Agentic AI: From LLMs to Autonomous Agents

Slide 1: Title

- **Agentic AI: The Next Frontier in AI**
- Presenter: *[Your Name]*, [Date]
- Audience: Software Engineers, AI Practitioners

Speaker Notes: Introduce yourself and the topic. Explain that today we explore "agentic AI," a new paradigm extending large language models. Highlight that this will connect LLM fundamentals to advanced autonomous systems. Outline that we'll cover LLM basics, their limitations, the agentic AI concept, and fintech examples.

Slide 2: Agenda

- **Overview:** Large Language Models (LLMs) and their evolution
- **LLM Limitations:** Why treat LLMs as tools vs agents
- **Agentic AI:** Definition and key features (autonomy, planning, memory, tool use)
- **Architecture:** Conceptual framework of an AI agent
- **Fintech Examples:** 1) Autonomous customer support, 2) Fraud detection pipeline
- **Implications:** What engineers need to consider
- **Conclusion & Outlook**

Speaker Notes: Walk through each agenda item. Emphasize flow: start with LLM basics, then transition to "what they can't do," introducing agentic AI as the answer. Preview that we'll end with concrete fintech scenarios and practical takeaways.

Slide 3: What Are LLMs?

- LLMs are **deep neural networks trained on massive text corpora** to predict the next word/token in a sequencesloanreview.mit.edu.
- They **leverage Transformer architecture**, using self-attention to capture relationships across an entire input sequencesloanreview.mit.edu.
- The model's **size (parameters)** and **training data** greatly affect capabilitysloanreview.mit.edu.
- Examples: GPT-3, GPT-4, BERT – used for translation, summarization, code generation, etc.

Speaker Notes: Explain that LLMs (Large Language Models) learn statistical patterns from large text datasets. The "next-token prediction" task is core – e.g., the model estimates the most likely next word. The Transformer architecture (Vaswani et al. 2017) uses attention mechanisms so that each output word is informed by all input positionssloanreview.mit.edu. Highlight that larger models trained on more data tend to perform better (performance scales with parameter count and data volumesloanreview.mit.edu). Mention notable models (GPT-3: 175B parameters in 2020, GPT-4 even larger/context >8K tokens). Stress LLM strength: fluent, contextually coherent text generation.

Slide 4: How LLMs Work & Evolve

- **Transformer encoder-decoder:** processes all input tokens in parallel, building contextual embeddingssloanreview.mit.edu.
- **Scale:** More parameters and data = better linguistic and factual accuracysloanreview.mit.edu.
- **Context window:** Early LLMs saw ~2K tokens (words); modern ones handle tens of thousands, giving short-term "memory"sloanreview.mit.edu.
- **Training:** Pretraining on web data followed by fine-tuning (and often RLHF) for task alignment.
- **Output:** LLMs are excellent at generation (text, code) and pattern recognition, but only as specified by their training data.

Speaker Notes: Dive slightly into the mechanics: Transformers use attention to relate all words, unlike older RNNs that did step-by-step processingsloanreview.mit.edu. This gives them a strong grasp of context. Point out the "scaling laws": e.g., GPT-3's breakthrough was 175 billion parameters. Newer models like GPT-4 further increase context length (some versions ~32K tokens). Emphasize that training has two stages: unsupervised pretraining on broad text and supervised/RL fine-tuning for safety/goals. Summarize that current LLMs can produce high-quality text but within the bounds of their training distribution.

Slide 5: Limitations of LLMs (as Tools)

- **Reactive, not proactive:** LLMs only respond to explicit prompts; they **cannot autonomously decide to start a task or pursue a goal**ibm.com.
- **Stateless (short-term "memory"):** No persistent state beyond the current context windowsloanreview.mit.edu. They don't remember previous interactions across sessions.
- **No inherent planning:** Generate text token-by-token without an explicit strategy or multi-step plan. They excel at single-step generation, but struggle with multi-step reasoning unless guided.
- **Hallucinations:** May produce confident-sounding but incorrect or nonsensical answers without factual grounding.
- **Limited World Understanding:** Their knowledge is fixed at training time (knowledge cutoff) and they lack real-time access to the external world unless connected to tools.

Speaker Notes: Explain that today's LLMs are very powerful generators but have critical gaps. They *wait* for user input (they won't start tasks on their own)ibm.com. They also lack long-term memory or state – each conversation is only as good as the context window sizesloanreview.mit.edu. Because of their design, they don't plan in a structured way; they guess the next word, not map out a sequence of actions. This leads to issues like hallucination – invention of details if the prompt doesn't constrain them, which is why we see occasional "bad answers." Stress that these limitations mean LLMs are great assistants (e.g., ChatGPT for coding, Q&A) but are not yet "autonomous agents."

Slide 6: What is Agentic AI?

- **Definition:** AI systems with *agency* – they can **autonomously make decisions and perform tasks toward goals** without continuous human oversightibm.com.
- **Goal-oriented:** Agentic AIs are **focused on achieving objectives** through multi-step reasoning, rather than only generating content.
- **Blends paradigms:** They combine LLM capabilities (language understanding/generation) with structured planning, statefulness, and control

logicibm.com.

- **Contrast:** Unlike a ChatGPT-style model (just Q&A), an agentic AI can **take initiative**, call APIs, and adapt to new situations.

- **Early examples:** Virtual assistants that schedule meetings, task-specific agents that monitor and act (e.g., automated devops scripts), and, in future, fully autonomous robots or software agents.

Speaker Notes: Introduce agentic AI as the "next step" – systems that do more than respond. Cite IBM: "Agentic AI describes AI systems that autonomously make decisions and act, pursuing complex goals with limited supervision"ibm.com. Emphasize *goal-directed behavior*: an agent is driven to achieve an objective (e.g., "maximize ad revenue", "invest client funds prudently"), and it sequences actions towards that. These agents still use LLMs and ML (for understanding tasks), but also incorporate components like planners and memory. Give a simple example: a virtual assistant that not only answers a flight booking question but actually checks prices, books tickets, and emails confirmations on its own initiative.

Slide 7: Key Features of Agentic AI

- **Autonomy:** Operates with minimal human prompting; can initiate and adapt actions independentlyibm.com.

- **Planning:** Decomposes goals into multi-step plans (e.g., sub-tasks) and reasons over future stepsblogs.nvidia.com.

- **Persistent Memory:** Maintains context and knowledge over time (short-term and long-term memory modules)cloud.google.com. This enables stateful behavior (e.g., remember past user preferences).

- **Tool/Environment Interaction:** Can invoke external tools, APIs, and databases to carry out tasksblogs.nvidia.com. For example, execute code, query live data, or send messages.

- **Learning & Adaptation:** Continuously improves via feedback or reinforcement learning; updates strategies based on new datafinextra.com. Agents evolve their behavior as they gather more experience.

Speaker Notes: Elaborate each feature:

- *Autonomy:* The system can choose what to do next. For instance, it might decide to check stock prices without being asked each time, based on a long-term objective.
- *Planning:* Unlike a bare LLM, an agent explicitly considers future steps. For example, it might plan to break “plan a trip” into “find flights, book hotel, arrange car rental” in sequence.
- *Memory:* Explain memory hierarchies. Cite Google Cloud: agents may have short-term memory (current conversation) and long-term memory (database of past interactions)cloud.google.com. This prevents repetitive questions and allows learning user preferences.
- *Tool Use:* Agents are connected. Example: If an agent can call an email API or a stock API, it can act in the world. Nvidia notes agents use external APIs under a planblogs.nvidia.com.
- *Learning:* Over time, the agent refines itself. The Finextra article highlights that agents “will evolve their detection patterns based on new data, continuously updating their understanding” in fraud detectionfinextra.com. Mention RL training as one method.

Slide 8: Agentic AI vs. Generative AI

- **Reactive vs Proactive:** Standard LLMs (Generative AI) are *reactive* – they wait for prompts. Agentic AI is *proactive* – it can take initiative to pursue its goalsibm.com.
- **Content vs Decisions:** Generative AI focuses on creating content (text, images) in response to input. Agentic AI focuses on **decision-making and problem-solving**, using content generation only as a subtask.
- **Single-turn vs Multi-turn:** A generative model answers one question at a time. An agent maintains a **dialogue or plan state over multiple turns**, keeping track of progress.
- **Environment Interaction:** Agents continuously interact with the environment; LLMs-as-tools typically do not (unless wrapped with extra code).
- **Example:** ChatGPT (gen AI) won’t spontaneously rebalance your investment portfolio. An agentic system could monitor your portfolio daily and execute trades according to your strategy, with minimal prompts.

Speaker Notes: Contrast clearly: Cite IBM’s point that agentic AI is *proactive* while gen AI is *reactive*ibm.com. Emphasize that a typical LLM session is “here’s a prompt – it generates text –

done." But agentic AI stays "alive", iteratively taking actions, checking results, and planning next steps. Illustrate with everyday analogy: ChatGPT needs a question; an agentic personal assistant can decide to send you reminders or book tickets before you ask. Highlight that this shift from "tool" to "agent" means a whole change in system design and use cases.

Slide 9: Conceptual Architecture of an AI Agent

- **Perception/Input:** Gathers data (user input, documents, sensor data) and preprocesses it (feature extraction)blogs.nvidia.com.
- **Reason/Planning:** A core LLM "brain" interprets goals and generates a plan. It may use techniques like chain-of-thought or RAG (retrieval-augmented generation) to form solutionsblogs.nvidia.com.
- **Memory Store:** Maintains context (short-term) and knowledge (long-term). This could be an in-memory buffer or an external database of facts and past interactionscloud.google.com.
- **Action/Tools:** Executes the plan by calling tools – e.g., code execution, database queries, or REST APIsblogs.nvidia.com. This step has built-in *guardrails* or validators to ensure safe execution.
- **Feedback Loop:** Monitors outcomes, gets new inputs, and feeds them back to the planner. This loop enables learning and adaptation over time.

Speaker Notes: Walk through a hypothetical agent architecture. For *Perception*, mention Nvidia's breakdown: the agent gathers and processes data (could be text, images, or structured data)blogs.nvidia.com. Then in *Reason/Plan*, the LLM formulates a strategy – "I have to do X, Y, then Z"blogs.nvidia.com. This often involves retrieving relevant info and reasoning in steps. The *Memory* bullet draws on Google Cloud's description of short-term/long-term/episodic memorycloud.google.com – e.g., storing past user conversations or a knowledge base. *Action* means the agent uses APIs (calls, queries) to carry out tasksblogs.nvidia.com, and we build checks (e.g., "only transfer up to $1000 automatically") as shown in Nvidia's example. Finally, the *Feedback* implies a loop: if an action fails or new data arrives, the agent replans, potentially learning (Nvidia's "Learn" stepblogs.nvidia.com supports this).

Slide 10: Example 1 – Autonomous Fintech Customer Support

- **Scenario:** An agentic assistant handles customer inquiries (balances, payments, troubleshooting) without human handoff.

- **Action Workflow:** It parses a query (e.g., "How much do I owe?"), retrieves account data via APIs, and responds or takes action. For example, it might "check the outstanding balance and suggest which account to debit for a payment" autonomouslyblogs.nvidia.com.
- **Multi-turn dialogue:** Keeps context (customer name, prior questions) in memory to follow up and clarify, rather than starting each response from scratch.
- **Tool integration:** Connects to banking systems (transaction API, CRM database, etc.) to execute tasks (initiate a transfer, set up reminders).
- **Benefits:** 24/7 personalized support, faster resolution, and humans see only the exceptions. Early adopters like banks are already using AI to personalize customer serviceblogs.nvidia.com.

Speaker Notes: Describe a fintech chatbot that truly *acts*. For instance, a user says, "Pay my credit card bill." The agent understands (LLM interprets intent), checks the current balance through the bank's API, confirms which of the user's accounts has sufficient funds, and then initiates the payment – all without further promptsblogs.nvidia.com. This contrasts with a static bot that might just answer with "Here's your balance." Note how memory is used: if the customer had previously named a preferred account, the agent recalls it. We cite Nvidia's example: the agent can wait for a user decision and then complete the transaction as askedblogs.nvidia.com. Also mention that companies (e.g., Bank of America's "Erica") are evolving in this direction, and enterprises are exploring Copilot-like assistants for financial servicesblogs.nvidia.com.

Slide 11: Example 2 – Fintech Fraud Detection Agent

- **Scenario:** A network of AI agents continuously monitors transactions and accounts for fraud, collaborating in real timefinextra.com.
- **Agent Roles:** For instance, *Agent 1* verifies user identity (biometrics/device data); *Agent 2* scores transaction risk using history and contextual data; *Agent 3* intervenes (pauses/flags) if risk is high; *Agent 4* (investigator) generates a report if neededfinextra.com.
- **Collaboration:** Agents share data and pass tasks along automatically. E.g., one agent's suspicious flag triggers another agent to compose an alert or report for human reviewfinextra.com.
- **Speed & Accuracy:** The entire pipeline operates in seconds to block fraud and minimize loss. Over time, agents continuously learn from new fraud patterns, reducing false positives and improving detectionfinextra.com.

- **Outcome:** A mostly autonomous fraud monitoring system that handles routine cases and escalates only the novel or severe ones to human analysts.

Speaker Notes: Paint the picture of real-time fraud defense. Use Finextra's vision: agents acting in concertfinextra.com. Walk through a credit card transaction example. Each agent's function is a bullet: identity check, risk evaluation, response, report. Emphasize there's no single monolithic model: it's a *multi-agent system*. Note that one agent can auto-generate suspicious activity reports with full contextfinextra.com. Mention that this process (as per the source) "happens within seconds" to stop a transaction if neededfinextra.com. Also highlight that unlike static rule engines, these agents learn: Finextra notes they update their detection patterns continuouslyfinextra.com. Conclude that such agentic pipelines could greatly reduce fraud losses and free analysts for more complex cases.

Slide 12: Implications for Software Engineers

- **System Integration:** Engineers must combine LLMs with databases, APIs, and custom code into a cohesive pipeline. It's not just a model call – it's a *system design* problem.

- **Tool & API Use:** Embed proper guardrails. For example, validate any API calls an agent makesblogs.nvidia.com (Nvidia suggests limits like "auto-approve transactions only under $X" to ensure safety).

- **Data Governance:** Ensure data privacy and compliance. Agents will access sensitive financial data, so enforce encryption, anonymization, and audit loggingfinextra.com.

- **Explainability & Auditing:** In fintech, actions must be explainable. Log the agent's decision process and maintain clear records for regulatorsfinextra.com.

- **Development Approach:** Start small and iterative. Pilot agents on low-risk tasks (e.g., internal reporting) and gradually scale upfinextra.com. Build monitoring to catch failures and involve humans in the loop during testing.

- **Skillsets:** Engineers should be familiar with techniques like RAG (for knowledge retrieval), Reinforcement Learning, and system orchestration tools (e.g., workflow engines, Docker, etc.).

Speaker Notes: Address practical considerations. Emphasize that building agentic AI is more than ML – it requires software engineering rigor. Engineers need to architect *multi-component systems* (LLM service, memory database, tool invocation modules). Citing Nvidia, mention using APIs under controlled conditions (guardrails)blogs.nvidia.com. Then discuss compliance: Finextra stresses ethics and regulation – any agentic system in finance must avoid bias and provide transparencyfinextra.com. Highlight best practices: incremental rollout and testing (as

Finextra recommends starting small)finextra.com. Stress the importance of logging and dashboards so human operators can understand agent decisions. This slide prepares engineers for the implementation challenges ahead.

Slide 13: Conclusion & Outlook

- **Summary:** Agentic AI adds **proactivity, planning, memory, and tool use** to the LLM paradigmblogs.nvidia.comibm.com. It transforms LLMs from passive responders into active problem-solvers.

- **Impact:** In fintech, this means automating complex workflows (like portfolio management or risk analysis) with reduced manual steps. Many experts predict autonomous agents will be central in finance by 2030finextra.com.

- **Challenges Ahead:** Ensuring these systems are safe, accurate, and aligned with regulations remains critical. We still need robust human-in-the-loop design, bias mitigation, and continuous validation.

- **Future Directions:** Expect advances in multi-agent coordination, persistent memory architectures, and better training (e.g., reinforcement learning, program synthesis).

- **Takeaway:** For software engineers, agentic AI is an emerging shift – stay informed and experiment with agent frameworks. The tools we build today will set the stage for tomorrow’s autonomous AI systems.

Speaker Notes: Recap the journey from LLMs to agents. Reinforce that agentic AI is an extension – powered by LLMs but structured differently. Cite NVIDIA’s notion of “next frontier” solving multi-step problemsblogs.nvidia.com and IBM’s point on proactivityibm.com. Point to the future: the Finextra piece envisions autonomous agents core to banking risk by 2030finextra.com, which underlines how fast this field may evolve. Acknowledge open issues (mention fairness, oversight). Encourage the team: learn and prototype with current tools (we have RL libraries, vector DBs, etc.) so we’re ready for the agentic era.