LLM-Assisted Design and Analytics of Next-Generation Optical Transport Networks

LLM-Assisted Design and Analytics of Next-Generation Optical Transport Networks

Imran Chowdhury Dipto*, Sanwal Zeb*, Muhammad Umar Masood*, Ihtesham Khan†,
Nelson Costa‡, João Pedro‡, Antonio Napoli§, Vittorio Curri*
*Politecnico di Torino, Torino, Italy
†Nokia Bell Labs, Murray Hill, USA
‡Nokia NI OND, Portugal
§Nokia NI OND, Germany
Email: imran.dipto@polito.it

***Abstract*—Optical networks are driving an increasing demand for network automation and intent-driven design, which requires translating high-level operator requests into network actions. In this paper, we present an LLM-based system for design and analytics of optical networks, where LLMs play the key role as reasoning agents, managing network analytics. The proposed framework integrates network queries with domain specific Python modules for topology generation, graph analytics, and equipment-aware referencing. Four open-source LLMs have been evaluated, namely Qwen-2.5-14B-Instruct, LLaMA-3.1-8B, Mistral-NeMo-12B, and Gemma-2-9B, using a benchmark of representative network engineering queries ranging in varying network tasks. Model outputs are assessed using a graded scoring criterion of full correctness, partial correctness, or hallucinations. The results show that the Qwen-2.5-14B-Instruct model achieved the highest overall accuracy of 96.97% with the lowest hallucination, while smaller models such as LLaMA and Gemma delivered competitive performance. These findings indicate that LLMs have strong potential to reliably support optical network design and analytics in real-world applications.**

***Index Terms*—Large Language Model, Qwen, LLaMA, Gemma, Optical networks, Network Design.**

I. Introduction

The rise of data-driven services, coupled with cloud-based and AI-enabled applications, is placing increasing pressure on optical transport networks. Optical infrastructures must support large traffic scaling and seamless integration of fronthaul, midhaul, and backhaul connectivity while enabling distributed edge computing [1], [2]. These needs are giving rise to the complexity of network designing and operations, requiring operators to consider topology design, routing, fiber constraints, and equipment management capabilities.

To address such challenges, optical networks have evolved toward programmable and software-defined architectures. Software-defined optical networks (SDNs) provide an abstraction and control mechanism to improve flexibility and automation [3]. However, as these systems mature, the complexity shifts toward interpreting high-level operator intents and translating them into executable network actions, particularly in the context of intent-based network and optical network-as-a-service (ONaaS) [4].

Traditional automation systems perform well when operator intents can be mapped to deterministic workflows; however, they struggle with ambiguous requests and multi-constraint reasoning spanning topology analytics and equipment-specific rules. Recently, LLMs have emerged as powerful general-purpose reasoning engines for natural language understanding and multi-step task execution [5].

In the context of intelligent network operations, LLMs are increasingly recognized for their potential role in intent interpretation, planning, and closed-loop operational support [6]. Despite this promising potential, the application of LLMs for optical transport network design and analytics, where they must support topology generation, graph analytics, and equipment management, remains largely unexplored.

Recent advances in large language models (LLMs) provide a promising approach to address this limitation. LLMs enable natural-language understanding and multi-step reasoning, allowing operators to interact with network systems using high-level queries [5], [6]. In optical networks, initial studies have explored LLM-based agents for specific tasks such as QoT estimation and performance optimization [7]. More broadly, tool-augmented LLMs have demonstrated the ability to combine reasoning with deterministic execution, improving reliability in complex workflows [8]–[10]. However, existing works primarily focus on isolated tasks and do not address intent-driven orchestration across multiple network analytics functions.

Motivated by these limitations, this paper presents an LLM-assisted framework for intent-driven optical network design and analytics. In the proposed system, LLMs act as reasoning agents that interpret network queries and orchestrate domain-specific Python modules for topology generation, graph analytics, and equipment-aware operations.

The remainder of this paper is organized as follows. Section II reviews the selected LLM architectures and justifies their choices for this research. Section III describes the proposed framework. Section IV presents and discusses the evaluation results. Finally, Section V concludes the paper with future directions of this work.

II. Large Language Models

LLMs are a powerful reasoning system capable of interpreting natural language and generating structured, task-

oriented outputs. Built on the transformer architecture and enhanced through instruction tuning and pre-trained on large datasets, they can understand complex queries, infer implicit constraints, and support flexible intent-driven workflow [11]. These features make LLMs ideal for optical network applications, where network operator intents can be translated into multi-step analytical operations. In this work, LLMs have been used as reasoning agents to facilitate network design and analytics.

A. Open-Source LLM Architectures

In this study, we have evaluated four open-source LLMs, all of which are based on the transformer architecture introduced in [12]. The self-attention mechanism enables these models to capture long-range dependencies and structured relationships within user inputs, which is essential for reasoning over complex network queries. Recent LLM architectures incorporate several refinements, including grouped-query attention, optimized feed-forward layers, and extended context windows, to improve computational efficiency and scalability [13]. These features enable them to understand subwords in sentences [14], [15] and are crucial for network automation for complex queries. For this study, we chose four open-source, instruction-tuned LLMs that can be deployed locally and run through the `Ollama` platform to support reproducibility.

1) LLaMA 3.1 (8B): LLaMA 3.1 (8B) is an open-weight LLM developed by Meta as part of the LLaMA 3 family, designed to balance reasoning capability with computational efficiency [16]. It incorporates an enhanced tokenizer and improved instruction tuning within a refined architecture, enabling reliable structured-output generation and code-oriented reasoning. These characteristics align well with the requirements of optical network design and planning, where high-level operator intents must be interpreted accurately and translated into deterministic analytical operations. The 8-billion (8B) parameter configuration further supports practical deployment in optical network management systems by enabling execution on a single graphics processing unit (GPU) while maintaining sufficient reasoning depth for intent-driven network automation workflows.

2) Mistral-NeMo (12B): Mistral NeMo (12B) is a larger open-weight language model developed by Mistral AI in collaboration with Nvidia, offering increased reasoning capacity compared to the LLaMA 3.1 (8B) model evaluated in this study [17], [18]. The model incorporates architectural enhancements such as grouped-query attention and efficient memory layouts, enabling effective processing of long prompts that involve complex instructions and multiple constraints. These capabilities are well aligned with intent-driven optical network automation, where user queries may require multi-step reasoning and consistent interaction with structured analytical tools. In the proposed framework, these characteristics contribute to improved reliability when generating tool invocations for complex network design and planning queries.

3) Gemma 2 (9B): Gemma 2 (9B) is an open-weight large language model from Google's Gemma family, designed for efficient deployment while maintaining strong instruction-following and reasoning capabilities [19]. The 9-billion-parameter configuration incorporates architectural refinements that improve performance relative to model size, making it suitable for resource-constrained environments. Within the context of this work, Gemma 2 (9B) serves as a lightweight alternative for optical network design and planning tasks, where operator intents must be interpreted accurately and mapped to structured analytical operations. Its reduced computational footprint enables faster inference and lower hardware requirements while still providing reliable performance for the evaluated network queries.

4) Qwen 2.5 (14B Instruct): Qwen 2.5 (14B Instruct) is an open-weight, instruction-tuned large language model developed by Alibaba, optimized for long-context reasoning and structured-output generation [20]. The model supports extended context windows, enabling reliable adherence to instruction constraints and robust handling of multi-parameter tool invocations. Such capabilities are well-suited to optical network automation, where accurate interpretation of operator requests is essential for transforming high-level intents into executable design and planning actions. With 14 billion parameters, Qwen 2.5 (14B Instruct) provides increased reasoning capacity and demonstrates consistently strong performance in intent understanding within the proposed framework, making it a high-accuracy candidate for autonomous optical network design and planning tasks.

These models are chosen as they provide balanced practical deployment considerations due to their diverse architectures. They can be executed locally, ensuring research reproducibility and data privacy. The set spans a range from 8B to 14B parameters, including models developed by different vendors. This ensures a systematic evaluation of accuracy, failure, and tool-use reliability, while aligning with our aim of assessing LLMs as reasoning agents for network designing and analytics.

III. Proposed Framework

This section presents the design, implementation, and evaluation methodology of the proposed LLM-based optical network design and analytics framework. The framework enables an LLM to operate as a reasoning and decision-making agent capable of interpreting natural-language queries related to optical networks and orchestrating a set of deterministic network analytics modules implemented in Python. Although these analytical functions are pre-built, the LLM retains autonomy in deciding which modules to invoke based on the user's intent and in determining how the resulting information is presented.

Fig. 1 illustrates the overall architecture of the proposed framework. The system follows an agent-driven design, with the LLM serving as the central reasoning agent. Network operators submit network-related queries, which are passed directly to the LLM. Based on the inferred intent, the LLM determines the required network operations like topology generation, path computation, visualization, etc., and selects the appropriate analytics functions to execute. The results are then formatted and presented to the user.

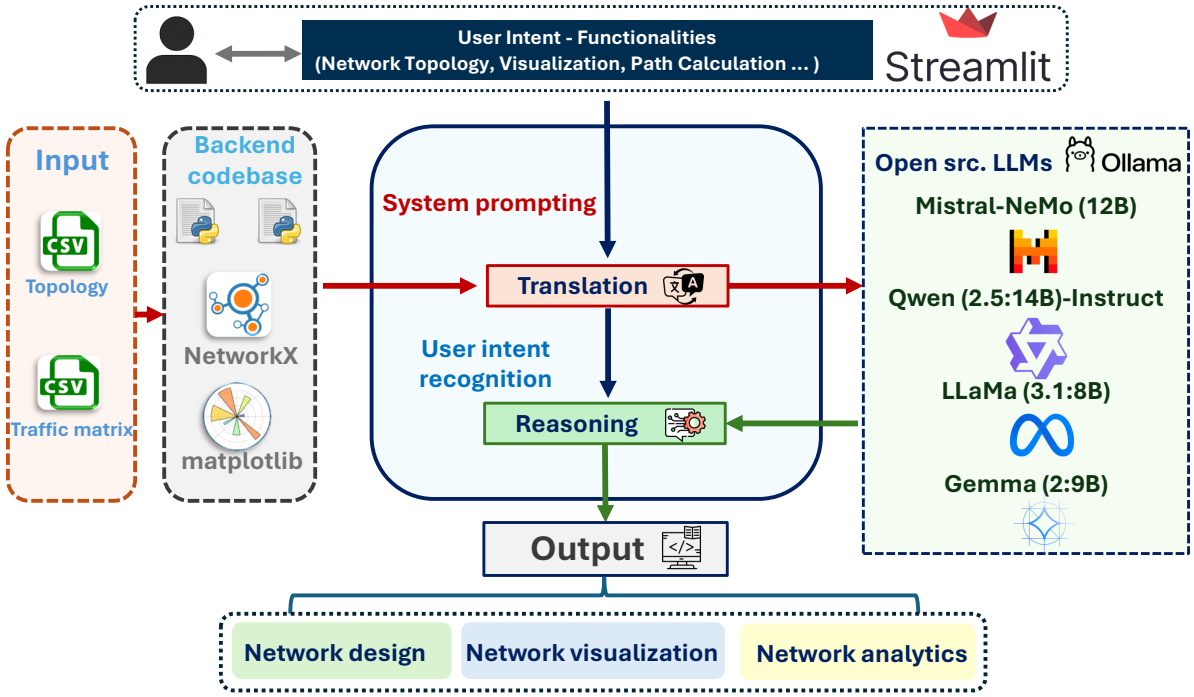


Fig. 1: LLM-assisted optical network design and analytics framework

The backend codebase consists of a lightweight set of domain-specific Python classes responsible exclusively for deterministic operations, such as the implementation of Dijkstra's algorithm for shortest path computation, and fiber reference modelling. These functions encapsulate domain knowledge that is not inherently available to the LLMs, as such information is not explicitly included in their pre-training.

We validated the framework by integrating these functionalities as deterministic backend modules for verification purposes. Network topology modelling and graph analytics are performed using the NetworkX library, ensuring reliable computation of paths, hop counts, and related metrics [21]. Visualization is implemented using Matplotlib, while the input section consists of CSV files containing node coordinates, link distance, and link classification. These CSV files serve as the data source, enabling accurate design and planning operations driven by the LLM models. A separate script implements the network equipment placement logic for reconfigurable optical add-drop multiplexer (ROADMs), transceivers, amplifiers, pre-amplifiers, and inline amplifiers (ILAs). This placement logic follows fixed engineering rules based on link distance and link type, thereby enforcing domain-specific behavior that the LLM can rely on during reasoning. Fiber reference information is stored in a dedicated file to support propagation-delay-related queries.

Integration of the input files and Python modules through an LLM agent script that acts as the management layer between the user interface and the codebase. The agent connects to the LLM models through `Ollama`, and a dedicated system prompt is used to enforce a strict JSON output format and instructions for interacting with the codebase, through which the LLM produces a structured intent plan containing the task type and relevant parameters. This plan is then mapped with the Python modules to generate the outputs. Finally, the output layer implemented in Streamlit enables user interaction with the LLM models and presents responses to network queries in structured formats such as visualizations and tables.

A. *Evaluation and Benchmarking*

To test our system for reliability and correctness, we devised a comprehensive benchmarking pipeline based on a fixed set of network queries covering basic to complex questions relating to topology generation, path computation, graph statistics, graph-level statistics, link analysis, equipment deployment, and propagation delay estimation. The evaluated queries were designed to reflect realistic operator-level requests. A dedicated Python script was developed to automatically collect and organize the responses generated by the LLMs into question-specific directories. All model outputs were subsequently reviewed and validated manually by a domain expert to assess correctness, consistency, and operational relevance.

Table I illustrates the categories of the network queries and examples of a representation of the different queries we asked the LLM models for evaluation. The categories include topology generation, path computation, graph and link statistics, equipment deployment, performance analysis, and constraint-based routing. These categories represent typical operator-level tasks in optical networks that include analytical operations and multi-constraint decision-making scenarios, demonstrating the practicality of the proposed framework.

B. *Scoring Metrics*

Each of the LLM models was evaluated using an accuracy metric defined as the fraction of benchmark queries for which

TABLE I: Categorization of evaluated network queries

Category	Example Queries
Topology Generation	Topology creation from CSV; topology visualization with link distances
Path Computation	Shortest path (Berlin–Munich); minimum-hop path (Berlin–Stuttgart)
Graph Statistics	Highest-degree node; average shortest-path length (hops)
Link Statistics	Average link distance; top-5 longest links with endpoints
Equipment Deployment	Equipment deployment; count of ROADMs and ILAs
Performance Analysis	Propagation delay (Berlin–Munich); delay minimization via fiber type
Constraint-Based Routing	Minimum-ILA path; paths satisfying ILA threshold constraints

the models produced a fully correct result. All responses were assigned one of three discrete scores based on the degree of correctness:

- A score of **1** (success) is assigned if the response fully satisfies the query intent and all output requirements.
- A score greater than **0** and less than **1** (partial correctness/hallucination) is assigned if the response is partially correct, incomplete, or inconsistent, thereby preventing full compliance with the expected result.
- A score of **0** (failure) is assigned if the response is entirely incorrect, non-executable, or missing.

Based on these scores, three complementary evaluation metrics are defined:

a) Failure Rate: The failure rate represents the proportion of queries for which the model produces a fully incorrect or non-executable response:

$$\text{Failure Rate} = \frac{\#(\text{score} = 0)}{N} \times 100 \tag{1}$$

b) Hallucination Rate: Hallucination is defined as the generation of partially correct, incomplete, or internally inconsistent outputs that do not fully satisfy the query intent. The hallucination rate is computed as:

$$\text{Hallucination Rate} = \frac{\#\,(0 < \text{score} < 1)}{N} \times 100 \tag{2}$$

c) Accuracy: Accuracy is defined as a strict metric that counts only fully correct responses. Partial correctness is explicitly excluded from accuracy to reflect the requirements of operational network environments where incomplete results are not acceptable:

$$\text{Accuracy} = \frac{\#(\text{score} = 1)}{N} \times 100 \tag{3}$$

IV. Performance Evaluation and Discussion

Table II summarizes the accuracy, hallucination, and failure rates for each evaluated model across all the Network Queries. The experimental results show that the Qwen-2.5-14B-Instruct model achieves the highest performance, with an accuracy score of 96.97%, a hallucination rate of 0%, and a failure rate of 3.03%. These results show a strong stability of this model for executing Python code and understanding strict output requirements across various network analytics tasks.

TABLE II: Comparison of open source LLMs

Model	Accuracy (%)	Hallucination (%)	Failure (%)
Qwen-2.5-14B-Instruct	96.97	0.00	3.03
Gemma-2-9B	78.79	9.09	12.12
LLaMA-3.1-8B	78.79	3.03	18.18
Mistral-NeMo-12B	69.70	15.15	15.15

Gemma-2-9B and LLaMA-3.1-8B both achieved 78.79% accuracy, with hallucination rates of 9.09% and 3.03%, respectively. These results indicate solid reasoning performance with occasional formatting or execution errors.

Mistral-NeMo-12B achieved the lowest accuracy of 69.70%, with the highest hallucination and failure rates among the compared models. The higher failure and partially correct outputs show reduced robustness in multi-step analytical reasoning tasks that require precise intent interpretation and structured tool calls. In this experiment, the model showed less consistent adherence to the strict JSON output format instructions, which led to incorrect results. Demonstrating that, despite its large size, it is less aligned for constrained tool-use scenarios compared to other models.

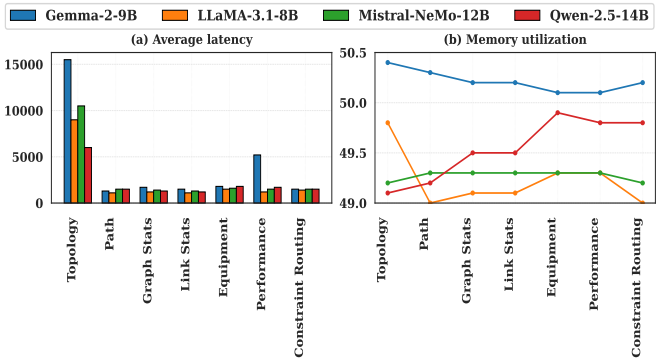
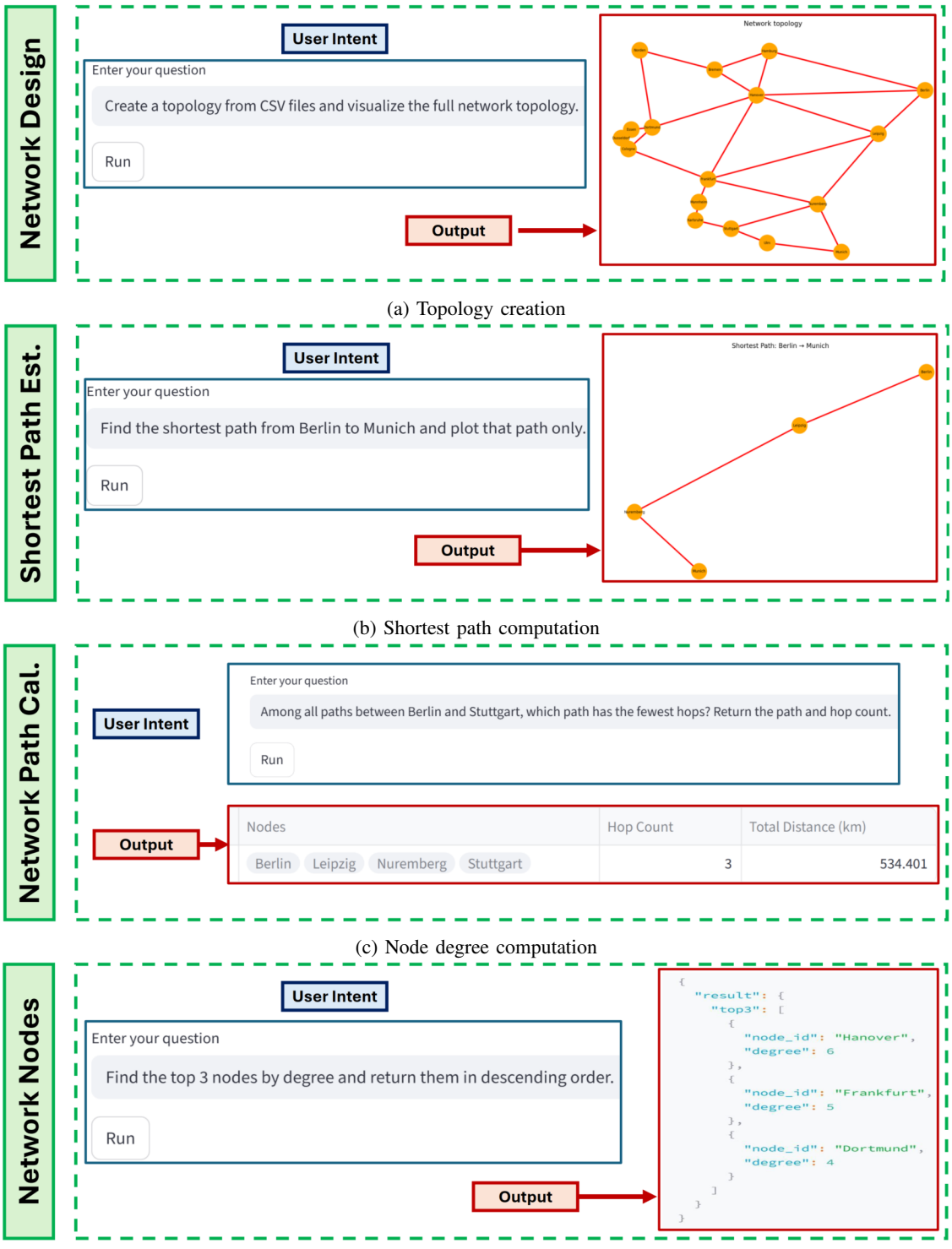


Fig. 2: Evaluation per category: (a) Average latency, (b) Memory utilization (%)

Fig. 2 shows inference latency variations across query categories, with tasks such as topology generation resulting in higher latency than analytical queries. While the Qwen model achieved the highest accuracy, it does not have the lowest latency. Also, the LLaMA model shows lower latency across most of the categories. Looking at part b of the figure, memory usage stays almost constant for all the models, where the LLaMA model uses significantly less memory than the Qwen model. These results indicate that despite the higher accuracy of the Qwen model, it has a trade-off against computational resources when compared to the smaller models.

A. Demonstration of Network Design and Analytics

We used the Qwen model in the Streamlit application to show the network design and analytics tasks. Fig. 3 covers four intents: topology generation from CSV (Figure 3a), shortest path (Figure 3b), minimum hop path (Figure 3c), and top-degree nodes. Figure 4 shows analytics tasks that include the longest links (Fig. 4a), ILA-minimized routing (Fig. 4b), and delay minimization with hollow-core fiber (Fig. 4c). Fig. 5



(a) Topology creation

(b) Shortest path computation

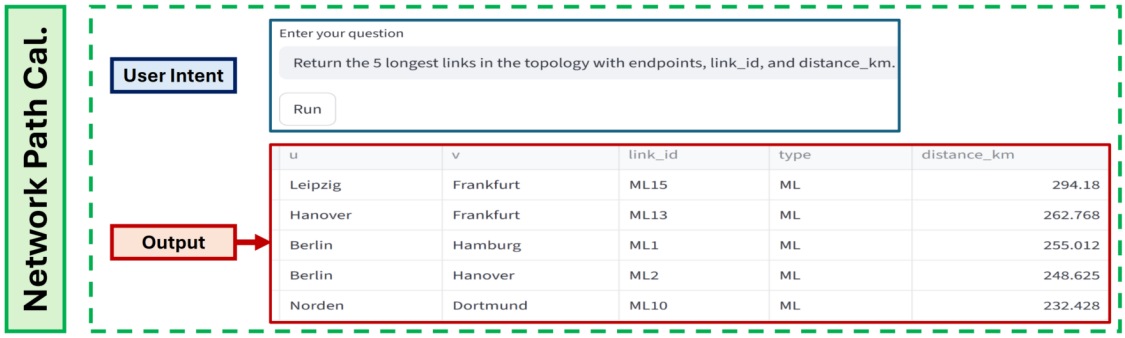
(c) Node degree computation

(d) Equipment deployment

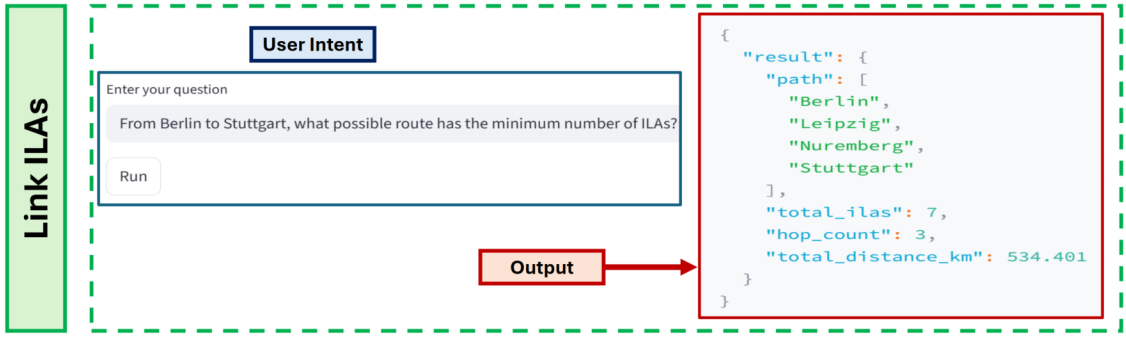
Fig. 3: LLM-assisted network design and analytics demonstrations - Part 1

presents equipment summarization. In all these tasks, the Qwen model correctly produced the results.

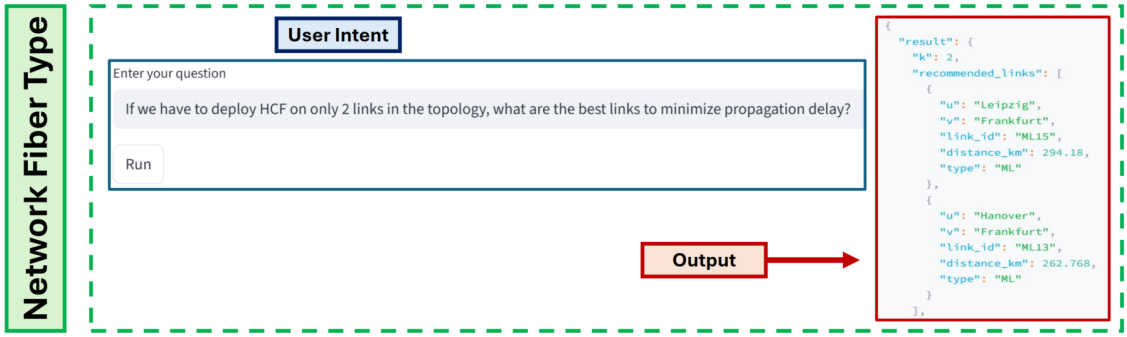
The correctness of these tasks demonstrates the Qwen models' strong understanding of interpreting the user queries and mapping them to the back-end codebase, and making the correct decisions leading to the delivery of reliable and actionable outputs for network operators. This further demonstrates that the proposed framework is suitable for a variety of network design and analytics tasks where the Qwen-2.5-14B-Instruct model showed strong accuracy. This model exhibits a strong interpretation of the back-end code base for graph operations, multi-constraint reasoning, aggregation, and



(a) Link (longest) information



(b) Route with ILAs constraint



(c) Minimizing delay

Fig. 4: LLM-assisted network design and analytics demonstrations - Part 2

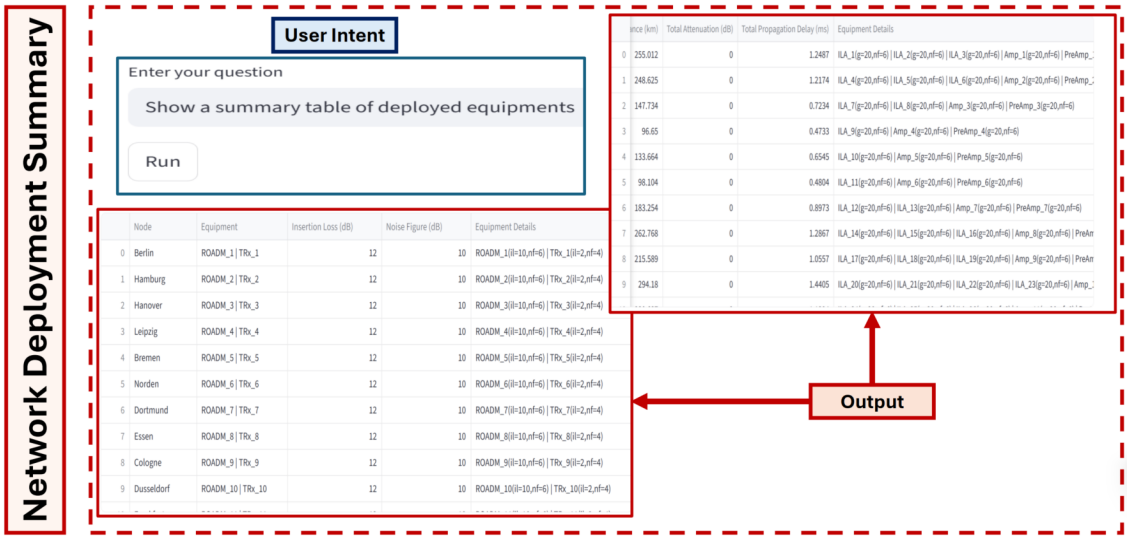


Fig. 5: Summary of deployed network

optimization. Hence, this model is ideal for deployment in LLM-driven applications for optical networks.

V. Conclusion and Future Work

We presented an LLM-based system for optical network design and analytics. We demonstrated LLMs as reasoning agents capable of interpreting human language to execute domain-specific network tasks. The results showed that the LLMs were effective in supporting various network tasks like network design, analytics, and equipment deployment. Among the evaluated models, the Qwen model was the best model for these tasks. These findings indicate that the Qwen model is feasible for deployment in real-world applications that can aid the network operators in performing design and analytics operations of optical networks.

Future work will focus on extending the proposed framework by integrating the GNPy library, enabling LLMs to analyze Quality of Transmission (QoT) impairments with higher accuracy. In addition, the incorporation of Retrieval-Augmented Generation (RAG) techniques will allow LLMs to leverage vendor-specific data sources, thereby enhancing their ability to support network operators in more informed and efficient planning decisions.

Acknowledgments

Funded by the European Union's Horizon Europe research and innovation programme under project NESTOR G.A. No. 101119983 and ALLEGRO G.A. No. 101092766, in collaboration with Nokia Optical Networks (Carnaxide, Portugal & Munich, Germany), & Nokia Bell Labs (Murray Hill, USA).

References

[1] J. G. A. et al., "What Will 5G Be?" *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1065–1082, 2014.

[2] S. V. et al., "Toward 6G Networks: A Survey on Enabling Technologies," *IEEE Communications Surveys & Tutorials*, 2020.

[3] A. S. Thyagaturu, A. Mercian, M. P. McGarry, M. Reisslein, and W. Kellerer, "Software defined optical networks (sdons): A comprehensive survey," *IEEE Communications Surveys & Tutorials*, 2016.

[4] Nokia Bell Labs, "Optical Network as a Service: Concepts and Implementations," 2021, white Paper.

[5] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong *et al.*, "A survey of large language models," *arXiv preprint arXiv:2303.18223*, vol. 1, no. 2, 2023.

[6] S. Long, J. Tan, B. Mao, F. Tang, Y. Li, M. Zhao, and N. Kato, "A survey on intelligent network operations and performance optimization based on large language models," *IEEE Communications Surveys & Tutorials*, vol. 27, no. 6, pp. 3915–3949, Dec. 2025.

[7] Y. Zhang, Y. Song, Y. Pang, S. Li, X. Jiang, Y. Wang, and D. Wang, "Design and evaluation of an llm-based agent for qot estimation and performance optimization in optical networks," *IEEE Open Journal of the Communications Society*, 2025.

[8] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," *arXiv preprint arXiv:2210.03629*, 2023.

[9] T. Schick *et al.*, "Toolformer: Language models can teach themselves to use tools," *arXiv preprint arXiv:2302.04761*, 2023.

[10] Y. Qin *et al.*, "Toolllm: Facilitating large language models to master 16000+ real-world apis," *arXiv preprint arXiv:2307.16789*, 2023.

[11] L. Ouyang *et al.*, "Training Language Models to Follow Instructions," *arXiv preprint arXiv:2203.02155*, 2022.

[12] A. Vaswani *et al.*, "Attention is All You Need," *Advances in Neural Information Processing Systems*, 2017.

[13] J. Ainslie *et al.*, "Gqa: Training generalized multi-query transformer models from multi-head checkpoints," *arXiv preprint arXiv:2305.13245*, 2023.

[14] R. Sennrich, B. Haddow, and A. Birch, "Neural Machine Translation of Rare Words with Subword Units," in *ACL*, 2016.

[15] T. Kudo and J. Richardson, "Sentencepiece: A Simple and Language Independent Subword Tokenizer and Detokenizer for Neural Text Processing," in *EMNLP*, 2018.

[16] A. Grattafiori *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[17] Mistral AI, "Mistral nemo," Mistral AI Technical Release, 2024, available at https://mistral.ai/news/mistral-nemo/.

[18] NVIDIA, "Power text generation applications with mistral nemo 12b running on a single gpu," NVIDIA Developer Blog, 2024, accessed: 2026-03-24. [Online]. Available: https://developer.nvidia.com/blog/power-text-generation-applications-with-mistral-nemo-12b-running-on-a-single-gpu/

[19] Gemma Team, "Gemma 2: Improving open language models at a practical size," *arXiv preprint arXiv:2408.00118*, 2024.

[20] A. Yang *et al.*, "Qwen2.5 technical report," *arXiv preprint arXiv:2412.15115*, 2024.

[21] M. U. Masood, S. Zeb, I. Khan, and V. Curri, "Evaluating capacity scaling strategies for cost-efficient optical network evolution," in *Optical Fiber Communication Conference (OFC)*, 2026, accepted paper.