AI-vOLT: Multi-Stage Agentic Translation from Operator Intent to Executable PON Procedures

Chansung Park, Yongwook Ra, Hwan Seok Chung
Optical Network Research Section
Electronics and Telecommunications Research Institute (ETRI)
Daejeon, Korea
{chansung18, nyw0242, chung}@etri.re.kr

***Abstract*—We propose AI-vOLT, a virtual OLT system integrating SEBA with large language models (LLMs) to enable intent-based PON provisioning through natural language interfaces. Manual provisioning in traditional Passive Optical Networks (PONs) remains resource-intensive and error-prone, limiting their adaptability to next-generation service demands. AI-vOLT addresses this challenge by translating high-level operator intent into low-level executable commands through a multi-stage agentic workflow of planning, shaping, and execution. We evaluate the framework across four representative provisioning scenarios and multiple LLM backends under varying contextual inputs. Experimental results show that AI-vOLT achieves near-perfect provisioning success ($\approx 99\%$) in simulated environments and is further validated on a 25G physical PON testbed. These findings confirm the practicality of AI-vOLT for reliable, language-driven automation of optical access networks.**

***Index Terms*—AI-vOLT, Large Language Model (LLM), Agentic Workflow, Intent-based Networking, Software Defined Networking (SDN)**

I. Introduction

The increasing demands of next-generation services—such as extended reality (XR), enhanced mobile broadband, and ultra-low-latency communication—are pushing optical access networks to become more dynamic and responsive. While Software-Defined Networking (SDN) has partially addressed the rigidity of traditional Passive Optical Networks (PONs) [1], manual provisioning and management tasks still remain resource-intensive and error-prone. Recent advances in Large Language Models (LLMs), trained on extensive corpora of natural language, have demonstrated remarkable capabilities in understanding, reasoning, and executing complex instructions. These capabilities introduce a new paradigm in network automation by enabling the translation of high-level human intent into system-level commands. To date, research on PON automation has primarily focused on rule-based orchestration [1]–[3] or classical machine learning [4], [5] approaches. The use of generative LLMs for intent-driven provisioning and control in PON systems remains underexplored, with limited prior research available.

In this paper, we demonstrate AI-powered virtual OLT (AI-vOLT), an intelligent vOLT system that combines the SDN Enabled Broadband Access (SEBA) [2] architecture with LLM-based agents [6] to automate PON provisioning. By employing natural language interfaces and prompt engineering, AI-vOLT interprets operator instructions and orchestrates network services without requiring infrastructure modifications. Furthermore, we evaluate multiple LLM backends—including both commercial and open-source models—across diverse operational workflows. This comprehensive study highlights their strengths and limitations under varied contextual inputs and demonstrates the feasibility of LLM-based intent-driven access network control.

II. System Architecture of AI-vOLT

A. Framework Overview: Components and Operational Flow

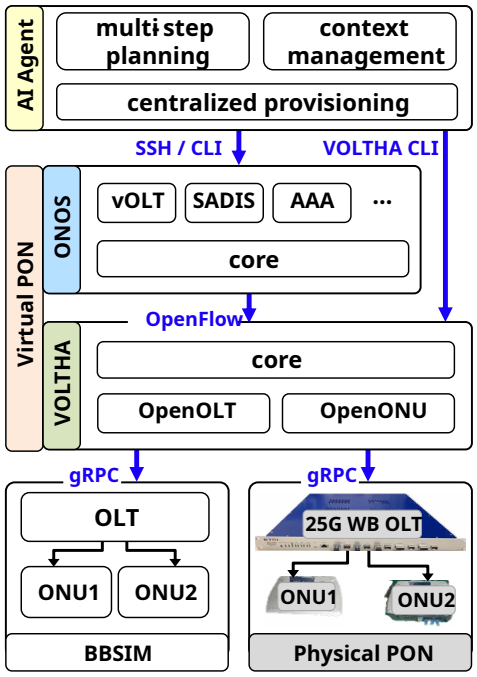


Fig. 1. System architecture of AI-vOLT, evaluated using BBSIM and deployed on a physical PON testbed.

The overall structure of AI-vOLT is shown in Fig. 1. The AI Agent serves as the central entry point for operator-issued instructions expressed in natural language. These instructions are processed and translated into operation commands. ONOS (Open Network Operating System) [7] functions as SDN controller, managing flow rules and device abstraction while

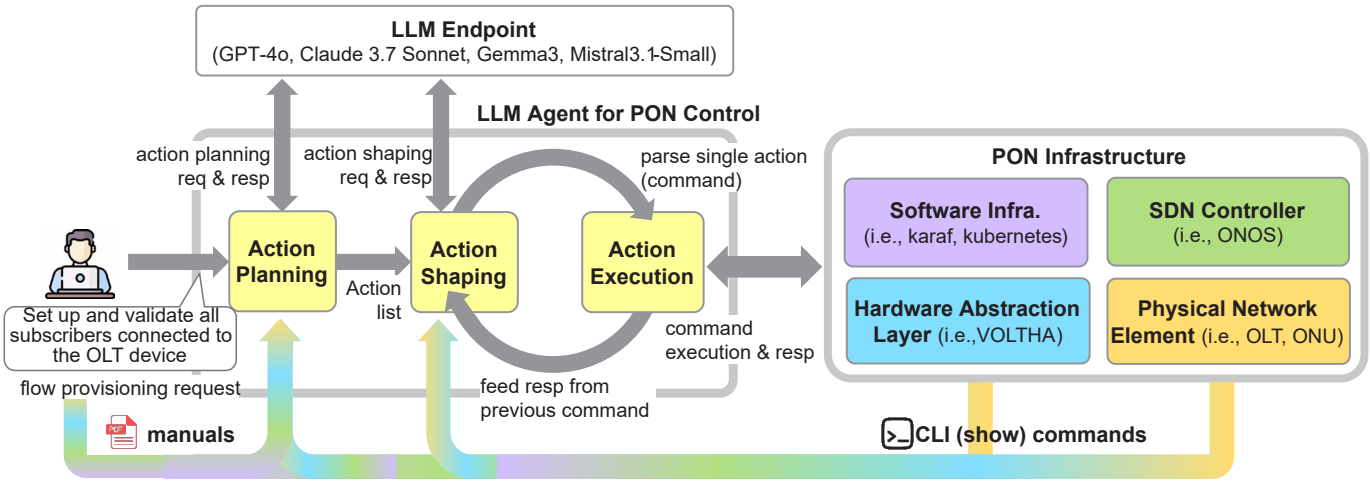


Fig. 2. Multi-stage agentic workflow of AI-vOLT for intent-based PON provisioning.

VOLTHA (Virtual OLT Hardware Abstraction) acts as the mediation layer between ONOS and access devices, exposing control over OLTs and ONUs. BBSIM (Broadband Simulator) emulates PON behavior and allows for rapid testing, while the physical PON setup includes actual OLT and ONU devices used for final validation.

Upon receiving an instruction from the operator, the AI Agent interprets the intent and generates a sequence of operation commands. Each command is routed to the appropriate subsystem (ONOS or VOLTHA) based on its execution target within the overall control flow. Depending on the nature of the command, the corresponding subsystem either directly interacts with the target environment (BBSIM or physical PON) or coordinates provisioning through intermediate layers. For example, flow rules configured in ONOS are propagated to VOLTHA via OpenFlow, and subsequently applied to access devices.

B. AI Agent: Translating Intent into Network-Level Actions

The AI-vOLT framework employs AI agent as an intelligent intermediary that translates operator instructions into executable actions. As illustrated in Fig. 2, the agent first gathers internal and external knowledge sources, where internal data include dynamic system states obtained via CLI outputs, and external knowledge such as operation manuals is provided in formats like PDFs and spreadsheets. This collected information serves as input to the LLM backends throughout the agent's workflow, supporting context-aware reasoning.

Building on the gathered knowledge, the agent executes a structured workflow consisting of three primary components:

1) Action planning: interprets operator instructions and formulates a high-level, descriptive execution plan that outlines the sequence of steps.

2) Action shaping: transforms each step of the execution plan into concrete, executable commands. The output of previously executed commands is incorporated as additional context to resolve dependencies between sequential operations within the workflow.

3) Action execution: issues the commands to the appropriate subsystems (ONOS or VOLTHA) and retrieves execution results. These results are fed back into the shaping step to enable adaptive prompting.

Through iterative interaction with the operational environment, The agent dynamically adapts its behavior throughout the planning, shaping, and execution stages. This closed loop mechanism enables responsive control, reduces manual overhead, and accelerates service provisioning, without requiring any changes to the existing PON infrastructure.

III. Experimental Setup

This study assesses the functionality and effectiveness of AI-vOLT across four representative operational scenarios: OpenOLT adapter creation, OLT activation, AAA, and flow installation. Initial evaluation is conducted using the BBSIM to ensure correctness and repeatability. Validated scenarios are then deployed on a physical PON testbed consisting of 1x25G white-box OLT and 2x25G ONUs to verify real-world applicability.

TABLE I
Overview of Evaluated Scenarios, Language Models, and Contextual Configurations in AI-vOLT Experiments

Component type	Evaluated items
Scenarios	• OpenOLT adapter creation • OLT activation • AAA • Flow installation
Commercial LLMs	• GPT-4o • GPT-4o-mini • Claude 3.5 Sonnet • Claude 3.7 Sonnet
Open-weight LLMs	• Gemma3-27B-IT • Mistral-3.1-Small-24B-IT
Context levels	• full context • w/o scenario • w/o scenario+term • no context
Target PON	• BBSIM (Broadband Simulator) • physical PON (1x25G WB OLT, 2x25G ONUs)

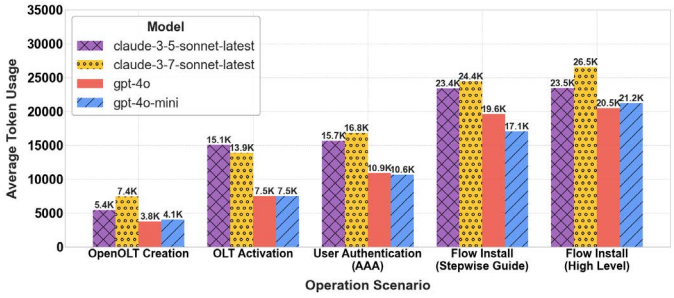


Fig. 3. Comparative average token usage per scenario across LLMs during AI-vOLT operation.

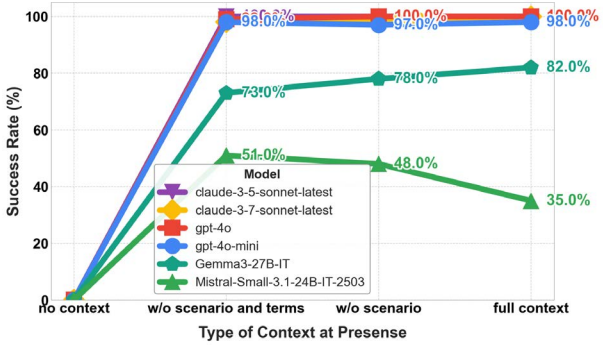


Fig. 4. Context sensitivity analysis of LLMs: Success rates under varying context levels.

We evaluate six LLMs to compare their performance in interpreting and executing operational workflows. The commercial models include *GPT-4o*, *GPT-4o-mini*, *Claude 3.5 Sonnet*, and *Claude 3.7 Sonnet*, while the open-weight models consist of *Gemma3-27B-IT* and *Mistral-3.1-Small-24B-IT*, both deployed on a 4×RTX 6000 Ada-equipped server. To emulate realistic operator input diversity, ten paraphrased prompt variants were generated per scenario using LLM-based rewriting. Each variant was tested ten times per model, resulting in 100 trials per model-scenario pair.

To evaluate model robustness under varying information granularity, four context levels are defined: (1) full context, which includes all relevant documents such as operational procedures and terminology guides; (2) w/o scenario, excluding procedural instructions; (3) w/o scenario+term, further omitting terminology documentation; and (4) no context, retaining only individual command descriptions. A summary of all evaluated components—including scenarios, model types, and context levels—is provided in Tab. I.

IV. Evaluation Results and Analysis

A. Token Efficiency and Task Success Analysis

To assess operational efficiency in addition to task accuracy, we examined the average token consumption per scenario. Token usage was accumulated throughout the closed-loop workflow of AI Agent, capturing all interactions with a LLM backend across planning, shaping, and execution stages. As shown in Fig. 3, Claude variants consistently consumed the most tokens, particularly in complex tasks such as flow installation, while maintaining strong success rates. In contrast, GPT-4 variants achieved similarly high performance with significantly fewer tokens, demonstrating greater efficiency.

These results suggest that token efficiency varies significantly across models and is not necessarily indicative of task performance. Excessively verbose generation patterns, observed in certain Claude variants, may introduce unnecessary inference overhead in practical deployments. Among the tested models, *GPT-4o-mini* offers a favorable balance between output compactness and reliability, making it a cost-effective option for repeated use.

We next analyzed how each model responds to varying levels of contextual information using the high-level flow installation task. This scenario requires approximately eight sequential operations and serves as a representative case for evaluating multi-step reasoning. As shown in Fig. 4, commercial models maintained high success rates ($\geq$ 97%) even under reduced context, indicating strong generalization. In contrast, the open-weight Gemma3 model showed substantial improvement as more context was provided, while Mistral-Small experienced performance degradation under full input, suggesting sensitivity to context complexity or potential overfitting.

Figure 5 presents the overall success rates across all four AI-vOLT operational scenarios. Commercial models—*Claude 3.5 Sonnet* (100%), *Claude 3.7 Sonnet* (99.0%), *GPT-4o* (99.7%), and *GPT-4o-mini* (97.7%)—demonstrated consistent performance across tasks. In contrast, open-weight models such as *Gemma3* and *Mistral Small* achieved lower average success rates of 77.7% and 44.7%, respectively.

These findings indicate that prompt and context design must be tailored to each model's capacity and tolerance. While commercial models are generally robust to ambiguity and operate reliably with minimal guidance, smaller or open-weight models may require concise, well-scoped inputs for optimal performance. In on-premise or cost-sensitive deployments where commercial models are unavailable, open-weight LLMs may require purpose-specific adaptation [8], [9].

We also assessed the cost implications of repeated inference

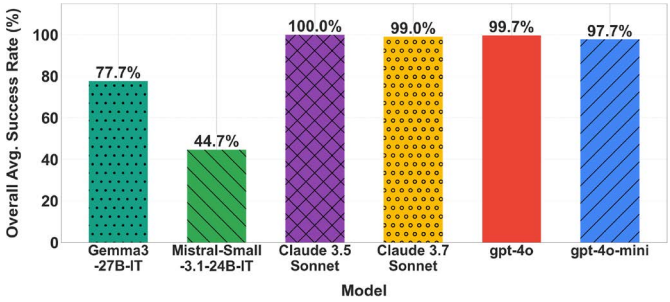
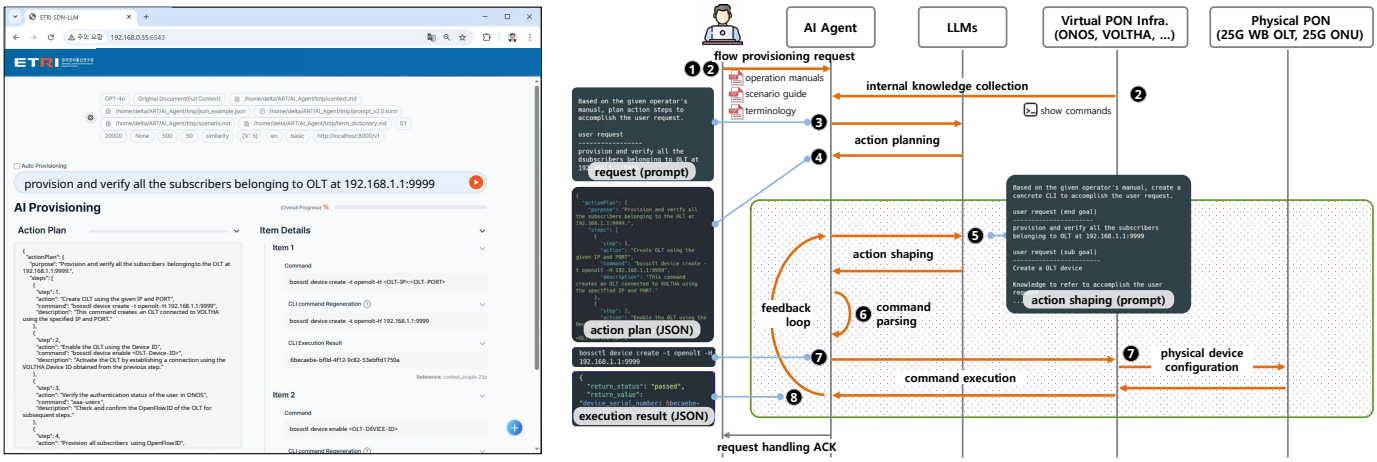


Fig. 5. Comparative task success rates across operational scenarios: Consistent performance of commercial LLMs versus variability in open-weight models.



(a) Web-based Provisioning Interface

(b) operational workflow of AI-vOLT on a physical PON testbed

Fig. 6. Demonstration of AI-vOLT on a physical PON testbed. (a) Web-based interface for submitting natural language requests and inspecting intermediate outputs. (b) End-to-end operational workflow of AI-vOLT during service provisioning over a 25G physical PON setup. Numbered steps illustrate the closed-loop control process, including knowledge acquisition, execution plan generation, command shaping, and real-time feedback handling.

using commercial APIs. Running *GPT-4o* for 100 executions consumed approximately 2 million tokens, incurring a cost of $10–$20. Scaling this to 3,000 executions would reach the cost of operating a self-hosted GPU cluster (e.g., approximately $45,000 for 4×RTX 6000 Ada). This comparison highlights the importance of considering long-term operational costs when selecting LLMs. While commercial APIs offer convenience and minimal setup overhead, cumulative expenses can quickly outpace one-time infrastructure investments. Model selection should therefore reflect a balance between reliability, scalability, and budget.

Our evaluation of AI-vOLT surfaces three key insights for LLM-based PON automation. First, task success depends strongly on how context is structured, highlighting the importance of prompt and input design. Second, token usage varies widely across models, impacting long-term cost. Third, while commercial models perform reliably out of the box, open-weight models in AI-vOLT require careful tuning for consistent operation. These findings inform practical decisions when deploying language-driven network control.

B. End-to-End Operational Validation

To verify operational feasibility of AI-vOLT beyond emulated environments, we applied the AI-vOLT framework to a physical PON testbed and executed the full flow installation scenario under real-world conditions. This experiment follows the same control logic previously tested on BBSIM, but targets actual 25G white-box OLT and ONU devices. Figure 6(a) triggered the flow installation operational sequence illustrated in Fig. 6(b), and the numbers present the demonstration workflow of AI-vOLT.

❶ The process begins when the operator issues a natural language request, such as provisioning a service flow between a specific OLT and ONU. ❷ The action planning component gathers internal knowledge (e.g., CLI-based status and configuration) from the underlying system and external knowledge (e.g., operation manuals in PDF format) manually uploaded by the operator. ❸ This combined knowledge is structured into a prompt—together with the user request and a predefined JSON output format—and sent to the LLM endpoint. The returned response is parsed to extract the action plan. ❹ The action plan includes a sequence of steps that decompose the original request into granular actions.

A processing loop is then established between the action shaping and action execution components to carry out the action plan: ❺ The action shaping component sequentially selects a step from the action plan, formulates a prompt based on the step and related knowledge, and queries the LLM to generate a concrete command. ❻ The resulting response is parsed to extract a valid executable command. ❼ This command is executed on the appropriate subsystem: a Karaf session is used for ONOS, while the VOLTHA CLI is directly invoked for VOLTHA-based tasks. ❽ The execution result—including pass/fail status and relevant output—is captured and fed back into the shaping component to inform the next step. This loop continues until all steps are completed or an error condition is encountered.

The successful execution of this workflow on physical infrastructure confirms that AI-vOLT can reliably translate high-level operator intent into low-level provisioning commands in practical deployment scenarios.

CONCLUSION

In this paper, we presented AI-vOLT, a language driven automation framework for SEBA based PON environments. By combining a multi stage agentic workflow with LLM backends, AI vOLT successfully translated operator intent into executable provisioning commands across diverse scenarios. Our experiments demonstrated near perfect provisioning success ($\approx 99\%$) across four representative scenarios (OpenOLT

adapter creation, OLT activation, AAA, and flow installation) with commercial LLMs. We further validated AI vOLT on a 25G physical PON testbed, achieving reliable end to end provisioning in real world conditions. While commercial models performed reliably out of the box, open weight models exhibited larger variance and typically require structured prompts or domain specific fine tuning to reach comparable stability. Overall, AI-vOLT effectively bridges operator intent and automated PON provisioning, providing a practical path toward scalable and intelligent automation in future optical access networks.

ACKNOWLEDGMENT

This work was supported by IITP grant funded by the Korea government(MSIT)[RS-2023-00215959, Development of Access Agnostic wired and wireless integrated optical access technology].

REFERENCES

[1] Y. Ra, C. Park, K. Hwang, K.-H. Doo, K. O. Kim, H. H. Lee, T. Cheung, J. Shin, and H. S. Chung, "Field Trial of Remotely Controlled Smart Factory based on PON Slicing and Disaggregated OLT," in *2022 European Conference on Optical Communication (ECOC)*. IEEE, 2022, pp. 1–3.

[2] S. Das, "From CORD to SDN enabled broadband access (SEBA)[Invited Tutorial]," *Journal of Optical Communications and Networking*, vol. 13, no. 1, pp. A88–A99, 2021.

[3] A. Leivadeas and M. Falkner, "A survey on intent-based networking," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 625–655, 2022.

[4] J. A. Hatem, A. R. Dhaini, and S. Elbassuoni, "Deep learning-based dynamic bandwidth allocation for future optical access networks," *IEEE Access*, vol. 7, pp. 97 307–97 318, 2019.

[5] L. Ruan, M. P. Dias, and E. Wong, "Enhancing latency performance through intelligent bandwidth allocation decisions: a survey and comparative study of machine learning techniques," *Journal of Optical Communications and Networking*, vol. 12, no. 4, pp. B20–B32, 2020.

[6] Z. Xi, W. Chen, X. Guo, W. He, Y. Ding, B. Hong, M. Zhang, J. Wang, S. Jin, E. Zhou *et al.*, "The rise and potential of large language model based agents: A survey," *Science China Information Sciences*, vol. 68, no. 2, p. 121101, 2025.

[7] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow, and G. Parulkar, "Onos: towards an open, distributed sdn os," in *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, ser. HotSDN '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 1–6. [Online]. Available: https://doi.org/10.1145/2620728.2620744

[8] F. Wang, J. Jiang, C. Park, S. Kim, and J. Tang, "Kasa: Knowledge-aware singular-value adaptation of large language models," in *International Conference on Learning Representations*, 2025, accepted at ICLR 2025. [Online]. Available: https://openreview.net/forum?id=OQqNieeivq

[9] C. Park, J. Jiang, F. Wang, S. Paul, and J. Tang, "LlamaDuo: LLMOps pipeline for seamless migration from service LLMs to small-scale local LLMs," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, W. Che, J. Nabende, E. Shutova, and M. T. Pilehvar, Eds. Vienna, Austria: Association for Computational Linguistics, Jul. 2025, pp. 33 194–33 215. [Online]. Available: https://aclanthology.org/2025.acl-long.1592/