1.0 Introduction

Traditional large language models (LLMs) offer powerful capabilities for text generation, but they are fundamentally limited to single-turn interactions, lacking the capacity for structured planning, autonomous action, and adaptive memory. With Agentic AI, systems can now not only generate outputs but also plan, act, adapt, and self-correct within dynamic environments. This works well for enterprises as they now seek solutions that can execute multi-step workflows, integrate with business tools, and operate under strict safety protocols.

NVIDIA Nemo is a comprehensive framework that extends the NeMo ecosystem beyond training and fine-tuning models to building full autonomous AI agents. Agentic NeMo offers the architecture, modularity, and scalability needed to transition LLMs into operational agents capable of real-world impact.

In this article, we review NVIDIA Nemo's agentic AI based capabilities.

2.0 Understanding Agentic AI

2.1 Definition

Agentic AI refers to systems designed with the capability to autonomously perceive goals, generate plans, execute multi-step actions, adapt based on environmental feedback, and reason over time.

Unlike traditional LLM deployments, where models respond statically to prompts, agentic systems engage in continuous cycles of:

- **Goal Identification:** Understanding the user's intent and defining the objectives of the interaction.
- **Planning and Reasoning:** Formulating a structured, step-by-step plan to achieve the identified goals, often involving complex reasoning and decision-making.
- **Action Execution:** Executing the planned steps by interacting with external systems via APIs, databases, and other tools, effectively carrying out real-world actions.
- **Memory Update:** Storing and updating information about the interaction, including past steps, intermediate results, and feedback, to maintain context and improve future actions.
- **Feedback Reflection:** Analyzing the outcomes of actions, gathering feedback from the environment or user, and using this information to refine future plans and actions.

2.2 Importance of Agentic AI

The growing importance and shift towards Agentic AI are driven by several critical enterprise needs that traditional LLMs cannot adequately address:

- **Enterprises Demand Autonomy:** Manual prompt engineering, while effective for simple tasks, is unsustainable and inefficient for complex workflows that require proactive decision-making, autonomous action execution, and continuous adaptation. Enterprises need AI systems that can operate independently, making decisions and taking actions without constant human intervention.
- **Real-World Environments Are Dynamic:** Static outputs from traditional LLMs are insufficient in dynamic environments where data, business conditions, or operational requirements are constantly changing. Agentic AI can adapt in real-time to these changes, ensuring that decisions and actions are always relevant and up-to-date.
- **Safety and Compliance Are Non-Negotiable:** In enterprise settings, autonomous agents must operate under strict guardrails to ensure security, regulatory compliance, ethical behavior, and data privacy. Traditional LLMs, which primarily focus on generating text, lack the built-in mechanisms to enforce these critical safeguards. Agentic AI systems can incorporate robust policy enforcement and validation at every step of their operation.

Agentic AI unlocks a new class of enterprise applications, such as automated customer service systems, real-time supply chain management, and intelligent financial analysis tools. These AI systems

not only answer questions but execute end-to-end business processes, interface with live systems, and learn from ongoing interaction (e.g., by adapting to user feedback, optimizing workflows, and refining decision-making models).

3.0 NVIDIA NeMO

3.1 Evolution of NeMo Towards Agentic AI

The NVIDIA NeMo framework initially focused on providing scalable training solutions for speech and natural language processing (NLP) models around 2019–2020. Since then, it has undergone a rapid and significant evolution, expanding its capabilities to address the needs of advanced AI systems. Here is a timeline of its key milestones:

- **2021 — Megatron-LM Integration:** Marked a major expansion, enabling support for training massive Large Language Models (LLMs) across distributed multi-node GPU clusters, significantly increasing model size and complexity.
- **2022 — Fine-Tuning and Efficient Adaptation:** Introduced crucial fine-tuning techniques like LoRA, QLoRA, and Prefix Tuning, allowing for efficient adaptation of pre-trained models to specific tasks and domains without full retraining.
- **2023 — Safety and Retrieval-Augmentation:** Expanded its ecosystem with the launch of NeMo Guardrails for

ensuring safety and compliance, and Retrieval-Augmented Generation (RAG) pipelines to enhance knowledge retrieval and accuracy.

- **2024 — Agentic Enablement**: Represented a pivotal shift, providing full orchestration of planning, memory, tool-use, and safety modules, transforming NeMo into a platform for building truly autonomous AI agents.

3.2 Core Building Blocks of Agentic NeMo

At the heart of Agentic NeMo is a tightly integrated system of specialized components.

- The foundation begins with **NeMo Foundation Models**, large Megatron-LM-based LLMs trained across massive GPU clusters. These models are fine-tuned using techniques such as LoRA, QLoRA, and Prefix-Tuning to adapt quickly to specific domains without requiring complete retraining. They provide the reasoning, planning, and language generation capabilities central to the agent's intelligence.
- Agentic NeMo integrates **Retrieval-Augmented Generation (RAG) Pipelines** that allow agents to retrieve up-to-date, task-relevant information from external vector databases like FAISS, RedisVector, or ElasticSearch. Instead of relying solely on static pretrained knowledge, NeMo agents

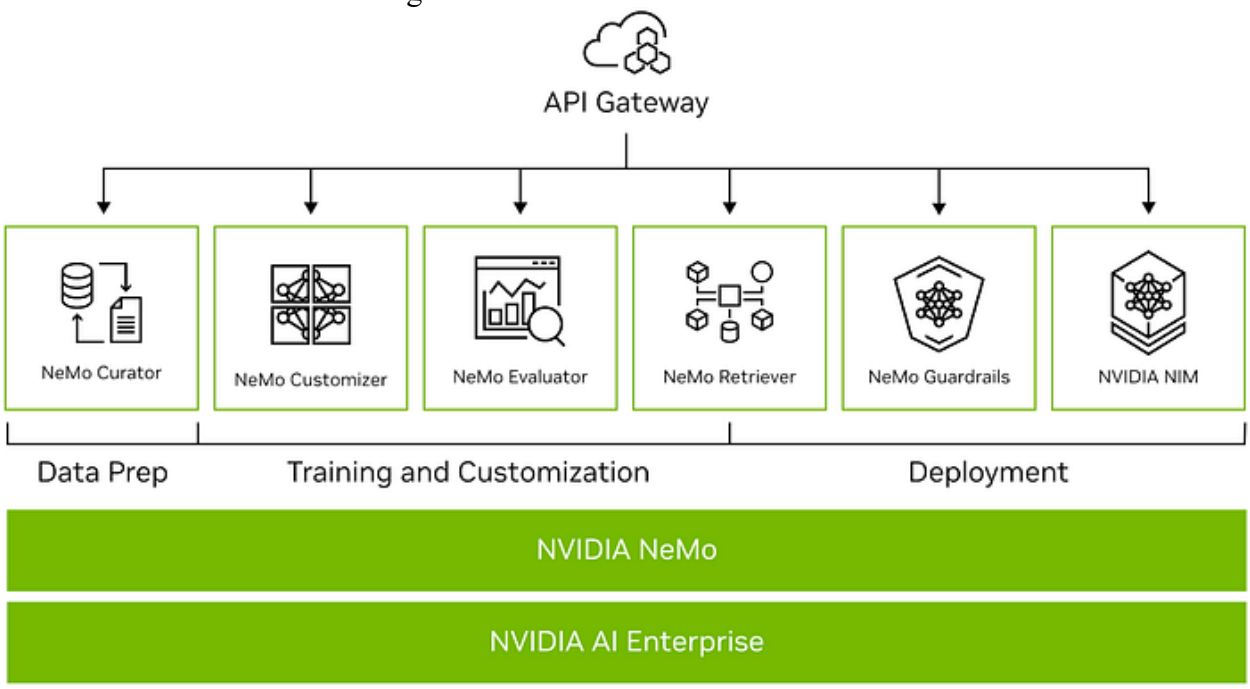
can dynamically pull documents, facts, or structured data in real time.

- With **Tool-Use and API Action Layer**, agentic NeMo agents can dynamically call external APIs, query databases, and trigger workflows using structured tool schemas (often defined using OpenAPI specifications). This layer transforms an agent from a passive responder into an active participant capable of executing complex, multi-step tasks across enterprise systems.
- Through **Memory and Context Management**, NeMo agents maintain evolving memory, tracking past interactions, intermediate results, and task progression across multi-turn conversations. Using short-term session memory or long-term user memory, these systems enable agents to adapt dynamically and reason over time, rather than treating each interaction statically.
- **NeMo Guardrails** enforce enterprise-grade policy and trust boundaries on both conversations and actions. They can validate outputs against safety standards, prevent unsafe or unauthorized tool use, and ensure that agents operate within pre-defined ethical and compliance frameworks. Guardrails work through both natural language policies and structured validation layers.

- Agentic NeMo leverages an **Inference Serving and Orchestration Layer** built around the NVIDIA Triton Inference Server and TensorRT-LLM. This infrastructure ensures low-latency, high-throughput serving of large LLMs across multiple GPUs and nodes, while integrating seamlessly into Kubernetes or cloud-native architectures.

4.0 Technical Architecture of an Agentic NeMo System

Press enter or click to view image in full size



4.1 Layered Architecture

Building a robust Agentic NeMo system involves the seamless orchestration of multiple specialized layers that work in concert to transform raw data into intelligent actions.

- **User Interface:** The User Interface serves as the primary entry point for end-users to interact with the agent. Whether through a chatbot, web portal, mobile application, or voice assistant, the interface is responsible for securely capturing user inputs and presenting agent outputs in a clear, accessible format.
- **API Gateway:** Upon receiving a user query, the API Gateway acts as a critical intermediary. It manages essential infrastructure tasks, including authentication, authorization, request routing, load balancing, and traffic management.
- **Planning and Reasoning Layer:** At the heart of the system lies the Planning and Reasoning Layer. Here, NeMo foundation models analyze user inputs to generate structured, multi-step plans instead of simple single-turn outputs. Leveraging techniques like chain-of-thought prompting, task decomposition, and advanced reasoning, this layer enables the agent to dynamically build actionable workflows and truly "think" through complex tasks.
- **Memory and Context Engine:** For agents to operate adaptively over time, persistent memory is essential. The Memory and Context Engine stores session information, past interactions, task progress, and intermediate outputs. It enables the agent to reference previous steps, maintain long-term objectives, and adjust plans dynamically as new information becomes available.

- **RAG Retrieval Layer:** The Retrieval-Augmented Generation (RAG) layer connects the system to external data sources, including vector databases like FAISS, RedisVector, and ElasticSearch. This layer fetches real-time, relevant knowledge that enriches the agent's reasoning.

- **Tool-Execution Engine:** Planning and retrieval alone are not enough; agents must be able to act. The Tool-Execution Engine empowers the agent to call external APIs, interact with databases, trigger business workflows, or generate documents. Agents dynamically select which tools to use based on the evolving context.

- **Guardrails Policy Enforcement:** Throughout the agent's workflow, safety and compliance are enforced by the Guardrails Policy Enforcement layer. Guardrails validate conversational outputs, monitor tool-use actions, and ensure topic constraints. They are applied dynamically at multiple checkpoints to guarantee that every action, retrieval, and output aligns with enterprise security, ethics, and operational policies.

- **Inference Serving Layer:** Finally, the Inference Serving Layer underpins the system's performance. Powered by NVIDIA Triton Inference Server and TensorRT-LLM, this layer delivers low-latency, high-throughput serving of large language models across multi-GPU deployments. It enables

real-time agentic behavior at scale and integrates seamlessly into Kubernetes, cloud-native, or hybrid infrastructures.

4.2 Performance Considerations

Building and operating an agentic AI system at scale introduces several unique performance challenges.

Latency Challenges in Agentic Workflows

Building agentic systems introduces multiple latency points that can significantly degrade user experience if not properly managed. Unlike single-turn LLM applications, agentic workflows involve multiple steps, or "hops," including planning, retrieval, tool execution, and Guardrails validation. Each additional step can contribute to measurable delays, compounding the overall latency. Primary sources of this latency include the LLM inference time, particularly for large models, retrieval delays from vector databases during Retrieval-Augmented Generation (RAG) operations, external API and tool invocation delays, and the validation overhead introduced by Guardrails.

Minimizing end-to-end latency is crucial for maintaining a fluid and responsive user experience. Addressing these various sources of delay is essential to ensure that agentic systems perform effectively in real-

time applications. By optimizing each stage of the agentic workflow, from inference to validation, developers can mitigate latency issues and provide a seamless interaction for users.

Optimization Techniques for Real-Time Performance

To address the inherent latency challenges in agentic workflows, NVIDIA's Agentic NeMo incorporates several optimization strategies aimed at achieving real-time performance.

- TensorRT-LLM Inference Acceleration plays a pivotal role by quantizing models to lower precision formats like FP8 or INT8 and applying graph optimizations, significantly reducing the LLM inference latency.
- DeepSpeed Integration further enhances efficiency by optimizing multi-GPU and multi-node communication, minimizing training and fine-tuning overheads through techniques like ZeRO-3.
- Additionally, Retrieval Caching is implemented to minimize RAG lookup times by storing frequently accessed documents and embeddings in caching layers.
- Parallel Execution is leveraged to asynchronously execute external tool calls and RAG retrievals, avoiding cumulative blocking delays.

- Lastly, Memory Sharding distributes agent memory across clusters, ensuring faster lookup and updating operations for large-scale deployments.

Scaling Up for Enterprise Workloads

For production deployments of Agentic NeMo, which often anticipate large volumes of concurrent users, horizontal scaling across multiple compute nodes becomes essential. This approach ensures that the system can handle increased load without compromising performance. To achieve this, several key strategies are employed.

- Inference Clustering involves deploying the NVIDIA Triton Inference Server across numerous GPUs and nodes, effectively distributing the inference workload and preventing bottlenecks.
- Load Balancing is implemented using intelligent routing mechanisms such as Envoy or Nginx, which evenly distribute incoming agent requests across the backend servers.
- Elastic Vector Databases are utilized to guarantee that vector stores and knowledge retrieval systems can scale horizontally, preventing them from becoming performance bottlenecks as the data and query volume increases.
- Finally, Real-Time Observability is crucial for maintaining the health and efficiency of the scaled system. Implementing

comprehensive monitoring tools allows for the tracking of critical metrics like latency, error rates, memory utilization, and throughput. This real-time data enables proactive scaling adjustments and rapid troubleshooting when issues arise.

5.0 Real-World Use Cases

Below are some common use-cases that can be

- **Healthcare AI Assistant:** An autonomous agent can be deployed to interface with live clinical trial databases, retrieve the latest oncology research papers, and summarize key findings into compliance-ready reports for healthcare providers. By leveraging RAG pipelines and Guardrails, the agent ensures that all content is accurate, up-to-date, and aligned with medical regulatory standards.
- **Financial Analysis Agent:** In financial services, an agentic system can monitor real-time financial APIs for stock market movements, retrieve corporate filings (such as SEC 10-Ks), analyze risk factors, and automatically generate financial risk assessments. These agents can dynamically adapt based on market volatility and deliver insights to analysts with minimal manual intervention.
- **IT Support Agent:** Agentic NeMo can power autonomous IT support agents that interact with internal ticketing systems like Jira or ServiceNow. These agents can classify support

tickets, recommend and execute potential resolutions, run diagnostic scripts, and escalate critical incidents when necessary. Integrated Guardrails ensure that only safe, authorized actions are executed in sensitive IT environments.

- **Multimodal Manufacturing Assistant:** In industrial settings, a multimodal agent can combine text and image processing to analyze equipment maintenance manuals and schematics. The agent can retrieve specifications, generate detailed repair workflows, and assist technicians in real-time by fusing visual and textual information. This boosts operational efficiency and minimizes downtime on production lines.

6.0 Conclusion

As the demand for autonomous, intelligent systems continues to rise, traditional LLM architectures are no longer sufficient to meet enterprise-grade requirements. Organizations need AI that can plan, act, adapt, and do so within rigorous safety and compliance boundaries. NVIDIA's Agentic NeMo stands at the forefront of this evolution. By tightly integrating powerful foundation models, dynamic knowledge retrieval (RAG), actionable tool-use APIs, persistent memory management, and robust policy enforcement via Guardrails, NeMo provides a comprehensive, production-ready stack for building autonomous agents. The modular design ensures that systems can be tailored to specific domains, scaled horizontally to support millions of

users, and optimized for real-time responsiveness through NVIDIA's high-performance compute ecosystem.