Bol. Soc. Paran. Mat.
©SPM – E-ISSN-2175-1188
SPM: www.spm.uem.br/bspm

(3s.) **v. 2026 (44) 5** : 1–21.
ISSN-0037-8712
doi:10.5269/bspm.82008

Tool-Augmented Agentic AI: A Survey on Composition, Selection and Integration *

Abdullah Jawwad Yousafi †, Nabiha Fatima and Mehmet Dik

ABSTRACT: Agentic AI systems enhanced by tool-use represent a shift in how large language models are designed and deployed. They are able to execute tasks that would otherwise be beyond the capabilities of standalone systems. They do this through multi-step reasoning, dynamic interaction with external resources, and by depending on external tools, ranging from APIs to simulators. In this survey, we dive into the rapidly expanding body of research surrounding tool-augmented Agentic AI. This survey examines recent research on tool-augmented agentic AI with a focus on how tools are composed, selected, and integrated into agent architectures. It provides a unifying taxonomy and actionable guidance for navigating this complex landscape. The goal is not to be exhaustive, but to synthesize our findings and relevant frameworks to provide structured guidance for researchers, practitioners and policymakers working on robust, safe and scalable Agentic AI.

Keywords: Agentic AI, Tool-Augmented AI, autonomous agents, large language models, intelligent systems, engineering decision support, computational intelligence, systematic review, architecture design.

Contents

1 Introduction **2**
1.1 Background and Motivation . . . 2
1.2 Importance of Tool Use . . . 3
1.3 Definition and Clarification . . . 3
1.4 Scope and Assumptions . . . 4
1.5 Methodology . . . 5
1.6 Limitations . . . 5
1.7 Objectives and Contributions . . . 5
1.8 Paper Organization . . . 6

2 Foundation of Tool Augmentation **6**
2.1 Evolution of Tool Integration in Agentic AI . . . 7
2.2 Taxonomy of External Tools in Agentic Systems . . . 7

3 Architectural Approaches and Methodologies in Tool Augmentation **8**
3.1 Composition . . . 8
3.2 Selection . . . 11
3.3 Integration . . . 13

4 Industry and Application Examples **15**
4.1 Research Applications . . . 15
4.2 Healthcare Applications . . . 15
4.3 Environment and Education . . . 16
4.4 Business Applications . . . 16

5 Challenges and Future Directions **16**
5.1 Current Limitations . . . 16
5.2 Future Areas of Advancement . . . 17

6 Conclusion **17**

* This paper was presented at the 9th International Conference of Mathematical Sciences (ICMS 2025), September 3–7, 2025, Maltepe University, Istanbul, Turkey.

† Corresponding author.

2020 *Mathematics Subject Classification:* 68T05, 68T42, 68T01.
Submitted February 25, 2026. Published April 09, 2026

1. Introduction

1.1. Background and Motivation

The breakneck speed of advancements in AI has caused Agentic AI to come to the forefront of this globally relevant field [1,11]. As AI systems became more integrated in real-world applications, their advancement and capability to interact with tools has become essential rather than optional [66]. Consequently, Agentic AI systems have seen an intensified research focus, with major labs and companies (eg., OpenAI, DeepMind, Google, Oracle, Nvidia) and industry consortia focusing on transformative applications emerging, for example, in scientific discovery (e.g., literature synthesis and hypothesis testing), adaptive supply chain management, and diagnostics in healthcare [57].

Agentic AIs are at the forefront of the paradigm shift from generative AI systems, demonstrating "agency" in complex situations, as opposed to being specialized for singular, basic tasks and performing better on benchmarks such as the ARC test [13]. They are defined as autonomous architectures capable of executing complex, multistep workflows through multimodal reasoning and self-correcting behaviour. The development of Agentic AI relies on the core functioning principles of goal-orientation, self governance and autonomous operation with low human interference, memory management and context inclusion/adaptation. They additionally possess the ability to interact with their environment, possess adaptability to unprecedented scenarios and errors, to plan and reason as well as perform multilayered thinking. They can perform multi-step tasks unable to be done by normal generative AI such as market research and analysis, travel planning, experimentation and report writing [60].

The integration of tools into AI systems draws its origins to the landmark Toolformer paper [62], which demonstrates how language models may self-teach tool access for enhanced capabilities. Upon this foundation came agentic frameworks like AutoGen [82] and LangGraph emerged in 2024, alongside tool-based agents such as EasyTool [86] and GenTopia [83]. Yet, despite substantial progress, research on agentic tool use remains fragmented, making it difficult to identify unifying principles or compare approaches systematically. To address this gap, we structure the survey around three recurring design concerns in tool-augmented agentic systems. Firstly, how tools are assembled. Secondly, how agents decide which tools to invoke. And thirdly, how external capabilities are incorporated into the overall architecture. Separating these concerns makes it easier to compare existing approaches and to understand why systems with similar components can exhibit different behaviours in practice.

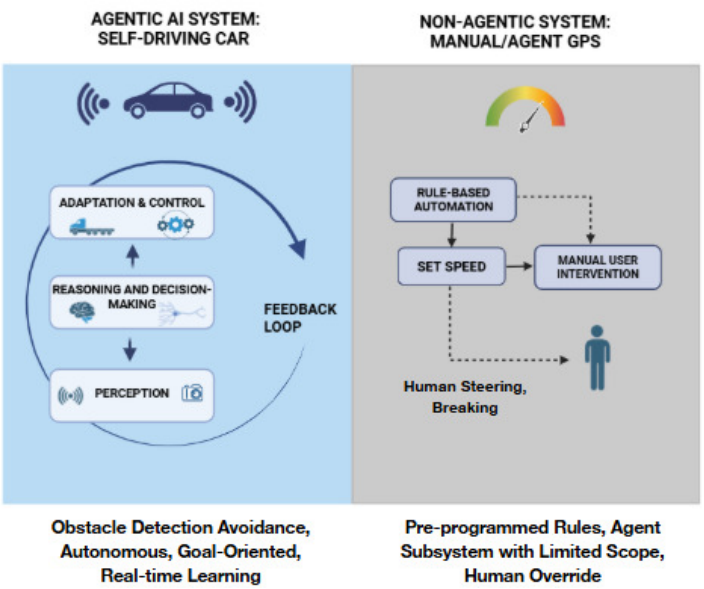

Figure 1: Illustration of the importance of tool use in agentic AI. Tool augmentation enables access to external computation, real-time data, and reliable execution mechanisms beyond the internal capabilities of standalone LLMs.

1.2. Importance of Tool Use

LLMs, and by extension Agentic AIs, by themselves have shown limited ability in tasks such as mathematical reasoning [35], real-time updates, and external interaction. Therefore, tool-based execution is a core capability that allows access to APIs and software such as Wolfram Alpha, as well as enabling real-world execution mechanisms such as updating Microsoft Excel sheets and Bing web searches [37,54]. The breakthrough paper, Google's "Toolformer" [62] demonstrated a significant performance gain on mathematical reasoning benchmarks when granted calculator access by training the LLM to embed function calls within parts of the text. Furthermore, by accessing these external tools and APIs, AIs are able to execute the retrieval of real-time data, run simulations, perform efficient searches and improve their overall task fulfillment and decision-making capabilities. Modern AI web search abilities have enabled it to extract real-time data to help users make informed decisions as well as act like a much more convenient version of google search. Indeed, google search now includes an "AI Overview" using its LLM Gemini as an adaptation to this. Figure 1 shows an example of an Agentic and non-Agentic system through a self-driving car system. The AI system in the first enables object-detection, avoidance and real-time learning to adapt as it drives, whereas the second stays manual and requires extensive human input, demonstrating the practical superiority of Agentic systems which are given access to external systems.

1.3. Definition and Clarification

In order to avoid conceptual fuzziness that has emerged in the modern AI boom, alongside acknowledging the large number of terms in this field, we must first understand the different topics within this field.

Table 1: Key terms and definitions used in this survey.

Term	Definition
Multi-Agent Systems	Architectures where ≥ 2 agents collaborate or interact [21].
Agentic AI	Autonomous architectures capable of executing complex, multistep workflows through multimodal reasoning and self-correcting behavior.
AI Agents	Specialized autonomous systems that perform narrow, single-domain tasks.
Tool Augmentation	The ability to interact with external systems and computational resources to fulfill tasks.
Orchestration	Stateful sequencing of tools to achieve complex tasks.

In this paper, we define AI Agents as separate from Agentic AI, due to fundamental architectural and performance differences [60], however we realize that both are essential to discuss in this paper seeing the extensive use of AI Agents solving. AI Agents may function as customer service chatbots, virtual assistants and general-purpose bots within Agentic AI. AI Agents are autonomous agents that perform specialized, single tasks which perform, learn and adapt within their specific domain and are usually standalone systems that excel in handling well-defined tasks. In contrast, Agentic systems consist of multiple AI agents that collaborate to achieve more complex, multi-tasked phenomena. Therefore, they differ in architecture, collaboration and the scope of their problem prioritization or report summarization, but their importance within tool augmentation in Agentic AI systems is not to be ignored, therefore they are mentioned in this paper sparingly. A formalized list of terms we use in this paper that are useful and their subsequent definitions are presented in Table 1.

1.4. Scope and Assumptions

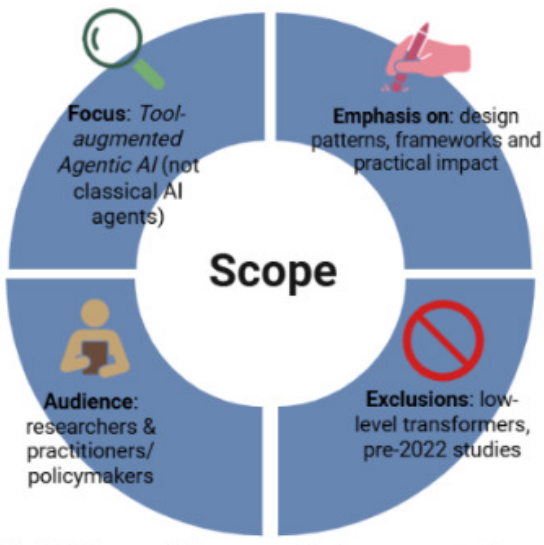

Figure 2: Layout of the scope of this survey.

The scope of this paper focuses primarily on Tool-Augmented Agentic AI systems, emphasizing their architectural frameworks, decision-making processes, and tool integrations that enhance their functionality. Our aim is to make this knowledge more forwardly accessible to academics and AI practitioners and inform Agentic AI tool developers in businesses, research and consumer domains. We assume readers have a fundamental understanding of deep learning, AI architectures, reasoning mechanisms and generative AI concepts as well as some knowledge of the literature.

We will focus on covering tool types, architectures, selection mechanisms, execution concerns through an analysis of a variety of digital tools to learn more about the titular topic. Our interest will lie more towards Agentic AI rather than AI Agents, due to the former being a more recent innovation and more applicable in real world business workflows and relevant for future AI research. Transformers and other in-depth systems will not be discussed owing to our primary focus on direct components that modulate decision-making and tool selection.

1.5. Methodology

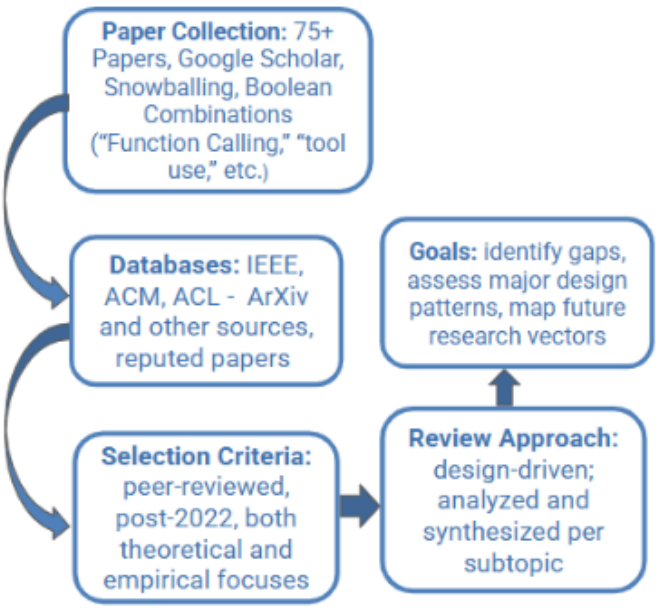

Figure 3: Overview of our literature review methodology, including search strategies, filtering criteria, and thematic synthesis.

Our paper review strategy was rooted in an established taxonomy of relevant concepts that stemmed from previous works in the domain. Initial search terms included "Agentic AI," "Classical Decision Theory," "function calling," "API selection," "tool use," and related terminology. These enabled the identification of key papers via snowballing, situating our work in the the context of existing literature while highlighting research gaps.

Boolean query combinations such as "LLM agents" AND "tool use" OR "decision theory" were employed to ensure comprehensive coverage of niche intersections. Each subsection of this survey corresponds to a separate literature stack, independently analyzed to extract methodological approaches and emerging design principles.

We exclude pre-2022 studies due to the paradigm shift brought by foundation models such as GPT-3, theoretical papers lacking empirical support, and non-peer-reviewed sources or outlets that do not maintain citation standards.

1.6. Limitations

The rapid advancement and near-exponential growth of this field means that readers must be aware of further breakthroughs and discoveries beyond the point this paper is written in 2025, those capabilities will not be covered here although we have specifically and comprehensively covered everything thus far. Additionally, even though some ethical considerations and societal and governance issues are discussed here owing to their relevance, that remains outside the scope of this article as our primary concern is to focus on the technical focuses and considerations of the topic of Agentic AI.

1.7. Objectives and Contributions

This paper aims to deliver a systematic survey of tool use in Agentic AI. We strive to fill in a noticeable gap in the current literature by offering a detailed systematic review of the terminologies and strategies in this sub-field. Moreover, we aim to further contribute to the understanding of AI, provide taxonomy to selection paradigms, analyse architectural trade-offs (between accuracy, efficiency, implementation,

etc) and identify and inform future research areas that may require work. We will document emergent challenges in this field. Our work aims to serve as an informer for researchers aiming to bridge gaps in this field and practitioners aiming to design architectures for deployment.

1.8. Paper Organization

The remainder of this paper is organized as follows:

- **Section 2** introduces the background and foundational concepts necessary for understanding tool selection within the broader context of artificial intelligence.
- **Section 3** examines the structure and architecture of agentic systems, with a focus on tool composition, selection, and integration.
- **Section 4** describes practical and industrial applications of tool-augmented agentic AI across various domains.
- **Section 5** discusses current challenges and limitations in this field and outlines potential directions for future research.
- **Section 6** concludes the paper by summarizing key findings and reflecting on the implications of the preceding analysis.

2. Foundation of Tool Augmentation

Despite their incredible potential, current standalone large language models (LLMs) face inherent limitations that constrain their ability to become autonomous, reliable, general-purpose agents. The most serious concerns are fixed context windows, finite token limits, and their lack of up-to-date, domain-specific knowledge [75,62]. External tools offer a solution to these limitations by providing deterministic, task-specific engines and computations that lie beyond the reasoning abilities of simple LLMs. Specialized engines like SymPy and domain-specific LLMs (e.g., BioBERT) offload computation-heavy or specialized tasks that are impractical for general-purpose models to compute internally [15,43,36]. The end result of tool use is LLMs treating tool access as a functional necessity for both reasoning and action [67].

In this section, we trace Agentic AI's evolution through history. We take a look at its core concepts and properties as they develop through history.

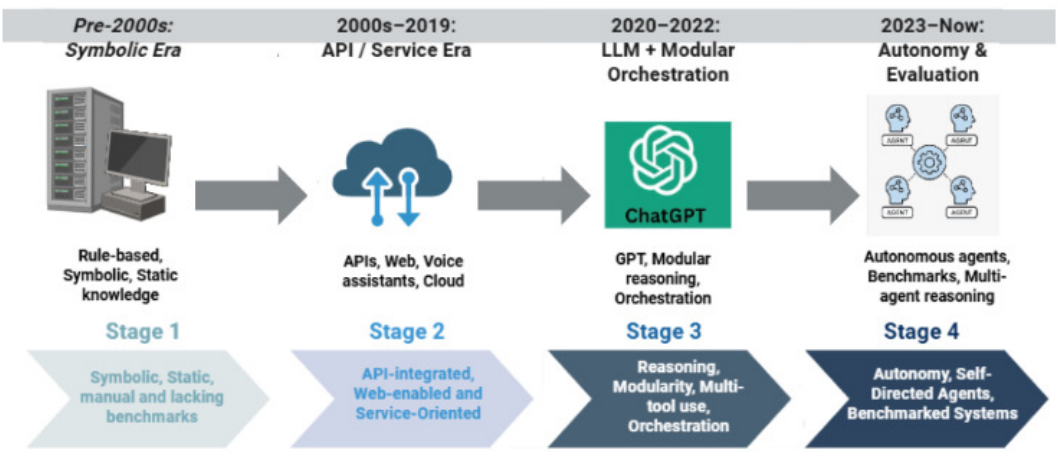

Figure 4: Historical evolution of tool integration in AI, spanning symbolic rule-based systems, web API access, plugin architectures, and modern tool-augmented LLM frameworks.

2.1. Evolution of Tool Integration in Agentic AI

The Symbolic Era (Pre-2000s). Early AI systems such as MYCIN [68] and DENDRAL [8] used static knowledge bases and remote procedural calls (RPCs) as early attempts to integrate external tools into AI. These systems relied on explicit rules and rigid, predefined logic, which limited their adaptability and ability to handle novel situations [29,59]. In the early 2000s, the emergence of Application Programming Interfaces (APIs), Service-Oriented Architectures (SOAs), and RESTful protocols [22] enabled basic external interactions such as retrieving real-time data and executing transactions. Early voice assistants (e.g., Siri, Alexa) incorporated limited web search functionality, but remained inflexible and highly constrained [27].

From Web Search to Self-Supervised Learning. The introduction of large language models such as GPT-3 and GPT-4 marked a turning point in tool use [7]. These models enabled more flexible forms of tool orchestration, including plugin-based systems in which external APIs, such as Wolfram Alpha, were invoked for symbolic computation. Delegating execution and retrieval to external systems reduced, but did not eliminate, limitations related to hallucinations and constrained context windows [62,75]. This transition reframed tool use as a central architectural concern rather than a peripheral enhancement.

Evolution of Tool Augmentation. The web search access provided by WebGPT [47] introduced the first flexible web search mechanism. It did this through integrating human-supervised reinforcement learning and command line interface interaction. This enabled LLMs to access up-to-date information, reduce factual errors, and provide verified, cited information.

Subsequent work moved toward modular system design. DeepMind's MRKL framework framed the language model as a coordinator to route subtasks towards specialized modules, influencing later approaches to tool use [32]. Google's LaMDA adopted similar principles in an industrial setting by embedding API calls directly within its dialogue system [72].

Later models focused on learning when tools should be invoked. ToolFormer [62] introduced a self-supervised mechanism that allowed models to decide whether to rely on internal knowledge or call external tools during generation. TALM extended this direction by scaling tool-use through supervised fine-tuning, treating tools as a reusable execution layer rather than a rigid process [51]. Reliability in tool invocation was addressed more explicitly by Gorilla, which leveraged abstract syntax tree (AST) validation to cut down on errors during tool use [53].

Evolution of Reasoning and Autonomous Execution. Early AI systems relied primarily on rule-based logic, which proved insufficient for layered agentic behavior. ReAct combined chain-of-thought reasoning with tool execution, enabling iterative planning and refinement [85]. Subsequent systems such as ART extended multi-step reasoning by reusing prior solutions and supporting execution pauses, which proved effective for frozen language models [50]. Open-source frameworks such as AutoGPT and BabyAGI further pushed autonomy by introducing recursive task decomposition, memory management, and iterative execution loops [46]. These developments exposed the growing need for robust orchestration mechanisms to manage increasingly complex agent behavior.

Evolution of Tool Evaluation. As agent autonomy increased, evaluation methodologies evolved beyond static rule checks. Benchmarks such as the Berkeley Function-Calling Leaderboard (BCFL) and APIBench introduced structured metrics for assessing correctness and efficiency in API invocation [53]. More recent simulators, including WebArena and AgentArena, provided interactive environments for evaluating agent behavior under realistic conditions [87,30]. The emergence of benchmarks such as AgentBench and τ-Bench reflects a shift toward evaluating multi-tool, multi-step reasoning under ambiguous and dynamic inputs [38,84].

2.2. Taxonomy of External Tools in Agentic Systems

The definition of tool includes any computational engine or software external to the agent's architecture that deterministically turns inputs into task-specific outputs. Agentic reasoning works differently compared to this, which enables actions beyond normal LLM abilities and helps prevent issues such as hallucination [15,75]. Distinct groups of tools can be integrated into frameworks and agents. The definition of tools encompasses:

- **APIs** (e.g., Google Search, Stripe payments): Application Programming Interfaces provide structured access to external services via predefined protocols. They reflect a transition from early RPC-style integrations to web APIs and now play a key role in LLM integration processes [23,4,19]. Examples include search APIs for real-time retrieval and payment APIs for executing financial transactions, both of which extend LLM capabilities.

- **Models** (specialized LLMs, classifiers, simulators): External models augment LLM capabilities in areas requiring specialized computation or representations. These include SymPy for symbolic solutions [43], vision models such as CLIP for image analysis [58], and domain-specific LLMs like BioBERT for biomedical text mining [36]. Their value lies in higher accuracy on niche tasks without overburdening the base model. For example, calculator-style tool access can improve performance on math and science problems [15].

- **Data processors** (Pandas for CSV, OpenCV for images): These perform structured transformations on large datasets that standard LLMs struggle with due to context limits. Examples include Pandas for tabular operations [42], OpenCV for image processing [5], and LibROSA for audio preprocessing [41]. Their essentiality stems from handling data volumes and operations that are otherwise incompatible with LLM-only processing [44].

- **Knowledge repositories** (Wolfram Alpha, PubMed crawlers): Repositories provide scaled information access through structured databases and search systems. These range from scientific literature access such as PubMed [39] to curated computational knowledge bases used for math and science retrieval. Unlike traditional search systems, agents can autonomously construct queries and prompts to retrieve task-relevant data [60].

- **Simulations and virtual environments** (PyBullet, NetLogo, CARLA) [14,74,17]: Simulated environments enable agents to test hypotheses and strategies in controlled settings. They range from lightweight frameworks such as OpenAI Gym [6] to higher-fidelity simulators that support embodied and sequential decision-making. These tools enable safe failure, iterative learning, and transfer from synthetic to real-world contexts.

- **Orchestration frameworks** (Temporal, AutoGPT): These meta-architectures structure how agents integrate tools, memory, and external services into pipelines and working services. Frameworks vary, but generally provide the stepping stones to scale LLMs from single-shot prompts to autonomous agents capable of multi-step execution with tool usage. They support tool-chaining, memory management, and reasoning across systems, while abstracting low-level orchestration logic so the agent can focus on task objectives [60].

3. Architectural Approaches and Methodologies in Tool Augmentation

Architectural choices are becoming increasingly important in determining how tools work reliably outside controlled settings. Decisions about how tools are composed, selected, and executed often matter more than anything else. To understand and compare these evolving systems, we segment this section into these four design pattern concerns, each addressing a distinct axis of capability.

3.1. Composition

This section outlines the architectural elements and interaction patterns that constitute reasoning, control, agents, tools, communication, and state such that complex multi-step tasks are executed reliably. A proper system typically separates reasoning from execution to ensure reliable and effective tool use [85].

The primary components of the system are usually as follows:

- **Core reasoning engine**: The primary natural language model responsible for intent understanding, plan generation, and synthesis. It converts goals into executable plans via chain-of-thought (CoT) prompting and serves as the "brain" of the system [81]. It may also produce and interpret structured outputs (e.g., JSON) for tool calls.

- **Agents**: Role-based software actors with constraints, capabilities, and interfaces. Agents may be specialized subprocesses or wrappers around APIs, and can communicate to delegate subtasks or exchange intermediate results.

- **Tool Registry**: A list of available tools with entries including the name, purpose, input schema, output schema, and execution context. The metadata, alongside example use cases, is what helps the system use tools effectively. Moreover, agentic systems typically consult the registry when deciding which tools to use. The registry usually includes:

  - Tool Name: A unique identifier for the tool.
  - Purpose/Functionality: A clear, concise explanation of what the tool achieves and when it is applicable.
  - Input Parameters: Specification of required and optional inputs, including parameter names, data types, and expected formats.
  - Output Specification: Defines the structure and semantics of returned results.
  - Potential Side Effects: Describes persistent or external changes triggered during execution.
  - Error Conditions: Defines common failure modes and/or expected errors.

- **Tool Metadata**: An "instruction manual" describing functionality, example executions, and tool constraints to enable reliable use. It can be provided in-context or injected via other mechanisms. Tool names, descriptions, I/O schemas, usage examples, error codes, and cost/latency profiles are common components. Metadata is essential for tool selection, error recovery, and optimization [28,78].

- **Memory and State Tracking**: Memory components range from centralized stores to local caches; design choices affect latency, privacy, and coordination.

- **Orchestrator / Coordinator**: Decides which agent/tool executes next and enforces execution policies, using modes including:

  - Hub-and-spoke: Single planner delegating to specialists; suited for auditability and deterministic governance.
  - Peer-to-peer: Agents negotiate and act; flexible but harder to control.
  - Pipeline: A fixed sequence for dataflow; simple and deterministic.
  - Hybrid: Planner with peer negotiation via substeps; a common practical compromise.

However, agentic models are based upon some fundamental key components: perception, reasoning, action, learning, and collaboration. Their interaction and placement in the overall architecture are illustrated in Figure 5. In practice, these features are sorted into specialized layers which deal with separate tasks [1]. The layers that deal with tool augmentation within Agentic AI are listed in the following:

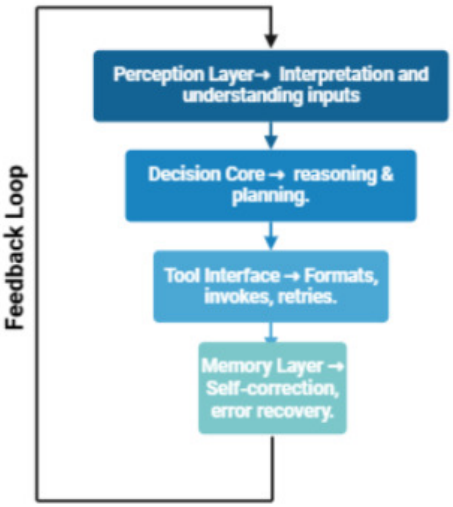

Figure 5: Layered agentic AI architecture, highlighting perception, decision core, tool interface and execution module, memory, and feedback mechanisms that enable iterative refinement.

- **Perception Layer**: Live interaction and real-time data collection, capturing both structured and unstructured inputs.
- **Decision Core**: Hosts reasoning, primarily LLM-driven, enhanced by algorithms or learning strategies that determine tool needs and actions.
- **Tool Interface & Execution Module**: Formats, dispatches, and manages tool interactions, parses results, and attempts retries on failure.
- **Memory Layer**: Houses semantic, procedural, episodic, and contextual memory; interacts with the rest of the system through read/write calls.
- **Feedback Loop**: Iterates self-correction and performance refinement using error signals, evaluation metrics, and tool outputs.

The agent's structural composition details how it is assembled, whereas tool composition describes its tool makeup and integration, such as metadata and internal code. Its structural composition follows recurring patterns that many systems share, as summarized in Table 2.

Table 2: Common structural composition patterns in agentic systems.

Pattern	Procedure	Benefit	Tradeoff
Planner-Based	Routing	Centralized, simple	Rigid, bottlenecked
Graph-Structured	Graph-based flows	Flexible, visual representation	Higher complexity
ReAct-Style	Reason + action looping	Effective for uncertain tasks	Debugging difficulty, oscillations
Multi-agent Hierarchy	Role delegation	Modular collaboration	Overhead and latency in scale

Structural composition defines the backbones of Agentic AI systems. They determine how problems are understood, actions are sequenced, and tools are orchestrated. Whether implicit or explicit, composition patterns affect transparency, reusability, and an agent's capacity for robust behavior. At the

time of writing, there is no single architectural design that all Agentic AIs use. Systems follow heavily experimental design patterns that share a few key similarities to enhance efficacy and performance.

Tool composition, on the other hand, refers to the logic by which external capabilities integrate into agentic reasoning and action loops. The agent's ability to route to, sequence, and integrate tools defines its modularity, utilization, and generalization potential. At the most basic level, some key terms in tool composition involve the following:

- **Tool Affordances**: Affordances is a term borrowed from ecological psychology, referring to the capabilities of an external tool as cognitively understood [70]. In agentic systems, affordances refer to the range of actions a tool can perform through invocation, which is crucial given the system's limited innate awareness of what capabilities it can access [31].

- **Tool Invocation and Routing**: Tool invocation refers to the process in which fuzzy inputs lead agentic systems to use a tool to accomplish a subtask via structured interfaces such as API calls or executable code [62]. Routing is the complementary process that selects the most suitable tool using techniques such as affordance-based heuristics, fuzzy matching, or learned decision policies [85].

- **Result Integration**: After invocation, agentic systems must ingest tool outputs, interpret their meaning, and incorporate them into the evolving plan or memory state. This includes error detection and context updates so outputs become usable for downstream steps.

- **Tool Chaining**: Multiple tools can be chained into a pipeline, passing outputs from upstream tools into downstream inputs (e.g., search → summarization → code execution). While powerful, chaining increases the risk of error propagation.

Another significant challenge is tool coordination, meaning ensuring tools are invoked in the correct order and failures are caught early to prevent cascades. Architectures vary significantly in robustness: some systems fail outright, whereas modular systems may replan and retry parts of the pipeline. Stronger state management and execution tracing significantly improve reliability in these scenarios [10].

3.2. Selection

Tool selection is a critical component of autonomy and functionality. Without robust and reliable tool selection strategies, Agentic AIs face overuse leading to reduced efficiency, increased costs, and erroneous results.

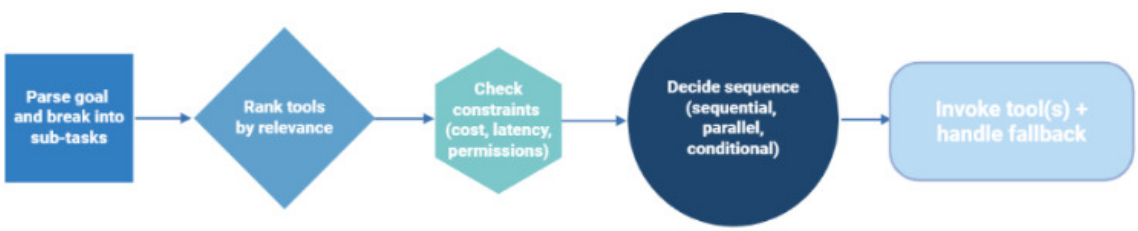

Figure 6: Typical tool workflow in an agentic system: from interpreting user goals, through tool selection and routing, to executing tools and integrating their outputs back into the agent's state and plan.

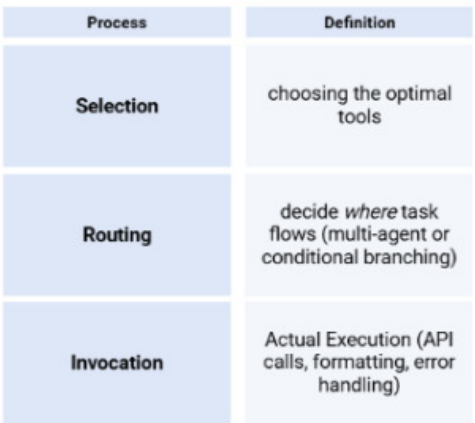

Figure 7: Defines three key processes in tool use: Selection (choosing the best tools), Routing (deciding task flows) and Invocation (executing tasks such as API calls and so forth.)

Within these systems, we distinguish clearly between the sequential stages of tool selection, routing and invocation due to their interchangeable role in tool execution and are depicted in Figure 7. Selection means the decision-making process that identifies which tool from a given set must be selected for a certain task or prompt. Selection occurs before action is taken and is worked out from tool affordances and metadata through the aforementioned processes of heuristics, learning methods, embedding similarities, and LLM-based reasoning. Routing governs distribution of "where" the task should go after decomposition between various modules, agents or tools. It includes conditional branching, agent delegation, or task segmentation. It is not necessarily a part of tool selection, and works majorly in multi-agent systems governed by LLMs. Invocation is the process often used synonymously with function calling that handles how the tool is called and includes formatting the inputs and outputs, calling the API and handling results. It is a low-level act that is critical for robust execution and failures here often cascade unless dealt with by built in safeguards and guardrails.

Tool selection strategies can be generalized into four main types, as shown laid out in Table 3: 1) embedding similarities, 2) learning-based selection, 3) heuristics, or 4) LLM based intelligence.

Table 3: Tool selection strategies and their respective benefits and limitations.

Selection Type	Method	Benefit	Tradeoff
Rule-based	If-then logic	Fast, deterministic execution	Rigid, lacks adaptability
Embedding-based	Semantic similarity matching	Scalable, zero-shot generalization	Limited functional awareness, weak context binding
Learning-based	Supervised training or RLHF	Learns optimal tool use over time	Requires training data, prone to forgetting
LLM-guided	Natural language reasoning for routing	Flexible multi-step planning and adaptation	High latency and computation cost

Learning-based selection typically follows two main approaches. The first is supervised approaches, in which LLMs are fine-tuned on datasets. The second is reinforcement learning with human feedback [48,64] in which agents learn which tools to use via rewards-based functions and human feedback [33]. The systems learn tool preferences from the results of reward functions, previous interactions, and system feedback. Challenges in this method include data scarcity for niche tools and forgetting during updates, alongside brittleness and opaqueness [33]. Heuristic routing involves rule-based prioritization (cost-latency trade-offs, percentage thresholds, etc.) This could include having a fallback for API failures, or retrace steps to avoid error propagation. These work best in resource-constrained environments or

as fallback mechanisms, as they contradict the principles of the dynamic systems that modern day Agentic AIs run on. By far one of the most used methods for tool routing and selection is through LLM intelligence. This is an incredibly flexible and variable method which handles implicit steps by task decomposition into smaller parts. It allows Agentic AI systems to plan out dynamic multi-step tool routings and selection. Embedding based tool selection filters optimize based on semantic similarity based on task descriptions and tool functionalities. Each tool's metadata (name, description, etc) is converted into a vector embedding and the user query is embedded into the same space. The highest scoring tool that surpasses the similarity threshold is invoked. It benefits through zero-shot generalization and computational efficiency with low latency, promoting scalability, however embeddings face issues with nuanced functional differences and parameter blindness as well as contextual insensitivity.

Accumulation of errors within the reasoning process may lead to the amplification of problems thus affecting the quality of results drastically, especially in complex reasoning processes with complex inference chains or multi-step operations. Backtracking and correcting errors without crashing the entire process or decreasing efficacy is a major challenge. Complex selection processes increase dynamic overhead and costs, and limit transparency. These issues pose a significant roadblock in its scalability and widespread adoption.

3.3. Integration

This subsection covers how Agentic AI systems fully incorporate tool calling into their structure and execute, alongside integrate, tool responses to overcome limitations to their capabilities. Execution encompasses the structured process of invocation, whereas integration refers to how tools are interpreted, assimilated and used to update and feed to the agent.

Invocation Pipeline: Execution mainly occurs through specialized invocation pipelines which includes the following stages, also illustrated in Figure 8:

- Input Formalization: Converts decomposed tasks and goals into interpretable tool inputs, which conform to defined schemas, such as API request formats or function call signatures, that have been fed to it beforehand.

- Tool Calling: The fundamental call that dispatches the formatted input through the interface (e.g., APIs) ensuring correct routing and call fidelity.

- Output Capture: Receival of tool responses to the system. These are often messy and unstructured and captured through predefined I/O streams or response handlers.

- Semantic/Output Parsing: Transforms raw outputs into structured and interpretable data that can be appended into the agent's memory and can be used to inform the next immediate actions and possibly be used for information and input for the next actions.

- Feedback loops: Performs assessment of utility, and success, of tool outputs through error detection, planning and continuation criteria and other various checks. This aids with backtracking as well as tool rerouting and replanning when outcomes deviate from expected results.

This invocation pipeline converts fuzzy reasoning and inputs into deterministic tool usage that maintain contextual coherence across calls and global states.

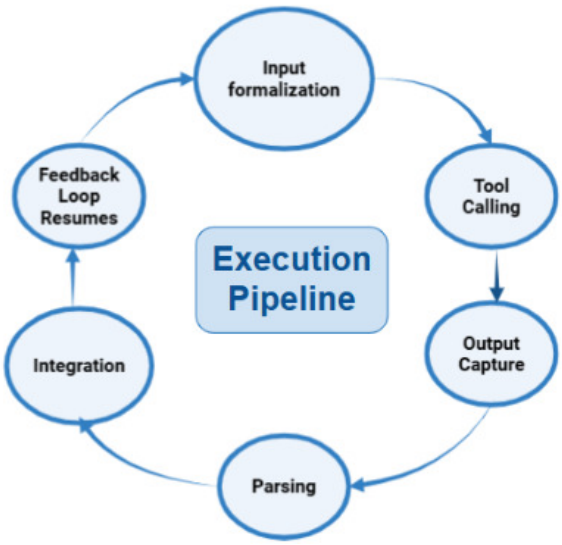

Figure 8: Execution workflow for tool invocation in agentic AI: input formalization, tool calling, output parsing, semantic integration, and feedback-driven replanning.

Tool outputs after execution often arrive as semi structured or unstructured data streams, such as JSON texts, HTML fragments or raw text, leading to rigorous parsing procedures before integration. Modern systems have multi-step parsing processes in which appropriate parsers are used for different formats and types (e.g., XML, YAML, or CSV) and if the text happens to be plaintext, then regex or template matchers are applied to extract key elements from text. Some other frameworks use pydantic or schema validators to ensure smooth workings in which parsed outputs are then matched against expected tool schemas in which missing or malformed fields are autocorrected or handled depending on the type of error. The meaning of the parsed output is then processed by "semantic grounding" through which it is embedded within an agent's current plan, task state, memory modules (either episodic or semantic) or its follow-up decision-making logic. Through this process, the tool working is effectively integrated and blended into tool reasoning and autonomous function.

Table 4: Common memory types in agentic systems and their characteristics.

Memory Type	Function	Benefit	Tradeoff / Example
Semantic	Tool schemas and embeddings	Fast retrieval and scalable access to structured knowledge	Static representation; e.g., recommender systems
Episodic	Execution logs and task history	Enables learning from failures and behavioral adaptation	Storage growth is unbounded; e.g., conversation memory
Procedural	Cached tool chains	Avoids repeated computation in known workflows	Becomes stale or domain-restricted over time
External	Persistent shared storage across agents	Supports collaboration and distributed execution	Network dependency and synchronization overhead

Tool-use is closely intertwined with memory and context handling, with some common memory types being shown in Table 4. As agents interact with dynamic environments and multi-step executions, memory modules become essential not only to serve as repositories but also as decision-making parts and for coherence. One foundational distinction within new models in literature lies between short-term and

long-term memories, an analogy to human cognitive processes. Working memory (short-term) is chosen to hold recent tool outputs, task states and intermediate reasoning steps, often through a scratchpad or token mechanism. Four main memory architectures are used in tool augmentations for things such as overcoming context-window constraints and enabling coherent tool coordination. Semantic memories [34] store tool schemas (APIs, constraints, affordances) in vector-encodings for retrieval during planning, while episodic memory logs tool execution records for failure instances and successes [16] and procedural memory creates caches for optimized tool chaining and reducing redundant computations in familiar tasks. External memories can also be used for instances such as for distributed structures where persistence and global states must be maintained.

Robust executions hang on the dependency of an invocation pipeline, which is the combined process of formatting tool inputs, calling the tool, parsing its output, and integrating the results back into memory. They must also handle retry logic, fallback strategies and rewind certain steps to catch cascading errors. For example ToolLLM [56] defines failure types in its metadata and uses error propagation fallbacks to save itself from errors. Coordination challenges may also persist throughout the process, especially within multi-agent architectures with distributed execution strategies. CrewAI and other orchestration frameworks utilize role abstraction and other collaboration/communication strategies to enable distributed task execution while maintaining global state coherence and avoiding tool conflicts.

Current challenges within integration include issues emerging from ambiguous or fuzzy tool outputs, error propagation from slight misalignments, computational latency and overhead due to inefficiencies and the general process, plan incoherence problems and non-deterministic LLM outputs making robust behaviour difficult to guarantee.

4. Industry and Application Examples

4.1. Research Applications

Literature has shown the ability of LLM-driven tools in aiding research processes. Experimental design, literature reviews, and report writing are examples of such areas. The most widely used LLM-driven agents include, LitSearch [73], ResearchArena [76], SciLitLLM [24], CiteME [9], Deep ResearchAgent [2] and Agent Laboratory [63].

There are apparent limitations when the demands of tool composition and integration become too high, however. For example, agent Laboratory demonstrates high success rates in areas such as data preparation and report writing, but showed a marked decline during literature review. This was due to effective tool selection being in high demand, being used between domain-specific database, citation searches and verification processes. This outcome reflects the difficulty of automating structured literature reviews, a high risk in domains such as the biosciences where high volume of resources are required.

4.2. Healthcare Applications

Agentic AI systems work by automating tools for diagnosis, decision support and patient management. Systems such as Sepsis Watch [65] which continuously analyzes diverse patient variables (EHRs, lab results, and medical imaging) to identify patterns altering immediate sepsis. The device automatically combines multiple systems nearly simultaneously to prove medically useful and an example of how Agentic AI could be potentially used for lifesaving applications. Similarly, PathAI collaborates with pathologists by pre-screening digital slides to highlight suspicious regions, integrating image analysis, pattern recognition, and historical patient data to help outcomes. Administrative systems also benefit from this, as agentic AI automates hospital scheduling, resource allocation, and claims processing which relieves clinicians from unnecessary overheads and enables them to focus on their patients.

However, unlike several other regions, the use of Agentic AI in lifesaving applications cannot afford errors. The United States spends 17.6% of its GDP on healthcare [52], with insurance and jobs in this industry having a major effect on the American economy and the lives of citizens [49]. Furthermore, the ethicality and legality of autonomous systems within this field is disputed, with legal claims and risks running high, all amounting to reasons why Agentic AI tools must run on maximum accuracy in this field. Agentic AI has been shown to reduce diagnostic errors [69], streamline workflows and improve patient

outcomes and aid in simulation scenario design [3]. However, algorithmic bias and need for extensive oversight and expertise remains an issue in this field [26,25].

4.3. Environment and Education

In climate science or environmental conservation and management, Agentic AI demonstrates the capacity to compose and integrate several tools for advanced tasks such as smart climate control [79], environmental monitoring, sophisticated climate modeling [18] and scientific discovery within the field [20,79]. These systems dynamically select between heterogeneous data sources including satellite imagery, sensor networks and simulation models to generate actionable insights and support complex decision-making. However, limitations in scalability and weak error-recovery mechanisms are some challenges that this domain of use exposes.

In educational settings, Agentic AI systems have been used to deliver personalized and adaptive learning experiences [12]. By coordinating tools such as curriculum planners and learner models, these systems can adjust content and pacing to individual students. This has been show to significantly improve engagement and learning outcomes [80]. However, this area shows tradeoffs as well. Benefits are accompanied by concerns related to bias and oversight. Aligning agentic educational systems with pedagogical goals remains a central challenge. Addressing these issues is essential if agentic AI is to be deployed responsibly and realize its full potential in educational contexts.

4.4. Business Applications

In organizational settings, agentic AI systems have demonstrated improved efficiency in tasks such as financial forecasting and customer support with tool usage [61]. Coordinating multiple tools dynamically is a key advantage of these systems. They reduce manual intervention by invoking databases and communication interfaces for longer processes as well. Empirical benefits include improved task completion rates and enhanced decision-making skills as compared to non-agentic approaches.

Yet, scalability remains a persistent constraint, as well as larger organizational sizes, deployment success rates tend to decline. The appeal of agentic systems comes from their ability to improve efficiency, enable partial automation of workflows and shift roles from operational to human-supervised. This effectively opens new avenues for revenue. However, these practices are frequently undermined by brittle tool selection strategies and weak integration protocols in practice. Small execution errors may also propagate through the pipeline when many different tools and tool types are chained together, leading to compounding failures that are difficult to troubleshoot and recover from. Addressing these deployment challenges is therefore necessary for agentic AI access sustained enterprise-wide use.

We can safely analyze that in order to realize the full potential of agentic AI we must overcome its limitations, ranging from challenges in orchestration to robustness. This observation reinforces the purpose of our survey aiding in developing methods for reliable methods for tool composition, selection and seamless integration to develop better, more robust agentic systems.

5. Challenges and Future Directions

This section identifies key challenges faced in the expansion in the field of Tool-Augmented Agentic AI and outlines some possible future vectors for research in this field.

5.1. Current Limitations

Latency problems that already exist with LLMs are enhanced by tool use [45]. For example, GPT web searches take several seconds or more, thus disrupting sequentially running workflows. In order to combat this issue, mitigating overuse and maintaining simplicity and prioritizing effectiveness in tool metadata is critical to combat this issue.

Tool Bias: Models, being trained by data mostly acquired online, exhibit opinions and characteristics from the set and thus may develop problematic biases. This extends to AI tools. Within hiring, lending, healthcare, or policing, agentic tools can amplify biases in datasets. This may lead to systematic discrimination against minorities or marginalized groups such as biased loan denial or racial profiling.

Tool Overuse and Underuse: Determining when and where to rely on external tools versus internal knowledge is a critical challenge in the modern day. It is a non-trivial question of balancing efficiency

with accuracy, and is especially important in agents, which are limited in reliably benchmarking the boundaries of their own knowledge, Around 30 percent of tool invocations are unnecessary. This severely increases computational overhead [55]. A model's overestimation of its own capabilities may lead to the reverse issue and to tool underuse and underperforming. Fine-grained benchmarks, verbal probing and uncertainty calibration techniques become essential to tackle these issues.

Lack of Standardized and Effective Benchmarking Methods: Real world scenarios and theoretical used in benchmarking techniques currently see a gap, although significant progress has been made in this area. This endangers scalability and generalizability. Furthermore, solid, quantitative and comprehensive evaluation methods are currently lacking in this area. Modern approaches that possess significant improvements such as ToolEval do not entirely reflect holistically better standards. In order to comprehensively test efficiency, precision, cost and practicality, new frameworks are needed.

Security and Privacy Challenges: Within Agentic AI, data privacy and security is a key risk. A single call can lead to agents potentially revealing sheer volumes of sensitive information. For example, "Plan a vacation" may entail agents having to check calendars, emails, bank accounts, and personalized mapping apps. Vulnerability is increased through improved memory management. Furthermore, privacy cover ups cannot be ensured for every single tool an agent may call. Emergent or inferred data, such as email histories and more, may be leaked even with privacy protections in place, making traditional consent models obsolete. Key protections include data minimization to load only necessary information, strict sandboxing and granular access control, privacy protection techniques such as federated learning, homomorphic encryption, and differential privacy, and updated regulatory techniques (GDPR, CCPA) tailored for Agentic AI data flows.

Even though these fields are seemingly the most significant barriers to the reliable use of Tool-Augmented Agentic AI today, they serve as the most promising directions for the research of tomorrow. Many of the limitations outlined above can be reframed as opportunities for targeted research and advancement.

5.2. Future Areas of Advancement

Tool Quality: The effort put into creating tools and their quality directly impacts their performance [78]. The quality of data descriptions in registries, metadata, and tool design features alongside examples given are key for proper tool calling. The breadth and accessibility of tools also impacts the range of user queries a model may be able to answer [40]. However, many existing datasets only cover a limited number of tools, and they also acquire tools from different sources such as public APIs and platforms like Hugging Face or OpenAI's plugin list which offer limited coverage, inconsistent formats and insufficient domain diversity [28,67,71].

Multi-Modal Tool Learning: A major area of improvement in current research is tool learning from mediums other than text, including images, audio, 3D and video in order to parse and interpret true user intent. Therefore, including multi-modality in Agentic AI tool use may aid in generating superior responses. Wang et al. [77] introduces MLLM-Tool, which is an excellent example theory put into practice, including multi-modal awareness and interpretation.

Evaluation Benchmarks and Standards: Current tool benchmarks only score models based on the usage of a few tools. For example, Berkeley's Function Calling Leaderboard (BCFL) [53] scores using only four tools at a time, which contrasts practical uses as system accuracy sharply decreases with an increased number of tools. Moreover, significant research can be made into making unified tool benchmarks that are reproducible, rigorous and comprehensive and also consider efficiency, precision, cost and practicality holistically.

6. Conclusion

In this paper, after surveying over 70+ papers, we present a comprehensive survey of Tool-Augmented Agentic AI. We explored the foundations, recent literature, composition, selection, integration, execution challenges, and future directions of research. We established that tool augmentation represents not merely an enhancement, but a fundamental architectural shift that embodies true autonomy, multi-step reasoning and dynamic external interaction. Through tool-augmentation, they overcome their inherent limitations of static knowledge, context window constraints and limited execution capabilities. We revealed their

diverse architectures through our analysis, ranging from planner-based systems to multi-agent hierarchies. We took apart different selection mechanisms, ranging from LLM-guided reasoning, embedding-based similarities, learned-policies, and heuristic routing. We noted their overlying tradeoffs in scalability, accuracy and computational overload. We examined the challenges that tool augmentation faces. This included result parsing, state management, and error recovery. We noted how this field, despite its significant progress, faces substantial challenges.

The future of Tool-Augmented Agentic AI points towards increasingly sophisticated, autonomous systems capable of taking on real-world problems across research, healthcare, business automation, and beyond. Looking ahead, future vectors of research emerge in improved tool quality, robust and comprehensive benchmarking and multi-modal learning and adaptation. By synthesizing the current landscape, identifying core challenges, and outlining critical research vectors, we hope this survey will serve as a thorough and invaluable resource for future researchers, policymakers and developers entering this field, ultimately drawing pathways for future research and deployment of not just powerful, but also reliable, efficient and ethically sound Tool-Augmented Agentic AI systems in the future.

Acknowledgments

This paper was presented at the 9th International Conference of Mathematical Sciences (ICMS 2025), held during September 3–7, 2025, at Maltepe University, Istanbul, Turkey.

References

1. Acharya, D. B., Kuppan, K., and Divya, B., *Agentic ai: autonomous intelligence for complex goals – a comprehensive survey*, IEEE Access, (2025).

2. Baek, J., Jauhar, S. K., Cucerzan, S., and Hwang, S. J., *Researchagent: iterative research idea generation over scientific literature with large language models*, Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the ACL, (2025).

3. Barra, F. L., Rodella, G., Costa, A., Scalogna, A., Carenzo, L., Monzani, A., and Corte, F. D., *From prompt to platform: an agentic AI workflow for healthcare simulation scenario design*, Advances in Simulation, 10(1), 29, (2025).

4. Basu, K., Abdelaziz, I., Chaudhury, S., Dan, S., Crouse, M., Munawar, A., Austel, V., Kumaravel, S., Muthusamy, V., and Kapanipathi, P., *Api-blend: a comprehensive corpora for training and benchmarking api llms*, Proceedings of the 62nd Annual Meeting of the ACL (Volume 1: Long Papers), (2024).

5. Bradski, G., *The opencv library*, Dr. Dobb's Journal: Software Tools for the Professional Programmer, 25(11), 120–123, (2000).

6. Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W., *Openai gym*, arXiv preprint arXiv:1606.01540, (2016).

7. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., and Askell, A., *Language models are few-shot learners*, Advances in Neural Information Processing Systems, 33, 1877–1901, (2020).

8. Buchanan, B. G., and Feigenbaum, E. A., *DENDRAL and Meta-DENDRAL: their applications dimension*, in Readings in Artificial Intelligence, pp. 313–322, Elsevier, (1981).

9. Burton, S., Basil, D. Z., Soboleva, A., and Nesbit, P., *Cite me! perspectives on coercive citation in reviewing*, Journal of Services Marketing, 38(7), 809–815, (2024).

10. Canese, L., Cardarilli, G. C., Di Nunzio, L., Fazzolari, R., Giardino, D., Re, M., and Spanò, S., *Multi-agent reinforcement learning: a review of challenges and applications*, Applied Sciences, 11(11), 4948, (2021).

11. Chan, A., Salganik, R., Markelius, A., Pang, C., Rajkumar, N., Krasheninnikov, D., Langosco, L., He, Z., Duan, Y., and Carroll, M., *Harms from increasingly agentic algorithmic systems*, Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency, (2023).

12. Christopoulos, A., Pellas, N., and Laakso, M.-J., *A learning analytics theoretical framework for STEM education virtual reality applications*, Education Sciences, 10(11), 317, (2020).

13. Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O., *Think you have solved question answering? try arc, the ai2 reasoning challenge*, arXiv preprint arXiv:1803.05457, (2018).

14. Coumans, E., and Bai, Y., *Pybullet, a python module for physics simulation for games, robotics and machine learning*, (2016).

15. Davis, E., and Aaronson, S., *Testing GPT-4 with Wolfram Alpha and Code Interpreter plug-ins on math and science problems*, arXiv preprint arXiv:2308.05713, (2023).

16. DeChant, C., *Episodic memory in ai agents poses risks that should be studied and mitigated*, 2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML), (2025).

17. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V., *CARLA: an open urban driving simulator*, Conference on Robot Learning, (2017).

18. Eyring, V., Gentine, P., Camps-Valls, G., Lawrence, D. M., and Reichstein, M., *AI-empowered next-generation multiscale climate modelling for mitigation and adaptation*, Nature Geoscience, 17(10), 963–971, (2024).

19. Feng, X., Shen, J., and Fan, Y., *REST: an alternative to RPC for web services architecture*, 2009 First International Conference on Future Information Networks, (2009).

20. Feng, Y., Yan, Y., Shi, K., and Zhang, Z., *Reducing carbon emission at the corporate level: does artificial intelligence matter?*, Environmental Impact Assessment Review, 114, 107911, (2025).

21. Ferber, J., and Weiss, G., *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*, Vol. 1, Addison-Wesley Reading, (1999).

22. Fielding, R. T., *Architectural Styles and the Design of Network-Based Software Architectures*, University of California, Irvine, (2000).

23. Fremantle, P., *A history and future of web APIs*, it – Information Technology, (2014).

24. Gade, F., Lund, O., and Mendoza, M. L., *Benchmarking zero-shot biomedical relation triplet extraction across language model architectures*, Proceedings of the 24th Workshop on Biomedical Language Processing, (2025).

25. Gridach, M., Nanavati, J., Abidine, K. Z. E., Mendes, L., and Mack, C., *Agentic ai for scientific discovery: a survey of progress, challenges, and future directions*, arXiv preprint arXiv:2503.08979, (2025).

26. Hinostroza Fuentes, V. G., Karim, H. A., Tan, M. J. T., and AlDahoul, N., *AI with agency: a vision for adaptive, efficient, and ethical healthcare*, Frontiers in Digital Health, 7, 1600216, (2025).

27. Hoy, M. B., *Alexa, Siri, Cortana, and more: an introduction to voice assistants*, Medical Reference Services Quarterly, 37(1), 81–88, (2018).

28. Huang, Y., Shi, J., Li, Y., Fan, C., Wu, S., Zhang, Q., Liu, Y., Zhou, P., Wan, Y., and Gong, N. Z., *Metatool benchmark for large language models: deciding whether to use tools and which to use*, arXiv preprint arXiv:2310.03128, (2023).

29. Jackson, P., *Introduction to Expert Systems*, Addison-Wesley, (1986).

30. Kadi, H. A., and Terzić, K., *Agent-Arena: a general framework for evaluating control algorithms*, arXiv preprint arXiv:2504.06468, (2025).

31. Kaptelinin, V., and Nardi, B., *Affordances in HCI: toward a mediated action perspective*, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, (2012).

32. Karpas, E., Abend, O., Belinkov, Y., Lenz, B., Lieber, O., Ratner, N., Shoham, Y., Bata, H., Levine, Y., and Leyton-Brown, K., *MRKL systems: a modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning*, arXiv preprint arXiv:2205.00445, (2022).

33. Kaufmann, T., Weng, P., Bengs, V., and Hüllermeier, E., *A survey of reinforcement learning from human feedback*, (2024).

34. Kumar, A. A., *Semantic memory: a review of methods, models, and current challenges*, Psychonomic Bulletin & Review, 28(1), 40–80, (2021).

35. Kvinge, H., Coda, E., Yeats, E., Brown, D., Buckheit, J., Scullen, S. M., Kennedy, B., Truong, L., Kay, W., and Joslyn, C., *Probing the limits of mathematical world models in LLMs*, ICML 2025 Workshop on Assessing World Models, (2025).

36. Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., and Kang, J., *BioBERT: a pre-trained biomedical language representation model for biomedical text mining*, Bioinformatics, 36(4), 1234–1240, (2020).

37. Li, D., Jiang, B., Huang, L., Beigi, A., Zhao, C., Tan, Z., and Liu, H., *From generation to judgment: opportunities and challenges of llm-as-a-judge*, Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pp. 2757–2791, (2025).

38. Liu, X., Yu, H., Zhang, H., Xu, Y., Lei, X., Lai, H., Gu, Y., Ding, H., Men, K., and Yang, K., *Agentbench: evaluating llms as agents*, arXiv preprint arXiv:2308.03688, (2023).

39. Lu, Z., *PubMed and beyond: a survey of web tools for searching biomedical literature*, Database, 2011, baq036, (2011).

40. Lyu, B., Cong, X., Yu, H., Yang, P., Qin, Y., Ye, Y., Lu, Y., Zhang, Z., Yan, Y., and Lin, Y., *Gitagent: facilitating autonomous agent with github by tool extension*, arXiv preprint arXiv:2312.17294, (2023).

41. McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., and Nieto, O., *librosa: audio and music signal analysis in python*, SciPy, 2015, 18–24, (2015).

42. McKinney, W., *Data structures for statistical computing in Python*, SciPy, 445(1), 51–56, (2010).

43. Meurer, A., Smith, C. P., Paprocki, M., Čertík, O., Kirpichev, S. B., Rocklin, M., Kumar, A., Ivanov, S., Moore, J. K., and Singh, S., *SymPy: symbolic computing in Python*, PeerJ Computer Science, 3, e103, (2017).

44. Mialon, G., Dessì, R., Lomeli, M., Nalmpantis, C., Pasunuru, R., Raileanu, R., Rozière, B., Schick, T., Dwivedi-Yu, J., and Celikyilmaz, A., *Augmented language models: a survey*, arXiv preprint arXiv:2302.07842, (2023).

45. Miao, X., Oliaro, G., Zhang, Z., Cheng, X., Jin, H., Chen, T., and Jia, Z., *Towards efficient generative large language model serving: a survey from algorithms to systems*, ACM Computing Surveys, 58(1), 1–37, (2025).

46. Nakajima, Y., *BabyAGI*, GitHub repository, (2023).

47. Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., and Saunders, W., *Webgpt: browser-assisted question-answering with human feedback*, arXiv preprint arXiv:2112.09332, (2021).

48. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., and Ray, A., *Training language models to follow instructions with human feedback*, Advances in Neural Information Processing Systems, 35, 27730–27744, (2022).

49. Paramarthalingam, K., Daniel, D. A., and Srinivasagopalan, M. L. N., *An integrative framework for evaluating healthcare insurance in public health equity and social justice*.

50. Paranjape, B., Lundberg, S., Singh, S., Hajishirzi, H., Zettlemoyer, L., and Ribeiro, M. T., *Art: automatic multi-step reasoning and tool-use for large language models*, arXiv preprint arXiv:2303.09014, (2023).

51. Parisi, A., Zhao, Y., and Fiedel, N., *Talm: tool augmented language models*, arXiv preprint arXiv:2205.12255, (2022).

52. Parui, P., and Prettner, K., *Public provision of healthcare and basic science: what are the effects on economic growth and welfare?*, Vienna University of Economics and Business, (2024).

53. Patil, S. G., Zhang, T., Wang, X., and Gonzalez, J. E., *Gorilla: large language model connected with massive apis*, Advances in Neural Information Processing Systems, 37, 126544–126565, (2024).

54. Peng, J.-L., Cheng, S., Diau, E., Shih, Y.-Y., Chen, P.-H., Lin, Y.-T., and Chen, Y.-N., *A survey of useful llm evaluation*, arXiv preprint arXiv:2406.00936, (2024).

55. Qian, C., Acikgoz, E. C., Wang, H., Chen, X., Sil, A., Hakkani-Tur, D., Tur, G., and Ji, H., *SMART: self-aware agent for tool overuse mitigation*, Findings of the ACL, (2025).

56. Qin, Y., Liang, S., Ye, Y., Zhu, K., Yan, L., Lu, Y., Lin, Y., Cong, X., Tang, X., and Qian, B., *Toolllm: facilitating large language models to master 16000+ real-world apis*, arXiv preprint arXiv:2307.16789, (2023).

57. Qiu, J., Lam, K., Li, G., Acharya, A., Wong, T. Y., Darzi, A., Yuan, W., and Topol, E. J., *LLM-based agentic systems in medicine and healthcare*, Nature Machine Intelligence, 6(12), 1418–1420, (2024).

58. Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., and Clark, J., *Learning transferable visual models from natural language supervision*, International Conference on Machine Learning, (2021).

59. Russell, S., and Norvig, P., *Artificial Intelligence: A Modern Approach*, Prentice-Hall, (1995).

60. Sapkota, R., Roumeliotis, K. I., and Karkee, M., *Ai agents vs. agentic ai: a conceptual taxonomy, applications and challenges*, arXiv preprint arXiv:2505.10468, (2025).

61. Sawant, P., *Agentic AI: a quantitative analysis of performance and applications*, Preprints, https://doi.org/10.20944/preprints202502.1647.v1, (2025).

62. Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., and Scialom, T., *Toolformer: language models can teach themselves to use tools*, Advances in Neural Information Processing Systems, 36, 68539–68551, (2023).

63. Schmidgall, S., Su, Y., Wang, Z., Sun, X., Wu, J., Yu, X., Liu, J., Liu, Z., and Barsoum, E., *Agent laboratory: using llm agents as research assistants*, arXiv preprint arXiv:2501.04227, (2025).

64. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O., *Proximal policy optimization algorithms*, arXiv preprint arXiv:1707.06347, (2017).

65. Sendak, M. P., Ratliff, W., Sarro, D., Alderton, E., Futoma, J., Gao, M., Nichols, M., Revoir, M., Yashar, F., and Miller, C., *Real-world integration of a sepsis deep learning technology into routine clinical care: implementation study*, JMIR Medical Informatics, 8(7), e15182, (2020).

66. Shavit, Y., Agarwal, S., Brundage, M., Adler, S., O'Keefe, C., Campbell, R., Lee, T., Mishkin, P., Eloundou, T., and Hickey, A., *Practices for governing agentic AI systems*, Research Paper, OpenAI, (2023).

67. Shen, Y., Song, K., Tan, X., Li, D., Lu, W., and Zhuang, Y., *Hugginggpt: solving ai tasks with chatgpt and its friends in hugging face*, Advances in Neural Information Processing Systems, 36, 38154–38180, (2023).

68. Shortliffe, E., *Computer-Based Medical Consultations: MYCIN*, Vol. 2, Elsevier, (2012).

69. Singh, H., Graber, M. L., Kissam, S. M., Sorensen, A. V., Lenfestey, N. F., Tant, E. M., Henriksen, K., and LaBresh, K. A., *System-related interventions to reduce diagnostic errors: a narrative review*, BMJ Quality & Safety, 21(2), 160–170, (2012).

70. Stoytchev, A., *Behavior-grounded representation of tool affordances*, Proceedings of the 2005 IEEE International Conference on Robotics and Automation, (2005).

71. Tang, Q., Deng, Z., Lin, H., Han, X., Liang, Q., Cao, B., and Sun, L., *Toolalpaca: generalized tool learning for language models with 3000 simulated cases*, arXiv preprint arXiv:2306.05301, (2023).

72. Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., and Du, Y., *Lamda: language models for dialog applications*, arXiv preprint arXiv:2201.08239, (2022).

73. Tieman, J. J., Lawrence, M. A., Damarell, R. A., Sladek, R. M., and Nikolof, A., *LIt. search: fast tracking access to Aboriginal and Torres Strait Islander health literature*, Australian Health Review, 38(5), 541–545, (2014).

74. Tisue, S., and Wilensky, U., *Netlogo: a simple environment for modeling complexity*, International Conference on Complex Systems, (2004).

75. Tonmoy, S., Zaman, S., Jain, V., Rani, A., Rawte, V., Chadha, A., and Das, A., *A comprehensive survey of hallucination mitigation techniques in large language models*, arXiv preprint arXiv:2401.01313, (2024).

76. Wan, H., Yang, C., Yu, J., Tu, M., Lu, J., Yu, D., Cao, J., Gao, B., Xie, J., and Wang, A., *DeepResearch Arena: the first exam of LLMs' research abilities via seminar-grounded tasks*, arXiv preprint arXiv:2509.01396, (2025).

77. Wang, C., Luo, W., Dong, S., Xuan, X., Li, Z., Ma, L., and Gao, S., *Mllm-tool: a multimodal large language model for tool agent learning*, 2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), (2025).

78. Wang, Z., Cheng, Z., Zhu, H., Fried, D., and Neubig, G., *What are tools anyway? a survey from the language model perspective*, arXiv preprint arXiv:2403.15452, (2024).

79. Wang, Z., Wang, L., Dounis, A. I., and Yang, R., *Multi-agent control system with information fusion based comfort model for smart buildings*, Applied Energy, 99, 247–254, (2012).

80. Ward, B., Bhati, D., Neha, F., and Guercio, A., *Analyzing the impact of AI tools on student study habits and academic performance*, 2025 IEEE 15th Annual Computing and Communication Workshop and Conference (CCWC), (2025).

81. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., and Zhou, D., *Chain-of-thought prompting elicits reasoning in large language models*, Advances in Neural Information Processing Systems, 35, 24824–24837, (2022).

82. Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., and Liu, J., *Autogen: enabling next-gen LLM applications via multi-agent conversations*, First Conference on Language Modeling, (2024).

83. Xu, B., Liu, X., Shen, H., Han, Z., Li, Y., Yue, M., Peng, Z., Liu, Y., Yao, Z., and Xu, D., *Gentopia: a collaborative platform for tool-augmented llms*, arXiv preprint arXiv:2308.04030, (2023).

84. Yao, S., Shinn, N., Razavi, P., and Narasimhan, K., *τ-bench: a benchmark for tool-agent-user interaction in real-world domains*, arXiv preprint arXiv:2406.12045, (2024).

85. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K. R., and Cao, Y., *React: synergizing reasoning and acting in language models*, The Eleventh International Conference on Learning Representations, (2022).

86. Yuan, S., Song, K., Chen, J., Tan, X., Shen, Y., Ren, K., Li, D., and Yang, D., *Easytool: enhancing llm-based agents with concise tool instruction*, Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the ACL, (2025).

87. Zhou, S., Xu, F. F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., Cheng, X., Ou, T., Bisk, Y., and Fried, D., *Webarena: a realistic web environment for building autonomous agents*, arXiv preprint arXiv:2307.13854, (2023).

Abdullah Jawwad Yousafi,
Corresponding Author,
Department of Mathematics and Computer Science,
Beloit College, 700 College Street, Beloit, Wisconsin 53511, USA.
ORCID: 0009-0004-8778-8909
E-mail address: yousafiaj@beloit.edu

and

Nabiha Fatima,
Lahore Grammar School 55 Main,
55 Main Gulberg, Lahore, Pakistan.
E-mail address: nabihafatimayousafi@gmail.com

and

Mehmet Dik,
Department of Mathematics, Computer Science and Physics,
Rockford University, 5050 E. State Street, Rockford, Illinois 61108, USA.
E-mail address: mdik@rockford.edu