Towards Network Agentic AI: Offloading Small Language Models to SmartNICs

Victor H. S. Lopes[1], Francisco G. Vogt[3], Ariel G. de Castro[3],
Fabricio E. R. Cesen[4], Fabio D. Rossi[2], Christian E. Rothenberg[3], Marcelo C. Luizelli[1]

[1]Universidade Federal do Pampa (UNIPAMPA), [2]Instituto Federal Farroupilha (IFFar)
[3]Universidade Estadual de Campinas (UNICAMP), [4]Telefonica Research

***Abstract*—Network agentic AI brings autonomous, adaptive decision-making to network management, but its effectiveness depends on fast and efficient reasoning. Today, this reasoning is commonly powered by Small Language Models (SLMs) running on CPUs or GPUs, which introduces latency and limits scalability. This paper investigates the feasibility of offloading the Agentic AI reasoning stage to SmartNICs. We evaluate a representative set of SLMs executed on an NVIDIA BlueField-3 SmartNIC and compare their performance against server-grade CPU and GPU deployments. Our results show that small SLMs achieve latency in the same order of magnitude as a data-center GPU, while substantially reducing host CPU usage and memory footprint. These findings highlight the potential of near-data reasoning to enable more efficient and scalable Agentic AI deployments.**

***Index Terms*—Agentic AI, SmartNIC, LLM, SLM, Offloading**

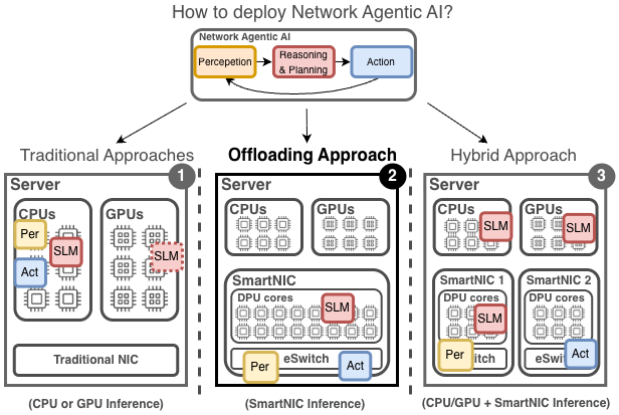

Figure 1: Simplified agentic AI deployment strategies.

I. Introduction

Agentic AI has emerged as a promising paradigm to enable autonomous decision-making in network management [1]–[3]. It differs from conventional AI methods [4], [5] by moving beyond passive pattern recognition toward autonomous, goal-driven behavior capable of reasoning, planning, and adapting over time. Traditional AI models typically learn static input–output mappings and require periodic retraining to remain effective, which reduces their applicability in dynamic environments. In contrast, an agentic AI integrates mechanisms such as memory, reflection, and multi-step planning that enable the system to assess context, anticipate consequences, and adjust actions proactively without human intervention [1].

The level of autonomy envisioned by agentic AI surpasses the reactive, rule-based automation of current AI-assisted network management systems [4]. Potential agentic-driven applications include self-directed troubleshooting and proactive performance management. In the former, an agentic AI can diagnose anomalies, reason about alternative root causes, validate corrective actions in a safe environment (e.g., a Network Digital Twin [6]), and apply effective remediation without relying on predefined playbooks. In the latter, the agentic AI continuously observes traffic dynamics, anticipates congestion or service degradation, and adjusts routing, scheduling, or resource allocation before service-level violations occur. These capabilities enable anticipatory, adaptive, and self-improving network control.

A core enabler of agentic AI is the reasoning component, responsible for interpreting observations, setting goals, devising plans, and making decisions. Reasoning commonly relies on advanced models, such as Large or Small Language Models (LLMs/SLMs) [7], [8], that process contextual information, generate explanations, and support multi-step planning. These models can be complemented by reinforcement learning techniques to allow the agent to learn from interactions with the environment and refine its decision-making over time.

Reasoning engines are typically deployed on centralized servers or cloud platforms, often backed by CPUs and GPUs [9], which introduces latency and limits the responsiveness required for time-sensitive networking and agent-based decisions. In this paper, we take the first step towards offloading the reasoning component of agentic AI to SmartNICs. Figure 1 illustrates possible deployment strategies for a Network agentic AI. In ❶, it illustrates the traditional agentic AI deployment to a cloud/edge server. In this deployment, the main agentic AI components run in the host CPU, while reasoning can be offloaded to the GPU when available. In ❷ all agentic AI components are offloaded to a SmartNIC, running either on the ARM cores (e.g., the reasoning) or in specialized hardware engines (e.g., the perception and action). Last, in ❸, it is still possible to combine both strategies and offload some of the agentic AI to the SmartNIC hardware (e.g., the perception and action), while others are balanced to ARM,

CPU, and GPU cores according to the demand.

Executing reasoning tasks on the data plane offers three main benefits for network operations. First, it reduces latency by executing decision-making directly where the agentic AI perceives the environment and enforces actions. This proximity allows the agentic AI to, for example, forecast traffic surges while processing packets and promptly adjust routing policies on network flows. As a result, it reduces the need for round-trip to external servers. Second, it decreases host CPU load and control-plane involvement, improving resource efficiency. Third, it enables fine-grained and time-sensitive actions at the packet/flow level, allowing the system to react to microbursts, anomalies, and policy changes with greater agility.

Previous efforts in offloading AI to the network (i.e., in-network AI) have shown that programmable data plane devices can host AI/ML tasks [10]. These studies explored different model architectures and accelerators (e.g., neural networks [11], [12], clustering [13], and decision trees [14]), but did not consider the use of SLMs as part of the in-network intelligence stack. LLMs, and to a lesser extent SLMs, usually require more computational resources than traditional ML models due to their transformer-based architectures, higher parameter counts, and token-based inference process [8]. In this work, we implement a SmartNIC-based reasoning prototype using a representative set of SLMs and evaluate it on an NVIDIA BlueField SmartNIC architecture. Our main goal in this preliminary study is to understand the feasibility deploying the reasoning stage of a network agentic AI system into SmartNICs (as illustrated in Figure 1 - ❷). Our results show substantial host CPU savings while maintaining inference latency comparable to a traditional host-based GPU deployment with an AI-grade GPU. Our main contributions can be summarized as follows: **(i)** performance evaluation of SLMs for different offloading strategies of agentic AI, **(ii)** open-source code of all experiments to foster reproducibility, and **(iii)** discussion of limitations associated with deploying network agentic AI.

II. Background and Related Work

A. Agentic AI

Agentic AI adopts a pipeline that enables autonomous and adaptive behavior throughout the decision-making process. The pipeline typically includes perception, reasoning, planning, action, and reflection stages that operate in a closed loop (see Figure 1 for a simplified overview). First, the agent collects context from the environment and interprets it to form a situational understanding (e.g., network load condition). Next, reasoning and planning modules evaluate possible actions, anticipate their effects, and select strategies aligned with predefined goals (e.g., to prevent QoE degradation, proactively change routing policies of video streaming sessions). After executing the chosen action, the agent observes outcomes, stores experiences in memory, and performs self-reflection to improve future decisions. This continuous learning loop enables the agent to evolve its behavior over time and remain effective in dynamic environments.

Unlike traditional multi-agent systems (MAS), which rely on coordination among specialized agents, agentic AI emphasizes goal-driven autonomy within each agent. MAS usually decomposes complex tasks into several agents with predefined roles, communication protocols, and cooperation strategies to achieve a collective objective. In contrast, agentic AI focuses on enhancing the cognitive capabilities of individual agents, enabling them to reason, adapt, and refine their own policies without relying on explicit inter-agent coordination. This distinction shifts the design paradigm from orchestrating interactions among specialized agents to empowering each agent with self-improving intelligence, making them suitable for SmartNIC offloading and taking autonomous decisions in the context of network management.

B. Serving Agentic AI

As discussed, the reasoning module of an agentic AI relies on language models. LLMs are commonly served through distributed inference architectures optimized for low latency, high throughput, and elastic scalability [9]. The inference engine loads the model, generates tokens, and manages hardware resources, often relying on GPUs, TPUs, or dedicated accelerators. Techniques such as model parallelism, quantization, and batching improve performance, while strategies like speculative decoding and prompt caching balance quality and responsiveness. This layer also integrates monitoring to track performance and resource usage. In contrast, SLMs can be served using lightweight and resource-efficient architectures that maintain responsiveness with minimal computational overhead. Their smaller memory footprint enables deployment on commodity hardware, edge servers, or SmartNICs, allowing low-latency inference close to data sources while still retaining strong reasoning and instruction-following performance, even with one to two orders of magnitude fewer parameters than large foundation models [15]. Serving stacks typically use quantized models, single-device execution, and compact batching to avoid queuing delays. Due to shorter context windows and faster token generation, SLMs offer interactive responses with limited infrastructure, making them suitable for on-device reasoning and latency-sensitive applications.

C. SmartNIC Landscape

Modern SoC-based SmartNICs – such as the NVIDIA BlueField-3 (BF-3) [16] – feature heterogeneous architectures that integrate programmable packet-processing pipelines (e.g., P4/DOCA [17]) with multi-core 64-bit ARM CPUs, on-board DRAM, and high-speed DMA engines. This combination enables flexible offloading of network and compute-intensive workloads while preserving line-rate packet forwarding. Structured telemetry – such as counters, sketches, and flow records – can be ingested and processed directly on the device to capture context, while the embedded cores make decisions that translate into adaptive updates to match-action tables or forwarding rules in real time. These capabilities make SmartNICs as a compelling platform for deploying lightweight

reasoning modules, enabling near-data intelligence and adaptive decision-making at line rate.

D. Related Work

In-network ML. A substantial body of research has investigated the offloading of machine learning models to programmable networks and SmartNICs [10]–[12], [14]. However, little progress has been made regarding the offloading of Language Models. Recent studies have begun exploring partial offload of the Language Model pipeline. NICTokenizer [18] offloads tokenization to SmartNICs, reducing CPU load and lowering latency in AI processing pipelines. In the same direction, OptimusNIC [19] targets large-scale LLM training by offloading optimizer state and parameter updates to SmartNICs, thereby freeing GPU resources and mitigating communication overhead.

Network agentic AI. Agentic AI has been applied to network management, with early contributions from the research community. Zhang *et al.* [20] employ LLM-driven generative agents for interactive system modeling and integrate a mixture-of-experts proximal policy optimization (MoE-PPO) scheme to enable adaptive and efficient transmission strategies in large-scale satellite communication networks. AgentNet [21] introduces multi-agent collaboration mechanisms that support communication, adaptation, and autonomous decision-making in networked systems. SANNet [22] proposes semantic-aware, cross-layer agent orchestration to enable real-time network control. More recently, Zhang *et al.* (2025) [23] developed an agentic contextual retrieval framework that enhances telecom planning by combining multi-source retrieval with structured reasoning and self-reflective validation.

Previous studies have focused on deploying traditional ML models or offloading isolated components of the LLM pipeline to SmartNICs. To the best of our knowledge, no existing work has investigated executing the reasoning core of an agentic AI system – particularly an SLM-based decision module – directly on SmartNIC hardware.

III. Agentic AI Reasoning Evaluation

We now turn our attention to evaluating a network agentic AI on a SmartNIC. Our evaluation focuses exclusively on the performance of SLM-based reasoning on different deployment targets: CPUs, GPUs, and SmartNICs. We compare a fully offloaded scenario (Figure 1 – ❷) to traditional CPU/GPU-based deployments. The goal is to assess how the execution platform may affect the efficiency of the reasoning process, without considering perception and downstream actuation or network control effects. The end-to-end performance evaluation and other deployment strategies are left for future work.

A. Agentic AI Reasoning

We consider agentic AI reasoning to be a set of open-source SLM models[1]. These SLM models are chosen with varying parameter sizes to capture diverse performance trade-offs on evaluated deployments: Gemma 2B $_{Q4\_0}$, Gemma 7B $_{Q4\_0}$, Gemma 2 9B $_{Q4\_0}$, Llama 3 8B $_{Q4\_0}$, Llama 3.2 1B $_{Q8\_0}$, Llama 3.2 3B $_{Q4\_K\_M}$, Qwen 2.5 0.5B $_{Q4\_K\_M}$, Qwen 2.5 1.5B $_{Q4\_K\_M}$, and Qwen 2.5 7B $_{Q4\_K\_M}$. Then, a common set of reasoning tasks was defined to ensure comparability across platforms, consisting of prompt analysis, inference, and generation of recommended actions for a fixed set of network scenarios. All models were executed using a consistent inference configuration, quantization level, and prompt set across platforms, allowing for a fair comparison of reasoning efficiency under identical workloads.

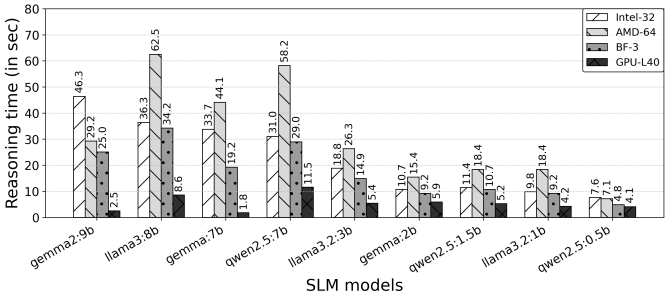

Figure 2: Reasoning time of a set of SLM models.

B. Testbed Setup & Baseline

Our testbed comprises servers equipped with Intel Xeon Silver 4216 processors (32 cores) and AMD Threadripper 3990X (64 cores) processors. The servers feature NVIDIA BlueField-2 and BlueField-3 SmartNICs, enabling programmable data-plane experimentation with different SoC generations. In addition, NVIDIA L40 GPUs (48 GB) and NVIDIA RTX 3090 GPUs (24 GB) are used to accelerate AI workloads and comparative evaluations across heterogeneous hardware.

We serve the SLMs using Ollama, a lightweight framework for deploying and running LLMs and SLMs on local servers. It loads quantized models to reduce memory and latency, manages token generation and resource allocation, while providing a hardware-agnostic runtime that minimizes implementation bias. Ollama exposes HTTP endpoints for seamless integration with external applications, enabling flexible and modular interaction with deployed models.

In our experiments, each model is first offloaded to the ARM cores of the NVIDIA BlueField-3 SmartNIC. Then, we trigger the same requests[2] for each SLM and process the corresponding model outputs. We compare its performance against two baselines: execution on the server CPU and execution on the server GPU. The average model response length across all SLMs is approximately 180 tokens. Statistical validation using a t-test shows that 30 repetitions per experiment provide at least 95% confidence.

C. Results

Reasoning time. We evaluate the reasoning time of different SLMs across multiple deployments. We define the reasoning

[1]The SLM models are available at https://huggingface.co/

[2]The prompts, scripts, and results are available at: https://github.com/lopesvictor1/SmartNIC-SLM-Serving

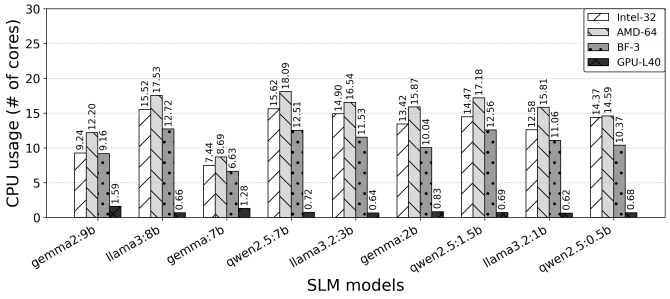

Figure 3: CPU Utilization.

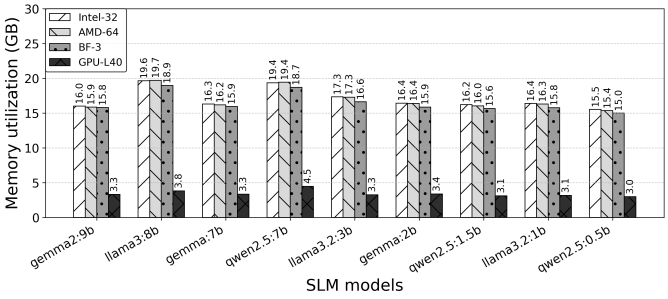

Figure 4: Memory footprint.

time as the total elapsed time from the moment the prompt is issued until the complete response is received. This metric measures agentic AI responsiveness, indicating how quickly it can interpret context and act autonomously in dynamic environments. Figure 2 shows the reasoning time for each SLM model and deployment.

Small SLM models can benefit more from the BlueField-3 architecture than from an AI-optimized data-center GPU. The difference in reasoning time between executing an SLM fully offloaded to the BlueField-3 and running it on the L40 GPU remains within the same order of magnitude for small models. For the `qwen2.5:0.5b` model, this gap is as low as 0.6 s (≈14%), and 3.3 s for `llama3.2:1b`. Such differences do not reflect their energy consumption, since the NVIDIA L40 consumes over twice as much power as the BlueField-3 [16], [24]. Another relevant observation is that the *BlueField-3 consistently outperforms CPU-based SLM reasoning*. In our experiments, CPU usage was limited to 16 cores to ensure a fair comparison. This performance advantage is largely attributed to the high-speed PCIe interface of the BlueField-3, which interconnects DDR5 memory with the ARM cores, although further investigation is required to fully characterize these effects. For larger SLMs (e.g., `gemma2:9b`), GPU-based execution remains faster, achieving up to a 9x speedup in the worst-case scenario.

Resource footprint. Figures 3 and 4 show the host CPU and memory utilization during SLM inference across deployments. *Server-side SLM execution demands a substantial amount of compute resources*. In our experiments, SLMs consumed more than 15 CPU cores exclusively for reasoning, and up to 17 GB of RAM. It is also worth noting that even when fully offloaded to a GPU, a non-negligible and fixed CPU and memory overhead remains to serve the model. In contrast, the BF-3 can execute SLMs using its own CPU and memory resources, thereby freeing server capacity for other applications. On Intel-32, SLMs used nearly half of the available cores, while on the AMD system, the usage reached approximately one quarter of the total cores. This resource efficiency is critical for scaling network agentic AI deployments, as reducing the host overhead enables multiple agents to operate concurrently and makes large-scale, distributed agentic reasoning more feasible across the network fabric.

SLM token generation. This metric reflects how quickly

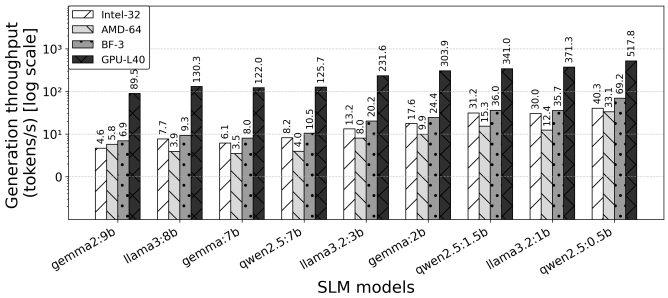

Figure 5: Tokens per second.

a model outputs tokens, indicating its reasoning and inference efficiency. GPU-L40 achieves the highest throughput, exceeding 500 tokens/s for 0.5b models and staying above 120 tokens/s for 7b models, with minimal – yet noticeable – host CPU and memory usage, demonstrating the effectiveness of GPU-optimized pipelines. CPU-only platforms (AMD-64 and Intel-32) scale poorly, dropping below 10 tokens/s for 7–9b models and approaching full core utilization. BF-3 consistently outperforms CPU-only platforms by 2-5×, reducing host CPU load via offloading. However, for small models (0.5-3b), BF-3's token throughput (18-73 tokens/s) remains 5-20× lower than GPU-L40. Despite this gap, SmartNIC deployment becomes viable for edge environments where eliminating host CPU/memory overhead and data transfer costs matters more than peak throughput, particularly when local, autonomous reasoning must coexist with other critical workloads.

IV. Final Remarks, Limitations and Future Work

This work provides a preliminary assessment of SmartNICs as a platform for executing SLM-based reasoning in network agentic AI. Our results establish a performance baseline for SmartNIC-based SLM agentic AI reasoning and demonstrate the feasibility of embedding autonomous intelligence directly within network infrastructure. The study, however, focused only on the reasoning stage, did not consider concurrent packet-processing workloads, and evaluated a single SmartNIC architecture, limiting generality. As future work, we plan to extend the evaluation to full agentic AI pipelines, assess co-execution with network tasks on the SmartNIC, and explore hybrid partitioning of agentic AI components across CPU, GPU, and SmartNIC resources.

REFERENCES

[1] Y. Xiao, G. Shi, and P. Zhang, "Toward agentic ai networking in 6g: A generative foundation model-as-agent approach," *IEEE Communications Magazine*, vol. 63, no. 9, pp. 68–74, 2025.

[2] R. Sapkota, K. I. Roumeliotis, and M. Karkee, "Ai agents vs. agentic ai: A conceptual taxonomy, applications and challenges," *Information Fusion*, vol. 126, p. 103599, 2026. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253525006712

[3] F. Jiang, C. Pan, L. Dong, K. Wang, O. A. Dobre, and M. Debbah, "From large ai models to agentic ai: A tutorial on future intelligent communications," 2025. [Online]. Available: https://arxiv.org/abs/2505.22311

[4] R. Boutaba, M. A. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano, and O. M. Caicedo, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," *Journal of Internet Services and Applications*, vol. 9, no. 1, p. 16, 2018. [Online]. Available: https://doi.org/10.1186/s13174-018-0087-2

[5] J. Xie, F. R. Yu, T. Huang, R. Xie, J. Liu, C. Wang, and Y. Liu, "A survey of machine learning techniques applied to software defined networking (sdn): Research issues and challenges," *IEEE Communications Surveys Tutorials*, vol. 21, no. 1, pp. 393–430, 2019.

[6] M. C. Luizelli, F. G. Vogt, P. S. S. De Souza, A. F. Lorenzon, R. I. T. Da Costa Filho, F. D. Rossi, R. N. Calheiros, and C. E. Rothenberg, "Diginet: Scaling up provisioning of network digital twin," in *2024 IEEE 10th International Conference on Network Softwarization (NetSoft)*, 2024, pp. 136–144.

[7] A. Plaat, A. Wong, S. Verberne, J. Broekens, N. van Stein, and T. Back, "Reasoning with large language models, a survey," *arXiv preprint arXiv:2407.11511*, 2024.

[8] P. Belcak, G. Heinrich, S. Diao, Y. Fu, X. Dong, S. Muralidharan, Y. C. Lin, and P. Molchanov, "Small language models are the future of agentic ai," 2025. [Online]. Available: https://arxiv.org/abs/2506.02153

[9] K. Qian, Y. Xi, J. Cao, J. Gao, Y. Xu, Y. Guan, B. Fu, X. Shi, F. Zhu, R. Miao, C. Wang, P. Wang, P. Zhang, X. Zeng, E. Ruan, Z. Yao, E. Zhai, and D. Cai, "Alibaba hpn: A data center network for large language model training," in *Proceedings of the ACM SIGCOMM 2024 Conference*, ser. ACM SIGCOMM '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 691–706. [Online]. Available: https://doi.org/10.1145/3651890.3672265

[10] R. Parizotto, B. L. Coelho, D. C. Nunes, I. Haque, and A. Schaeffer-Filho, "Offloading machine learning to programmable data planes: A systematic survey," *ACM Comput. Surv.*, vol. 56, no. 1, Aug. 2023. [Online]. Available: https://doi.org/10.1145/3605153

[11] M. Saquetti, R. Canofre, A. F. Lorenzon, F. D. Rossi, J. R. Azambuja, W. Cordeiro, and M. C. Luizelli, "Toward in-network intelligence: Running distributed artificial neural networks in the data plane," *IEEE Communications Letters*, vol. 25, no. 11, pp. 3551–3555, 2021.

[12] I. P. Lamb, P. A. P. R. Duarte, J. Marques, M. C. Luizelli, L. P. Gaspary, A. Tavares, R. A. Ferreira, Cunha, J. R. F. Azambuja, and W. Cordeiro, "Distributed graph neural networks in programmable data planes," in *NOMS 2025-2025 IEEE Network Operations and Management Symposium*, 2025, pp. 01–09.

[13] L. Cannarozzo, T. B. Morais, P. S. S. de Souza, L. R. Gobatto, I. P. Lamb, P. A. P. R. Duarte, J. R. F. Azambuja, A. F. Lorenzon, F. D. Rossi, W. Cordeiro, and M. C. Luizelli, "Spinner: Enabling in-network flow clustering entirely in a programmable data plane," in *NOMS 2024-2024 IEEE Network Operations and Management Symposium*, 2024, pp. 1–9.

[14] B. Bütün, D. De Andres Hernandez, M. Gucciardo, and M. Fiore, "Dune: Distributed inference in the user plane," in *IEEE INFOCOM 2025 - IEEE Conference on Computer Communications*, 2025, pp. 1–10.

[15] X. Zhan, A. Goyal, Y. Chen, E. Chandrasekharan, and K. Saha, "Slm-mod: Small language models surpass llms at content moderation," *arXiv preprint arXiv:2410.13155*, 2024.

[16] NVIDIA, "Nvidia bluefield-3 dpu data sheet," https://www.nvidia.com/en-us/networking/products/data-processing-unit/bluefield-3/, 2022, [Accessed: 2025-11-06].

[17] ——, "Nvidia doca," https://docs.nvidia.com/doca/sdk/index.html, 2025, [Accessed: 2025-11-06].

[18] S. Galvankar, S. Mehta, and S. Choi, "Nictokenizer: Tokenizer offload to a smartnic," in *Proceedings of the ACM SIGCOMM 2025 Posters and Demos*, 2025, pp. 79–81.

[19] A. Rebai and M. Canini, "Optimusnic: Offloading optimizer state to smartnics for efficient large-scale ai training," in *Proceedings of the 5th Workshop on Machine Learning and Systems*, 2025, pp. 176–182.

[20] R. Zhang, H. Du, Y. Liu, D. Niyato, J. Kang, Z. Xiong, A. Jamalipour, and D. I. Kim, "Generative ai agents with large language model for satellite networks via a mixture of experts transmission," *IEEE Journal on Selected Areas in Communications*, 2024.

[21] Y. Xiao, G. Shi, and P. Zhang, "Towards agentic ai networking in 6g: A generative foundation model-as-agent approach," *arXiv preprint arXiv:2503.15764*, 2025.

[22] Y. Xiao, H. Zhou, X. Li, Y. Gao, G. Shi, and P. Zhang, "Sannet: A semantic-aware agentic ai networking framework for multi-agent cross-layer coordination," *arXiv preprint arXiv:2505.18946*, 2025.

[23] R. Zhang, S. Tang, Y. Liu, D. Niyato, Z. Xiong, S. Sun, S. Mao, and Z. Han, "Toward agentic ai: generative information retrieval inspired intelligent communications and networking," *arXiv preprint arXiv:2502.16866*, 2025.

[24] NVIDIA, "Nvidia l40 gpu product brief," https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/datasheets/L-40/product-brief-L40.pdf, 2025, [Accessed: 2025-11-10].

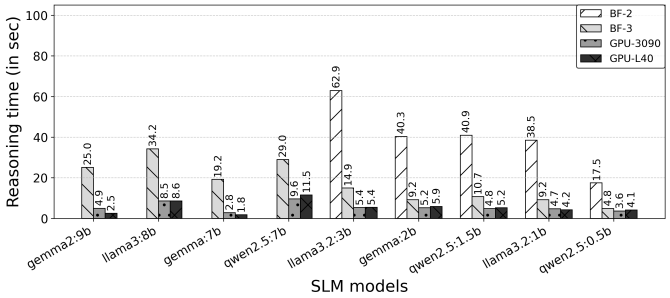

Figure 6: Reasoning time of a set of SLM models.

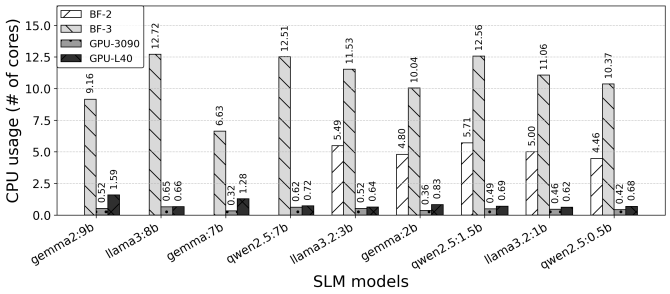

Figure 7: CPU Utilization.

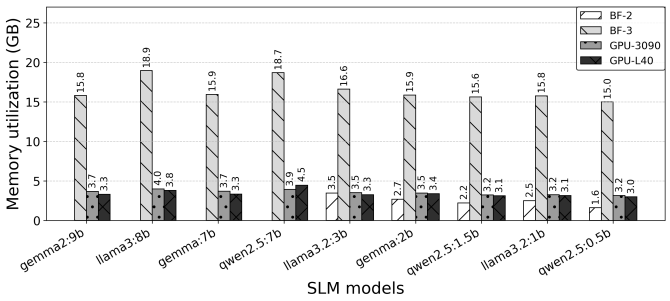

Figure 8: Memory footprint.

APPENDIX A
ADDITIONAL RESULTS

This appendix presents additional performance results obtained from different hardware platforms: NVIDIA BlueField-2 (BF-2) and NVIDIA GeForce RTX 3090 (GPU-3090). The main results from the BlueField-3 (BF-3) and NVIDIA L40 (GPU-L40) were used for reference. These results complement the main evaluation in Section III by including earlier-generation SmartNICs and high-end consumer GPU for a broader comparison. While BF-3 and GPU-based systems could execute all SLMs in the benchmark, the limited memory capacity of BlueField-2 prevented running models larger than 3 billion parameters. Consequently, BF-2 results are reported only for the smaller SLM configurations.

A. Reasoning Time

Figure 6 compares total reasoning time across BlueField and GPU generations. The BlueField-3 SmartNIC achieves a substantial leap in inference performance over BlueField-2, reducing total reasoning time by roughly an order of magnitude across most models. For example, while BF-2 required 40–80 s to complete 1–3 B parameter models, BF-3 executes the same workloads in under 10–20 s. When compared to GPUs, BF-3 now operates in the same order of magnitude as the RTX 3090, with small and mid-sized models completing only 1.5–2× slower, a major improvement from BF-2's 10× gap.

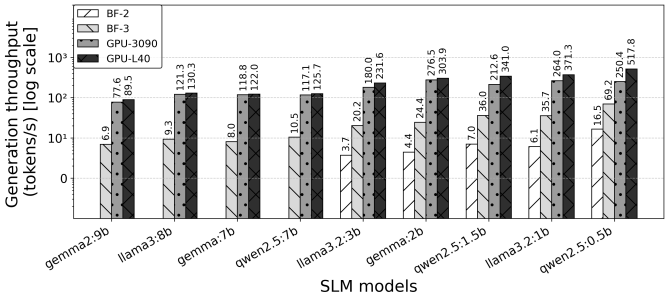

Figure 9: Tokens per second.

DPU-based inference and consumer GPUs, while datacenter-class GPUs like the L40 still maintain clear dominance in raw throughput.

B. Resource Footprint

Figure 7 compares CPU utilization during inference. On the BF-2, all eight embedded Arm cores reach near-full saturation, reflecting the limited compute available for inference. The RTX 3090 shows host CPU usage between 18–63%, while the L40 reaches 62–159%, likely due to datacenter-grade drivers and more aggressive scheduling. Overall, while DPU inference fully occupies on-board cores, it leaves the host CPU largely free for concurrent network or application tasks.

Figure 8 shows memory consumption during inference. The BF-2 leverages its onboard DDR4 system memory (typically 16 GB), ranging from 850 MB to 3 GB of active use depending on model size, while the RTX 3090 consumes host-side GPU memory (ranging from 3.6 GB to 8.2 GB). This illustrates how DPU-based inference isolates memory utilization from the host system.

C. Generation Throughput

As shown in Figure 9, the RTX 3090 achieves generation throughputs exceeding 250 tokens/s for small SLMs, demonstrating the advantage of dedicated GPU compute pipelines. The BF-2, by contrast, sustains throughputs below 15 tokens/s even for lightweight models, limited by both memory fre-