A Context-Aware Retrieval Framework for Dynamic Network Management

Franc Pouhela[1], Hans D. Schotten[1 2]
[1]*German Research Center for Artificial Intelligence (DFKI GmbH)*
[2]*University of Kaiserslautern-Landau (RPTU), Germany*
Email: {franc.pouhela | schotten}@dfki.de

***Abstract*—Artificial Intelligence (AI) is increasingly recognized as a transformative technology across industries, and the mobile communication community is actively investigating its potential impact for the future of mobile communication. A central challenge in mobile network management is the effective utilization of contextual data to enable adaptive behavior, context-aware service provisioning, dynamic resource allocation, and resilience. This paper introduces a context-aware retrieval framework that leverages the Natural Language Processing (NLP) capabilities of Large Language Models (LLMs) to deliver precise, context-specific network information for enhanced decision-making. The proposed framework is evaluated in a virtual network simulation, demonstrating its effectiveness and strong performance in adaptive network management scenarios.**

***Index Terms*—6G, AI, RAG, LLM, Mobile Communication**

I. Introduction

The evolution of mobile communication systems from 4G towards 5G and the envisioned 6G era has fundamentally transformed the way networks are designed, deployed, and managed. Increasing demand for high data rates, Ultra-Reliable Low-Latency Communication (URLLC), and massive connectivity drives networks to become more heterogeneous and dynamic. These requirements are amplified in vertical domains such as Industry 4.0, connected healthcare, and autonomous mobility, where reliable and adaptive services are mission-critical. Consequently, network management has become a central challenge, as operators must cope with complex topologies, volatile user behavior, and fluctuating environmental conditions.

Traditional network management relies heavily on static rule-based systems and centralized control mechanisms. While effective in relatively stable environments, these approaches fail to capture the dynamic and context-dependent nature of next-generation networks. For example, mobility management in heterogeneous networks often depends on pre-configured thresholds for handovers, leading to suboptimal performance when conditions deviate from expected patterns [1]. Similarly, resource allocation schemes that do not account for temporal or spatial context cannot guarantee efficient utilization of scarce resources, particularly under mission-critical traffic demands. These limitations highlight the urgent need for management frameworks that can operate in a context-aware and adaptive manner using Artificial Intelligence (AI).

AI has emerged as a key enabler in addressing these challenges. By leveraging data-driven models, networks can predict traffic dynamics, learn mobility patterns, and optimize resource usage. Context-awareness plays a crucial role in this vision, as it provides the semantic layer that bridges raw network measurements with actionable knowledge. For instance, integrating context information such as user location, device type, and environmental factors into decision-making has been shown to improve handover performance, reduce service disruptions, and enhance resilience [2], [3]. However, existing AI-based approaches remain constrained by two major issues: (i) the reliance on domain-specific models and hand-crafted features, which limits generalizability across scenarios, and (ii) the lack of mechanisms to efficiently retrieve and apply knowledge from diverse and unstructured data sources.

Recent breakthroughs in Natural Language Processing (NLP), particularly with Large Language Models (LLMs), open a new direction for tackling these challenges. Trained on massive and diverse corpora, LLMs exhibit remarkable capabilities in contextual reasoning, knowledge retrieval, and semantic understanding. While initial applications of LLMs in networking have focused on code generation, zero-touch configuration, and orchestration [4], [5], [6], their potential as context-aware retrievers for network knowledge has remained underexplored. Unlike conventional machine learning models that operate on fixed features, LLMs can flexibly process heterogeneous data, integrate multiple contextual signals, and retrieve relevant knowledge on demand.

This paper introduces a *context-aware retrieval framework* that leverages the NLP capabilities of LLMs to enhance adaptive network management. The core idea is to treat network management problems not only as optimization tasks but also as retrieval tasks, where the key challenge is identifying the most relevant contextual information to guide decisions. By embedding contextual data into natural language queries, the framework enables LLMs to return precise, context-specific knowledge that can directly inform resource allocation, mobility management, and service provisioning strategies.

The main contributions of this work are summarized as follows: We identify the limitations of existing context-aware and AI-driven approaches for mobile network management, highlighting the gap in retrieval-based solutions. We propose a novel framework that employs LLMs as context-aware

knowledge retrievers, enabling precise and adaptive decision support in dynamic environments. We design and evaluate the framework in a smart factory simulation, showing its effectiveness in improving network context data retrievals.

The remainder of this paper is organized as follows. Section II reviews related work on context-aware network management and LLM applications in networking. Section III presents the proposed framework in detail. Section IV describes the simulation setup and evaluation methodology and discusses results and insights, while Section V concludes the paper and outlines future research directions.

II. Related Work

Context-aware network management has a rich foundation. Early frameworks for mobile context-management employed generic context handling layers for mobile applications, decoupling reasoning from application logic to reduce disruption [7], [8]. Context-aware service composition was explored in mobile environments, enabling dynamic adaptation based on situational parameters [9], [10]. Platform-level context-awareness for mobile data management further advanced adaptability by managing context transparency across network layers [11].

Mobility management in heterogeneous and next-generation networks benefited from context and Machine Learning (ML): reinforcement learning improved handover throughput and fairness in HetNets [1]; AI-driven frameworks in 6G context-aware mobility reduced handover failures and enhanced Quality of Experience (QoE) [12].

Recent methods integrate context-aware data into network prediction and control. Satellite imagery augments performance estimation in networks to generalize across regions and address cold-start issues [3]. In edge-enabled systems, learning-based context-aware scheduling and resource allocation support adaptive industrial Internet of Things (IoT) [2].

Generative AI applications in network management are emerging. One approach uses LLMs to generate code for network tasks while preserving explainability and privacy [4]. LLMs have been applied to automatic configuration generation and zero-touch intent-based management [5]. In wireless orchestration, LLMs orchestrate multi-model workflows in dense environments [6]. For active queue management, LLMs are distilled to manage latency and congestion adaptively [13].

III. System Architecture

We present a modular context-aware retrieval framework that converts raw network telemetry and semantic context into precise, executable knowledge for adaptive network control. The framework has four functional blocks: (i) network data collector, (ii) context retriever (dispatch + executor), (iii) reasoning engine, and (iv) control signal sink.

Fig. 1 illustrates the high-level architecture and the retriever internals respectively. The retriever uses an LLM to synthesize structured queries *cyphers* and a fast executor to fetch grounded facts from a heterogeneous Knowledge Graph (KG). The reasoner uses retrieved facts plus the live context to produce actions or recommendations. [2], [4] provide precedent for integrating learned models into resource allocation and automation; our novelty is casting the problem as retrieval + reasoning with explicit grounding and execution.

Formally let the system state at decision epoch t be $s_t = (m_t, c_t)$ where $m_t \in \mathcal{M}$ is measured telemetry (KPIs, traces) and $c_t \in \mathcal{C}$ is semantic context (device type, location, mission). Let the knowledge corpus be $\mathcal{D} = \{d_j\}$ containing heterogeneous items (time series segments, policies, topology fragments, textual logs). The retrieval task is a mapping

$$R : \mathcal{C} \times \mathcal{D} \to \mathcal{K}, \qquad \mathcal{K} = \{k_1, \ldots, k_K\},$$

where R returns the top-K grounded facts most relevant to (m_t, c_t). We implement R as a two-stage pipeline: a lightweight index for candidate selection followed by an LLM-based dispatcher that generates a structured cypher query and a guarded executor that performs database queries and returns grounded results. The LLM interface is used in two distinct roles. The reasoner LLM consumes a system prompt embedding the ontology, the current context c_t, and retrieved facts $\mathcal{K}$ to produce either a high-level action a_t or a policy update. The cypher generator LLM receives the same system prompt plus a targeted retrieval intent from the dispatcher module and emits a structured query q_{cypher}. The dispatcher/executor handshake follows the loop in Fig. 1: dispatcher $\rightarrow LLM_{cypher} \rightarrow$ executor $\rightarrow$ grounded answer $\rightarrow$ dispatcher $\rightarrow$ return to reasoner. This separation isolates generative capacity (cypher synthesis) from execution and grounding, reducing hallucination risk and enabling auditability.

Implementation details follow the block diagram. The Network Data Collector ingests telemetry, aggregates it into short windows, and produces structured events. The context retriever houses two sub-modules shown in Fig. 1: *Dispatcher* constructs the LLM prompt from (system prompt + knowledge ontology + query intent) and forwards it to LLM_{cypher}. The *Executor* executes the generated cypher against the knowledge store and returns structured answers. The reasoning engine consumes the answers and executes the policy π_θ either deterministically (rule logic) or via an ML policy (e.g., a small transformer or Reinforcement Learning (RL) agent) guided by the LLM output. Control signals are dispatched to RAN/Core subsystems or fed to a human operator for verification.

IV. Implementation and Evaluation

To evaluate our approach, we decided to build a network simulator tailored for factory-scale 5G deployments rather than relying on abstracted analytical models or heavy network simulators such as ns-3. Our implementation is realized in JavaScript with the `Three.js` library, allowing for real-time visualization and intuitive inspection of network dynamics. The simulator models both the physical environment and communication aspects: a bounded factory floor with walls, obstacles, and workstations; multiple gNodeBs operating on

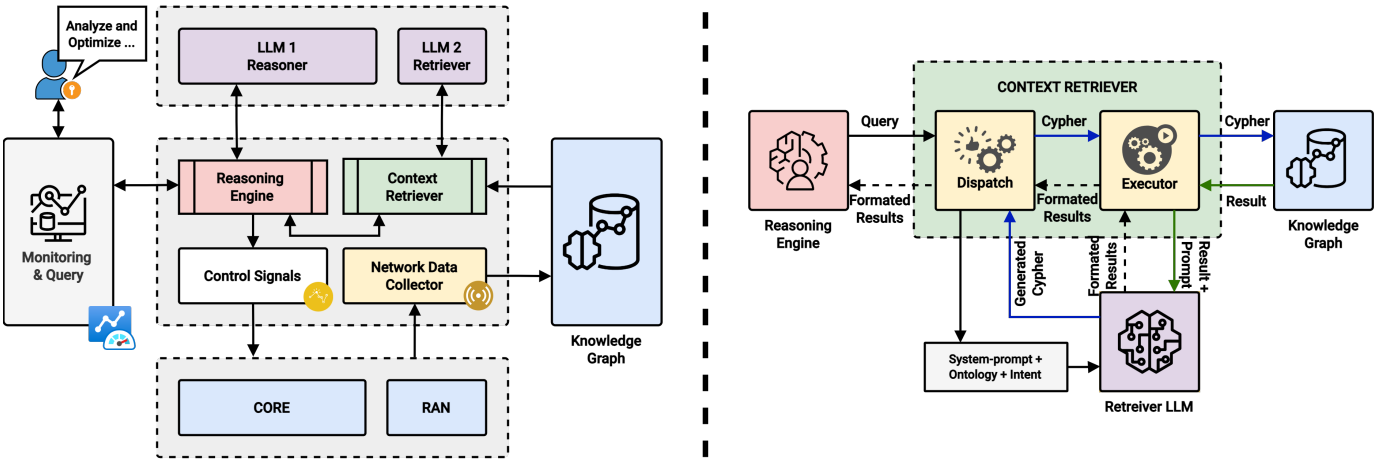

Fig. 1: Context Management Architecture

heterogeneous frequency bands and vendors; and Automated Guided Vehicles (AGVs) executing mobility and task-oriented behavior (Fig. 2). The source code is present on GitHub and can be found under [14].

Unlike purely stochastic models, this approach provides a dual perspective. First, it captures radio-layer performance through path-loss, shadowing, Signal-to-Interference-plus-Noise Ratios (SINR), and throughput estimation. Second, it embeds operational realism by incorporating AGV task cycles, battery constraints, collision avoidance, and dynamic rerouting. Connectivity and handover logic follow Reference Signal Received Power (RSRP)-based decision rules, while communication quality is mapped into color-coded SINR categories for visual traceability.

This hybrid environment enables the evaluation of network coverage, mobility, and interference effects in scenarios close to actual smart factories, while still being lightweight and configurable. Moreover, by exporting structured state to external databases, the simulator supports integration with graph-based analytics for further performance evaluation.

a) Factory Model: The simulated factory is a bounded square area of size $L \times L$ with walls of height H. The environment includes randomly placed obstacles, static workstations, and charging stations. Collidable objects are maintained for Line-of-Sight (LoS) and shadowing calculations.

b) Base Stations: Up to four gNodeBs are positioned near the factory quadrants. Each base station is modeled as a mast of height h_{bs} with panel antennas oriented at 120° separation. Each gNodeB is assigned a vendor from $\{\text{Ericsson}, \text{Nokia}, \text{Huawei}, \text{Samsung}\}$ and a frequency band from $\{n78 : 3.5\,\text{GHz}, n257 : 28\,\text{GHz}, n258 : 26\,\text{GHz}\}$. Transmission power is set at $P_{\text{tx}} = 23$ dBm.

c) AGVs: Or mobile User Equipments (UEs), initialized with random starting positions. Each AGV has a finite battery and task assignment (e.g., parts transport, material supply). When battery levels drop below 20%, AGVs autonomously reroute to the charging station. Otherwise, tasks correspond to navigating between workstations. Movement is governed by velocity vectors with collision detection via raycasting. If blocked for > 2s, an AGV is re-routed to a new target.

d) Channel Model: The received signal power RSRP is computed as

$$P_{\text{rx}}(d) = P_{\text{tx}} - \big(L_0 + 10\,n\,\log_{10}\tfrac{d}{d_0} + S\big),$$

where $L_0 = 32.45$ dB is the free-space reference loss at $d_0 = 1$ m, $n = 2.5$ is the path-loss exponent, and $S = 10$ dB is an additional shadowing penalty when LoS is obstructed. Noise power is given by

$$N = N_0 B, \quad N_0 = -174\ \text{dBm/Hz}, \quad B = 20\ \text{MHz}.$$

The SINR for AGV i connected to Base Station (BS) j is

$$\text{SINR}_{i,j} = \frac{P_{\text{rx},i,j}}{\sum_{k \neq j} P_{\text{rx},i,k} + N}.$$

Throughput is estimated via Shannon capacity with efficiency factors:

$$R_i = \min\Big(20 \cdot \log_2(1 + \text{SINR}_{i,j}),\ 400\Big) \cdot \xi,$$

where $\xi \in [0.8, 1.0]$ is a random multiplicative efficiency.

e) Mobility and Handover: AGVs perform continuous connectivity evaluation. A handover is triggered if a neighboring BS provides

$$\Delta\text{RSRP} > M, \quad M = 3\ \text{dB},$$

relative to the currently serving BS. Connections are visually represented by dynamic colored lines, where color encodes SINR quality (*excellent*: > 15dB, *good*: 5–15dB, *fair*: -5–5dB, *poor*: < -5dB).

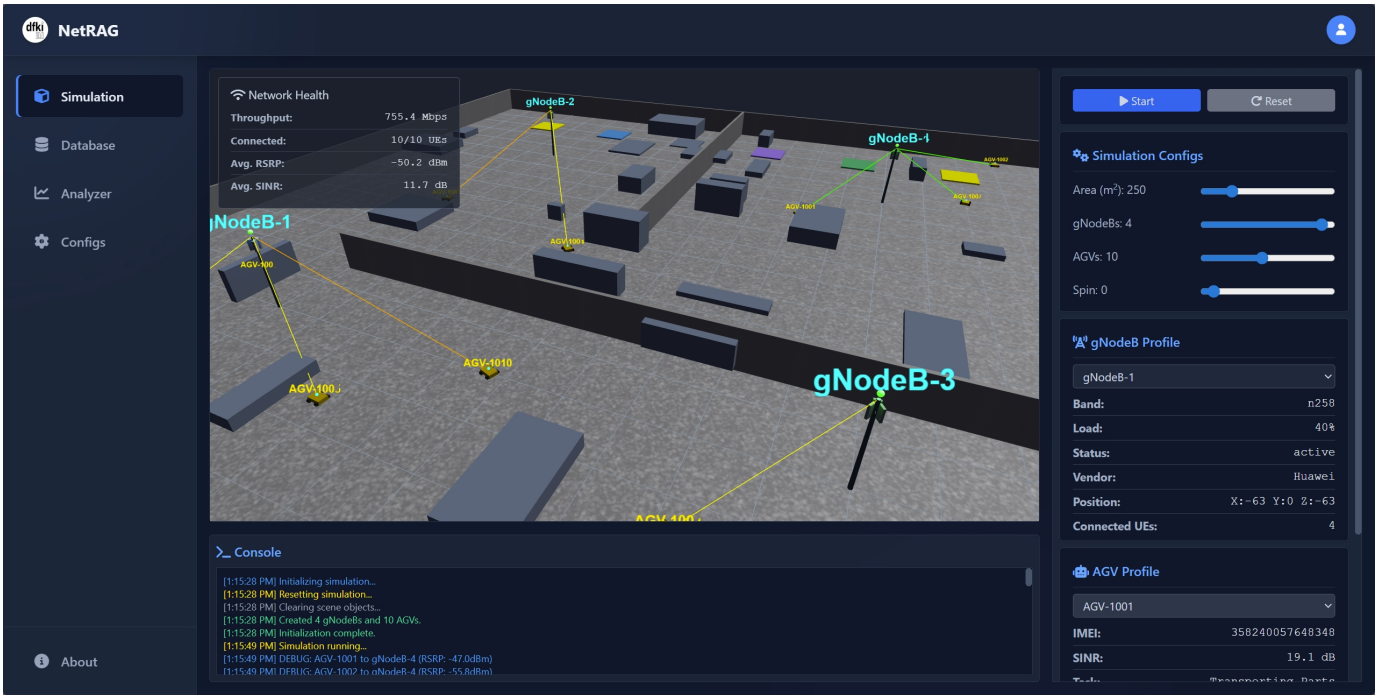

Fig. 2: Smart factory network simulator [14]

f) Knowledge Integration: Every second ($1s$), a structured representation of the changes occurred on the gNodeBs and AGVs states is exported to a knowledge graph via REST API calls. The framework employs Neo4j [15], a graph database system tailored for storing and managing data in a graph structure. Neo4j is optimized for handling intricate, highly interconnected datasets, facilitating efficient querying and analysis of relationships between nodes. It utilizes Cypher Query Language (CQL), a declarative query language, and is commonly applied in domains such as social networks, recommendation systems, and more.

```
You are an expert Cypher query generator for a Neo4j
graph database. Your specific domain of expertise is a 5G
telecommunications network. Your primary function is to
convert natural language questions into precise, efficient,
and syntactically correct Cypher queries that can be
executed directly against the database. [...].

### GRAPH NODES:
- gNodeB: Represents a base station.
  - Attributes: 'id', 'load', 'band', 'status', [...]
- AGV: Represents a User Equipment.
  - Attributes: 'id', 'imei', 'speed', 'rsrp', [...]
- ...

### RELATIONSHIPS:
- '(gNodeB)-[:OPERATES_ON]->(Band)'
- '(AGV)-[:CONNECTED_TO]->(gNodeB)'
- '...'

Based on the instructions and knowledge graph description
above, generate the cypher query for this instruction:
```

Listing 1: System prompt for cypher generation

Fig. 3 illustrates the KG from our study, showcasing different entities and their relationships. Each node and relationship possesses underlying attributes not shown in the visualization. These attributes can be seen in the system prompt snippet depicted in Listing 1.

g) LLM Integration: This study utilizes Google Gemini LLMs (`2.0 Flash, 2.5 Pro`) for cypher query generation and reasoning tasks. With their extensive context windows of `1 million tokens`, Gemini models are particularly capable of processing complex and large inputs. An initial effort to deploy a local LLM service within our network for privacy purposes was constrained by computational resources. This limitation ultimately guided us toward a simulation.

A. Results Evaluation

In order to evaluate the system, we conducted a series of reasoning tasks by submitting queries and allowing the system to process them in order to generate answers. Table 1 lists some of the queries submitted to the Retrieval-Augmented Generation (RAG) system. Both LLM models accurately generated cypher queries that were executed against the knowledge base and subsequently used for reasoning. The results in Fig. 4 indicate that Gemini 2.0 Flash is sufficient for this task while additionally providing lower response latency. The evaluation also incorporated a repeatability test to assess the stability of model outputs. With the temperature fixed at `0.1`, we anticipated highly consistent behavior, which was confirmed by the results in Fig. 4, where both models achieved high repeatability scores.

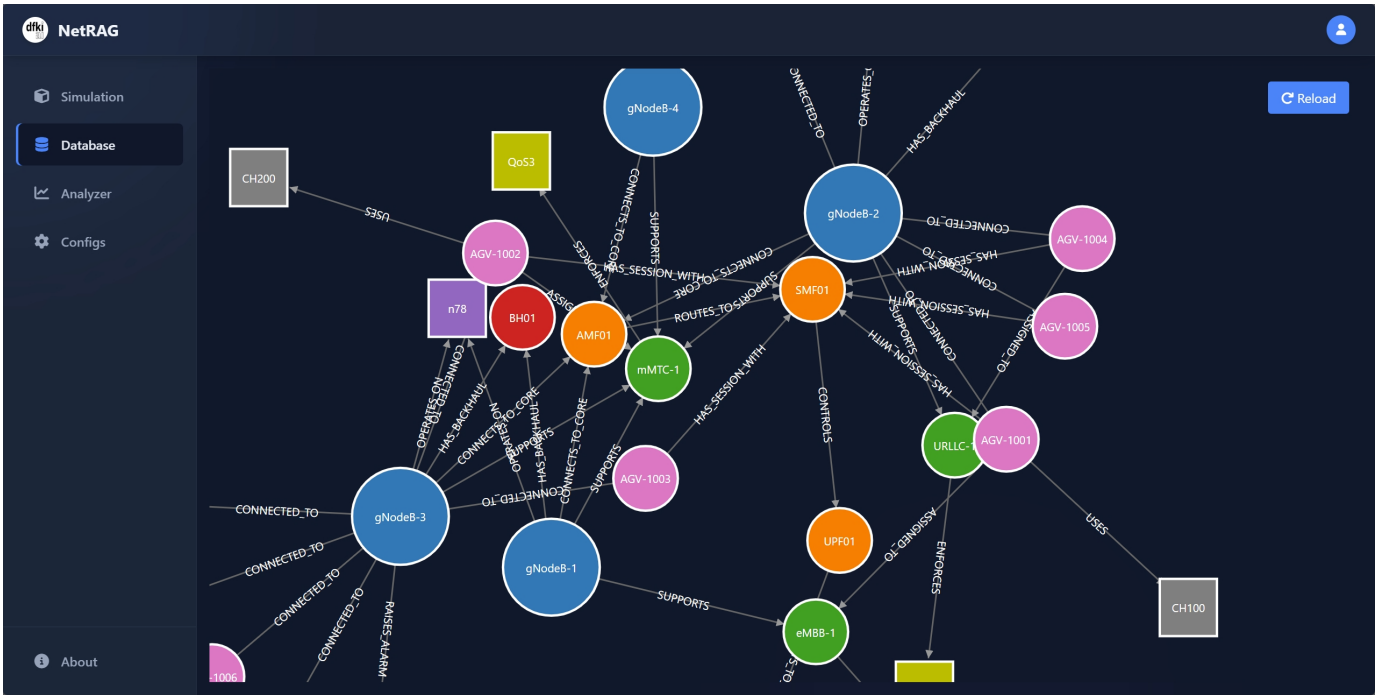

Fig. 3: Knowledge graph view

Query 1:	**Analyze the data of AGV-1001**	
Gemini 2.0 Flash	`MATCH (agv:AGV id: 'AGV-1001')-[r]->(n) RETURN agv, type(r), n`	✔
Gemini 2.5 Pro	`MATCH (agv:AGV id: 'AGV-1001') RETURN agv`	✔
Query 2:	**How many AGVs are connected to the gNodeB-1?**	
Gemini 2.0 Flash	`MATCH (agv:AGV)-[:CONNECTED_TO]->(gnb:gNodeB id: 'gNodeB-1') RETURN count(agv) AS nbrAgv`	✔
Gemini 2.5 Pro	`MATCH (agv:AGV)-[:CONNECTED_TO]->(gnb:gNodeB id: 'gNodeB-1') RETURN count(agv) AS nbrAgv`	✔
Query 3:	**Which channel is currently used by AGV-1002?**	
Gemini 2.0 Flash	`MATCH (agv:AGV id: 'AGV-1002')-[:USES]->(channel:Channel) RETURN channel.id, channel.label, channel.bandwidth_MHz`	✔
Gemini 2.5 Pro	`MATCH (agv:AGV id: 'AGV-1002')-[:USES]->(ch:Channel) RETURN ch.id, ch.label, ch.bandwidth_MHz`	✔
Query 4:	**Which network slice is currently used by the AGV-1005?**	
Gemini 2.0 Flash	`MATCH (agv:AGV id: 'AGV-1005')-[:ASSIGNED_TO]->(slice:Slice) RETURN slice.id, slice.label`	✔
Gemini 2.5 Pro	`MATCH (agv:AGV id: 'AGV-1005')-[:ASSIGNED_TO]->(slice:Slice) RETURN slice.id, slice.label`	✔
Query 5:	**How many clients are connected to the broker hosted at IP address 192.168.2.177?**	
Gemini 2.0 Flash	`MATCH (s:Service) RETURN s.id, s.label, s.description`	✔
Gemini 2.5 Pro	`MATCH (s:Service) RETURN s.id, s.label, s.description`	✔

TABLE 1: Generated cypher query results

1) Potential Improvements: Key engineering trade-offs include latency, grounding, and scalability. Naive LLM-only retrieval provides strong semantic robustness but incurs inference latency t_{LLM} and a risk of hallucinations. Latency can be mitigated via hybrid indexing: Approximate Nearest Neighbor (ANN) selection for candidate retrieval followed by a single LLM call for cypher generation. For scalability, sharded indices and caching of recent $(c_t, \mathcal{K})$ pairs are essential. Query complexity is $O(\log |\mathcal{D}|)$ for ANN retrieval plus $O(k)$ for execution, where k is the number of returned items. The dominant runtime remains $t_{\text{LLM}} + t_{\text{exec}}$. Security and reliability controls are essential. Prompts must comply with a defined privacy budget, with differential privacy [10] applied to embedding construction and access controls enforced on the knowledge graph. For high-impact actions, a confidence threshold combined with human-in-the-loop approval can be employed. Logging, immutable audit trails, and replayable cypher outputs provide support for post-hoc analysis and compliance.

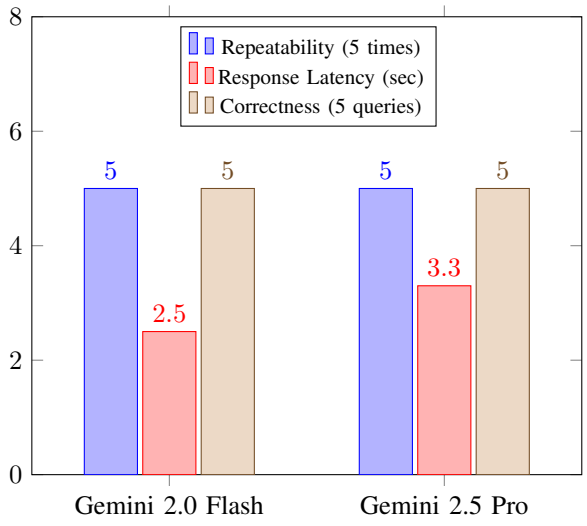

Fig. 4: Retrieval results evaluation

2) Notable Limitations: LLM hallucination remains a residual risk when the ontology or stored facts are stale. Freshness requires continuous ingestion and timestamped facts; stale data can produce incorrect actions even if retrieval was correct. The approach also depends on the quality of the ontology and prompt engineering. Finally, real-time constraints in strict URLLC settings may preclude on-path LLM calls; in such cases local distilled models or fallback rule sets must be used.

V. Conclusion and Future Work

We presented a modular retrieval+reasoning framework that converts telemetry m_t and semantic context c_t into grounded facts $\mathcal{K} = R(c_t, \mathcal{D})$ and actionable outputs for adaptive network control. The two-stage retriever isolates generative synthesis from execution, improving grounding and auditability while keeping retrieval complexity $O(\log|\mathcal{D}|) + O(k)$ and run-time dominated by $t_{\text{LLM}} + t_{\text{exec}}$. A lightweight, real-time 5G factory simulator validated feasibility and integration with a Neo4j knowledge graph; commercial LLMs achieved high correctness and repeatability but incurred measurable latency. Principal limitations are stale ontology/facts, and hard real-time constraints for URLLC. Practical mitigations include cached and sharded indices, confidence thresholds with human-in-the-loop for high-impact actions, differential-privacy embeddings, and local distilled models as low-latency fallbacks. Future work must quantify control-loop stability under t_{LLM} delays, formalize safety/confidence bounds, and demonstrate end-to-end performance on hardware testbeds. The approach bridges symbolic grounding and generative reasoning, enabling auditable adaptive control while exposing clear engineering trade-offs in latency, freshness, and safety.

Acknowledgment

The authors acknowledge the financial support by the German *Federal Ministry for Education and Research (BMFTR)* within the projects Open6GHub {16KISK003K}, Open6GHub+ {16KIS2402K} & KIOps6G {16KIS2398}.

References

[1] Simsek, M., Bennis, M., and Güvenc, I., "Context-aware mobility management in hetnets: A reinforcement learning approach," *arXiv preprint arXiv:1505.01625*, 2015.

[2] Liao, H., Zhou, Z., Zhao, X., Zhang, L., Mumtaz, S., Jolfaei, A., and Bashir, A., "Learning-based context-aware resource allocation for edge-computing-empowered industrial iot," *IEEE Internet of Things Journal*, vol. 7, pp. 4260–4277, 2019.

[3] Shibli, A. and Zanouda, T., "Context-aware mobile network performance prediction using network & remote sensing data," *arXiv preprint arXiv:2405.00220*, 2024.

[4] Mani, S., Zhou, Y., Hsieh, K., Segarra, S., Chandra, R., and Kandula, S., "Enhancing network management using code generated by large language models," *arXiv preprint arXiv:2308.06261*, 2023.

[5] Lira, O., Caicedo, O., and da Fonseca, N., "Large language models for zero touch network configuration management," *arXiv preprint arXiv:2408.13298*, 2024.

[6] Abdallah, A., Albaseer, A., Celik, A., Abdallah, M., and Eltawil, A., "Netorchllm: Mastering wireless network orchestration with large language models," *arXiv preprint arXiv:2412.10107*, 2024.

[7] Chen, P., Sen, S., Keng Pung, H., Xue, W., and Choong Wong, W., "A context management framework for context-aware applications in mobile spaces," *International Journal of Pervasive Computing and Communications*, vol. 8, no. 2, pp. 185–210, 2012.

[8] Pouhela, F., Krummacker, D., and Schotten, H. D., "Towards 6G Networks," in *A Context Management Architecture for Decoupled Acquisition and Distribution of Information in Next-Generation Mobile Networks*, ser. ITG, vol. 157, VDE. IEEE, 5 2023. [Online]. Available: https://www.researchgate.net/publication/373328855_A_Context_Management_Architecture_for_Decoupled_Acquisition_and_Distribution_of_Information_in_Next-Generation_Mobile_Networks

[9] Lee, C., Ko, S., Lee, S., Lee, W., and Helal, S., "Context-aware service composition for mobile network environments," in *Ubiquitous Intelligence and Computing (UIC)*, vol. 4611. Springer, 2007, pp. –.

[10] Pouhela, F., Sanon, S. P., and Schotten, H. D., "Ngna 2023," in *A Differential Privacy Approach for Context-Aware Service Provisioning in Mobile Networks*, 12 2023. [Online]. Available: https://www.researchgate.net/publication/377307205_A_Differential_Privacy_Approach_for_Context-Aware_Service_Provisioning_in_Mobile_Networks

[11] Norrie, M., Signer, B., Grossniklaus, M. *et al.*, "Context-aware platform for mobile data management," *Wireless Networks*, vol. 13, pp. 855–870, 2007.

[12] Chittipedhi, K., Abbas, H., Meharunnisa, S., Rashmi, S., Annapurna, D., and Udayakumar, R., "Context-aware mobility management in 6g-ready mobile internet networks," *Journal of Internet Services and Information Security*, vol. 15, no. 2, pp. 641–651, 2025.

[13] Satish, D., Pokhrel, S. R., Kua, J., and Walid, A., "Distilling large language models for network active queue management," *arXiv preprint arXiv:2501.16734*, 2025.

[14] Pouhela, F. Netrag 5g/6g. [Online]. Available: https://github.com/Madsycode/sim-network-llm

[15] Neo4j. Genai apps, grounded in your data. [Online]. Available: https://neo4j.com/