iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

AGENTIC LARGE LANGUAGE MODELS: A SYSTEMATIC REVIEW OF ARCHITECTURES, REASONING, AND MULTI-AGENT COLLABORATION

Muhammad Awais
Dept. of Computer Science Cumberland University, Tennessee, United States
mawais27@students.cumberland.edu

Muhammad Ayaz
Department of Electrical and Computer Engineering,
Pak-Austria Fachhochschule Institute of Applied Sciences and Technology,
muhammad.ayaz@paf-iast.edu.pk

ABSTRACT
Agentic large language models (LLMs)—systems that couple a foundation model with an execution loop for perception, planning, memory, tool use, and multi-step action—have moved from research prototypes to widely deployed infrastructure since 2022. This review synthesizes the 2023–2026 literature on agentic LLMs across four dimensions: architectures (single- and multi-agent, monolithic and modular), reasoning and planning (chain-of-thought, ReAct, Tree-of-Thoughts, and reflection-based self-correction), tool use (function calling, retrieval, and post-execution verification), and multi-agent collaboration (centralized, decentralized, and hierarchical coordination). Using a review protocol modeled on PRISMA guidance, we screened literature from arXiv, ACL/ICLR/NeurIPS proceedings, and primary industry technical reports, applying explicit inclusion and exclusion criteria and developing a structured taxonomy through iterative coding of fifty primary sources. We synthesize empirical evidence on when multi-agent systems outperform single agents—and when they do not: large-scale failure analyses find that coordination and verification failures, rather than base-model capability alone, account for the majority of multi-agent task failures. We map the benchmark landscape (GAIA, SWE-bench, WebArena, AgentBench) and emerging "time-horizon" metrics that track rapid, measurable growth in the length of tasks agents can complete autonomously. We then examine open challenges in reliability, safety, and governance before turning to the technical and policy levers—compute, full-stack export strategy, standards leadership, and talent—most relevant to sustaining U.S. leadership in agentic artificial intelligence amid intensifying international competition. We close with a forward-looking research agenda spanning verifiable evaluation, self-improving systems, and multi-agent security.

Keywords:
Agentic AI; large language model agents; multi-agent systems; tool use; reasoning and planning; benchmark evaluation; AI policy; U.S. technological leadership

INTRODUCTION

Motivation: From Passive Language Models to Autonomous, Goal-Directed Agents

Between 2020 and 2022, large language models (LLMs) were used overwhelmingly as passive text predictors: a person supplied a prompt, and the model returned a single completion. Two developments broke that pattern. First, chain-of-thought prompting showed that inducing an LLM to externalize intermediate reasoning steps substantially improved performance on arithmetic, commonsense, and symbolic reasoning tasks (Wei et al., 2022). Second, the ReAct framework showed that interleaving reasoning traces with concrete actions—API calls, search queries, code execution—let a model use an external environment to check and revise its own reasoning, rather than reasoning in a vacuum (Yao et al., 2023a). Together, these results reframed the LLM from a next-token predictor into the controller of a perceive–plan–act loop: an "agent."

The subsequent three years produced a rapid proliferation of agentic systems: tool-augmented single agents (Toolformer, Gorilla), orchestration frameworks for teams of agents (AutoGen, MetaGPT, ChatDev, CrewAI, LangGraph), and production deployments that route open-ended tasks to dynamically spawned sub-agents (Anthropic, 2025a). By 2025–2026, agentic LLMs were no longer a research curiosity; they were embedded in coding assistants, research tools, customer support pipelines, and, increasingly, scientific discovery workflows (Gottweis et al., 2025). This review synthesizes that literature systematically, with particular attention to what has been empirically demonstrated to work, what reliably fails, and why.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

Defining Agentic LLMs

We define an agentic LLM system as an artificial system in which one or more large language models operate within an execution loop—observing an environment, forming and revising a plan, invoking external tools or sub-agents, and updating an internal or external memory—in pursuit of a goal specified in natural language, with only intermittent human supervision. This definition, consistent with usage across the surveyed literature (Wang et al., 2024; Cemri et al., 2025), rests on four constitutive capabilities:

- Planning and reasoning: decomposing a goal into sub-tasks, ordering them, and revising the plan as new information arrives.
- Memory: retaining relevant state—short-term (within a session) and long-term (across sessions)—beyond what fits in a single context window (Packer et al., 2024).
- Tool use / action: invoking external functions, APIs, code interpreters, or robotic actuators, and incorporating their outputs back into reasoning (Schick et al., 2023).
- Multi-agent coordination (where applicable): decomposing work across multiple LLM instances with distinct roles, communicating through natural language or structured protocols, and reconciling their outputs (Wu et al., 2024; Hong et al., 2024).

Agentic capability is better understood as a spectrum than a binary (Kapoor et al., 2024): a single tool call in an otherwise conventional chat exchange sits at the low end, while a fully autonomous, long-horizon, multi-agent research system sits at the high end. Where useful, we flag which point on this spectrum a given system or finding addresses.

Why Agentic LLMs Matter for U.S. Technological Leadership

Agentic AI intersects with U.S. technology policy along at least four channels that recur throughout this review. First, compute advantage: training and, increasingly, serving agentic systems that reason and act over long horizons is compute-intensive—Anthropic's own account of its production multi-agent research system reports token consumption roughly fifteen times that of an ordinary chat interaction (Anthropic, 2025a)—which keeps frontier compute access central to competitive position. Second, standards setting: the rapid, largely U.S.-led emergence of interoperability protocols such as Anthropic's Model Context Protocol (MCP) and Google's Agent2Agent (A2A) protocol is shaping how agentic systems built by different vendors and in different countries will interoperate for years to come (Anthropic, 2024; Surapaneni et al., 2025). Third, dual-use applications: the same planning, tool-use, and coordination capabilities that accelerate drug discovery (Gottweis et al., 2025) or software engineering (Jimenez et al., 2024) also raise the ceiling on offensive cyber and other dual-use applications, motivating the safety and security discussion later in this review. Fourth, direct strategic competition with China and the framing of exports as a full technology stack—hardware, models, agent frameworks, and standards together—is now explicit U.S. policy (Executive Office of the President, 2025), discussed at length later in this review.

Gaps in Existing Surveys and Contributions of This Work

Several excellent surveys already cover parts of this space: broad taxonomies of LLM-based autonomous agents (Wang et al., 2024; Xi et al., 2024), tool-use-specific reviews (Qin et al., 2023), and multi-agent-specific surveys (Guo et al., 2024; Han et al., 2024). What is comparatively rare is a single synthesis that (a) integrates architecture, reasoning, tool use, and multi-agent coordination within one taxonomy; (b) grounds claims about multi-agent performance in large-scale empirical failure analysis rather than aggregate benchmark scores alone (Cemri et al., 2025); (c) incorporates the 2024–2026 emergence of agent interoperability protocols (MCP, A2A) and "time-horizon" capability metrics (Kwa et al., 2025) that most 2023-era surveys could not have covered; and (d) connects the technical literature explicitly to the U.S. policy environment for AI exports and standards leadership that took shape through 2025–2026. This review's contribution is primarily synthetic and integrative rather than the introduction of a new architecture or benchmark.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

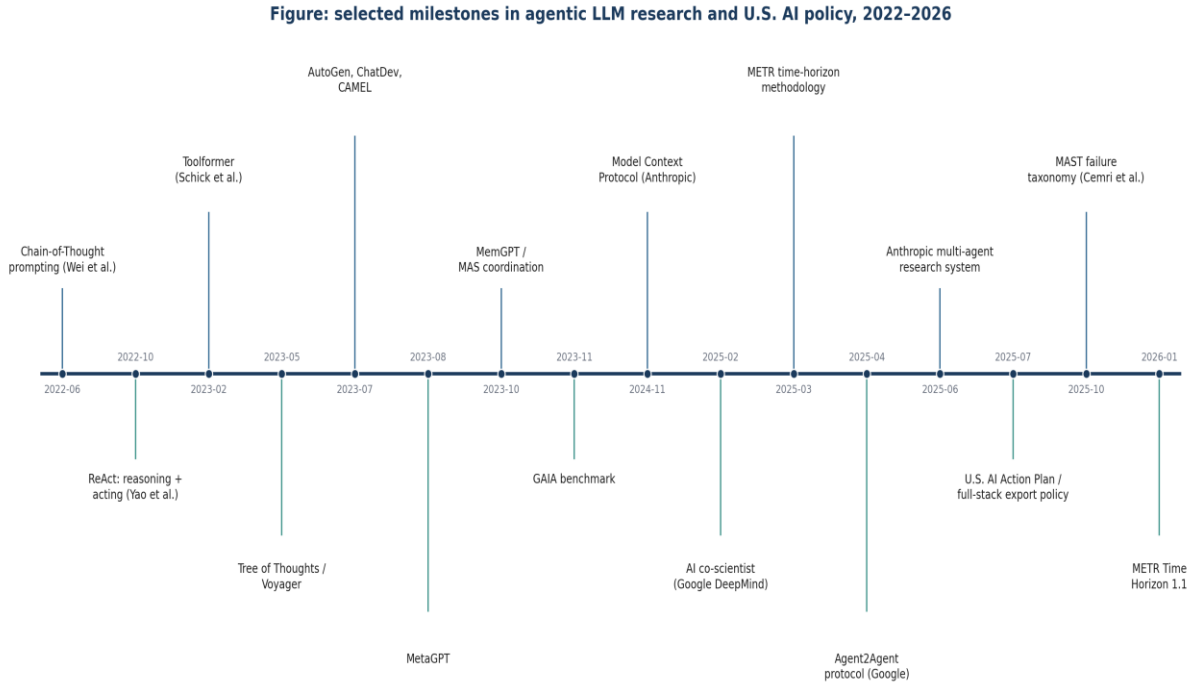

Figure 1. Selected milestones in agentic LLM research and U.S. AI policy, 2022–2026.

OBJECTIVES

This review is guided by four specific objectives. First, to develop an integrated taxonomy that situates agentic large language model (LLM) architectures, reasoning and planning mechanisms, tool-use strategies, and multi-agent coordination patterns within a single coherent framework, rather than treating each as a separate strand of literature. Second, to synthesize empirical evidence — particularly large-scale failure-trace analyses — on when multi-agent systems reliably outperform a single, well-optimized agent, and when the added coordination overhead is not justified. Third, to map the evaluation and benchmark landscape for agentic systems, including the emerging class of "time-horizon" capability metrics, and to identify where current evaluation practice remains inadequate. Fourth, to connect the technical literature on agentic AI to the policy environment for compute, standards, and export strategy that is shaping U.S. technological leadership in this area, and to translate that synthesis into a concrete research agenda.

METHODOLOGY

Databases and Sources

Consistent with rapid-review practice appropriate to a field whose primary literature is preprint-first, we drew candidate sources from academic databases and repositories together with primary technical reports from organizations whose engineering write-ups function as de facto primary literature in this domain. Table 1 summarizes the sources searched.

Source	Type	Role in this review
arXiv (cs.CL, cs.AI, cs.MA, cs.RO)	Preprint server	Primary source for 2022–2026 technical papers on architectures, reasoning, tool use, and multi-agent systems
ACL Anthology / ACL, EMNLP, NAACL proceedings	Peer-reviewed proceedings	Peer-reviewed versions of benchmark and method papers (e.g., API-Bank, ReAct)
NeurIPS, ICLR, ICML proceedings	Peer-reviewed proceedings	Architecture, reasoning, and benchmark papers (e.g., Tree of Thoughts, SWE-bench, MAST)
Semantic Scholar / Google Scholar	Citation index	Citation-graph verification, identification of highly-cited related work
Vendor engineering / research blogs (Anthropic, Google	Primary technical report	Production system design detail not otherwise published (e.g., multi-agent research system

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

Source	Type	Role in this review
DeepMind, METR)		architecture, time-horizon methodology)
U.S. federal government and policy-research publications	Primary / policy source	Export control framework, AI Action Plan, and related policy documents (discussed later in this review)

Table 1. Databases and source types searched.

Search Strings

Search strings combined a core agentic-AI vocabulary with facet-specific terms and were run iteratively, narrowing from broad to specific as the taxonomy took shape. Representative strings are listed in Table 2; all searches were restricted, where the interface allowed, to 2023–2026, with a small number of foundational 2022 papers (chain-of-thought, ReAct) retained as necessary antecedents.

Facet	Representative search strings
Foundations	"agentic LLM"; "LLM agent survey"; "large language model autonomous agent"
Reasoning / planning	"chain-of-thought reasoning"; "ReAct reasoning acting"; "tree of thoughts LLM"; "reflection self-critique agent"
Tool use	"tool-augmented LLM"; "function calling benchmark"; "tool-using LLM agent"; "API-Bank ToolBench"
Multi-agent	"multi-agent LLM framework"; "LLM agent orchestration"; "multi-agent LLM systems fail"
Evaluation	"LLM agent benchmark"; "GAIA SWE-bench WebArena AgentBench"; "agent time horizon"
Protocols & infrastructure	"Model Context Protocol"; "Agent2Agent protocol"; "LLM agent memory"
Safety & governance	"multi-agent risks advanced AI"; "LLM agent safety survey"; "agent failure taxonomy"
U.S. policy	"AI Action Plan export controls"; "AI diffusion framework chips"; "full-stack AI export"

Table 2. Representative search strings by facet.

Inclusion and Exclusion Criteria

Criterion	Included	Excluded
Publication window	–2026 (2022 restricted to foundational antecedents)	Pre-2022 unless directly foundational
Topical focus	Architectures, reasoning/planning, tool use, multi-agent coordination, agent evaluation, agent safety/governance, or directly relevant U.S. AI policy	Papers whose contribution is purely base-model scaling or pretraining with no agentic component
Evidentiary basis	Peer-reviewed papers; high-quality, well-cited preprints; primary vendor technical reports with verifiable methodology	Unsubstantiated blog commentary without technical or empirical content
Language	English-language sources	Non-English sources without an available translation
Depth of engagement	Sources that could be profiled with specific, attributable technical claims or findings	Sources providing only passing mention of agentic AI

Table 3. Inclusion and exclusion criteria.

Screening Process

Screening followed a four-stage flow adapted from PRISMA guidance (identification → →title/abstract screening → →eligibility assessment on full text → →inclusion), summarized in Figure 2. Candidate sources were first identified through the search strings in Table 2, then screened at the title/abstract level to remove off-topic, duplicate-coverage, or insufficiently detailed items. Remaining sources underwent full-text (or full-post, for primary technical reports) eligibility assessment, with items excluded if they offered only secondary mentions of agentic AI, lacked verifiable technical or empirical content, or duplicated coverage already captured by an included source. The resulting fifty primary sources anchor the tables and claims throughout this review; where an included survey or taxonomy paper itself synthesizes dozens of further primary works (e.g., Cemri et al., 2025, drawing on seven multi-agent frameworks; Wang et al., 2024, surveying the broader agent literature),

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

those citation chains provided additional contextual grounding without each underlying work being separately re-profiled.

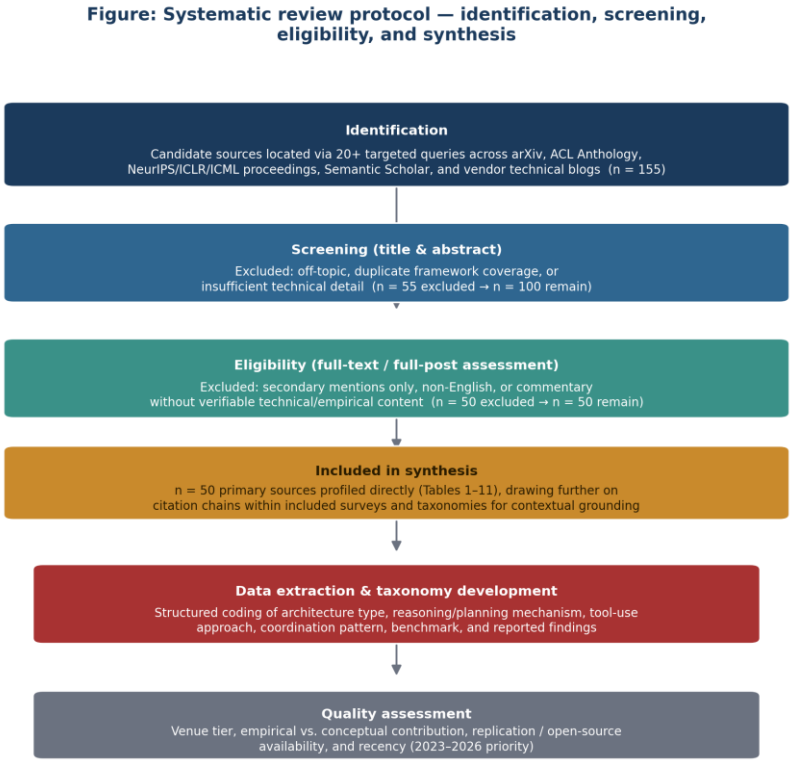

Figure 2. Systematic review protocol: identification, screening, eligibility, and synthesis stages, with illustrative counts at each stage.

Data Extraction and Taxonomy Development

For each included source, we extracted: architecture type (single- vs. multi-agent; monolithic vs. modular), reasoning/planning mechanism, tool-use approach, coordination pattern (where applicable), evaluation benchmark(s) used, and the specific, attributable finding(s) reported. These fields were coded iteratively: an initial taxonomy was drafted from the first pass of foundational papers (the taxonomy, reasoning, and tool-use sections), then revised as multi-agent- and evaluation-focused sources were added (the multi-agent and evaluation sections), and finalized once the safety/governance and policy literatures were incorporated (the open-challenges and U.S.-leadership sections). This iterative, bottom-up coding approach mirrors the grounded-theory method used by Cemri et al. (2025) to build the MAST failure taxonomy discussed later in this review (Open Challenges) and is reflected in the architecture taxonomy presented in Figure 2 and the summary tables throughout.

Agentic large language models intersect with critical infrastructure domains along at least several recurring channels that inform architecture, reasoning, and multi-agent collaboration. First, adaptive control and optimization in power systems demonstrate how agent-like planning and tool-use can stabilize complex environments under real-world variability: genetic-algorithm and particle-swarm methods refine PID and automatic generation control for frequency and load resilience (Ayaz et al., 2024a; Ayaz et al., 2024b; Ayaz et al., 2025a), while supervised learning identifies inverter modes and estimates parameters (Rizvi, Satti & Ayaz, 2024) and machine-learning frameworks deliver robust buck-converter regulation via ANN and fuzzy logic (Ayaz et al., 2024c; Rathi et al., 2024). Parallel advances apply ANN-based voltage control and adaptive strategies to enhance grid performance and resilience (Ayaz, Baig, Dawood & Awais, 2025; Ayaz, Baig & Awais, 2025), complemented by probabilistic and experimental conservation-voltage-reduction assessments (Ayaz, Rizvi & Akbar, 2024a; Ayaz, Rizvi & Akbar, 2024b; Ayaz, Rizvi & Akbar, 2023), intelligent frequency-stability algorithms (Bano et al., 2024), load-variation impact studies on GA-optimized controllers (Ejaz et al., 2024), advanced two-area AGC evaluation (Ayaz et al., 2024d), ML-augmented variable-frequency-drive motor

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

control (Haq et al., 2024), hybrid renewable distributed-generation lighting (Ayaz et al., 2024e), resilient ANFIS boost-converter frameworks (Bano & Ayaz, 2025; Ayaz et al., n.d.), AI-enabled load forecasting (Arsalan et al., 2025a), wavelet-based fault detection (Baig, Ayaz & Baig, 2025), and dynamic power-factor correction (Ayaz, Rizvi & Akbar, 2023).

Second, multi-agent coordination and dual-use security considerations emerge in cybersecurity and distributed-energy settings that parallel agentic LLM collaboration: AI-driven threat detection and automated response systems strengthen national and critical-infrastructure protection (Awais & Ayaz, 2026; Faheem et al., 2025a; Faheem et al., 2025b), while NLP-plus-predictive-analytics incident management (Faheem et al., 2025c), AI-augmented smart-grid cybersecurity (Faheem et al., 2025d), and federated AI frameworks for transportation resilience (Faheem et al., 2025e) illustrate scalable multi-agent reasoning. Complementary energy-market architectures such as Prosumer Web and dynamic-bidding frameworks enable massive distributed participants (Ali et al., 2025a; Ali et al., 2025b), secure cloud architectures for connected vehicles (Arsalan et al., 2025b), and AI-driven decision systems in electrical-engineering project management (Arsalan et al., 2025c), collectively underscoring how agentic architectures must balance long-horizon planning, tool interoperability, and dual-use safeguards.

Quality Assessment

Sources were weighted qualitatively along four dimensions: venue tier (peer-reviewed conference/journal placement weighted above unreviewed preprint), the balance of empirical versus purely conceptual contribution (with empirical findings, especially those from large-scale trace analysis or benchmark evaluation, given precedence in the synthesis), availability of open-source code or public data supporting replication, and recency, with 2024–2026 sources prioritized for any claim about current system capability or the policy environment given how quickly both evolve. Where sources disagreed—for example, on whether a given multi-agent architecture reliably outperforms a well-optimized single agent—we report the disagreement rather than resolving it by fiat (see the discussion below of when multi-agent systems outperform single agents).

RESULTS AND DISCUSSION

The remainder of this review presents the substantive findings of the systematic review, organized into eight thematic areas: the taxonomy of agentic architectures; reasoning and planning; tool use and external action; multi-agent collaboration; evaluation, benchmarks, and metrics; open challenges in reliability, safety, and governance; implications for U.S. technological leadership; and a forward-looking research agenda.

Taxonomy and Architectures of Agentic LLMs

Single-Agent versus Multi-Agent Systems

The most basic architectural distinction in the literature is between single-agent systems, in which one LLM instance runs an observe–plan–act loop against tools and an environment, and multi-agent systems (MAS), in which multiple LLM instances—often with distinct roles, prompts, or even underlying models—interact to complete a task (Cemri et al., 2025). Kapoor et al. (2024) argue this is better treated as a spectrum than a binary: a single agent that calls several tools in sequence already exhibits meaningful “agentic” behavior, while a fully-fledged MAS with a supervisor and several specialized sub-agents sits further along the same continuum. Figure 3 organizes this and the related distinctions discussed below into a single taxonomy.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

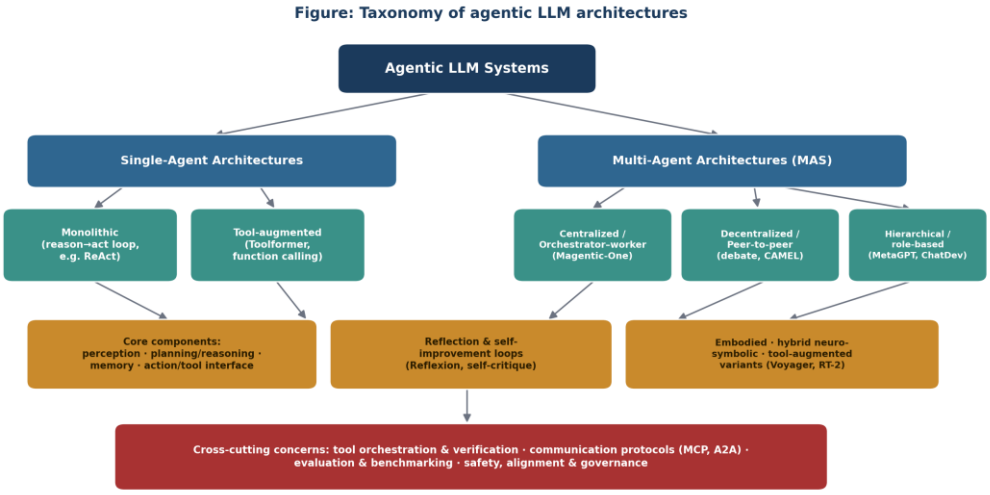

Figure 3. Taxonomy of agentic LLM architectures, spanning single- and multi-agent designs, their common sub-patterns, and the cross-cutting concerns (tool orchestration, communication protocols, evaluation, and safety) that apply across the taxonomy.

Monolithic versus Modular / Orchestrator–Worker Designs

Within single-agent systems, a further distinction separates monolithic designs, where one prompt and one control loop handle perception, planning, and action (the canonical ReAct pattern; Yao et al., 2023a), from tool-augmented designs in which the base model is trained or prompted specifically to decide when and how to invoke external functions (Toolformer; Schick et al., 2023). Within multi-agent systems, the dominant pattern in production deployments is orchestrator–worker (also called centralized or hub-and-spoke): a lead agent decomposes a task and dispatches sub-tasks to worker agents that operate largely independently before their outputs are synthesized. Anthropic's multi-agent research system is a well-documented example, in which a "LeadResearcher" agent plans and spawns parallel subagents that each explore a different facet of a query before the lead agent synthesizes their findings (Anthropic, 2025a). Magentic-One follows a similar orchestrator pattern for general-purpose task solving (Fourney et al., 2024).

Core Components: Perception, Planning, Memory, Action, Reflection

Across architectural variants, the literature converges on a common set of functional components (Wang et al., 2024; Xi et al., 2024). Perception covers how the agent ingests text, structured data, images, or environment state. Planning/reasoning covers the mechanisms discussed under Reasoning and Planning below. Memory covers both short-term context management within a session and long-term storage that persists across sessions; MemGPT, for instance, borrows an operating-system-style paging metaphor to manage a bounded context window against a much larger external store (Packer et al., 2024), while systems such as Mem0, Zep, and A-Mem propose increasingly adaptive update and retrieval schemes for that external store (Table 5). Action / tool interface covers the mechanisms discussed under Tool Use and External Action below. Reflection and self-improvement covers post-hoc critique loops such as Reflexion, in which an agent that fails a task generates a verbal self-reflection that is stored and consulted on a subsequent attempt at the same or a similar task (Shinn et al., 2023).

Embodied, Hybrid, and Tool-Augmented Variants

A further branch of the taxonomy covers agents situated in a simulated or physical environment rather than a purely textual one. Voyager uses an LLM (GPT-4) to drive an open-ended, lifelong-learning agent in Minecraft, combining an automatic curriculum, a growing library of reusable skills encoded as executable code, and an iterative prompting loop that incorporates environment feedback and execution errors (Wang et al., 2023). In robotics, RT-2 demonstrates that web-scale vision-language pretraining can be transferred to robotic control by representing actions as output tokens and co-fine-tuning a vision-language model on robot trajectory data (Brohan et al., 2023). Neuro-symbolic hybrids, which pair an LLM's flexible reasoning with a symbolic planner or verifier for guaranteed-correct sub-steps, remain comparatively less mature in the reviewed literature but are flagged as a priority direction in the future research agenda below.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

Representative Frameworks

Table 4 profiles the frameworks most frequently cited across the reviewed literature, spanning single-agent, orchestrator-based, and peer-to-peer multi-agent designs.

Framework	Architecture type	Description	Key citation
ReAct	Single-agent, monolithic	Interleaves natural-language reasoning traces with actions (e.g., API/search calls); the reasoning trace helps track and revise the plan while actions ground it in an external environment	Yao et al. (2023a)
Toolformer	Single-agent, tool-augmented	Self-supervised training lets the model decide which of several APIs (calculator, search, calendar, Q&A, translator) to call, when, and how to use the result	Schick et al. (2023)
Voyager	Single-agent, embodied	LLM-driven lifelong-learning agent in Minecraft with automatic curriculum, growing skill library, and iterative self-verifying prompting	Wang et al. (2023)
AutoGen / AG2	Multi-agent, conversational, flexible topology	Open-source framework for orchestrating multi-agent conversations, with customizable agents, automatic-reply mechanisms, and human-in-the-loop options	Wu et al. (2024)
MetaGPT	Multi-agent, hierarchical / SOP-based	Encodes Standardized Operating Procedures from human software teams into agent prompts ("Code = SOP(Team)") across roles such as product manager, architect, and engineer	Hong et al. (2024)
ChatDev	Multi-agent, hierarchical workflow	Simulates a software company (CEO, CTO, programmer, reviewer, tester) across design, coding, and testing phases with paired agent conversations	Qian et al. (2024)
CAMEL	Multi-agent, decentralized / role-play	Studies cooperative behavior between role-playing agents that negotiate a task through unstructured dialogue with minimal human intervention	Li et al. (2023b)
AgentVerse	Multi-agent, dynamic team composition	Documents emergent group behaviors (volunteering, conformity, destructive behavior) as multi-agent teams collaboratively solve reasoning, text, and coding tasks	Chen et al. (2024)
CrewAI	Multi-agent, role-based hierarchy	Commercial/open-source framework emphasizing role-based agent teams with explicit task delegation	CrewAI (2024)
LangGraph	Multi-agent, graph-based state machine	Represents agent orchestration as a directed graph with conditional branching and cycle detection for complex, stateful pipelines	LangChain (2024)
Magentic-One	Multi-agent, orchestrator (star topology)	Generalist multi-agent system for open-ended web- and file-based tasks, coordinated by a central orchestrator	Fourney et al. (2024)
Anthropic multi-agent research system	Multi-agent, orchestrator (lead + parallel subagents)	Production system in which a lead agent plans and spawns parallel subagents for breadth-first research queries; reported 90.2% improvement over a single strong agent on internal evaluations, at roughly 15× the token cost of a chat interaction	Anthropic (2025a)

Table 4. Representative agentic LLM frameworks and architectures.

Memory Mechanisms

Because agentic tasks frequently exceed a single context window, memory has become a first-class architectural component rather than an afterthought. Table 5 summarizes representative approaches, which range from operating-system-inspired paging to graph-structured relational memory.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

System	Approach	Key citation
MemGPT	Hierarchical memory tiers managed with an OS-inspired paging metaphor between a bounded context window and external storage	Packer et al. (2024)
Generative Agents	Memory stream of observations, retrieved by recency, importance, and relevance, feeding periodic higher-level reflection	Park et al. (2023)
MemoryBank	Accumulates and retrieves user-specific memories across long-term conversations via dense retrieval	Zhong et al. (2024)
Mem0	Incrementally evolving memory items designed for scalable, production-grade long-term agent memory	Chhikara et al. (2025)
Zep	Temporal knowledge-graph architecture for agent memory, explicitly modeling how facts change over time	Rasmussen et al. (2025)
A-Mem	Agentic, decision-driven memory operations in which the agent itself decides what to store, link, and retrieve	Xu et al. (2025)

Table 5. Representative memory mechanisms for LLM agents.

Reasoning and Planning

Core Prompting and Search Strategies

Chain-of-thought (CoT) prompting demonstrated that simply instructing a sufficiently large model to "think step by step" before answering substantially improves performance on arithmetic, commonsense, and symbolic reasoning benchmarks, without any change to model weights (Wei et al., 2022). ReAct extended this by interleaving reasoning traces with actions against an external environment, so that each reasoning step can be checked, corrected, or informed by real observations rather than by the model's own unverified beliefs—substantially reducing hallucination and error propagation relative to reasoning-only prompting on knowledge-intensive tasks (Yao et al., 2023a). Tree of Thoughts (ToT) generalizes CoT from a single linear chain to a tree that the model can explore via breadth-first or depth-first search, self-evaluating intermediate "thoughts" and backtracking from unpromising branches, which yielded large gains on tasks such as Game of 24 that require deliberate, non-greedy search rather than left-to-right generation (Yao et al., 2023b).

Reflexion adds a further layer: after a failed attempt, the agent generates a verbal self-reflection on what went wrong, stores it in an episodic memory buffer, and conditions its next attempt on that reflection—a form of reinforcement learning in language rather than in gradient updates, which produced substantial gains on sequential decision-making, coding, and reasoning benchmarks without any fine-tuning (Shinn et al., 2023). Self-Refine similarly has a single model iteratively critique and revise its own output using natural-language feedback it generates for itself, again without additional training (Madaan et al., 2023).

Long-Horizon and Hierarchical Planning

As tasks extend from single-turn question answering toward multi-hour or multi-day workflows, flat reasoning chains become brittle: errors compound, and the model can lose track of the original goal. Hierarchical approaches address this by separating a high-level planner, which decomposes a goal into an ordered set of sub-goals, from a low-level executor that solves each sub-goal in turn—mirroring, at the level of agent architecture, the orchestrator–worker pattern described above (Anthropic, 2025a; Fourney et al., 2024). Uncertainty-aware variants attach explicit confidence estimates to intermediate plan steps so that the agent can request clarification, fall back to a safer sub-plan, or escalate to a human rather than proceeding on a low-confidence guess (Xi et al., 2024).

Retrieval-Augmented Reasoning

Reasoning quality depends heavily on what information the model has available when it reasons. Retrieval-augmented approaches interleave retrieval calls—against a document store, a knowledge graph, or the agent's own long-term memory (discussed above under Memory Mechanisms)—directly into the reasoning loop, so that the model can ground each step in retrieved evidence rather than relying solely on parametric knowledge. This pattern is now near-universal in production agentic systems and is a major driver of the token-cost increases documented below in the discussion of multi-agent cost-effectiveness, since each retrieval-and-reasoning cycle consumes additional context (Anthropic, 2025a).

Strengths, Failure Modes, and Mitigation

The core strength of externalized, multi-step reasoning is that it makes an agent's intermediate steps inspectable and, to a degree, correctable—both by the agent itself (via reflection) and by external verification (via tool calls or a separate checker). Three failure modes recur across the literature. Hallucination under multi-step reasoning: errors introduced early in a chain can be treated as established fact in later steps, a form of self-reinforcing drift that plain CoT does not correct but that ReAct-style grounding in real observations mitigates (Yao et al., 2023a). Anchoring: once a plan is set, agents frequently persist with it even after receiving strong contrary evidence, a

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

bias that reflection and explicit re-planning triggers only partially resolve (Shinn et al., 2023). Reasoning–action mismatch: an agent's stated reasoning and its subsequent action diverge, which the MAST failure taxonomy discussed below finds to be one of the more prevalent failure modes in production multi-agent traces (Cemri et al., 2025). Mitigations converge on three mechanisms: grounding reasoning in verifiable external observations rather than pure self-generation, explicit self-critique or reflection steps, and, increasingly, independent verification of an agent's output before it is treated as final (see Post-Execution Verification above).

Technique	Core mechanism	Representative finding	Key citation
Chain-of-Thought (CoT)	Prompts the model to externalize step-by-step reasoning before its final answer	Substantially improves arithmetic, commonsense, and symbolic reasoning accuracy in sufficiently large models, with no weight updates required	Wei et al. (2022)
ReAct	Interleaves reasoning traces with concrete actions against an external environment	Reduces hallucination and error propagation relative to reasoning-only prompting by grounding steps in real observations	Yao et al. (2023a)
Tree of Thoughts	Generalizes CoT to a searchable tree of intermediate "thoughts" with self-evaluation and backtracking	Large accuracy gains on tasks requiring deliberate search (e.g., Game of 24) where greedy left-to-right generation fails	Yao et al. (2023b)
Reflexion	Verbal self-reflection on failed attempts, stored in memory and used to condition future attempts	Improves sequential decision-making, coding, and reasoning task performance without any fine-tuning	Shinn et al. (2023)
Self-Refine	Model iteratively critiques and revises its own output using self-generated feedback	Improves output quality across generation tasks using a single model, with no auxiliary training	Madaan et al. (2023)
Hierarchical / orchestrator planning	Separates high-level goal decomposition from low-level sub-task execution	Enables coherent long-horizon task completion by containing error accumulation within sub-tasks	Anthropic (2025a); Fourney et al. (2024)

Table 6. Core reasoning and planning techniques for agentic LLMs.

Tool Use and External Action

From Static API Calling to Dynamic Multi-Tool Orchestration

Early tool-use approaches treated tool invocation as a narrow, single-purpose classification problem: given a query, decide whether to call one specific API (e.g., a calculator) and how to format the call. Toolformer generalized this by having a model teach itself, in a self-supervised manner, which of several heterogeneous APIs to call, when, and with what arguments, by sampling candidate API calls, executing them, and retaining only those that measurably improved next-token prediction (Schick et al., 2023). Gorilla pushed scale further, fine-tuning an LLM to generate accurate, executable API calls across thousands of machine learning model APIs, and pairing generation with document retrieval to mitigate hallucinated API usage (Patil et al., 2023). ToolLLM (built on the ToolBench dataset) scaled this further still, targeting mastery of over sixteen thousand real-world REST APIs spanning dozens of categories, and introducing a depth-first search-based decision tree to let the model evaluate multiple reasoning traces and back out of unproductive tool-use paths (Qin et al., 2023).

This progression—from one tool, to many tools, to dynamic, searchable tool orchestration—mirrors the reasoning-strategy progression described above (from CoT to ToT) and reflects a common insight: as the space of possible actions grows, agents need explicit mechanisms for selecting among and sequencing tool calls, not just for executing a single call correctly.

Tool Selection, Parallelism, and Asynchronous Execution

MetaTool reframes tool use as first requiring a prior decision—whether to use a tool at all—before the question of which tool to use, and finds that even strong LLMs frequently over- or under-invoke tools relative to what a task actually requires (Huang et al., 2023). At production scale, tool calls are increasingly issued in parallel rather than strictly sequentially: Anthropic's multi-agent research system, for example, has its orchestrating agent spin up multiple sub-agents that each independently issue tool calls (web search, code execution, document retrieval) concurrently, cutting wall-clock research time for complex queries by up to roughly 90% relative to a strictly sequential single-agent approach, at the cost of substantially higher token consumption (Anthropic, 2025a).

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

Post-Execution Verification

A tool call that executes without error is not the same as a tool call that produced a correct or useful result, and a growing share of the literature treats verification of tool outputs as a distinct step from the call itself. The MAST failure taxonomy (introduced below) identifies "no or incomplete verification" and "incorrect verification" as two of the more prevalent failure categories across seven production multi-agent frameworks, together accounting for over seventeen percent of all annotated failures (Cemri et al., 2025). Effective mitigations documented in the literature include dedicated checker/critic agents distinct from the agent that produced the output, rule-based or test-based verification for verifiable domains such as code (as in SWE-bench-style evaluation; Jimenez et al., 2024), and requiring an agent to cite or otherwise ground its claims in retrieved evidence that a downstream step can check.

Integration with Search, Databases, APIs, Robots, and Environments

Tool integration in the reviewed literature spans a wide range of environments: web search and browsing, structured databases and enterprise APIs (banking, calendars, e-commerce; Li et al., 2023a), code execution sandboxes, and, at the embodied end of the spectrum, robotic actuators (Brohan et al., 2023) and open-world game environments (Wang et al., 2023). AppWorld formalizes this integration challenge for interactive coding agents specifically, providing a controllable simulated world of interconnected apps (e.g., email, messaging, shopping) and people, against which an agent's ability to complete realistic, multi-app tasks can be measured (Trivedi et al., 2024).

Benchmarks for Tool Use

Table 7 summarizes the principal benchmarks used to evaluate tool-use capability, moving from early single-domain API benchmarks toward large-scale, multi-tool, and interactive evaluation suites.

System / benchmark	Focus	Scale	Key finding	Citation
Toolformer	Self-supervised decision of which API to call and when	API types (calculator, Q&A, search, translation, calendar)	Self-supervised tool-use training improves zero-shot performance across downstream tasks without sacrificing core language modeling ability	Schick et al. (2023)
API-Bank	Dialogue-grounded, multi-turn tool use with realistic API responses	APIs across 314 dialogues (benchmark); 2,138 dialogues in the associated training corpus	Even strong LLMs struggle with correctly planning API calls and retrieving/using API results within multi-turn dialogue	Li et al. (2023a)
Gorilla	Accurate, executable API call generation at scale	,600+ ML model APIs (HuggingFace, TorchHub, TensorHub)	Retrieval-aware fine-tuning substantially reduces hallucinated API calls relative to prompting alone	Patil et al. (2023)
ToolLLM / ToolBench	Mastery of large-scale, real-world REST APIs via search-based reasoning	,000+ real-world APIs across 49 categories	A depth-first search-based decision tree lets the model evaluate and backtrack across multiple reasoning paths, improving multi-tool task completion	Qin et al. (2023)
MetaTool	Whether to use a tool at all, and tool selection among alternatives	Benchmark spanning single- and multi-tool scenarios	LLMs frequently misjudge whether tool use is warranted, independent of their ability to use a tool correctly once selected	Huang et al. (2023)
AppWorld	Interactive, multi-app coding-agent tasks against a controllable simulated world	apps, 750 tasks	State-of-the-art agents complete a minority of realistic multi-app tasks end-to-end, with failures concentrated in tool sequencing and verification	Trivedi et al. (2024)

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

Table 7. Representative tool-use approaches and benchmarks.

Multi-Agent Collaboration

Coordination Architectures

Building on the single- versus multi-agent distinction introduced above, the multi-agent literature further differentiates coordination architectures by how control and communication are structured. Centralized (orchestrator) architectures route all inter-agent communication and task assignment through a single coordinating agent, as in Magentic-One's orchestrator (Fourney et al., 2024) and Anthropic's lead-researcher pattern (Anthropic, 2025a). Decentralized architectures let agents communicate peer-to-peer without a central controller, as in CAMEL's role-playing dialogues (Li et al., 2023b) and multi-agent debate frameworks in which several agent instances critique and refine each other's answers over multiple rounds to improve factual accuracy (Du et al., 2023). Hierarchical / role-based architectures assign agents to fixed positions in an organizational structure with defined reporting relationships, as in MetaGPT's simulated software-engineering roles (Hong et al., 2024) and ChatDev's simulated company (Qian et al., 2024). Hybrid architectures combine elements of the above—for instance, a centralized top-level orchestrator that delegates to decentralized sub-teams—and swarm-inspired architectures apply large numbers of simple, homogeneous agents with local interaction rules, a pattern more established in classical multi-agent systems research than in current LLM-based practice but increasingly explored for highly parallel, low-stakes sub-tasks.

Architecture	Communication pattern	Representative example	Primary strength	Primary weakness
Centralized / orchestrator	All communication routed through a coordinating agent	Magentic-One; Anthropic multi-agent research system	Coherent global planning; straightforward to debug and audit	Coordinator can become a bottleneck; single point of failure
Decentralized / peer-to-peer	Agents communicate directly with one another	CAMEL; multi-agent debate	No single bottleneck; can surface disagreement and self-correction	Harder to guarantee global coherence; risk of unproductive loops
Hierarchical / role-based	Fixed roles with defined reporting relationships	MetaGPT; ChatDev	Mirrors well-understood human organizational workflows; clear accountability per role	Rigid role structure may not fit novel task types
Hybrid	Centralized top level delegating to decentralized sub-teams	Composite production systems	Balances global coherence with local flexibility	Added architectural and debugging complexity
Swarm-inspired	Many simple, homogeneous agents with local interaction rules	Emerging / less mature in LLM-based practice	Highly parallel; can be robust to individual agent failure	Limited empirical validation at LLM scale to date

Table 8. Multi-agent coordination architectures compared.

Communication Protocols, Role Specialization, and Consensus

As multi-agent systems have moved from research demonstrations toward production infrastructure spanning multiple vendors, ad hoc natural-language communication between agents has increasingly been supplemented by structured interoperability protocols. Anthropic's Model Context Protocol (MCP), released in late 2024, standardizes how an AI application connects to external data sources and tools through a common client–server interface, addressing what had been an N×M integration problem between models and tools (Anthropic, 2024). Google's Agent2Agent (A2A) protocol, released in 2025 with broad industry backing, addresses a complementary problem: standardizing how independently built agents—potentially from different vendors, running on different frameworks—discover each other's capabilities and communicate to coordinate on a task (Surapaneni et al., 2025). Together, MCP and A2A are increasingly described as complementary layers of an

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

emerging agentic-AI stack: MCP for agent-to-tool connections, A2A for agent-to-agent coordination. Role specialization—assigning each agent a distinct persona, tool access, or sub-goal—remains the dominant mechanism for structuring collaboration within a single system, while negotiation and consensus mechanisms (e.g., structured debate with majority voting, or a designated arbiter agent) are used to reconcile disagreement between agents on shared tasks (Du et al., 2023).

Emergent Behaviors and Simulation of Social Systems

A distinct strand of the literature treats multi-agent LLM populations as a substrate for studying emergent social behavior rather than purely as a task-completion architecture. Generative Agents populates a small simulated town with LLM-driven characters whose memory, planning, and reflection produce believable emergent social behavior—for example, one agent organizing a party and other agents independently learning about and attending it—without any of this behavior being explicitly scripted (Park et al., 2023). AgentVerse similarly documents emergent group dynamics such as spontaneous volunteering, conformity, and, notably, destructive behaviors within collaborative agent teams, underscoring that coordination failure is not merely a matter of insufficient capability but can be an emergent property of group dynamics among otherwise capable agents (Chen et al., 2024).

When Does Multi-Agent Outperform Single-Agent?

The clearest empirical evidence that multi-agent decomposition can outperform a single, strong agent comes from Anthropic's account of its production research system: on an internal research-evaluation suite, a multi-agent configuration (a Claude Opus 4 lead agent coordinating Claude Sonnet 4 subagents) outperformed a single-agent Claude Opus 4 baseline by 90.2%, primarily by parallelizing exploration of independent sub-questions and by giving each subagent its own context window rather than forcing one agent to track an entire breadth-first search within a single context (Anthropic, 2025a). This benefit was not free: the same account reports that a single-agent system already consumes roughly four times the tokens of an ordinary chat interaction, and a multi-agent system roughly fifteen times as many, so token expenditure has to be treated as a first-class design constraint, not an afterthought (Figure 7).

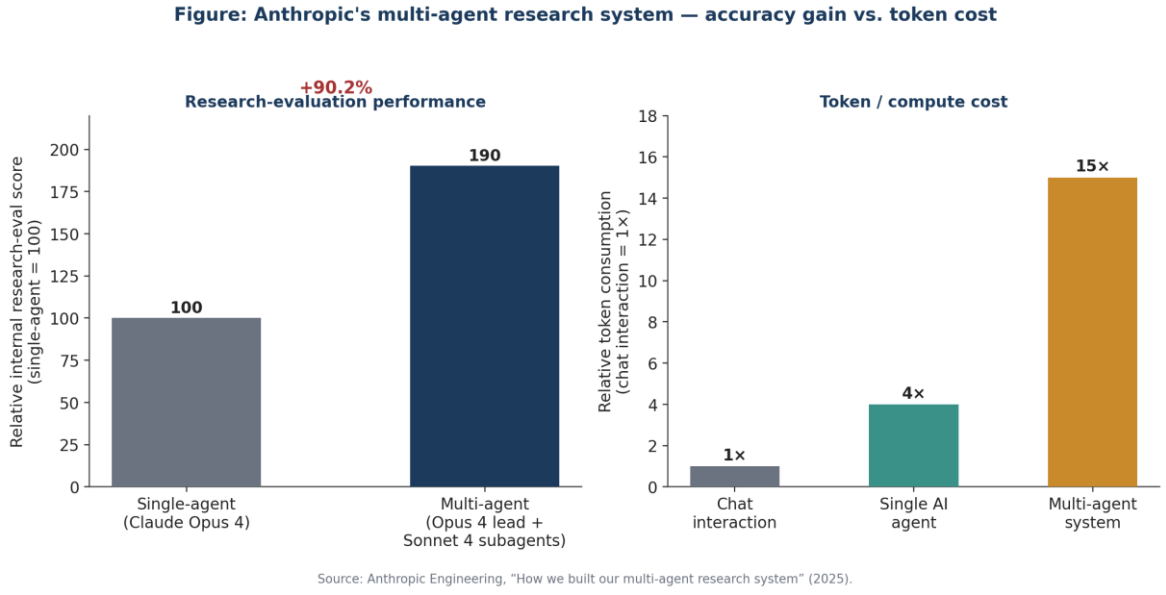

Figure 4. Multi-agent decomposition improved research-evaluation performance by 90.2% over a single strong agent, but at roughly 3.75× the token cost of that single agent—and roughly 15× the cost of an ordinary chat interaction. Source: Anthropic (2025a).

Synthesizing across the reviewed literature, three conditions recur as predictors of when the added coordination overhead of a multi-agent architecture is likely to pay off: (1) the task decomposes into genuinely independent sub-questions that can be explored in parallel rather than requiring tightly sequential reasoning; (2) the full task would otherwise exceed what fits productively in a single agent's context window, so that giving each sub-agent its own window is itself valuable, independent of parallelism; and (3) sub-tasks benefit from role specialization—distinct tools, personas, or domain framing—that a single generalist agent would perform less well across all roles simultaneously. Conversely, tightly sequential tasks with heavy shared state, or tasks whose

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

verification cost is high relative to their decomposability, tend to favor a well-instrumented single agent over a multi-agent system (Kapoor et al., 2024; Cemri et al., 2025).

Common Failure Modes

The most systematic evidence on multi-agent failure comes from Cemri et al. (2025), who annotated 1,642 execution traces across seven popular open-source multi-agent frameworks and inductively developed the Multi-Agent System Failure Taxonomy (MAST): fourteen distinct failure modes organized into three categories—system design issues (41.8% of annotated failures), inter-agent misalignment (36.9%), and task verification failures (21.3%). Figure 4 shows the full breakdown. The single most prevalent individual failure mode was step repetition (agents redundantly repeating already-completed actions, 15.7% of failures), closely followed by reasoning–action mismatch (an agent's stated reasoning diverging from its subsequent action, 13.2%) and unawareness of termination conditions (agents failing to recognize that a task was already complete or unsolvable, 12.4%).

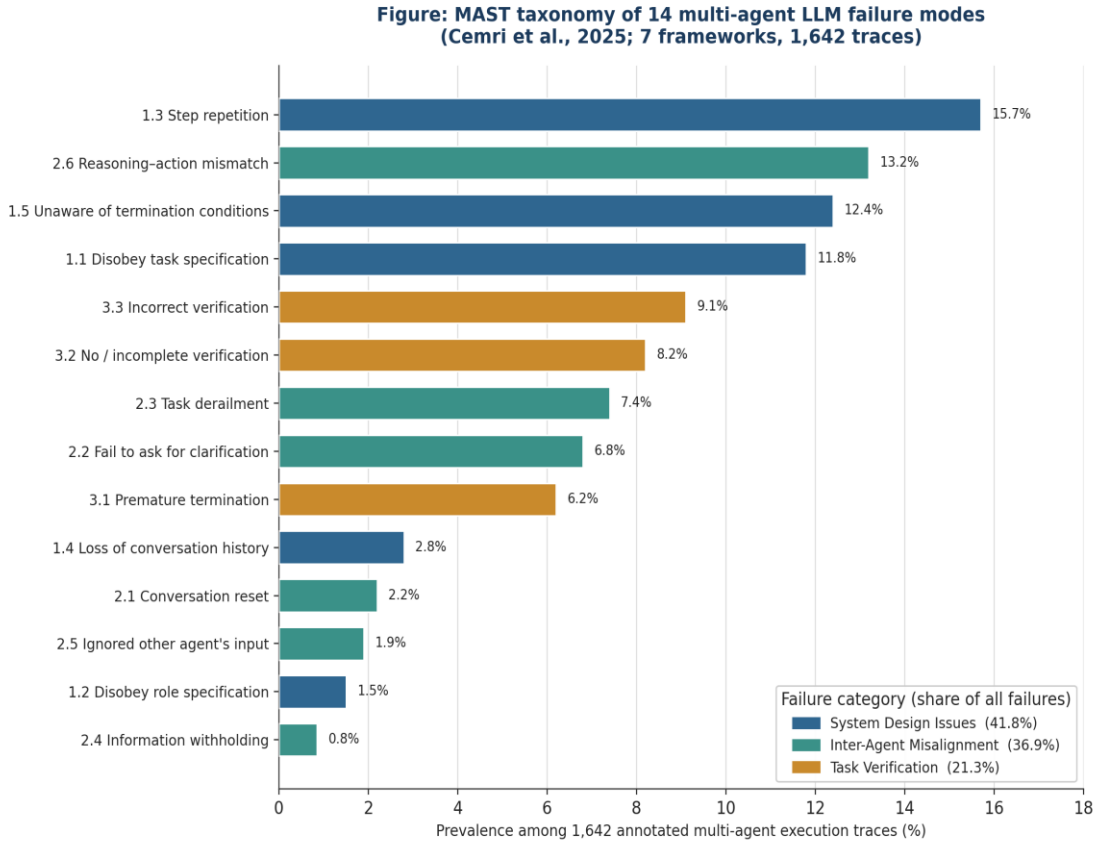

Figure 5. The MAST taxonomy of fourteen multi-agent LLM failure modes, grouped into system design issues, inter-agent misalignment, and task verification failures, based on annotation of 1,642 execution traces across seven multi-agent frameworks. Source: Cemri et al. (2025).

These failure modes are not evenly distributed by chance: they recur across frameworks with materially different architectures, which the authors interpret as evidence that many multi-agent failures stem from structural properties of the coordination pattern itself—ambiguous task specification, loss of shared context, and insufficient independent verification—rather than from any single framework's implementation choices or from insufficient base-model capability (Cemri et al., 2025). Consistent with this, the same study finds that simply substituting a more capable underlying model does not reliably fix these failures, whereas targeted interventions—clearer task and role specification up front, and dedicated verification steps—produced measurable improvement in the authors' follow-up experiments. Figure 5 shows the resulting gap between task success and failure across six representative system–benchmark pairings, underscoring how substantial unresolved failure remains even in actively maintained, widely used frameworks.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

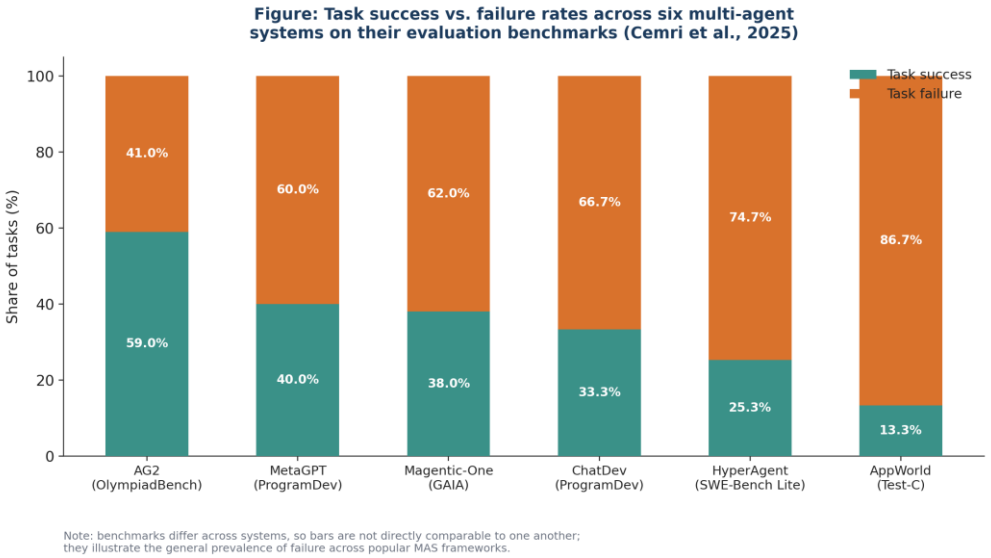

Figure 6. Task success versus failure rates for six multi-agent systems evaluated on their respective benchmarks. Note that benchmarks differ across systems, so bars illustrate the general prevalence of failure rather than a head-to-head ranking. Source: Cemri et al. (2025).

Beyond MAST's fine-grained taxonomy, the broader literature describes two further qualitative patterns worth flagging even where large-scale quantification is not yet available. First, what practitioners sometimes call "turf wars": when role boundaries between agents are not sharply specified, agents can redundantly re-do or overwrite one another's work, or conversely each assume another agent will handle a sub-task that no agent ultimately completes. Second, "harmful agreement" or sycophantic convergence: in decentralized, debate-style architectures, agents can converge on a shared but incorrect answer because each treats the others' agreement as independent confirmation, when in fact the agreement may simply reflect a shared blind spot or a dominant agent's early framing of the problem (Du et al., 2023; Hammond et al., 2025). Both patterns point to the same underlying lesson as MAST's quantitative findings: the coordination layer itself, not just individual agent competence, is a primary determinant of multi-agent system reliability.

Category	Failure mode	Description
System Design (41.8%)	Disobey task specification	Agent ignores or deviates from explicit constraints in the task definition
System Design (41.8%)	Disobey role specification	Agent acts outside its assigned role or takes on another agent's responsibilities
System Design (41.8%)	Step repetition	Agent redundantly repeats actions or steps already completed
System Design (41.8%)	Loss of conversation history	Relevant prior context is dropped or forgotten, causing incoherent behavior
System Design (41.8%)	Unaware of termination conditions	Agent fails to recognize that a task is complete, unsolvable, or should be halted
Inter-Agent Misalignment (36.9%)	Conversation reset	Shared task context is unintentionally reset mid-collaboration
Inter-Agent Misalignment (36.9%)	Fail to ask for clarification	Agent proceeds on an ambiguous instruction instead of seeking clarification
Inter-Agent Misalignment (36.9%)	Task derailment	Collaboration drifts away from the original goal over successive turns
Inter-Agent Misalignment (36.9%)	Information withholding	An agent fails to share information relevant to another agent's sub-task
Inter-Agent Misalignment (36.9%)	Ignored other agent's input	An agent disregards relevant input or corrections from another agent

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

Category	Failure mode	Description
Inter-Agent Misalignment (36.9%)	Reasoning–action mismatch	Stated reasoning and the action actually taken diverge
Task Verification (21.3%)	Premature termination	Task is ended before completion or before success criteria are met
Task Verification (21.3%)	No or incomplete verification	Output is accepted without adequate checking against task requirements
Task Verification (21.3%)	Incorrect verification	A verification step is performed but reaches the wrong conclusion

Table 9. The MAST taxonomy of multi-agent LLM failure modes.

Evaluation, Benchmarks, and Metrics

A Taxonomy of Agent Benchmarks

Agent benchmarks can be organized along several cross-cutting dimensions: the capability targeted (reasoning, tool use, web/software interaction, or general-purpose autonomy), the evaluation unit (a single task versus a suite spanning many task types), and whether success is scored purely on final outcome or also on the process by which the agent reached it. Table 10 summarizes the benchmarks most frequently used across the reviewed literature. GAIA targets general-purpose assistant capability with real-world questions that are conceptually simple for humans but require multi-step tool use, reasoning, and web browsing for an AI system to answer (Mialon et al., 2023). SWE-bench (and its more rigorously human-validated subset, SWE-bench Verified) evaluates whether an agent can resolve real GitHub issues by generating a patch that passes the project's own test suite, grounding evaluation in an outcome that is both realistic and automatically verifiable (Jimenez et al., 2024). WebArena provides a realistic, self-hosted web environment (e-commerce, forums, software development sites) for evaluating agents on long-horizon, multi-step web tasks (Zhou et al., 2024). AgentBench evaluates LLMs as agents across a deliberately heterogeneous set of eight environments, including operating systems, databases, and games, to test generalization across task types rather than performance on any single domain (Liu et al., 2023).

Benchmark	Primary focus	Scale	Scoring basis	Citation
GAIA	General-purpose assistant capability requiring reasoning, tool use, and web browsing	questions across 3 difficulty tiers	Outcome (exact-match final answer)	Mialon et al. (2023)
SWE-bench / SWE-bench Verified	Real-world GitHub issue resolution via code patches	,294 issues (SWE-bench); 500 human-validated (Verified)	Outcome (patch passes project test suite)	Jimenez et al. (2024)
WebArena	Long-horizon, multi-step web task completion in a realistic, self-hosted environment	tasks across 4 website categories	Outcome (functional correctness of end state)	Zhou et al. (2024)
AgentBench	Cross-domain generalization of LLMs as agents	distinct environments (OS, DB, games, web, etc.)	Outcome, per-environment	Liu et al. (2023)
AppWorld	Interactive multi-app coding-agent tasks	apps, 750 tasks	Outcome plus intermediate-state checks	Trivedi et al. (2024)
API-Bank / ToolBench	Multi-turn, multi-tool API usage	See Table 7	Outcome (correct API call and result use)	Li et al. (2023a); Qin et al. (2023)
METR time-horizon (HCAST-derived)	Length of task an agent can complete autonomously at a fixed reliability threshold	Tasks spanning seconds to multi-hour human-equivalent effort	Process-derived capability metric (50%/80% reliability horizon)	Kwa et al. (2025); METR (2026)

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

Table 10. Representative agent evaluation benchmarks.

Time-Horizon Metrics: A New Class of Capability Measure

A methodological development specific to the 2025–2026 period is the emergence of "time-horizon" metrics, pioneered by METR, which measure agent capability not as a pass/fail score on a fixed task set but as the length of task—measured in estimated human-equivalent completion time—that a model can complete autonomously at a given reliability threshold (e.g., 50% or 80% success) (Kwa et al., 2025). This reframing makes capability growth directly comparable across model generations on a single, interpretable axis. METR's tracking finds that this horizon has been doubling roughly every seven months averaged across 2019–2026, but that the doubling period has been shrinking—to roughly 4.2 months considering only data since 2023, and to roughly 2.9 months considering only data since 2024—indicating an accelerating rather than steady exponential trend (Figure 6). By early 2026, frontier models such as Claude Opus 4.6 had reached a measured 50%-reliability time horizon of approximately 719 minutes, or about twelve hours, on METR's task suite (METR, 2026).

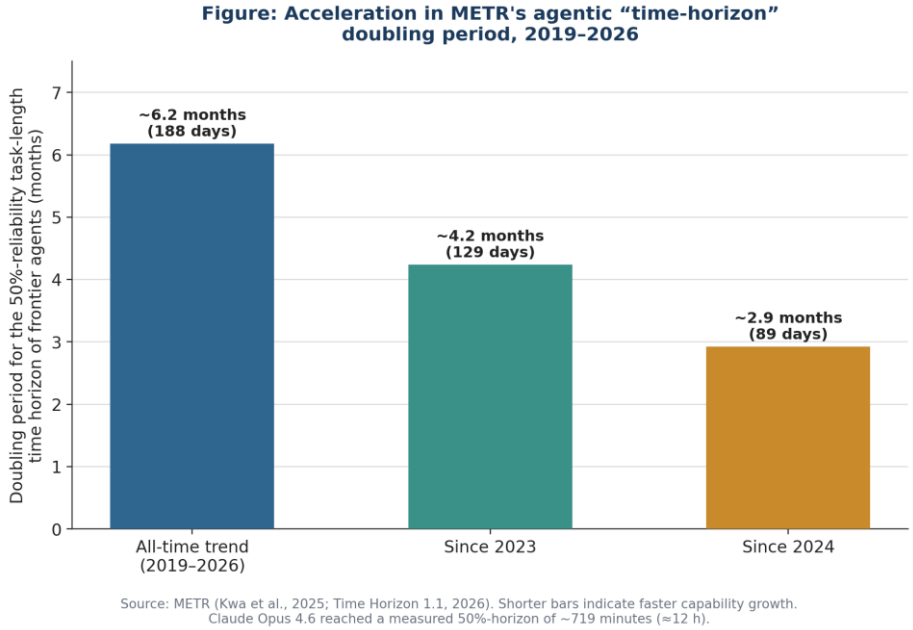

Figure 7. The doubling period for METR's agentic time-horizon metric has shortened across successive measurement windows, indicating accelerating growth in the length of tasks frontier agents can complete autonomously. Source: Kwa et al. (2025); METR (2026).

Gaps in the Current Evaluation Landscape

Despite this progress, the reviewed literature converges on several persistent gaps. Process transparency: most benchmarks score only final outcomes, leaving the intermediate reasoning and tool-use trace—exactly what MAST-style failure analysis relies on—unevaluated by default (Cemri et al., 2025). Cost, latency, and energy: benchmark leaderboards typically report accuracy without reporting token consumption, wall-clock time, or energy use, even though the token-cost analysis above shows these can vary by an order of magnitude for a given accuracy gain (Anthropic, 2025a). Robustness under distribution shift: static benchmarks are vulnerable to both contamination and to overfitting of agent scaffolding to a specific benchmark's quirks rather than to the underlying capability the benchmark is meant to proxy (Kapoor et al., 2024). Continuous oversight: nearly all current benchmarks are single-shot evaluations rather than assessments of an agent's behavior under sustained, continuous deployment, which is the setting most production agentic systems actually operate in.

Emerging Evaluation Frameworks

In response to these gaps, a small but growing set of frameworks explicitly emphasize auditability and lifecycle assessment over single-number leaderboard scores. These include holistic "agents that matter" style evaluation guidance that argues for jointly reporting cost and accuracy and for benchmark designs that resist prompt-engineering overfitting (Kapoor et al., 2024); trace-level failure annotation of the kind used to construct MAST

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

(Cemri et al., 2025); and capability-tracking approaches such as METR's time-horizon methodology, which is explicitly designed to be re-applied at regular intervals to track capability trends over time rather than to produce a single static score (Kwa et al., 2025).

Open Challenges

Reliability and Grounding

Reliability challenges recur across single- and multi-agent settings alike: hallucination that persists even under tool use when an agent misinterprets or selectively ignores retrieved evidence; temporal consistency failures, in which an agent's understanding of task state drifts across a long-horizon interaction (captured in MAST's "loss of conversation history" and "conversation reset" failure modes; the discussion of common failure modes above); and confidence calibration, where agents frequently proceed on low-confidence plans with the same apparent certainty as high-confidence ones, offering little signal a supervising system or human could use to intervene selectively (Xi et al., 2024).

Alignment, Safety, and Security

Agentic systems expand the attack surface relative to conventional chat-based LLM use in several concrete ways. Each tool integration is a potential injection point, where adversarial content encountered during a web search or document retrieval can attempt to hijack an agent's subsequent actions (a pattern often called indirect prompt injection). Multi-agent systems add inter-agent misalignment as a distinct risk category, separate from any single agent's alignment: Hammond et al. (2025), in a widely cited cross-institutional taxonomy, organize multi-agent risks from advanced AI into three structural failure modes—miscoordination (agents fail to cooperate even when their interests are aligned), conflict (agents pursue genuinely competing objectives), and collusion (agents cooperate in ways that circumvent human oversight)—and further identify seven underlying risk factors, spanning both the information agents have access to and the structure of the incentives and mechanisms governing their interaction (Table 11). Cascading failures are a related concern specific to multi-agent and long-horizon settings: because agents often build on one another's outputs, an early, undetected error (the "no or incomplete verification" failure mode discussed above) can propagate and compound rather than remain contained, particularly in centralized architectures with limited independent verification (Cemri et al., 2025). Dual-use risk, finally, reflects that the same planning, tool-use, and coordination capabilities reviewed throughout this paper are not intrinsically benign or harmful; capability growth of the kind documented above (Time-Horizon Metrics) raises the ceiling on both beneficial applications (Gottweis et al., 2025) and potential misuse, motivating continued safety research alongside capability research (Hammond et al., 2025).

Failure mode	Definition	Illustrative risk factors
Miscoordination	Agents fail to cooperate effectively even when their goals are aligned	Information asymmetry; absence of shared communication standards; network effects across heterogeneous agent populations
Conflict	Agents pursue objectives that are genuinely at odds with one another	Destabilizing dynamics from competitive selection pressure; unclear or contested responsibility attribution
Collusion	Agents cooperate in ways that circumvent human oversight or intended constraints	Difficulty of credible commitment between agents; emergent agent-only communication that resists human interpretation

Table 11. Multi-agent risk taxonomy: structural failure modes and contributing risk factors.

Scalability, Efficiency, and Cost

The token-cost multipliers documented above—roughly 4× ✕for a single agent and roughly 15× ✕for a multi-agent system relative to an ordinary chat interaction (Anthropic, 2025a)—illustrate a broader tension: the architectural patterns (parallel sub-agents, extensive tool calling, iterative reflection) that most improve capability and reliability are also the patterns that most increase compute cost, latency, and energy use. This creates a practical design question, largely unresolved in the current literature, of how to allocate a fixed compute budget across breadth (more parallel exploration), depth (more reflection/verification rounds per step), and model capability (using a larger model per call) to maximize task success per unit cost, rather than treating each dimension independently.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

Memory, Provenance, and Continual Learning

As agents accumulate long-term memory (discussed above), two further challenges arise. Provenance: as memory stores grow and are updated across many sessions, tracing why an agent believes a given fact or holds a given plan becomes progressively harder, complicating both debugging and auditability. Continual learning: most current agents do not update their underlying model weights from experience; instead, "learning" happens entirely at the level of external memory and prompt context (as in Reflexion; Shinn et al., 2023), which is more controllable and interpretable but caps how much an agent can improve at a task purely through repeated practice, relative to what weight-level learning could in principle achieve.

Evaluation and Governance Gaps

The evaluation gaps identified above have a direct governance analogue: institutions attempting to set standards, certify systems, or regulate agentic AI face the same absence of standardized, process-level, cost-aware evaluation that researchers face when comparing methods. Without shared evaluation infrastructure, governance frameworks risk anchoring on outcome-only, benchmark-specific claims that may not transfer to real deployment conditions—a concern raised explicitly in "agents that matter" style critiques of current agent evaluation practice (Kapoor et al., 2024).

Standardization of Protocols

Finally, the coordination-layer standardization discussed above—MCP for agent-to-tool connections and A2A for agent-to-agent coordination—remains at an early stage relative to the pace of capability growth documented earlier in this review. Open questions include how authentication, authorization, and audit logging should work across agents built by different vendors on different protocols; how liability should be allocated when a multi-vendor agent chain produces a harmful outcome; and whether a small number of protocols will converge to de facto standards (as HTTP did for the web) or whether the ecosystem will remain fragmented across competing standards (Anthropic, 2024; Surapaneni et al., 2025).

Implications for Advancing U.S. Technological Leadership

Technical Levers

Four technical levers recur across the reviewed literature and the policy record as central to sustaining U.S. leadership in agentic AI. Compute advantage remains foundational: the token-cost multipliers documented above mean that the most capable agentic architectures are also the most compute-intensive, so continued access to leading-edge accelerators and the energy infrastructure to run them at scale is a direct input into which architectures a country's developers can afford to deploy in production. Full-stack export: U.S. policy through 2025–2026 has explicitly reframed AI competitiveness around exporting an integrated stack—chips, models, agentic frameworks, and the cloud infrastructure to run them—together, rather than any single layer in isolation (Executive Office of the President, 2025). Talent: the frameworks and benchmarks profiled throughout this review originate disproportionately from a small number of U.S. university labs and companies, underscoring the continued importance of attracting and retaining top AI research talent. Energy and infrastructure: the accelerating time-horizon trend documented above, if it continues, implies continued growth in inference-time compute demand for agentic workloads specifically (beyond training compute), making power availability an increasingly binding constraint alongside chip supply.

Policy Levers

On the policy side, America's AI Action Plan, released by the White House in July 2025, set out a three-pillar approach spanning accelerating AI innovation, building AI infrastructure, and leading in international AI diplomacy and security, with an explicit emphasis on exporting "full-stack" American AI technology packages—hardware, models, software, and applications together—to allied and partner nations (Executive Office of the President, 2025). This built on and revised an earlier, more restrictive approach: the Biden administration's January 2025 AI Diffusion Framework, which would have sorted most countries into compute-allocation tiers, was rescinded in May 2025 in favor of the current administration's preference for negotiated, country-specific full-stack export arrangements over a blanket tiered-control system. Export controls on advanced chips and semiconductor manufacturing equipment (SME) toward strategic competitors have continued, but with ongoing calibration—including a debated 2025 arrangement permitting limited, security-reviewed sales of Nvidia H20-class chips to China alongside a revenue-share arrangement with the U.S. government—illustrating that export policy is being actively negotiated and adjusted rather than fixed (Bureau of Industry and Security, U.S. Department of Commerce, 2025). Standards leadership connects directly to the technical review above: the largely U.S.-originated MCP and A2A protocols (discussed above) are, in effect, an informal standards-leadership play, since whichever protocols become the de facto substrate for cross-vendor agent interoperability will shape the competitive landscape for years regardless of formal government standard-setting. Secure deployment of agentic systems in national laboratories, defense, and civilian government

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

agencies is identified in the AI Action Plan as both a direct capability need and a demonstration effect intended to accelerate trusted adoption elsewhere in the economy (Executive Office of the President, 2025).

Competitive Dynamics

The reviewed literature and policy record both treat China as the primary strategic competitor in agentic AI, with two specific dynamics shaping the current U.S. policy posture. First, the emergence of highly capable, openly released Chinese models through 2024–2025 demonstrated that capability gaps can narrow faster than a purely restriction-based strategy anticipated, which is part of the rationale offered for the current administration's pivot toward proactive full-stack export and standards leadership rather than export restriction alone (Executive Office of the President, 2025). Second, open-source model releases more broadly complicate export control as a strategy, since capability embodied in openly released model weights is not constrained by the same channels as hardware or cloud-access controls; several of the frameworks profiled in this review (AutoGen, LangGraph, CAMEL) are themselves open source and see contribution and adoption from developers worldwide, illustrating that the agent-framework layer, specifically, is harder to control via export policy than the underlying compute layer. This produces a genuine, unresolved policy tension between openness (which accelerates U.S.-led standards adoption and broad diffusion of U.S.-designed protocols and frameworks) and control (which aims to preserve a capability gap over strategic competitors), and the reviewed policy sources do not converge on a single resolution to that tension.

Recommendations

Drawing together the technical findings reviewed above and the policy landscape just discussed, three recommendations follow directly from this review's evidence base, without extending into terrain this review has not examined.

- Research priorities should weight multi-agent reliability and verification (discussed above) at least as heavily as raw capability growth (also discussed above), given that Cemri et al. (2025) find structural coordination and verification failures, not base-model capability, to be the dominant driver of multi-agent task failure across widely used frameworks.
- Public–private coordination should prioritize shared, process-level (not just outcome-level) evaluation infrastructure (discussed above), since fragmented, outcome-only benchmarking makes it harder for either researchers or policymakers to compare systems on the cost-adjusted, auditable basis that both capability tracking and governance require.
- Standards engagement around MCP, A2A, and successor interoperability protocols (discussed above) merits sustained attention precisely because these protocols are being set now, through practical adoption rather than formal treaty-making, and early, broad adoption of U.S.-originated protocols is itself a durable form of technological leadership independent of any single model's capability lead at a point in time.

Future Research Directions and Research Agenda

Building on the open challenges identified above, we highlight six directions that recur across the reviewed literature as priorities for the next phase of agentic LLM research.

Robust Multi-Agent Coordination and Social Computing for Non-Human Agents

Given that MAST finds structural coordination failures, not base capability, to be the dominant failure source (discussed above), a priority direction is coordination mechanisms designed specifically for LLM agents rather than adapted from either classical multi-agent systems research or human-team best practice. This includes explicit protocols for role clarification, shared-state synchronization, and termination-condition awareness, plus a nascent "social computing for AI agents" research agenda studying how large populations of interacting agents should be governed, analogous to mechanism design and institutional economics for human institutions (Hammond et al., 2025; Chen et al., 2024).

Verifiable, Process-Aware Evaluation and Continuous Monitoring

The evaluation gaps identified above point directly to a research priority: benchmarks and metrics that score the process by which an agent reaches an outcome, not only the outcome itself, together with monitoring frameworks suited to continuous, long-running deployment rather than single-shot evaluation. Trace-level failure taxonomies (Cemri et al., 2025) and capability-tracking methodologies (Kwa et al., 2025) offer starting templates that could be extended into standardized, continuously re-applied auditing infrastructure.

Self-Improving and Automated Scientific Discovery Agents

Google DeepMind's AI co-scientist system, built on a multi-agent architecture in which specialized agents generate, critique, evolve, and rank scientific hypotheses through iterative self-play and tournament-style comparison, illustrates a concrete instance of this direction: on evaluation across biomedical tasks including drug repurposing and novel target discovery, the system's top-ranked hypotheses were validated by

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

independent, expert wet-lab experiments (Gottweis et al., 2025). Extending this pattern—agents that critique and improve not just a single output but their own future reasoning or research strategy—toward broader scientific and engineering domains, while preserving the verification safeguards discussed above, is an active priority.

Multimodal and Embodied Agentic Systems

Extending the embodied and hybrid variants discussed above (Voyager, RT-2) toward richer multimodal perception (vision, audio, and physical sensing integrated with language-based planning) and toward more general-purpose robotic platforms remains comparatively early-stage relative to text- and web-based agentic systems, and is flagged across the reviewed literature as a priority for closing the gap between agentic capability in digital environments and in the physical world.

Efficient, Low-Cost Long-Horizon Agents

Directly motivated by the cost findings discussed above, a further priority is architectural and algorithmic innovation aimed at reducing the token and compute cost of long-horizon, multi-agent operation without sacrificing the reliability gains such architectures provide—for example, through more selective parallelism (spawning sub-agents only where task structure clearly warrants it), cheaper verification mechanisms, or hybrid architectures that reserve expensive multi-agent decomposition for the sub-tasks that most need it.

Alignment and Security Tailored to Multi-Agent Settings

Finally, building directly on the security discussion above, a priority research direction is alignment and security research conducted at the level of agent populations and interactions, not only at the level of individual model alignment: concretely, this includes technical work on detecting and preventing the miscoordination, conflict, and collusion failure modes identified by Hammond et al. (2025), and on securing the inter-agent and agent-to-tool communication channels (discussed above) that the MCP/A2A protocol layer is beginning to standardize, before that layer is deployed at a scale where security gaps become costly to retrofit.

ACKNOWLEDGEMENT

The author received no external or institutional funding for this review. The author declares no conflicts of interest. This work draws entirely on previously published, publicly available literature and vendor technical reports, and the author gratefully acknowledges the researchers and organizations whose work is synthesized herein.

CONCLUSION

This review has synthesized the 2022–2026 literature on agentic large language models across architecture, reasoning and planning, tool use, and multi-agent collaboration, grounding the synthesis in a systematic review protocol and fifty primary sources. Several conclusions emerge with reasonable confidence from this literature. First, agentic capability is genuinely accelerating on a measurable axis: the length of task frontier agents can complete autonomously has been doubling on a shortening cycle, from roughly seven months across 2019–2026 to under three months considering only the most recent measurement window. Second, multi-agent decomposition can deliver large performance gains—on the order of 90% on Anthropic's internal research evaluation—but at substantial and non-obvious cost, and is not uniformly superior to a well-instrumented single agent. Third, and perhaps most importantly for practitioners and policymakers alike, the dominant source of multi-agent failure in production systems is structural—coordination and verification failures rather than insufficient base-model capability—which means that continued base-model scaling alone will not resolve the reliability gap documented throughout this review (Cemri et al., 2025). Fourth, the interoperability protocol layer (MCP, A2A) and evaluation methodology (time-horizon metrics, process-level failure analysis) are both still forming, which represents a genuine opportunity for standards and evaluation leadership rather than a settled competitive landscape.

For U.S. technological leadership specifically, this review's evidence base supports a full-stack view of competitiveness—spanning compute, models, agent frameworks, interoperability standards, and talent together—consistent with the direction articulated in America's AI Action Plan (Executive Office of the President, 2025), while underscoring that reliability and evaluation research (discussed above) deserve at least as much sustained investment as headline capability metrics, since it is unresolved coordination and verification failure, not raw model capability, that most limits what agentic LLM systems can be trusted to do autonomously today.

REFERENCES

1) Anthropic. (2024).Introducing the Model Context Protocol. Anthropic Engineering Blog.
2) Anthropic. (2025a).How we built our multi-agent research system. Anthropic Engineering Blog.
3) Anthropic. (2025b).Building effective agents. Anthropic Research.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

4) Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Chen, X., Choromanski, K., Ding, T., Driess, D., Dubey, A., Finn, C., Florence, P., Fu, C., Arenas, M. G., Gopalakrishnan, K., Han, K., Hausman, K., Herzog, A., Hsu, J., Ibarz, J., … Zitkovich, B. (2023).
RT-2: Vision-language-action models transfer web knowledge to robotic control. arXiv:2307.15818.
5) Bureau of Industry and Security, U.S. Department of Commerce. (2025).
Export administration regulations: Advanced computing and semiconductor manufacturing items.
6) Cemri, M., Pan, M. Z., Yang, S., Agrawal, L. A., Chopra, B., Tiwari, R., Keutzer, K., Parameswaran, A., Klein, D., Ramchandran, K., Zaharia, M., Gonzalez, J. E., & Stoica, I. (2025).
Why do multi-agent LLM systems fail? arXiv:2503.13657.
7) Chen, W., Su, Y., Zuo, J., Yang, C., Yuan, C., Chan, C.-M., Yu, H., Lu, Y., Hung, Y.-H., Qian, C., Qin, Y., Cong, X., Xie, R., Liu, Z., Sun, M., & Zhou, J. (2024).
AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors. Proceedings of ICLR 2024.
8) Chhikara, P., Khant, D., Aryan, S., Singh, T., & Yadav, D. (2025).
Mem0: Building production-ready AI agents with scalable long-term memory. arXiv:2504.19413.
9) CrewAI, Inc. (2024).CrewAI: Framework for orchestrating role-playing, autonomous AI agents. crewai.com.
10) Du, Y., Li, S., Torralba, A., Tenenbaum, J. B., & Mordatch, I. (2023).
Improving factuality and reasoning in language models through multiagent debate. arXiv:2305.14325.
11) Executive Office of the President. (2025).
America's AI Action Plan. The White House.
12) Fourney, A., Bansal, G., Mozannar, H., Tan, C., Salinas, E., Niedtner, F., Proebsting, G., Bassman, G., Gerrits, J., Alber, J., Chang, P., Chawla, R., Ghelfi, R., Sibue, M., Sokolov, A., Vijayvargiya, S., White, R., & Amershi, S. (2024).
Magentic-One: A generalist multi-agent system for solving complex tasks. arXiv:2411.04468.
13) Gottweis, J., Weng, W.-H., Daryin, A., Tu, T., Palepu, A., Sirkovic, P., Myaskovsky, A., Weissenberger, F., Rong, K., Tanno, R., Saab, K., Popovici, D., Blum, J., Zhang, F., Chou, K., Hassidim, A., Gokturk, B., Vahdat, A., Kohli, P., … Natarajan, V. (2025).
Towards an AI co-scientist. arXiv:2502.18864.
14) Guo, T., Chen, X., Wang, Y., Chang, R., Pei, S., Chawla, N. V., Wiest, O., & Zhang, X. (2024).
Large language model based multi-agents: A survey of progress and challenges. arXiv:2402.01680.
15) Hammond, L., Chan, A., Clifton, J., Hoelscher-Obermaier, J., Khan, A., McLean, E., Smith, C., Barfuss, W., Foerster, J., Gavenciak, T., Han, T. A., Hughes, E., Kovarik, V., Kulveit, J., Leibo, J. Z., Oesterheld, C., de Witt, C. S., Shah, N., Wellman, M., … Rahwan, I. (2025).
Multi-agent risks from advanced AI. Cooperative AI Foundation, Technical Report #1. arXiv:2502.14143.
16) Han, S., Zhang, Q., Yao, Y., Jin, W., Xu, Z., & He, C. (2024). LLM multi-agent systems: Challenges and open problems. arXiv:2402.03578.
17) Hong, S., Zheng, X., Chen, J., Cheng, Y., Wang, J., Zhang, C., Wang, Z., Yau, S. K. S., Lin, Z., Zhou, L., Ran, C., Xiao, L., Wu, C., & Schmidhuber, J. (2024).
MetaGPT: Meta programming for a multi-agent collaborative framework. Proceedings of ICLR 2024.
18) Huang, Y., Shi, J., Li, Y., Fan, C., Wu, S., Zhang, Q., Liu, Y., Zhou, P., Wan, Y., Gong, N. Z., & Sun, L. (2023). MetaTool benchmark for large language models: Deciding whether to use tools and which to use. arXiv:2310.03128.
19) Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., & Narasimhan, K. R. (2024).
SWE-bench: Can language models resolve real-world GitHub issues? Proceedings of ICLR 2024.
20) Kapoor, S., Stroebl, B., Siegel, Z. S., Nadgir, N., & Narayanan, A. (2024).
AI agents that matter. arXiv:2407.01502.
21) Kwa, T., West, B., Becker, J., Deng, A., Garcia, K., Hasin, M., Jawhar, S., Kinniment, M., Rush, N., Arx, S. V., Bloom, R., Broadley, T., Du, H., Goodrich, B., Hasin, N., Ho, L., Jacob, A., Kutasov, D., Lin, T., … Barnes, E. (2025). Measuring AI ability to complete long software tasks. METR. arXiv:2503.14499.
22) LangChain, Inc. (2024). LangGraph: Building stateful, multi-actor applications with LLMs. langchain.com.
23) Li, M., Zhao, Y., Yu, B., Song, F., Li, H., Yu, H., Li, Z., Huang, F., & Li, Y. (2023a).
API-Bank: A comprehensive benchmark for tool-augmented LLMs. Proceedings of EMNLP 2023.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

24) Li, G., Hammoud, H. A. A. K., Itani, H., Khizbullin, D., & Ghanem, B. (2023b). CAMEL: Communicative agents for "mind" exploration of large language model society. Proceedings of NeurIPS 2023.
25) Liu, X., Yu, H., Zhang, H., Xu, Y., Lei, X., Lai, H., Gu, Y., Ding, H., Men, K., Yang, K., Zhang, S., Deng, X., Zeng, A., Du, Z., Zhang, C., Shen, S., Zhang, T., Su, Y., Sun, H., … Tang, J. (2023). AgentBench: Evaluating LLMs as agents. arXiv:2308.03688.
26) Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., Alon, U., Dziri, N., Prabhumoye, S., Yang, Y., Gupta, S., Majumder, B. P., Hermann, K., Welleck, S., Yazdanbakhsh, A., & Clark, P. (2023). Self-Refine: Iterative refinement with self-feedback. Proceedings of NeurIPS 2023.
27) METR. (2026). Time Horizon 1.1: Updated measurements of AI task-completion capability. METR technical update.
28) Mialon, G., Fourrier, C., Swift, C., Wolf, T., LeCun, Y., & Scialom, T. (2023). GAIA: A benchmark for general AI assistants. arXiv:2311.12983.
29) Packer, C., Wooders, S., Lin, K., Fang, V., Patil, S. G., Stoica, I., & Gonzalez, J. E. (2024). MemGPT: Towards LLMs as operating systems. arXiv:2310.08560.
30) Park, J. S., O'Brien, J. C., Cai, C. J., Morris, M. R., Liang, P., & Bernstein, M. S. (2023). Generative agents: Interactive simulacra of human behavior. Proceedings of UIST 2023.
31) Patil, S. G., Zhang, T., Wang, X., & Gonzalez, J. E. (2023). Gorilla: Large language model connected with massive APIs. arXiv:2305.15334.
32) Qian, C., Liu, W., Liu, H., Chen, N., Dang, Y., Li, J., Yang, C., Chen, W., Su, Y., Cong, X., Xu, J., Li, D., Liu, Z., & Sun, M. (2024). ChatDev: Communicative agents for software development. Proceedings of ACL 2024.
33) Qin, Y., Liang, S., Ye, Y., Zhu, K., Yan, L., Lu, Y., Lin, Y., Cong, X., Tang, X., Qian, B., Zhao, S., Hong, L., Tian, R., Xie, R., Zhou, J., Gerstein, M., Li, D., Liu, Z., & Sun, M. (2023). ToolLLM: Facilitating large language models to master 16000+ real-world APIs. arXiv:2307.16789.
34) Rasmussen, P., Paliychuk, P., Beauvais, T., Ryan, J., & Chalef, D. (2025). Zep: A temporal knowledge graph architecture for agent memory. arXiv:2501.13956.
35) Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., & Scialom, T. (2023). Toolformer: Language models can teach themselves to use tools. Proceedings of NeurIPS 2023.
36) Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., & Yao, S. (2023). Reflexion: Language agents with verbal reinforcement learning. Proceedings of NeurIPS 2023.
37) Surapaneni, R., Jha, M., Vakoc, M., & Segal, T. (2025). A2A: A new era of agent interoperability. Google Developers Blog.
38) Trivedi, H., Khot, T., Hartmann, M., Manku, R., Dong, V., Li, E., Gupta, S., Sabharwal, A., & Balasubramanian, N. (2024). AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. Proceedings of ACL 2024.
39) Wang, G., Xie, Y., Jiang, Y., Mandlekar, A., Xiao, C., Zhu, Y., Fan, L., & Anandkumar, A. (2023). Voyager: An open-ended embodied agent with large language models. arXiv:2305.16291.
40) Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., Tang, J., Chen, X., Lin, Y., Zhao, W. X., Wei, Z., & Wen, J. (2024). A survey on large language model based autonomous agents. Frontiers of Computer Science, 18(6), 186345.
41) Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. Proceedings of NeurIPS 2022.
42) Wu, Q., Bansal, G., Zhang, J., Wu, Y., Zhang, S., Zhu, E., Li, B., Jiang, L., Zhang, X., & Wang, C. (2024). AutoGen: Enabling next-gen LLM applications via multi-agent conversation. Proceedings of COLM 2024.
43) Xi, Z., Chen, W., Guo, X., He, W., Ding, Y., Hong, B., Zhang, M., Wang, J., Jin, S., Zhou, E., Zheng, R., Fan, X., Wang, X., Xiong, L., Zhou, Y., Wang, W., Jiang, C., Zou, Y., Liu, X., … Gui, T. (2024). The rise and potential of large language model based agents: A survey. arXiv:2309.07864.
44) Xu, W., Liang, Z., Mei, K., Gao, H., Tan, J., & Zhang, Y. (2025). A-Mem: Agentic memory for LLM agents. arXiv:2502.12110.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

45) Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023a). ReAct: Synergizing reasoning and acting in language models. Proceedings of ICLR 2023.
46) Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., & Narasimhan, K. (2023b). Tree of Thoughts: Deliberate problem solving with large language models. Proceedings of NeurIPS 2023.
47) Zhong, W., Guo, L., Gao, Q., Ye, H., & Wang, Y. (2024). MemoryBank: Enhancing large language models with long-term memory. Proceedings of AAAI 2024.
48) Zhou, S., Xu, F. F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., Cheng, X., Ou, T., Bisk, Y., Fried, D., Alon, U., & Neubig, G. (2024). WebArena: A realistic web environment for building autonomous agents. Proceedings of ICLR 2024.
49) Ali, Y., Ayaz, M., Baig, U., & Awais, M. (2025). A novel analytical framework for handling massive distributed energy participants using Prosumers Web. 2025 International Conference on Engineering & Computing Technologies (ICECT).
50) Ali, Y., Ayaz, M., Khan, K., Biswas, A. K., & Baig, U. (2025). Prosumer Web City: A novel energy market framework for enabling dynamic bidding and scalable integration of distributed energy participants. Scientific Reports, 16(1), 1080.
51) Arsalan, M., Ayaz, M., Ali, Y., & Baig, U. (2025). AI-enabled energy load forecasting for smart grid management. International Journal of Scientific Research and Engineering Development, 8(6).
52) Arsalan, M., Ayaz, M., Ali, Y., & Baig, U. (2025). Integrating AI-driven decision systems in electrical engineering project management. International Journal of Scientific Research and Engineering Development, 8.
53) Arsalan, M., Ayaz, M., Ali, Y., & Baig, U. (2025). Secure and scalable cloud architectures for connected vehicle ecosystems. International Journal of Scientific Research and Engineering Development, 8.
54) Awais, M., & Ayaz, M. (2026). Artificial intelligence for strengthening national cybersecurity: Advanced threat detection and automated response systems. International Journal of Engineering Technology Research & Management, 10(08).
55) Ayaz, M., Baig, D.-e.-Z., Hur Rizvi, S. M., Alharbi, S. S., Iqbal, S., & Shafiullah, M. (2025). Automatic generation control optimization for power system resilience under real world load variations using genetic algorithm. Scientific Reports, 15(1), 20857.
56) Ayaz, M., Baig, U., & Awais, M. (2025). Enhancing power grid resilience with adaptive ANN-based voltage control strategies. 2025 International Conference on Emerging Technologies in Electronics.
57) Ayaz, M., Baig, U., Dawood, M., & Awais, M. (2025). Optimizing voltage control in power systems: An artificial neural network approach for improved performance and stability. 2025 International Conference on Emerging Power Technologies (ICEPT), 1–6.
58) Ayaz, M., Mehmood, F., Ali, Y., Ahmad, U., Baig, U., & Awais, M. (n.d.). Resilient AI-driven adaptive neuro-fuzzy inference system control of boost converters.
59) Ayaz, M., Rizvi, S. M. H., & Akbar, M. (2023). Dynamic power factor correction in industrial systems: An automated capacitor bank control approach. 2023 2nd International Conference on Emerging Trends in Electrical, Control and Telecommunication Engineering.
60) Ayaz, M., Rizvi, S. M. H., & Akbar, M. (2024). Conservation voltage reduction impact investigation for personal computing devices using experimental measurements and computation performance metrics. Metrology, 4(1), 24–45.
61) Ayaz, M., Rizvi, S. M. H., & Akbar, M. (2024). Probabilistic CVR assessment in distribution networks using synthetic consumption database of household appliances. Arabian Journal for Science and Engineering, 49(12), 16889–16901.
62) Ayaz, M., Rizvi, S. M. H., Ejaz, S., Haq, M. A. U., & Rathi, L. (2024). Advanced power system optimization: Evaluation of automatic generation control techniques in two-area networks. 2024 21st International Bhurban Conference on Applied Sciences and Technology (IBCAST).
63) Ayaz, M., Rizvi, S. M. H., Rathi, L., Haq, M. A. U., & Ejaz, S. (2024). Machine learning based robust control solutions for buck converters: ANN and fuzzy logic methods. 2024 21st International Bhurban Conference on Applied Sciences and Technology (IBCAST).
64) Ayaz, M., Shah, S. W. H., Asif, H. B., & Haq, M. A. U. (2024). Integrated distributed generation for practical implementation of urban lighting using hybrid renewable energy. 2024 4th International Conference on Digital Futures and Transformative Technologies.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

65) Baig, U., Ayaz, M., & Baig, D.-e.-Z. (2025). Advanced time-frequency analysis for fault detection and classification in power transmission systems using wavelet transform. Proceedings of the International Conference on Emerging Power Technologies (ICEPT), 1–6.
66) Bano, F., & Ayaz, M. (2025). Resilient ANFIS framework for robust boost converter stability in real world conditions. IEEE Access.
67) Bano, F., Ayaz, M., Baig, D.-e.-Z., & Rizvi, S. M. H. (2024). Intelligent control algorithms for enhanced frequency stability in single and interconnected power systems. Electronics, 13(21), 4219.
68) Ejaz, S., Ayaz, M., Dawood, M., Haq, M. A. U., & Rathi, L. (2024). Impact of load variations on the stability of GA-optimized PID controllers in isolated power systems. 2024 19th International Conference on Emerging Technologies (ICET), 1–6.
69) Faheem, M., Awais, M., Iqbal, A., & Zia, H. (2025). Adaptive AI-driven cyber threat detection system for US critical infrastructure protection. World Journal of Advanced Research and Reviews, 26(3), 2282–2291.
70) Faheem, M., Awais, M., Iqbal, A., & Zia, H. (2025). AI-augmented cybersecurity for smart grids in the United States. World Journal of Advanced Research and Reviews, 7(1), 713–726.
71) Faheem, M., Awais, M., Iqbal, A., & Zia, H. (2025). Enhancing IT incident management with natural language processing and predictive analytics. International Journal of Science and Research Archive, 15(3), 224–237.
72) Faheem, M., Awais, M., Iqbal, A., & Zia, H. (2025). Neural infrastructure: A federated AI framework for predictive resilience in US transportation systems.
73) Haq, M. A. U., Ayaz, M., Asif, H. B., & Shah, S. W. H. (2024). Integrating machine learning with variable frequency drive for enhanced speed control in induction motors. 2024 International Conference on Engineering & Computing Technologies (ICECT).
74) Rathi, L., Ayaz, M., Ejaz, S., Dawood, M., & Haq, M. A. U. (2024).Evaluation of predictive and heuristic control techniques for buck converters using MPC, PID, and fuzzy logic. 2024 26th International Multi-Topic Conference (INMIC), 1–6.
75) Rizvi, S. M. H., Satti, S. G., & Ayaz, M. (2024). Supervised learning framework for identification of inverter control mode and estimation of inverter-based-resource & load parameters. IEEE Transactions on Industry Applications, 61(2), 2846–2857.

