formula is scaled so that $r_t^{acc} = 1$ corresponds to the maximal improvement, $r_t^{acc} = -1$ indicates a maximal loss, and $r_t^{acc} = 0$ means no change.

2.4. Efficiency

The efficiency component of the reward evaluates how effectively the agent generates prompts to interact with external knowledge sources. It aims to minimize unnecessary prompts and promotes the formulation of concise, high-quality prompts that faster lead to actionable responses. The efficiency score r_t^{eff} is computed based on the number T of prompt-response iterations required, considering the cost C (i.e., needed time) per one prompt-response iteration and a maximal cost (time) threshold $C_{\max}$ for the whole dialogue (i.e., allowed maximal time to get the intended final response). The efficiency is computed as follows:

$$r_t^{eff} = \frac{C_{max} - T \cdot C}{C_{max} + T \cdot C}, \qquad (6)$$

where:

- $T \cdot C$ is the actual total time cost of the prompt-response iterations;
- C_{max} is the maximum time allowed for the entire process resulting in the intended response.

Formula (6) ensures that the efficiency score is bounded within [-1, 1] interval. When $T = 0$, indicating perfect efficiency (no prompting needed), the reward is 1. As $T \cdot C$ increases, approaching the maximum allowed time C_{max} the reward decreases reaching 0. Further growth of T to infinity will result to the reward decrease towards -1, reflecting inefficiency.

Now we have to define the unified knowledge reward $r_t^{K_{ext}}$ for formula (1) as combination of the following components:

- r_t^{prompt} (formula (2)) : relevance (primary) or the prompt-goal alignment, which ensures the agent formulates queries that are focused on solving the task;
- $r_t^{responce}$ (formula (3)): relevance (secondary) or the response-goal alignment, which evaluates how useful the external knowledge is for advancing the goal;
- r_t^{act} (formula (4)): actionability, which promotes queries and responses that lead to clear and actionable decisions;
- r_t^{acc} (formula (5)): accumulation (Learning Gain) incentivizes the agent to not only achieve its immediate goal but to learn in a way that benefits future decision-making, optimizing long-term performance;
- r_t^{eff} (formula (6)): efficiency, which rewards minimizing unnecessary or inefficient queries, promoting fast and direct knowledge acquisition.

The integration formula is as follows:

$$r_t^{K_{ext}} = \omega_{prompt} \cdot r_t^{prompt} + \omega_{response} \cdot r_t^{response} + \omega_{act} \cdot r_t^{act} + \omega_{acc} \cdot r_t^{acc} + \omega_{eff} \cdot r_t^{eff}, \quad (7)$$

where $0 < \omega_{prompt}, \omega_{response}, \omega_{act}, \omega_{acc}, \omega_{eff} < 1$, and $\omega_{prompt} + \omega_{response} + \omega_{act} + \omega_{acc} + \omega_{eff} = 1$.

This combined knowledge reward ensures that the agent is incentivized not just to query external knowledge but to do so purposefully, efficiently, and with a clear focus on improving both immediate and long-term decision-making in line with its goals. This makes OPRA-RL agents more effective and adaptive in complex environments.

Parameter λ from formula (1) balances the influence of the external knowledge reward ($r_t^{K_{ext}}$) relative to the environment reward (r_t^{env}). It controls how much weight is given to the knowledge aspect in the overall reward computation. The weights ($\omega_{prompt}, \omega_{response}, \omega_{act}, \omega_{acc}, \omega_{eff}$) from formula (7) determine the relative importance of individual knowledge components within $r_t^{K_{ext}}$. Their equal to 1

sum ensures a normalized contribution to r_t^{Kext}.

The value of weighting factor λ in Equation (1) can be adjusted empirically to reflect the characteristics of a specific domain. For instance, higher λ may be useful in environments where external knowledge significantly boosts performance (e.g., sparse reward settings), while lower λ can help stabilize learning in highly reactive environments with dense feedback. This balance allows OPRA-RL to flexibly adapt its learning priorities based on external guidance versus direct interaction. The selection of the five reward components—prompt relevance, response relevance, actionability, learning gain (accumulation), and efficiency—with corresponding weights is conceptually motivated by the OPRA-RL framework's emphasis on the quality of querying. Each component represents a distinct dimension of prompt effectiveness: relevance ensures alignment with goals, actionability focuses on decision support, accumulation encourages strategic learning, and efficiency prevents wasteful querying. These criteria were chosen to reflect the minimal but sufficient dimensions of useful external communication for real-time agents. The associated weights allow the designer to prioritize these dimensions differently depending on application context. For example, high-stakes domains may emphasize actionability and learning gain, while high-speed systems may prefer prompt efficiency. These weights can be chosen through expert tuning, meta-learning, or optimization strategies such as Bayesian search or grid search. Their sum is normalized to one to preserve the interpretability and stability of the combined reward signal during training.

3. Balancing Internal and External Knowledge in OPRA-RL Framework

The diversity in querying external sources, such as ChatGPT, within OPRA-RL opens up several distinct models and approaches that impact learning, rewards, and policies. OPRA-RL integrates a reward function that balances traditional environmental rewards with external knowledge rewards, guided by the parameter λ, which emphasizes components like relevance, actionability, accumulation, and efficiency. Let us consider some essential variations of the OPRA-RL framework:

- *Querying for additional observations (ChatGPT as auxiliary sensor)*: The agent asks for observations that its own sensors or environment model cannot capture. For example, the agent might lack certain sensor capabilities and rely on ChatGPT to describe relevant external conditions (like remote sensors, unseen dynamics, or data from another environment). ChatGPT serves as an auxiliary sensor. The reward in this model is influenced by the relevance and efficiency of the additional observations in improving decision-making. Effective integration of these observations into enriched state representations can lead to improved rewards through better actions, reflecting accumulation and actionability of external inputs.
- *Querying for additional context (knowledge of the environment)*: The agent seeks knowledge about the environment's current state or future trends that it is unaware of, but ChatGPT may provide. ChatGPT acts as a dynamic knowledge base, filling gaps in the agent's understanding with relevant and actionable context. The reward reflects the relevance and utility of the information retrieved, emphasizing how effectively the agent crafts queries to acquire useful knowledge. By aligning external knowledge with internal decision-making processes, the agent maximizes rewards, balancing accumulation and actionability in its strategies.
- *Querying for decision-making support*: The agent prompts ChatGPT for suggested actions or guidance based on its observations and goals. ChatGPT acts as an advisor, and the agent balances between following this advice and relying on its own learned policy. This aligns with the OPRA-RL approach, where action selection depends on λ, controlling the blend of policy-driven decisions and external advice. The reward here captures both the quality of the agent's execution and the effectiveness of the advice, reflecting the interplay of internal competence and external utility. The agent refines its querying and execution strategy to maximize rewards, gradually adjusting its reliance on external advice versus its internal policy as it gains confidence and competence. This dynamic adjustment ensures that the agent becomes more autonomous over time while still utilizing external expertise if needed, embodying the OPRA-RL principles.

In all these models, including their possible hybrids, rewards are distributed to reflect both, the internal competence (how well the agent's learned policy contributes to achieving tasks based on its own observations) and the external utility (the effectiveness of prompting ChatGPT for valuable observations, context, or advice, highlighting relevance, actionability, and efficiency). These components introduce a specific reward structure, emphasizing not only task performance but also the quality of external interactions. This diversity enables OPRA-RL to support varied learning dynamics: enhancing environmental perception, strategic foresight, or decision-making by combining internal learning with external insights. Each model demonstrates OPRA-RL's adaptability for domains requiring robust integration of environment-driven and knowledge-driven strategies.

Let us consider examples for each of the OPRA-RL options (models) within an industrial context (e.g., monitoring, diagnostics, maintenance, and hybrid case) to illustrate the collaborative decision-making between the agent and ChatGPT. These examples also highlight the reward structures based on how the agent queries ChatGPT and incorporates the resulting information.

Querying for additional observations (smart transportation example): In a smart city transportation system, an agent manages a fleet of autonomous buses. Each bus is equipped with sensors for local traffic density and weather conditions (e.g., rain or fog). However, the system lacks visibility into conditions on adjacent highways or nearby urban areas that could impact routes or schedules. When faced with partial information, the agent queries ChatGPT for additional observations by synthesizing publicly accessible real-time data (e.g., traffic cameras, weather stations, and satellite feeds) via APIs and reports. For example, the agent detects slow movement in its assigned route but does not know if it is a local issue or part of a broader congestion trend. ChatGPT aggregates relevant external sensor data, confirming severe congestion due to a highway accident. The agent uses this external observation to reroute buses efficiently. The reward reflects the agent's ability to identify relevant queries and its successful route optimization using combined internal and external observations, balanced by λ.

Querying for additional context (industrial diagnostics example): Assume that, in a smart factory, an agent monitors the performance of a robotic assembly line using embedded diagnostics. While it can detect real-time inefficiencies, it lacks access to historical maintenance data and usage patterns, which are critical for diagnosing recurring issues. For instance, the agent observes a robotic arm's efficiency declining due to reduced joint speed. Unable to determine the root cause with its onboard system, it queries ChatGPT for contextual knowledge. ChatGPT retrieves historical data from the factory's central database, including past maintenance logs and records of similar robots' performance under different operating conditions. ChatGPT reveals that the robotic arm has a history of wear-related issues during peak operation cycles, especially in the joint mechanism. Combining its own observations with this additional context, the agent identifies the issue as joint fatigue and schedules a targeted repair. The reward reflects both the agent's diagnostic success based on internal data and the efficiency gained through contextual knowledge from ChatGPT. The total reward is weighted by the λ parameter, emphasizing the role of external context in optimizing decisions.

Querying for decision-making support (predictive maintenance example): In an industrial plant, an agent monitors machinery to optimize maintenance schedules based on real-time sensor data (e.g., temperature, pressure, usage cycles, etc.). While the agent learns a policy to determine preventive maintenance timing, it occasionally encounters complex scenarios requiring strategic trade-offs. For instance, the agent observes increased wear in a machine component but is unsure if immediate maintenance is necessary. To refine its decision, it queries ChatGPT with its observations, recent trends, and operational goals (e.g., minimize downtime and costs). ChatGPT analyzes this context and recommends delaying maintenance until after the next production cycle, citing low immediate risk and aligning with operational efficiency goals. The agent follows this advice and successfully avoids unnecessary downtime while scheduling timely maintenance. If the agent's original decision achieves a positive outcome, it receives some base reward. If ChatGPT's guidance improves decision quality (e.g., minimizing disruptions), the agent gains an additional external advice reward, weighted by λ. This demonstrates how the agent integrates external advice to enhance its decision-making policy and adapt to complex industrial conditions.

Hybrid model in OPRA-RL framework (factory equipment diagnosis and maintenance example): In the OPRA-RL framework, an agent is capable of dynamically employing all three querying models—additional observations, additional context, and decision-making support—within a single mission. This hybrid approach allows the agent to adaptively manage tasks by combining its internal capabilities with external expertise to optimize outcomes. Consider, for example, an agent, which is tasked with diagnosing and addressing a fault in a robotic assembly line. During the first stage (querying for additional observations), the agent detects irregular vibrations in a conveyor motor but lacks detailed acoustic data. It queries ChatGPT, which accesses readings from remote high-resolution acoustic sensors. ChatGPT confirms abnormal noise levels, suggesting early-stage bearing degradation. Assume that for leveraging external observations, the agent receives +0.4 reward. During the second stage (querying for additional context) and aiming to refine its diagnosis, the agent requests historical failure logs and maintenance trends for similar motors. ChatGPT provides data showing that similar issues often stem from overheating, particularly during peak production hours. Assume that successfully incorporating context for precise diagnostics earns the agent an additional reward of +0.6. Assume that during the third stage (querying for decision-making support), the agent needs to decide between immediate maintenance (causing downtime) and delaying repairs. It queries ChatGPT for recommendations based on production goals and operational efficiency. ChatGPT advises delaying repairs by one shift, as the risk of failure remains low. The agent implements this advice and schedules maintenance during off-peak hours,

optimizing downtime and cost. Assume that following ChatGPT's optimal decision guidance grants +0.8. The agent's total reward reflects the successful integration of all three querying models to handle the task efficiently, calculated by weighing individual rewards and their impact on overall task performance. This hybrid approach showcases the agent's adaptive capability to balance its own knowledge and actions with external expertise, achieving superior outcomes.

Each example illustrates distinct querying intentions in OPRA-RL framework:

- Additional observations involve querying for inaccessible sensory data, enhancing environmental awareness and diagnostic accuracy. External information enriches state representation, directly influencing action selection and policy updates.
- Additional context queries retrieve historical or external knowledge to improve strategic understanding, shaping long-term strategies over immediate actions.
- Decision-making support queries seek direct advice to optimize specific actions, balancing the agent's learned policy with external guidance.
- Hybrid approaches integrate all three, contributing to learning and mission execution.

These scenarios highlight ChatGPT's collaborative role in industrial decision-making, emphasizing OPRA-RL's focus on internal learning and external knowledge integration.

4. OPRA-RL as a Q-Learning Algorithm

As it was mentioned above, in RL, the Q-value (or action-value) represents the expected cumulative reward an agent can obtain from a given state s_t by performing a specific action a_t and then following an optimal policy. It is a critical concept as it helps the agent evaluate and compare possible actions to select the one that maximizes the long-term rewards.

As we have seen already, the general RL formulation for Q-values can be expressed as:

$$Q(s_t, a_t) = \mathbb{E}\left[r_t + \gamma \cdot \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) \mid s_t, a_t\right].$$

Here, the expectation operator $\mathbb{E}$ reflects the probabilistic nature of state transitions and rewards.

Q-learning [12] is a widely used, model-free RL algorithm that learns the Q-values iteratively without requiring a model of the environment [13]. Its main advantage is the ability to learn optimal action policies directly from experience.

In Q-learning, the Q-value is updated using a temporal difference learning approach:

$$Q_{new}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \cdot \left[r_t + \gamma \cdot \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right],$$

where:

- α (learning rate): determines the rate at which new information is incorporated relative to prior knowledge;
- γ (discount factor): assigns weight for future rewards while integrating both internal and externally influenced strategies.

In OPRA-RL, the Q-learning paradigm is extended to incorporate external knowledge queries. This integrates observations, contextual insights, and decision support via prompts (e.g., to ChatGPT). The modified Q-value update in OPRA-Q-Learning is:

$$Q_{new}(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \cdot \left[r_t^{env} + \lambda \cdot r_t^{K_{ext}} + \gamma \cdot \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \varepsilon) - Q(s_t, a_t)\right], \quad (8)$$

where:

- r_t^{env} : reward from the environment;
- $r_t^{K_{ext}}$: reward from knowledge prompts, scaled by λ;
- ε : exploration rate governing the balance between exploring new actions and exploiting learned ones.

Specific roles of hyperparameters in OPRA-Q-Learning are as follows:

- α determines the weight given to new information, encompassing both environmental observations and external prompts. A higher α prioritizes new updates, while a lower α ensures gradual integration for stable learning;
- γ balances immediate versus long-term rewards, encouraging strategic planning. Higher values favor long-term planning, while lower values focus on short-term gains;

- ε promotes exploration of novel actions and queries, preventing the agent from getting stuck in local optima by balancing exploration and exploitation.

Therefore, in OPRA-Q-Learning, the hyperparameters govern updates from both observed and prompted data, ensuring that the Q-value reflects a balance of environmental interactions and external advisory knowledge. Through this integration of traditional RL principles with modern capabilities like prompting, OPRA-Q-Learning ensures robust decision-making in dynamic environments by balancing environmental interactions and external advisory knowledge.

5. Testing OPRA-Q-Learning through the OPRA-POKER Game

This section evaluates the OPRA-Q-Learning framework using the "OPRA-POKER" game, which we introduce in this paper as a partially observable environment designed to simulate strategic decision-making with external advice. The game focuses on balancing the costs of information gathering, rewards, and optimal actions, illustrating the adaptability of the OPRA framework. Consider the main components of the game logic written below and visualized in Figure 1.

State space:

Three face-down cards (A, B, and C), each with two possible states: {A1, A2}, {B1, B2}, and {C1, C2}. Each combination (e.g., A1, B2, C1) represents a unique state with predefined rewards, ranging from -100 to +100, with an overall zero sum across all combinations.

Agent actions:

- Open all at once: Fully reveals all three cards at a cost of -10 points, providing complete observability. (Using only this action indicates an "extreme policy").
- Skip iteration: Ends the round early, incurring a loss of -3 points.
- Open 1st card (A): Observes card A at a cost of -1 point. The next steps depend on its state.
- Ask ChatGPT (card B): Queries ChatGPT for the state of card B for -5 points. This option is available only after observing card A.
- Open card C: Reveals the third card after learning the states of A and B, with no cost.

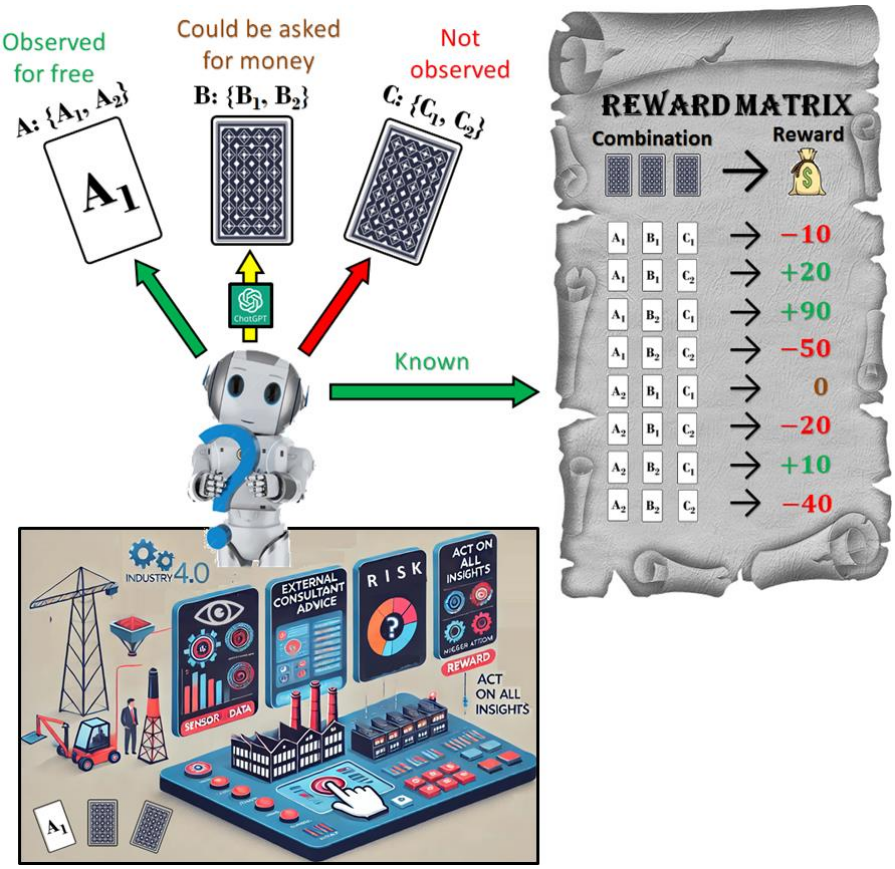

Figure 1. Logic of OPRA-POKER as an OPRA-RL/OPRA-Q-Learning simulation game visualized

Reward mechanism:

The agent receives the predefined reward for fully observing all cards, with observation (A and B) and advice (ChatGPT) costs subtracted.

Learning dynamics:

- Observation (O): The agent observes card A.
- Prompt (P): The agent queries ChatGPT for card B's state.
- Response (R): ChatGPT provides the state of card B.
- Action (A): The agent decides whether to open card C or skip the iteration.

Trade-offs:

- Information gathering: The agent must decide when to observe or query external advice.
- Risk management: The agent balances the costs of gathering information against potential rewards or losses.
- *Note*: The choice between observation, prompting, or decision support is not hardcoded but emerges from learning. The agent treats these query types as distinct actions in the Q-learning process, each associated with a different cost and informational benefit. As Q-values evolve, the agent implicitly learns when a low-cost observation is sufficient, when a higher-cost prompt is justified, and when the "Open All" shortcut is too risky. Thus, query selection is governed by experience-based optimization rather than static rules.

Implementation:

- Environment setup: The reward table for all $2^3 = 8$ states is initialized with rewards summing to zero. Actions include "Open all," "Skip," "Open A," "Ask ChatGPT," and "Open C."
- Algorithm: A Q-learning algorithm updates the action-value function based on three key hyperparameters as shown in previous section (formula (8)).
- Evaluation: The agent learns optimal strategies over 1000 episodes. Performance metrics include cumulative rewards and action selection frequencies.
- *Note*: The tested hyperparameter configurations (learning rate α, discount factor γ, exploration rate ε) were selected to examine how varying learning dynamics influence the agent's performance under partial observability and cost-sensitive querying.

The OPRA-Q-Learning framework was tested across three runs (RUN-I, RUN-II, RUN-III) using randomly initialized 0-sum reward matrices. Each run consisted of eleven experiments (each repeated three times), assessing the learned policy against an extreme policy that consistently chose the "Open All" action, resulting in the worst-case total reward (-10,000). In this context, subtracting the worst-case outcome serves as a normalization step: it highlights how much better the learned policy performs relative to an uninformed baseline, rather than focusing on absolute reward values. Since the "Open All" policy leads to the most severe penalty in expectation, it provides a meaningful anchor to interpret improvements. The experiments evaluated how varying hyperparameters α (learning rate), γ (discount factor), and ε (exploration rate) influenced total rewards.

The results are summarized in Table 1, Table 2, and Table 3, which detail the total rewards of both policies, along with the learned policy's advantage over the extreme policy. Across all runs, the learned policy consistently outperformed the extreme policy, particularly with hyperparameter configurations that balanced exploration and exploitation effectively.

Note: For these tables, we selected a representative subset of 11 state combinations to highlight diverse reward dynamics without overwhelming the search space. This choice facilitates visualizing learning trends while preserving tractability. Using finer granularity such as 0.01 steps would result in over 9 million combinations, motivating future work based on function approximation and deep RL. Results shown within each cell in the tables are the average of three runs with different random seeds. This experiment is designed as a conceptual validation rather than a statistical performance benchmark.

RUN-I Results (Table 1):

- Reward values in the reward matrix ranged from −58 to 99, with a standard deviation (σ) of 52.1.
- The learned policy outperformed the extreme policy in all experiments, with advantages ranging from 2,244 to 9,450.
- The best result was achieved with parameters ($\alpha = 0.15, \gamma = 0.95, \varepsilon = 0.1$), yielding a total reward of –550 and an advantage of 9,450 over the extreme policy. This means that winning configuration of parameters proved most effective across multiple runs, and attributed to the combination of stable learning (low α), long-term planning (high γ), and reduced reliance on random exploration (low ε), which together encourage cautious, information-efficient strategies that maximize cumulative reward.

- Lower exploration rates (ε) and higher discount factors (γ) contributed significantly to better performance.

Table 1. Summary of experimental results for OPRA-Q-Learning testing. RUN-I on the following randomly initialized 0-sum reward matrix: {(0, 0, 0) → -58, (0, 0, 1) → 0, (0, 1, 0) → 33, (0, 1, 1) → 7, (1, 0, 0) → 99, (1, 0, 1) → -57, (1, 1, 0) → 32, (1, 1, 1) → -56}. Range of the rewards: [-58, 99]. Standard deviation of the rewards $\sigma \approx 52.1$.

Experiment	α	γ	ε	Total sum of rewards (learned policy)	Total sum of rewards (extreme policy)	Advantage of learned policy over extreme policy
1	0.1	0.9	0.1	-6711	-10000	3289
2	0.2	0.8	0.2	-1995	-10000	8005
3	0.3	0.7	0.3	-1419	-10000	8581
4	0.4	0.6	0.4	-1736	-10000	8264
5	0.5	0.5	0.5	-4442	-10000	5558
6	0.6	0.4	0.6	-2778	-10000	7222
7	0.7	0.3	0.7	-6112	-10000	3888
8	0.8	0.2	0.8	-5522	-10000	4478
9	0.9	0.1	0.9	-5487	-10000	4513
10	1.0	0.0	1.0	-7756	-10000	2244
11 - best	**0.15**	**0.95**	**0.1**	**-550**	**-10000**	**9450**

RUN-II Results (Table 2):

- Reward values ranged from −89 to 102, with a standard deviation (σ) of 63.2.
- The learned policy demonstrated substantial advantages over the extreme policy, with the best advantage reaching 11,222.
- Optimal parameters ($\alpha = 0.15, \gamma = 0.95, \varepsilon = 0.1$) resulted in a total reward of 1,222, the highest among all configurations.
- Intermediate learning rates ($\alpha = 0.2$ to 0.5) and moderate exploration rates ($\varepsilon = 0.1$ to 0.4) led to consistently strong results.
- The performance sharply declined for high exploration rates ($\varepsilon \geq 0.6$).

Table 2. Summary of experimental results for OPRA-Q-Learning testing. RUN-II on the following randomly initialized 0-sum reward matrix: {(0, 0, 0) → -70, (0, 0, 1) → -7, (0, 1, 0) → 59, (0, 1, 1) → -89, (1, 0, 0) → 60, (1, 0, 1) → -23, (1, 1, 0) → -32, (1, 1, 1) → 102}. Range of the rewards: [-89, 102]. Standard deviation of the rewards $\sigma \approx 63.2$.

Experiment	α	γ	ε	Total sum of rewards (learned policy)	Total sum of rewards (extreme policy)	Advantage of learned policy over extreme policy
1	0.1	0.9	0.1	196	-10000	10196
2	0.2	0.8	0.2	702	-10000	10702
3	0.3	0.7	0.3	468	-10000	10468
4	0.4	0.6	0.4	992	-10000	10992
5	0.5	0.5	0.5	1137	-10000	11137
6	0.6	0.4	0.6	1011	-10000	11011
7	0.7	0.3	0.7	-1682	-10000	8318
8	0.8	0.2	0.8	-3811	-10000	6189
9	0.9	0.1	0.9	-8514	-10000	1486
10	1.0	0.0	1.0	-3868	-10000	6132
11 - best	**0.15**	**0.95**	**0.1**	**1222**	**-10000**	**11222**

RUN-III Results (Table 3):

- Reward values ranged from −86 to 56, with a standard deviation (σ) of 41.3.
- The learned policy achieved the highest advantage of 10,979 with parameters ($\alpha = 0.3, \gamma = 0.7, \varepsilon = 0.3$).
- An alternative set of parameters ($\alpha = 0.15, \gamma = 0.95, \varepsilon = 0.1$), which was the winner of previous runs, achieved a slightly lower advantage of 9,805, highlighting the flexibility of the framework.

- Moderate parameter configurations ($\alpha = 0.2$ to 0.4, $\gamma = 0.6$ to 0.8, $\varepsilon = 0.2$ to 0.4) yielded the most robust outcomes.
- Performance declined rapidly when the exploration rate exceeded $\varepsilon = 0.5$, suggesting diminishing returns from exploration.

Table 3. Summary of experimental results for OPRA-Q-Learning testing. RUN-III on the following randomly initialized 0-sum reward matrix: {(0, 0, 0) → -26, (0, 0, 1) → 29, (0, 1, 0) → -86, (0, 1, 1) → 56, (1, 0, 0) → 34, (1, 0, 1) → -19, (1, 1, 0) → 8, (1, 1, 1) → 4}. Range of the rewards: [-86, 56]. Standard deviation of the rewards $\sigma \approx 41.3$.

Experiment	α	γ	ε	Total sum of rewards (learned policy)	Total sum of rewards (extreme policy)	Advantage of learned policy over extreme policy
1	0.1	0.9	0.1	191	-10000	10191
2	0.2	0.8	0.2	668	-10000	10668
3 - best	**0.3**	**0.7**	**0.3**	**979**	**-10000**	**10979**
4	0.4	0.6	0.4	715	-10000	10715
5	0.5	0.5	0.5	-1831	-10000	8169
6	0.6	0.4	0.6	-2410	-10000	7590
7	0.7	0.3	0.7	-3051	-10000	6949
8	0.8	0.2	0.8	-4527	-10000	5473
9	0.9	0.1	0.9	-5649	-10000	4351
10	1.0	0.0	1.0	-6310	-10000	3690
11	0.15	0.95	0.1	-195	-10000	9805

In all the experiments, the ranges selected for α, γ, and ε all fall within the standard [0.0, 1.0] interval commonly used in RL. This inclusive interval allows exploration of both extreme behaviors (e.g., purely reactive updates, or fully greedy exploitation) and more balanced strategies. In particular, intermediate values of ε ($0.2 \div 0.4$) consistently produced the best results across our experiments. This aligns with RL theory: such values balance the exploration–exploitation trade-off, where too little exploration ($\varepsilon \approx 0.0$) can lead to premature convergence on suboptimal policies, and too much (ε close to 1.0) causes erratic, unfocused learning. By tuning within [0.0, 1.0], we could empirically confirm that moderate exploration allows the agent to benefit from both environmental signals and external prompts without destabilizing learning.

General observations:

- Lower exploration rates ($\varepsilon \leq 0.4$) combined with higher discount factors ($\gamma \geq 0.7$) generally produced better results across all runs.
- The learning rate (α) influenced convergence speed but had a secondary impact compared to γ and ε.
- Variability in reward distributions (e.g., wider ranges and higher σ) correlated with higher potential advantages for the learned policy over the extreme policy.
- The framework consistently produced significant advantages over the extreme policy, demonstrating its robustness and adaptability to different reward matrices.
- The OPRA-Q-Learning framework successfully optimized policies in diverse scenarios. Future research could explore adaptive parameter tuning to further enhance performance and reduce sensitivity to initial parameter configurations

Learning trends discovered from the experiments:

- Cumulative rewards: As the agent balances observation and advice costs with state-dependent rewards, cumulative rewards increase.
- Optimal strategies: The agent gradually prefers querying ChatGPT during high-stakes rounds.

Additional observations:

- The best result was achieved due to contribution from an external advice, highlighting the value of integrating external knowledge.
- The OPRA-Q-Learning framework is effective in partially observable environments with decision costs, as demonstrated by the OPRA-POKER game. The agent successfully balances exploration with exploitation of external advice, which makes the framework suitable for complex, dynamic tasks found in Industry 4.0 environments.

- OPRA-Q-Learning is applicable to tasks requiring a trade-off between exploration, exploitation, and external knowledge, such as manufacturing optimization, predictive maintenance, and resource management in Industry 4.0.

Conclusive remark regarding the experiments:

The OPRA-Q-Learning framework proves effective for adaptive decision-making tasks that incorporate external knowledge sources. The OPRA-POKER game serves as a controlled demonstration of trade-offs in a stylized environment. While it does not aim to rival competitive multi-agent Poker AIs (such as DeepMind's or Libratus), it provides an abstraction to study adaptive decision-making under uncertainty and cost constraints—elements found in real-world applications.

It is important to emphasize that the primary aim of our study is not to deliver statistically optimized or benchmark-level performance, but rather to provide a proof-of-concept demonstration of the OPRA-Q-Learning framework in action. Each value reported in Tables 1–3 reflects the average outcome over three independent runs with different random seeds. Given that our focus lies in illustrating qualitative behaviors — specifically, the benefit of incorporating informed prompting over blind decision-making — we have intentionally avoided full-scale significance testing. In our experiments, we do not simulate the actual latency of external queries. Each interaction is treated as a symbolic call to an oracle-like LLM with assumed instantaneous response. Future work should incorporate realistic delays, batch scheduling, and asynchronous architectures to better assess OPRA-RL in latency-sensitive environments. All these aligns with the early-stage and illustrative nature of our contribution, and avoids conveying a false sense of precision or generalizability. Future work will extend this foundation to more comprehensive experimental protocols with statistical testing and larger-scale reproducibility analysis.

OPRA-POKER is not just a game (Industry 4.0 example):

OPRA-POKER looks like a simplified game designed and presented through the metaphor of cards to illustrate the core principles of the OPRA-Q-Learning framework, such as balancing exploration, exploitation, and external advice. This abstraction intentionally avoids multi-agent complexity found in standard Poker, focusing instead on individual decision processes under uncertainty. Future work can investigate competitive dynamics or connections to frameworks such as *Student of Games* [14], which combine guided search, self-play learning, and game-theoretic reasoning to address both perfect and imperfect information games in multi-agent settings. While the game offers a basic environment to demonstrate the core OPRA-RL concepts, it serves as a foundation for more complex, real-world scenarios. In industrial settings, tasks often involve dynamic, partially observable environments with higher stakes, more sophisticated decision-making, and diverse external inputs. Thus, OPRA-Q-Learning can be adapted to tackle such challenges, making it highly relevant to fields like manufacturing, resource management, and predictive maintenance in Industry 4.0, like it is conceptually shown in Figure 1.

The structural parallels between OPRA-POKER and industrial scenarios are intentional and central to the framework's generalizability. In both contexts, the agent operates under partial observability, must manage information acquisition costs, and navigate complex reward landscapes influenced by latent states. Just as the OPRA-RL agent must decide whether to observe cards or ask for external advice, an industrial agent must determine whether to sense a machine's status, consult an expert system, or act with limited knowledge. This is particularly applicable in smart factories where decisions must be made with incomplete sensor data, time-sensitive actions, and costly diagnostic queries. Therefore, the experimental findings in OPRA-POKER—especially regarding learned strategies to query selectively and exploit advice—are expected to transfer to real-world settings. The card metaphor simplifies these dynamics for clarity, but the learning principles and decision trade-offs directly mirror challenges in industrial automation.

To demonstrate this, consider the following simple industrial example. Like OPRA-POKER, this example involves hidden states, progressive information gathering, and reward trade-offs, but it is grounded in the practical context of resource allocation in a smart factory. Note that this example can be scaled to include more resources (akin to adding more "cards"), a more complex state-space, or advanced reward structures, reflecting the intricate dynamics of real-world industrial systems. The resulting reward landscape could be viewed as a high-dimensional function, motivating future integration of deep RL for function approximation and generalization across states.

In a smart factory, a production scheduler agent optimizes the use of three key resources—Material A, Machine B, and Technician C—to fulfill manufacturing orders under conditions of uncertainty. Each resource's state (availability, readiness, or cost) determines the efficiency of the production process, and the scheduler must balance the costs of gathering information with the potential rewards of making informed decisions. The factory's operational state is dynamic, and the reward system reflects zero-sum

trade-offs: gains from efficient production are balanced by losses from penalties (e.g., delays, waste, or underutilization).

Assume the following decision flow:

1. Starting the scheduling task:

- Proceed immediately: Blindly allocate Material A, Machine B, and Technician C, facing high uncertainty but with potential for immediate rewards (for an unlikely case if all resources are optimal).
- Investigate material A's availability: Pay a small cost (e.g., querying the inventory system) to gain information about Material A, reducing some uncertainty.
- Postpone the task: Skip scheduling this order, minimizing immediate penalties but potentially losing future opportunities.

2. After investigating Material A and based on its availability:

- Request external expertise for Machine B: Consult an external diagnostic tool (at a higher cost) to assess Machine B's readiness. This additional insight can help evaluate whether Technician C is likely to be the limiting factor.
- Proceed with scheduling: Allocate Machine B and Technician C immediately, based on Material A's availability. This strategy assumes that Technician C is likely in a favorable state, but the uncertainty about C introduces a calculated risk.
- Abandon task: Skip scheduling altogether.

3. After assessing machine B and being armed with knowledge about Material A and Machine B, the scheduler must weigh the remaining risk associated with Technician C:

- Finalize decisions: Allocate resources and schedule the order, accepting the residual uncertainty about Technician C's state.
- Adapt the task: If the information about Machine B indicates potential misalignment with Technician C, refine the scheduling to mitigate risks (e.g., reschedule for another shift).

One may see that Technician C (Card C) remains hidden throughout most of the decision-making process, representing the element of residual uncertainty. The scheduler's awareness of Machine B (Card B) indirectly guides expectations about Technician C's state due to correlations observed in past operations (e.g., certain machine issues often require a specific skill set from Technician C).

For example:

- If Machine B is in a suboptimal state, Technician C is likely critical; and blind allocation poses higher risks.
- If Machine B is fully functional, the likelihood of Technician C's state influencing the task diminishes, making it safer to proceed without further investigation.

Such a scenario could be easily mapped to OPRA-POKER:

- Material A, Machine B, and Technician C correspond to Cards A, B, and C in OPRA-POKER.
- Observing Material A is equivalent to revealing Card A, with a small cost.
- Requesting advice about Machine B matches asking ChatGPT about Card B, incurring a higher cost.
- Technician C (like Card C) introduces residual uncertainty, where decisions must consider correlated risks and rewards.

This analogy captures the intrigue and decision-making dynamics of OPRA-POKER in a real-world Industry 4.0 setting. The production scheduler must manage resource constraints, operational risks, and trade-offs between information costs and potential reward maximization.

Similar logic applies broadly to scenarios like:

- Production line optimization, where resource availability varies dynamically.
- Predictive maintenance, where diagnostic tools guide decisions under incomplete knowledge of system states.
- Supply chain management, balancing costs and risks in allocating limited resources.

The correlation between Machine B and Technician C in this example highlights how incremental knowledge acquisition helps to address the most significant risks while leaving some decisions to intuition or learned policy.

6. OPRA-RL within the Related Work Context

RL is a central paradigm in AI, where agents interact with environments to maximize cumulative rewards through sequential actions [8]. It differs from other ML methods, like supervised learning, by emphasizing exploration, long-term planning, and adaptability in uncertain environments. This makes RL valuable in areas such as robotics [15], autonomous vehicles [16], and complex games [17], where long-term strategies are critical. Historically, RL has evolved from psychological concepts like operant

conditioning [18] and the law of effect [19], alongside contributions from control systems theory [20] (Bellman, 1957), which principles laid the basics for dynamic optimization and decision-making models like the Bellman equation.

Recent RL breakthroughs, powered by deep learning and vast datasets, have led to significant applications in real-world challenges, from gaming to industrial automation [21, 22]. RL's capacity to adapt and optimize under uncertainty is driving advancements in smart manufacturing and Industry 4.0, where it plays a key role in optimizing resource allocation, predictive maintenance, and production efficiency.

In this context, OPRA-RL introduces a novel approach by integrating external knowledge, such as advice from language models like ChatGPT, into RL frameworks. This hybrid approach enhances decision-making in dynamic, partially observable environments, offering a promising direction for industrial applications.

6.1. Modern RL approaches vs OPRA-RL

OPRA-RL integrates external expertise, such as ChatGPT, into traditional RL frameworks, improving robustness, flexibility, and adaptability. By balancing external advice costs with exploration and exploitation, OPRA-RL excels in dynamic, partially observable, and high-stakes environments, making it well-suited for Industry 4.0 applications. Most notable and relevant RL-driven frameworks to be compared to OPRA-RL are as follows.

6.1.1. Deep Q-Learning (DQL)

DQL [23] uses deep neural networks to approximate Q-values, enabling scalability in complex environments. However, it struggles in partially observable environments and lacks external input [24]. OPRA-RL addresses this by incorporating external advice to guide decision-making, balancing exploration with the cost of external knowledge.

6.1.2. Proximal Policy Optimization (PPO)

PPO [25] optimizes policies through stabilized updates, ensuring robust performance in continuous and discrete action spaces. While PPO performs well in a variety of tasks, it does not integrate external domain knowledge. OPRA-RL enhances PPO by incorporating external knowledge to improve decision quality, particularly in dynamic or less understood environments, reducing the need for extensive environment modeling.

6.1.3. Soft Actor-Critic (SAC)

SAC [26] maximizes reward-entropy trade-offs, promoting exploration in stochastic environments. However, its dual Q-networks can be computationally intensive, and exploration can be inefficient in sparse reward settings [27]. OPRA-RL enhances SAC by using external advice to guide exploration, improving policy learning and reducing inefficiency in sparse environments.

6.1.4. AlphaZero

AlphaZero [17] combines RL with Monte Carlo Tree Search to learn optimal strategies in well-defined environments. While it excels in structured tasks like board games, it is computationally expensive and impractical in uncertain or real-time settings. OPRA-RL, on the other hand, incorporates external advice, enabling efficient decision-making in uncertain, partially observable environments without exhaustive tree searches.

6.1.5. Hierarchical RL (HRL)

HRL [28] decomposes tasks into high-level policies and low-level actions, enabling multi-level decision-making and long-horizon task handling. However, it often relies on predefined subgoals, which can be challenging to identify. OPRA-RL addresses this by dynamically defining subgoals using external advice, improving adaptability and performance in long-horizon tasks.

6.1.6. Multi-Agent RL (MARL)

MARL [29] handles collaborative or competitive tasks with multiple agents. However, it faces challenges such as coordination complexity and non-stationary dynamics [30]. OPRA-RL mitigates these challenges by leveraging external prompts for coordination, reducing learning complexity and enhancing adaptability in multi-agent environments.

6.1.7. Model-Based RL (MBRL)

MBRL [31] uses predictive models to guide decision-making, supporting data efficiency and long-term planning. However, it struggles in dynamic, unpredictable environments where model accuracy is crucial. OPRA-RL complements MBRL by integrating external advice, allowing agents to address uncertainties in real-time without relying on highly accurate predictive models.

6.2. Analogy between Knowledge-Informed ML and OPRA-RL as Knowledge-Informed RL

Knowledge-Informed ML (KIML) and OPRA-RL share the philosophy of augmenting data-driven learning with external knowledge to enhance efficiency, adaptability, and interpretability, positioning OPRA-RL as Knowledge-Informed RL (KIRL).

The KIML framework [32] integrates domain knowledge into ML models, such as physics-informed neural networks (PINNs) [33], which use physical constraints to guide training. This reduces dependency on large datasets and mitigates overfitting, improving robustness in data-scarce environments. Similarly, OPRA-RL integrates external prompts (e.g., from ChatGPT) into RL, addressing limited experience or incomplete observations. External knowledge in KIRL guides exploration, reduces uncertainty, and prioritizes high-reward actions, decreasing the need for trial-and-error and enabling more efficient learning with fewer samples. KIRL's adaptability mirrors KIML's generalization, as domain knowledge helps OPRA-RL adapt quickly in dynamic environments, enhancing decision-making. In this way, OPRA-RL exemplifies the potential of KIRL, enabling faster learning, improved adaptability, and greater robustness, particularly in high-stakes Industry 4.0 applications.

7. Conclusions

This paper introduces the OPRA-RL framework, which integrates RL with external knowledge prompting to address key limitations in dynamic, high-stakes, and partially observable environments. Incorporating structured knowledge queries and responses into the decision-making process enables OPRA-RL agents to navigate complex scenarios with enhanced adaptability, interpretability, and efficiency. By embedding prompts and responses as integral components of the learning loop, the framework addresses the challenge of limited observations, allowing agents to dynamically fill knowledge gaps using external expertise. This approach is particularly suited to scenarios where domain knowledge evolves or remains partially accessible during training. A dual-reward mechanism—rewarding both action selection and effective prompting—strikes a balance between leveraging internal policies and external advice, accelerating learning while maintaining alignment with optimal decision-making strategies. The experiments demonstrate OPRA-RL's potential to address real-world challenges, including applications in Industry 4.0, where agents must operate in environments characterized by uncertainty, high stakes, and dynamic constraints. The integration of RL with external knowledge sources opens new avenues for applications in healthcare, manufacturing, and autonomous systems.

Several important limitations and practical constraints must be acknowledged. First, OPRA-RL depends on external advice sources such as LLMs, which are known to occasionally produce biased, inconsistent, or factually incorrect outputs. In this regard, OPRA-RL incorporates outcome-based reward attribution to dynamically assess and learn the utility of such advice, allowing agents to adjust the weighting or disregard misleading responses over time. However, the framework currently lacks mechanisms for proactive verification or ensemble querying across multiple external sources, which could further improve robustness. Second, real-time deployment of OPRA-RL in time-sensitive settings introduces computational latency concerns. In our current simulation, external advice is modeled abstractly with negligible runtime cost. However, in practice, querying an LLM via an API introduces nontrivial delays (typically, between 0.5–2 seconds per call, depending on load, prompt size, and model configuration). While tolerable in offline or semi-autonomous planning settings, this may be problematic for real-time control loops. One mitigation strategy is to limit advice frequency via learned query policies, using confidence thresholds or epistemic uncertainty to decide when external help is warranted. Caching previously helpful advice and parallelizing query pipelines are also promising optimization avenues. Third, we acknowledge that no quantitative latency benchmarks or system-level performance evaluations have been reported. This is because our study is exploratory in nature and focuses on demonstrating qualitative decision improvements arising from informed prompting. A full engineering analysis, including profiling end-to-end inference pipelines under various deployment conditions (e.g., cloud API vs edge caching), is left for future work. Finally, while OPRA-RL shows conceptual generalizability, further empirical studies are required to validate its scalability across domains with high-dimensional state spaces, multi-agent interaction, or strict real-time requirements.

Building on the OPRA-RL framework, future research will explore its enhancement through integration with the context-driven balanced RL paradigm. This direction aligns with prior work [34] emphasizing the balance of external operational goals and internal health or maintenance needs. The goal is to extend OPRA-RL's capacity for dynamic decision-making by incorporating context-awareness to improve adaptability and resilience in complex environments. The proposed integration envisions agents capable of leveraging external observations, prompting expertise, and balancing contextual rewards across internal and external domains. By transforming RL into a context-aware supervised learning task, agents will not only learn optimal policies but also adaptively harmonize competing goals. This involves utilizing multidimensional contextual attributes—derived from both external environments (e.g., operational metrics) and internal states (e.g., health indicators)—to iteratively guide decisions. Additionally, integrating contextual drivers into OPRA-RL's multifaceted reward structure will enable agents to dynamically weigh trade-offs between intrinsic and extrinsic motivations. Experimental validation will focus on refining the decision-making game to simulate real-world challenges, allowing agents to interact with dynamic external environments while considering their internal states. This extension aims to position OPRA-RL as a robust framework for intelligent systems, bridging self-learning with context-driven adaptability and addressing the demands of Industry 4.0, 5.0, and beyond.

While challenges persist, OPRA-RL showcases the potential to blend RL with external knowledge systems effectively. By shifting the paradigm from purely autonomous learning to knowledge-informed decision-making, this framework supports smarter, more adaptable agents capable of tackling complex real-world challenges. We believe OPRA-RL and its extensions represent a meaningful contribution to the evolving landscape of intelligent systems, inspiring future research and practical innovations in knowledge-augmented AI.

Author Contributions

Methodology, conceptualization, and formal analysis V.T. and O.V.; implementation, experimentation, and validation O.T. and A.T.; writing, review, and editing O.V., O.T., and V.T.; supervision and administration, V.T.

Funding

This research received no external funding.

Conflict of Interest Statement

The authors declare no conflicts of interest.

Data Availability Statement

This study did not generate or analyze any datasets.

References

1. Leng, J., Zhong, Y., Lin, Z., Xu, K., Mourtzis, D., Zhou, X., ... & Shen, W. (2023). Towards Resilience in Industry 5.0: A Decentralized Autonomous Manufacturing Paradigm. *Journal of Manufacturing Systems*, *71*, 95-114. https://doi.org/10.1016/j.jmsy.2023.08.023
2. Lee, J., Ni, J., Singh, J., Jiang, B., Azamfar, M., & Feng, J. (2020). Intelligent Maintenance Systems and Predictive Manufacturing. *Journal of Manufacturing Science and Engineering*, *142*(11), 110805. https://doi.org/10.1115/1.4047856
3. Longo, F., Nicoletti, L., & Padovano, A. (2022). New Perspectives and Results for Smart Operators in Industry 4.0: A Human-Centered Approach. *Computers & Industrial Engineering*, *163*, 107824. https://doi.org/10.1016/j.cie.2021.107824
4. Golovianko, M., Terziyan, V., Branytskyi, V., & Malyk, D., (2023). Industry 4.0 vs. Industry 5.0: Co-existence, Transition, or a Hybrid. *Procedia Computer Science, 217*, 102-113. Elsevier. https://doi.org/10.1016/j.procs.2022.12.206
5. Kumpulainen, S., & Terziyan, V. (2022). Artificial General Intelligence vs. Industry 4.0: Do They Need Each Other? *Procedia Computer Science, 200*, 140-150. Elsevier. https://doi.org/10.1016/j.procs.2022.01.213
6. Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., ... & Xie, X. (2024). A Survey on Evaluation of Large Language Models. *ACM Transactions on Intelligent Systems and Technology*, *15*(3), 1-45. https://doi.org/10.1145/3641289
7. Terziyan, V., Vitko, O., & Terziyan, O. (2025). A Conceptual Design of Industrial Asset Maintenance System by Autonomous Agents Enhanced with ChatGPT. *Intelligent and Sustainable Manufacturing, 2*(1), 10008. https://doi.org/10.70322/ism.2025.10008
8. Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT press.
9. Li, C., Zheng, P., Yin, Y., Wang, B., & Wang, L. (2023). Deep Reinforcement Learning in Smart Manufacturing: A Review and Prospects. *CIRP Journal of Manufacturing Science and Technology*, *40*, 75-101. https://doi.org/10.1016/j.cirpj.2022.11.003

10. Jain, S. M. (2022). Hugging Face. In: *Introduction to Transformers for NLP: With the Hugging Face Library and Models to Solve Problems* (pp. 51-67). Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-8844-3_4
11. Devika, R., Vairavasundaram, S., Mahenthar, C. S. J., Varadarajan, V., & Kotecha, K. (2021). A Deep Learning Model Based on BERT and Sentence Transformer for Semantic Keyphrase Extraction on Big Social Data. *IEEE Access*, *9*, 165252-165261. https://doi.org/10.1109/ACCESS.2021.3133651
12. Watkins, C. J., & Dayan, P. (1992). Q-Learning. *Machine Learning*, *8*, 279-292. https://doi.org/10.1007/BF00992698
13. Al-Tamimi, A., Lewis, F. L., & Abu-Khalaf, M. (2007). Model-Free Q-Learning Designs for Linear Discrete-Time Zero-Sum Games with Application to H-Infinity Control. *Automatica, 43*(3), 473-481. https://doi.org/10.1016/j.automatica.2006.09.019
14. Schmid, M., Moravčík, M., Burch, N., Kadlec, R., Davidson, J., Waugh, K., ... & Bowling, M. (2023). Student of Games: A Unified Learning Algorithm for Both Perfect and Imperfect Information Games. *Science Advances, 9*(46), eadg3256. https://doi.org/10.1126/sciadv.adg3256
15. Kober, J., Bagnell, J. A., & Peters, J. (2013). Reinforcement Learning in Robotics: A Survey. *The International Journal of Robotics Research*, *32*(11), 1238-1274. https://doi.org/10.1177/0278364913495721
16. Aradi, S. (2020). Survey of Deep Reinforcement Learning for Motion Planning of Autonomous Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, *23*(2), 740-759. https://doi.org/10.1109/TITS.2020.3024655
17. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., ... & Hassabis, D. (2018). A General Reinforcement Learning Algorithm that Masters Chess, Shogi, and Go through Self-Play. *Science*, *362*(6419), 1140-1144. https://doi.org/10.1126/science.aar6404
18. Skinner, B. F. (1971). Operant Conditioning. *The Encyclopedia of Education*, *7*, 29-33. New York: Macmillan and Free Press.
19. Thorndike, E. L. (1927). The Law of Effect. *The American Journal of Psychology*, *39*(1/4), 212-222. https://doi.org/10.2307/1415413
20. Bellman, R. (1957). *Dynamic Programming*. Princeton, NJ: Princeton University Press.
21. Font, J. M., & Mahlmann, T. (2018). Dota 2 Bot Competition. *IEEE Transactions on Games*, *11*(3), 285-289. https://doi.org/10.1109/TG.2018.2834566
22. Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., ... & Zhang, S. (2019). Dota 2 with Large Scale Deep Reinforcement Learning. *arXiv preprint arXiv:1912.06680*. https://doi.org/10.48550/arXiv.1912.06680
23. Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., Horgan, D., Quan, J., Sendonaris, A., Osband, I., Dulac-Arnold, G., Agapiou, J., Leibo, J., & Gruslys, A. (2018). Deep Q-learning From Demonstrations. *Proceedings of the AAAI Conference on Artificial Intelligence*, *32*(1), 3223–3230. https://doi.org/10.1609/aaai.v32i1.11757
24. Fan, J., Wang, Z., Xie, Y., & Yang, Z. (2020). A Theoretical Analysis of Deep Q-learning. *Proceedings of Machine Learning Research, 120*, 486-489. https://proceedings.mlr.press/v120/yang20a.html
25. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347*. https://doi.org/10.48550/arXiv.1707.06347
26. Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. *Proceedings of Machine Learning Research, 80*, 1861-1870. https://proceedings.mlr.press/v80/haarnoja18b
27. Du, R., Wu, J., & Gao, Y. (2024). Dual-Q Network Deep Reinforcement Learning-Based Computation Offloading Method for Industrial Internet of Things. *The Journal of Supercomputing*, *80*(17), 25590-25615. https://doi.org/10.1007/s11227-024-06425-x
28. Barto, A. G., & Mahadevan, S. (2003). Recent Advances in Hierarchical Reinforcement Learning. *Discrete Event Dynamic Systems*, *13*, 341-379. https://doi.org/10.1023/A:1025696116075
29. Tan, M. (1993). Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents. In: *Proceedings of the Tenth International Conference on Machine Learning* (pp. 330-337). San Francisco, CA: Morgan Kaufmann Publishers Inc. https://web.media.mit.edu/~cynthiab/Readings/tan-MAS-reinfLearn.pdf
30. Zhang, K., Yang, Z., & Başar, T. (2021). Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms. *Studies in Systems, Decision and Control, 325*, 321-384. Springer, Cham. https://doi.org/10.1007/978-3-030-60990-0_12
31. Moerland, T. M., Broekens, J., Plaat, A., & Jonker, C. M. (2023). Model-Based Reinforcement Learning: A Survey. *Foundations and Trends® in Machine Learning*, *16*(1), 1-118. http://dx.doi.org/10.1561/2200000086
32. Von Rueden, L., Mayer, S., Beckh, K., Georgiev, B., Giesselbach, S., Heese, R., ... & Schuecker, J. (2021). Informed Machine Learning–A Taxonomy and Survey of Integrating Prior Knowledge into Learning Systems. *IEEE Transactions on Knowledge and Data Engineering*, *35*(1), 614-633. https://doi.org/10.1109/TKDE.2021.3079836
33. Cuomo, S., Di Cola, V. S., Giampaolo, F., Rozza, G., Raissi, M., & Piccialli, F. (2022). Scientific Machine Learning through Physics–Informed Neural Networks: Where We are and what's Next. *Journal of Scientific Computing*, *92*(3), 88. https://doi.org/10.1007/s10915-022-01939-z
34. Terziyan, V., & Vitko, O. (2025). Context-Aware Machine Learning for Smart Manufacturing. *Procedia Computer Science*, *253*, 25-36. Elsevier. https://doi.org/10.1016/j.procs.2025.01.066