Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.067857

REVIEW

A Review of AI-Driven Automation Technologies: Latest Taxonomies, Existing Challenges, and Future Prospects

Weiqiang Jin[1,2], **Ningwei Wang**[1], **Lei Zhang**[3], **Xingwu Tian**[1], **Bohang Shi**[1] **and Biao Zhao**[1,*]

[1]School of Information and Communications Engineering, Xi'an Jiaotong University, iHarbour, Xi'an, 710049, China
[2]Department of Electrical and Electronic Engineering, The University of Hong Kong, Pokfulam Road, Hong Kong, 999077, China
[3]School of Intelligence Science and Technology, Beijing University of Civil Engineering and Architecture, Beijing, 102616, China

*Corresponding Author: Biao Zhao. Email: biaozhao@xjtu.edu.cn

Received: 14 May 2025; Accepted: 10 July 2025; Published: 30 July 2025

ABSTRACT: With the growing adoption of Artifical Intelligence (AI), AI-driven autonomous techniques and automation systems have seen widespread applications, become pivotal in enhancing operational efficiency and task automation across various aspects of human living. Over the past decade, AI-driven automation has advanced from simple rule-based systems to sophisticated multi-agent hybrid architectures. These technologies not only increase productivity but also enable more scalable and adaptable solutions, proving particularly beneficial in industries such as healthcare, finance, and customer service. However, the absence of a unified review for categorization, benchmarking, and ethical risk assessment hinders the AI-driven automation progress. To bridge this gap, in this survey, we present a comprehensive taxonomy of AI-driven automation methods and analyze recent advancements. We present a comparative analysis of performance metrics between production environments and industrial applications, along with an examination of cutting-edge developments. Specifically, we present a comparative analysis of the performance across various aspects in different industries, offering valuable insights for researchers to select the most suitable approaches for specific applications. Additionally, we also review multiple existing mainstream AI-driven automation applications in detail, highlighting their strengths and limitations. Finally, we outline open research challenges and suggest future directions to address the challenges of AI adoption while maximizing its potential in real-world AI-driven automation applications.

KEYWORDS: AI-driven automation techniques and systems; artificial general intelligence (AGI); LLMs; robotic process automation (RPA)

1 Introduction

1.1 Research Backgrounds of AI-Driven Automation

The evolution of artificial intelligence (AI) [1–3] technologies has advanced significantly in recent years, bringing about a new era of AI-driven autonomous tools and automation system [4–7]. Numerous advanced AI-driven automation technologies [8–11] have been developed to address specific challenges within various aspects of human production and daily life. These advancements have enabled organizations to optimize operational efficiency, reduce costs, and enhance productivity across a variety of practical application scenarios, from healthcare [12–14] to finance [15–17]. AI-driven automation systems [18–20] are now capable of automating repetitive and time-consuming tasks, allowing human workers to focus on more efficient activities such as strategic planning and creative problem solving [21–23].

1.1.1 Advancements of AI-Driven Automation

Specifically, AI-driven automation encompasses a wide range of technologies [6,9,24,25], which together streamline complex operations, enhance task-specific decision-making capabilities, and significantly boost practical operational efficiency. Among the various approaches to AI-driven automation, several stand out as particularly influential in driving the field forward, including Robotic Process Automation (RPA) [14,26,27], No-code platforms [28–30], and Large Language Models (LLMs) [31,32].

(1) Firstly, Robotic Process Automation (RPA) [26,33] is a key technique in AI-driven automation, utilizing software robots to perform repetitive tasks traditionally handled by humans. RPA is particularly prevalent in industries such as finance, healthcare, and manufacturing, where efficiency and error reduction are critical [34,35].

(2) Meanwhile, No-code platforms [28,36–38] are another important approach in AI-driven automation, further broadening the accessibility of automation technologies. They enable users without technical expertise to design complex workflows and automation systems, democratizing the development of AI solutions. With intuitive interfaces that allow for the creation of complex workflows without requiring coding skills, these platforms empower users to build customized automation solutions [39–42].

(3) Lastly, LLMs [43–45] have achieved exceptional success across a wide range of tasks, from natural language processing (NLP) [46,47] to decision support systems [48–50]. By utilizing vast amounts of textual data, these models generate accurate and contextually relevant outputs, making them indispensable in applications such as chatbots [51,52], document automation [53–56], and customer service [10,57,58].

In recent years, among them, the most transformative advances in AI-driven automation is the integration of LLMs into task-execution and production processes [59,56,60]. The integration of LLMs and traditional workflow automation techniques dramatically improves the effectiveness and efficiency of human production processes such as customer service [10,57,58], document processing [2,53,54], and decision-making support [61–63], allowing people and organizations to develop more intelligent systems capable of understanding complex tasks and delivering optimized solutions.

Overall, AI-driven automation has quickly become a transformative force across various industries, offering tremendous potential in improving efficiency, accuracy, and scalability.

1.1.2 Challenges in AI-Driven Automation

However, as the technology continues to evolve, AI-driven automation faces several significant technical challenges that hinder its widespread adoption and effective implementation [56,64,65]. These challenges, ranging from data governance [2,54,66] to model interpretability [13,64,67] and deployment complexity [20,55,56,68], present barriers that need to be addressed to fully realize the potential of AI-driven automation.

Specifically, AI-driven automation faces multiple technical hurdles: (1). Data quality and governance issues [35,69,70], including noise, privacy, and compliance; (2). Model interpretability gaps in "black-box" systems [64,67,71], critical for high-stakes domains; (3). Generalization limitations in unseen scenarios [72–74]; (4). Scalability demands [66,75,76] for large-scale, low-latency deployment; (5). Algorithmic bias perpetuating unfair outcomes [77–79]; (6). Integration complexity with legacy systems [14,17,80]; (7). Real-time processing requirements [18,81,82]; and (8). Security vulnerabilities to adversarial attacks [58,73,76].

Addressing these issues is essential for advancing the AI-driven automation research. Overcoming these challenges will enable AI-driven automation to reach its full potential, leading to more efficient and effective systems across industries.

1.2 Motivations of This Work and Our Contributions

While numerous innovative AI automation-related works, particularly with the advanced LLMs [77,83,84] and Robotic Process Automation (RPA) [26,54,82], have introduced various novel technical concepts and knowledge in the research of AI-driven automation. Despite these successes and increasing attention, a comprehensive review of the current AI-driven automation developments still remains limited [75,77,85,86]. Moreover, these latest developments have introduced many innovative framework and approaches that were not covered in earlier literature [67,75,86], highlighting the need for more updated and structured overview and arrangement for these literature.

Given the accelerating technology progress in AI-driven automation [6,25,70,76], in this paper, we are motivated to present a novel taxonomy that categorizes and unifies the various methods, tools, and applications. We believe that this will help subsequent researchers with a better and clearer understanding of the key technical features and workflows of AI-driven automation [17,75]. We believe this work will contribute to the development of AI-driven automation, offering insights into how different components/modules interact and drive overall performance [69,82]. This review aims to help researchers identify emerging trends, and highlight critical gaps that require further exploration to advance the frontiers of AI-driven automation.

In summary, in this survey, we make the following key contributions:

[1]. Firstly, we propose a novel taxonomy to classify the different methods used in AI-driven automation. This different taxonomy provide clarity on the strengths and weaknesses of each categories and how they can be applied in various practical application scenarios.

[2]. Then, we also systematically evaluate current mainstream tools and systems used for AI-driven automation, discussing their strengths and limitations, and offering insights into the kinds of approaches.

[3]. Finally, we discuss the major challenges facing the field, including data privacy, ethical concerns, and the lack of regulatory frameworks, and propose strategies for overcoming these obstacles.

[4]. Meanwhile, we suggest promising areas for future research of AI-driven automation, particularly in improving LLM transparency, developing more efficient RPA techniques, and addressing regulatory challenges in the AI space.

1.3 Methodology for Literature Selection

To ensure a comprehensive and unbiased selection of relevant literature, we employed a systematic search strategy across multiple academic databases, including IEEE Xplore[1], ACM Digital Library[2], ScienceDirect[3], SpringerLink[4], and arXiv[5]. The search query was designed to capture key concepts related to AI-driven automation tools, their functionalities, and applications.

Specifically, our survey adopts a systematic approach to literature selection, prioritizing peer-reviewed journal articles, conference proceedings, and high-impact preprints We focus exclusively on publications that address AI-driven automation tools, their technical architectures, and real-world applications, ensuring alignment with our taxonomy development goals. The selected works were published between 2010 and 2025, a period that captures both foundational advancements in AI-automation techniques [26,27,62] This timeframe was chosen to reflect the rapid evolution of the field, including breakthroughs such as transformer-based models (e.g., GPT-4 [45]).

[1]IEEE Xplore: https://ieeexplore.ieee.org/Xplore/home.jsp (accessed on 09 July 2025).
[2]ACM Digital: https://dl.acm.org/ (accessed on 09 July 2025).
[3]ScienceDirect: https://www.sciencedirect.com/ (accessed on 09 July 2025).
[4]Springer Nature Link: https://link.springer.com/ (accessed on 09 July 2025).
[5]arXiv: https://arxiv.org/ (accessed on 09 July 2025).

The initial screening of titles and abstracts refined the candidate pool from over 3000 articles to the most relevant works. Among over 250 core references selected, approximately 10% originate from 2010 to 2018, representing seminal works on RPA infrastructure and early NLP integration. The period 2019 to 2021 accounts for 35% of references, coinciding with the emergence of LLM-powered automation frameworks. The majority at 55% derive from 2022 to 2025. This ensures balanced coverage of historical foundations and representative advancements while minimizing recency bias through rigorous citation analysis and validation.

1.4 Survey Roadmap

This survey is systematically organized to provide a comprehensive analysis of AI-driven automation tools, their challenges, and future directions. The structure is as follows: Section 1 introduces the transformative impact of AI-driven automation across industries, outlines the scope of this survey, and presents its organizational framework. Section 2 establishes systematic taxonomies to classify AI-driven automation tools into four architectural paradigms: rule-based, LLM-driven, multi-agent cooperative, and hybrid models, further categorizing them by functionality, execution mechanisms, and human-AI collaboration modes. Section 3 conducts a comparative analysis of state-of-the-art tools (e.g., Manus, OWL, LangChain, UiPath), emphasizing their applicability in domains like healthcare, finance, and manufacturing. Section 4 examines persistent challenges, including data governance gaps, model interpretability deficits, and scalability bottlenecks, while proposing mitigation strategies. Section 5 explores emerging trends (e.g., self-debugging workflows, AGI integration, and quantum-resistant security) and forecasts future research trajectories. Section 6 concludes by synthesizing key insights and advocating for interdisciplinary collaboration to align technological advancements with ethical and societal values.

2 A Comprehensive Technological Taxonomy of AI-Driven Automation

Recent advances of *Machine Learning* (ML) [87–89] and *Artificial Intelligence* (AI) [72,78,83,90] have led to a proliferation of AI-driven automation tools, each designed to address specific challenges in workflow optimization, decision-making, and task execution [91–94]. As these tools grow increasingly diverse in their architectures, functionalities, and applications, there is a pressing need for a systematic taxonomy to categorize and compare their capabilities [95–98]. Currently, the lack of a unified framework makes it difficult for researchers and practitioners to evaluate the suitability of these tools for different use cases [99–102]. Thus, this section aims to address this gap by proposing several structured classifications from different dimensions of various AI-driven automation tools, analyzing their key characteristics, and highlighting their roles in modern automation ecosystems.

As illustrated in Fig. 1, these tools exhibit diverse architectural paradigms and functional specializations. The proposed taxonomy systematically organizes these systems across five key dimensions: (1) architectural foundations (Section 2.1), (2) core functionalities (Section 2.2), (3) task flow management (Section 2.3), and (4) application domains (Section 2.4).

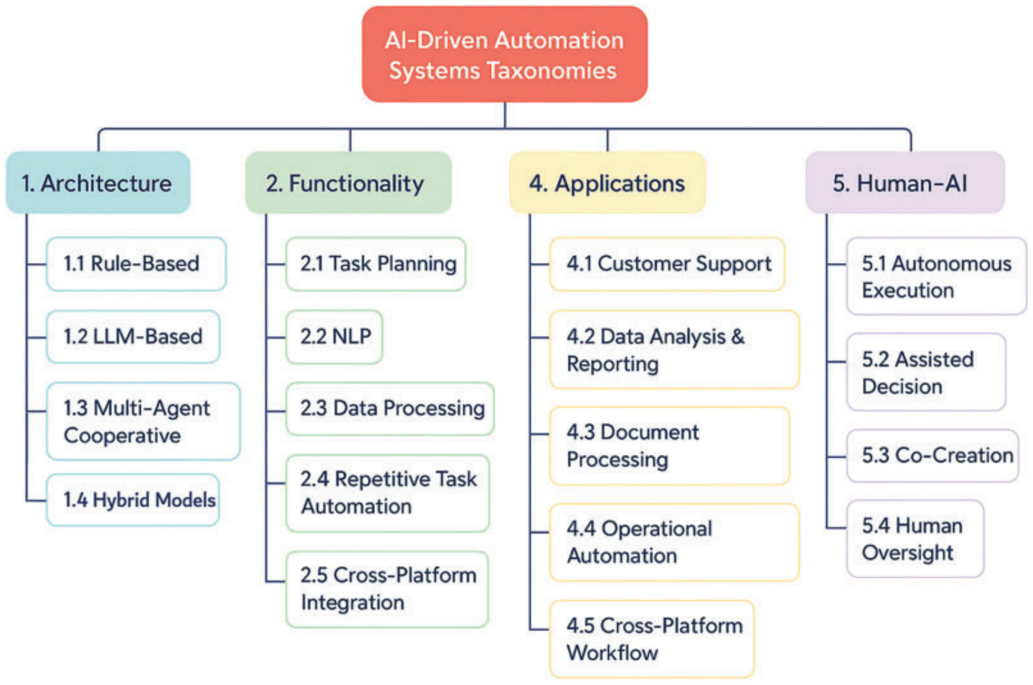


Figure 1: The taxonomy visualization of architectural approaches for AI-driven automation

2.1 Classification on Architecture of AI-Driven Automation Tools

The architectural landscape of modern AI-driven automation tools can be categorized into four distinct paradigms, including Rule-based [2,15,54,103], LLM-based [59,72,78,87], Multi-Agent-based [48,91–93] and Hybrid Models [7,33,55,86], each addressing specific operational requirements and complexity levels [6,96–98]. The architecture classification of automation technologies is outlined in Table 1.

Table 1: Categories divided by architecture of AI-driven automation tools

AI-driven automation categories	Descriptions	Examples
Rule-based	These tools rely on pre-defined rules and logic to perform tasks. The rules dictate actions based on conditions.	UiPath [104], ChatbotRev [105–107] HeyTAP [108], ChatIoT [109], RecRules [110]
LLM-based	These tools use LLMs to understand and generate natural language, automating tasks related to text and decision-making.	LangChain [10,111,112], Auto-GPT [113], GPT Researcher [44], OpenManus [114], BabyAGI [115], AgentGPT [31], SuperAGI [116]
Multi-agent cooperative	These systems use multiple autonomous agents to collaboratively work on tasks, with task decomposition and coordination between agents.	Manus [117–119], OWL [120], MetaGPT [60], AutoGen [91], AgentVerse [65], CrewAI [121], ChatDev [47], TradingGPT [16]

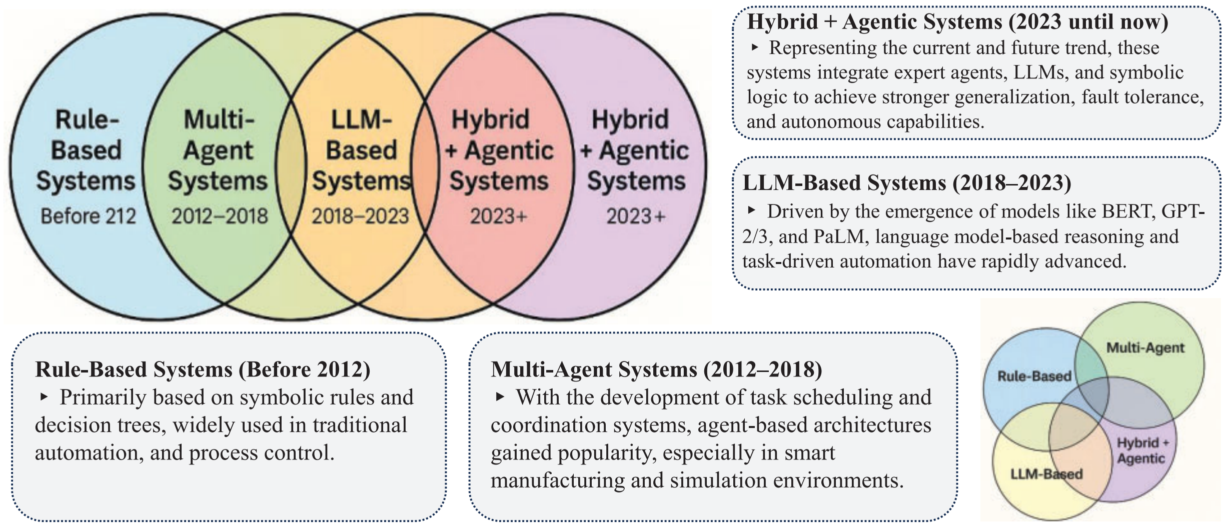
(Continued)

Table 1 (continued)

AI-driven automation categories	Descriptions	Examples
Hybrid models	Combining RPA, AI, and machine learning models, hybrid tools use both pre-programmed rules and AI decision-making for task execution.	UiPath AI Fabric [104,106,107], AirSlate [122], Zapier [30], SmartFlow [55], LMRPA [33], AutoAgent [123], EconAgent [32]

Specifically, 1. *Rule-Based Systems* [11,103,54] represent the foundational layer of automation architectures, employing deterministic logic through predefined condition-action rules. 2. *LLM-Based Systems* [46,77,87,124] mark a paradigm shift toward cognitive automation, leveraging LLMs to process unstructured data and execute language-centric tasks. 3. *Multi-Agent Systems* [125–127] introduce distributed intelligence through coordinated autonomous agents, as implemented in platforms like Manus [114,117,119] and OWL [120]. 4. *Hybrid Models* [7,33,55] combine the strengths of rule-based and AI-driven approaches, exemplified by UiPath AI Fabric's integration of RPA with machine learning.

Fig. 2 illustrates the evolutionary landscape of automation system architectures over time. It highlights four major phases, Rule-Based, Multi-Agent, LLM-Based, and Hybrid Agentic, alongside their timeframes and defining characteristics [91,94–96]. This shows how each paradigm emerged, developed, and influenced the next, reflecting shifts in both technical capability and system design philosophy. Notably, these paradigms are not isolated; rather, they exhibit significant overlaps and integration points, emphasizing that no single approach exists independently of the others in practice [5,6,75].


Rule-Based Systems Before 212
Multi-Agent Systems 2012–2018
LLM-Based Systems 2018–2023
Hybrid + Agentic Systems 2023+
Hybrid + Agentic Systems 2023+
Rule-Based
Multi-Agent
LLM-Based
Hybrid + Agentic

Hybrid + Agentic Systems (2023 until now)
‣ Representing the current and future trend, these systems integrate expert agents, LLMs, and symbolic logic to achieve stronger generalization, fault tolerance, and autonomous capabilities.

LLM-Based Systems (2018–2023)
‣ Driven by the emergence of models like BERT, GPT-2/3, and PaLM, language model-based reasoning and task-driven automation have rapidly advanced.

Rule-Based Systems (Before 2012)
‣ Primarily based on symbolic rules and decision trees, widely used in traditional automation, and process control.

Multi-Agent Systems (2012–2018)
‣ With the development of task scheduling and coordination systems, agent-based architectures gained popularity, especially in smart manufacturing and simulation environments.

Figure 2: The evolutionary trajectory and integration of automation architectures: *From rule-based systems to hybrid agentic intelligence*

As shown in Fig. 3 (1), this is a typical structure of rule-based automation systems [11,51,105]. This architecture is centered around deterministic processing driven by predefined logic. The system accepts structured and pre-validated input from the user or an external source. These inputs are fed into a logic engine, which applies rule-based conditions-often expressed in IF-THEN format-for pattern matching [108,110]. Each rule is associated with a deterministic action, such as sending an alert or executing an API call [59,80,128].

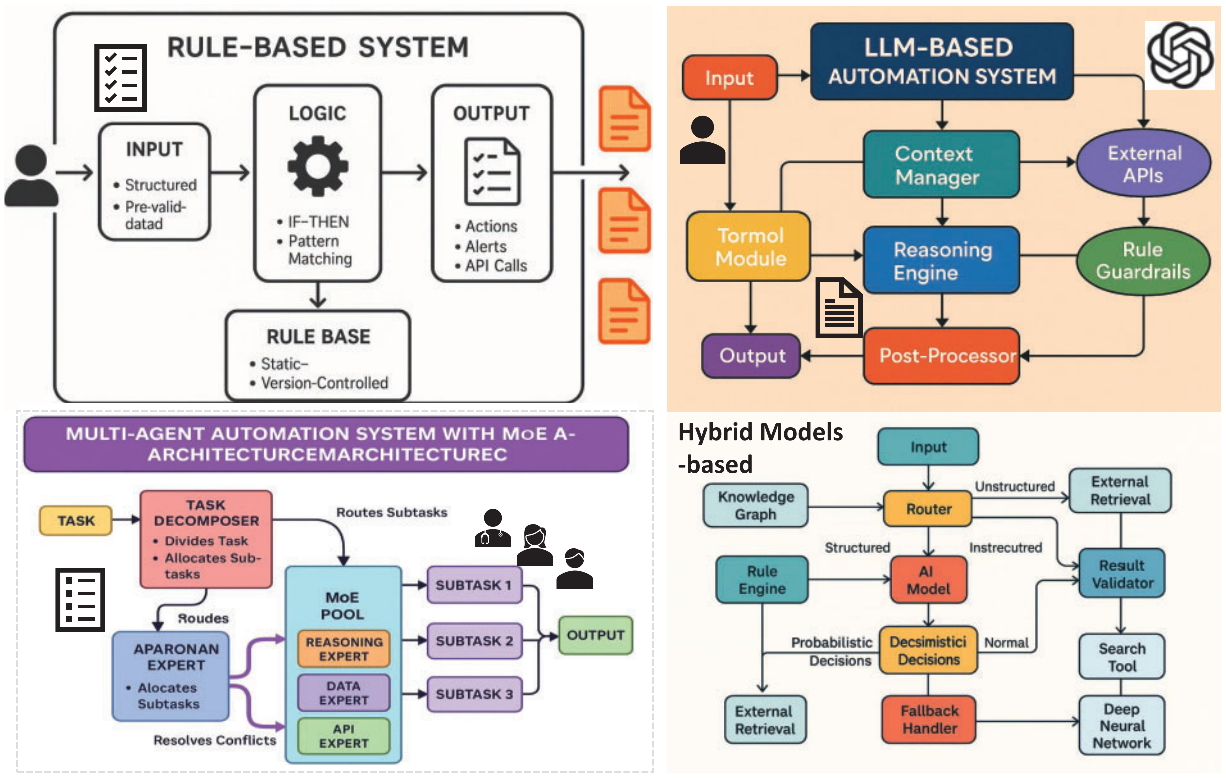


Figure 3: The representative architectural prototypes of four major categories of existing automation systems: (1) Rule-based systems [105,109,110], (2) Multi-agent systems [65,129,130] with Mixture-of-Experts (MoE) [131,132], (3) Multi-agent-based automation decision architectures [133–136], and (4) Hybrid models [7,33,55] integrating symbolic rules with AI reasoning and fallback mechanisms

The rule base in this like automation system is a separate, static, and version-controlled module, decoupled from the logic execution layer [51,108]. This ensures maintainability and traceability of the rule set over time. The outputs are standardized and include actionable signals like alerts or service calls. The model emphasizes the non-probabilistic, mechanical nature of such systems, with fixed logical flows and strict dependency on exact condition matching. This kind of architecture is highly interpretable, though it requires manual effort to maintain and update the rule repository [109,137].

Fig. 3 (2) illustrates a typical architecture prototype of a LLM-based automation system. The architecture adopts a modular design that clearly delineates core processing flows and external interaction interfaces, enabling an end-to-end intelligent mapping from input to output.

Firstly, the *input module* receives natural language requests or structured data from the user and forwards them to the *Context Manager*, which maintains dialogue history, task context, and multi-turn interaction states-essential for preserving semantic continuity and task awareness [70]. Next, the *Memory Module* organizes both short-term and long-term information, allowing fast retrieval of contextual data and knowledge fragments. On top of this, the system can integrate *Retrieval-Augmented Generation* (RAG) [46,138–140], leveraging external vector databases for semantic search to enhance the model's knowledge base. The *Reasoning Module* is responsible for understanding intent, making logical decisions, and invoking tools [5,141,142]. It interacts with *External APIs* to query databases or perform calculations. Finally, outputs are then passed through *Rule Guardrails* [105,109] to ensure compliance with business rules, ethical standards, and formatting constraints. Finally, the *Post-Processor Module* [102,143] sanitizes the raw output, applies safety filters, and formats the response before it is delivered to the user through the *output module*, ensuring clarity and reliability.

Fig. 3 (3) presents a hierarchical architecture for multi-agent automation systems that integrates dynamic task decomposition with a Mixture-of-Experts (MoE) [131,132] execution layer. The system begins with a main task input, which is parsed by a Task Decomposer module responsible for dividing the task into coherent subtasks based on task semantics and dependency graphs. These subtasks are then routed to an MoE pool containing specialized expert agents, such as reasoning experts [61,78,90], data processing agents [53,144,145], and API orchestration agents [59,79,80,128].

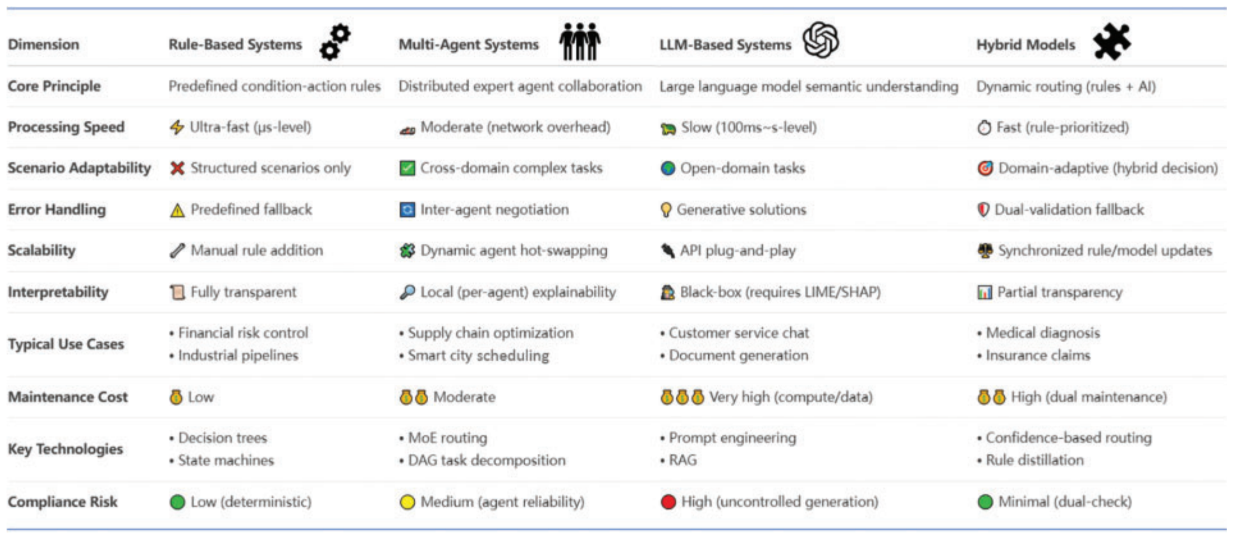
Routing decisions are made based on task embeddings and gating networks that dynamically assign subtasks to the most suitable experts [131,146–148]. Subtasks are executed in parallel, and their results are aggregated into a unified output. Conflict resolution and validation mechanisms ensure consistency and robustness. This architecture supports modular scalability, fault tolerance, and efficient parallelism, making it well-suited for complex, heterogeneous, and cross-domain automation scenarios [39,55,91,94].

Fig. 3 (4) depicts a hybrid model-based automation architecture that combines rule-based systems with AI models to achieve adaptive and resilient decision-making [7,33,86,129]. The system begins by routing user input-either structured or unstructured-through a decision router. Structured data is directed to a deterministic rule engine, while unstructured or context-dependent input is processed by an AI model [59,72,78,90]. The outputs from both branches are validated through a result validation module that ensures consistency and accuracy.

Depending on the decision type (probabilistic or deterministic), results are either accepted or passed to a fallback handler when anomalies or inconsistencies are detected. The fallback mechanism includes model switching, rule simplification, or escalation to human review [7,83,98]. A knowledge graph [149] supports both the router and AI model in contextual understanding, and external retrieval modules assist in enriching or verifying the input [132,144,145,150]. This hybrid approach balances the explainability and speed of symbolic logic with the flexibility and adaptability of neural models, making it suitable for high-stakes or regulated automation domains [33,55,86,129].

Furthermore, Fig. 4 presents a comparative architecture overview of four representative categories of AI-driven automation system. The figure is designed to highlight the core structural patterns and processing flows underlying each class, including rule-based systems, multi-agent automation systems, LLMs-based decision frameworks, and hybrid-AI integrated models [91,94–96]. The visualization contextualize the diversity of design philosophies and operational mechanisms across these automation systems [39,47,55,151]. This comparison supports the analysis of trade-offs in scalability, interpretability, robustness, and domain alignment across different automation paradigms.

Overall, the architectural progression from rule-based systems to hybrid systems and LLM-based systems reflects an industry-wide transition from mechanical task replication to cognitive process optimization [54,72,78,87]. Current research focuses on developing systems that self-adaptive dynamically select appropriate architectural paradigms based on real-time workflow analysis [97,99,146,147].



Dimension	Rule-Based Systems	Multi-Agent Systems	LLM-Based Systems	Hybrid Models
Core Principle	Predefined condition-action rules	Distributed expert agent collaboration	Large language model semantic understanding	Dynamic routing (rules + AI)
Processing Speed	⚡ Ultra-fast (μs-level)	Moderate (network overhead)	Slow (100ms~s-level)	⏱ Fast (rule-prioritized)
Scenario Adaptability	❌ Structured scenarios only	✅ Cross-domain complex tasks	🌐 Open-domain tasks	🎯 Domain-adaptive (hybrid decision)
Error Handling	⚠ Predefined fallback	Inter-agent negotiation	💡 Generative solutions	🛡 Dual-validation fallback
Scalability	✏ Manual rule addition	🧩 Dynamic agent hot-swapping	API plug-and-play	Synchronized rule/model updates
Interpretability	📜 Fully transparent	🔎 Local (per-agent) explainability	Black-box (requires LIME/SHAP)	📊 Partial transparency
Typical Use Cases	• Financial risk control • Industrial pipelines	• Supply chain optimization • Smart city scheduling	• Customer service chat • Document generation	• Medical diagnosis • Insurance claims
Maintenance Cost	💰 Low	💰💰 Moderate	💰💰💰 Very high (compute/data)	💰💰 High (dual maintenance)
Key Technologies	• Decision trees • State machines	• MoE routing • DAG task decomposition	• Prompt engineering • RAG	• Confidence-based routing • Rule distillation
Compliance Risk	🟢 Low (deterministic)	🟡 Medium (agent reliability)	🔴 High (uncontrolled generation)	🟢 Minimal (dual-check)

Figure 4: A comprehensive framework comparison of AI-driven automation systems

2.2 *Classification on Functionality of AI-Driven Automation Tools*

The functional landscape of AI-driven automation tools can be systematically organized into five principal categories, including *Task Planning, NLP, Data Processing, Repetitive Task Automation*, and *Cross-Platform Integration* [5,6,95]. Each represents distinct technical approaches to workflow automation [69,75]. This classification, presented in Table 2, reveals the evolving capabilities of modern automation systems from isolated task execution to integrated cognitive platforms.

Table 2: Functional classification of AI-driven automation tools

Functionality	Technical characteristics	Representative systems
Task planning	Hierarchical task decomposition using graph neural networks.	Manus [117–119], OpenManus [114], LangChain [10,111,152,153], ToRA [154], AutoAgent [123], TaskMatrix.AI [23], LLM-Planner [155]
Natural Language Processing (NLP)	Transformer-based semantic parsing and multi-modal interaction.	GPT-4 [43,45,156], DeepAI [157], Hugging Face [158,159], MM-REACT [79], Whisper [160,161], CLIP [162], airSlate [122], LLaMA [88,89], Docling [53]
Cognitive data processing Repetitive task automation	Cognitive ETL pipelines with automated feature engineering. Vision-enabled RPA with contextual adaptation.	MetaGPT [60], OWL [120], UiPath AI Fabric [106,107] UiPath [106], Zapier [30], AirSlate [122] SmartFlow [55], LMRPA [33], BreachSeek [163]

(Continued)

Table 2 (continued)

Functionality	Technical characteristics	Representative systems
Cross-platform integration	Neural API mapping and protocol translation.	OWL [120], UiPath [106], Zapier [30], ToolLLM [59], AutoGen [91], AgentVerse [65], CrewAI [121], Airtable [28], Voiceflow [37]

As shown in Fig. 5 (1), **Task Planning Architecture** enables intelligent workflow automation through dynamic goal decomposition and adaptive execution. The system first parses natural language objectives into structured representations using LLMs. A Graph Neural Network (GNN) [164] then hierarchically decomposes tasks while modeling dependencies between sub-tasks as directed edges [165,166]. The architecture features three innovation layers: (1) *Resource-Aware Allocation* that dynamically assigns compute resources (GPU/CPU) based on sub-task requirements [25,27,167], (2) *Reinforcement Learning Scheduler* optimizing execution order through real-time performance feedback [16,85,168], and (3) *Fault-Tolerant Monitoring* triggering granularity adjustments when success rates drop below thresholds [39,56,136].

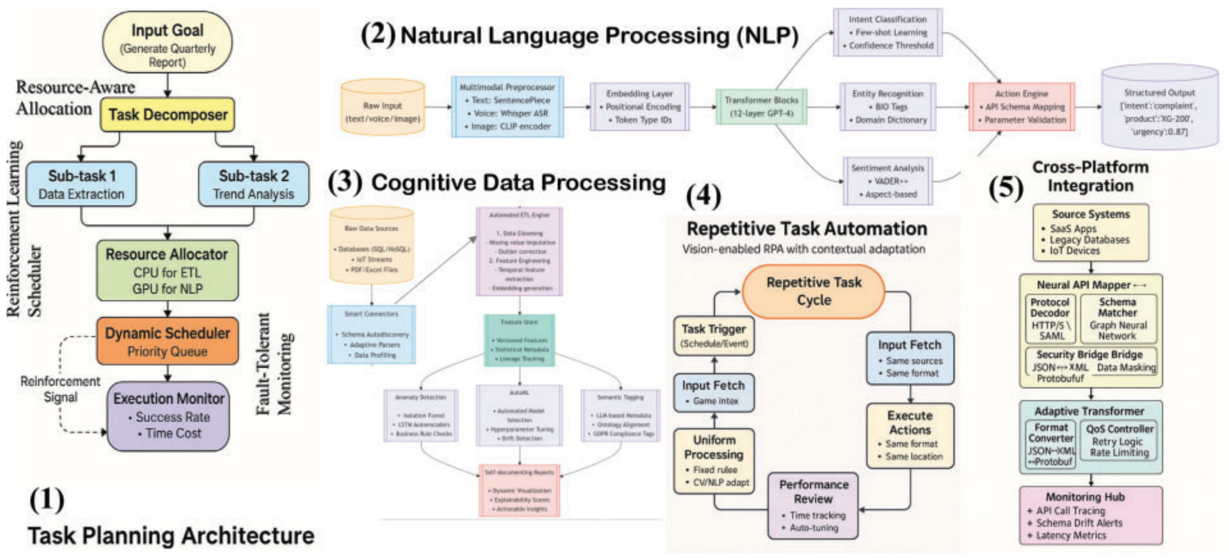

Input Goal (Generate Quarterly Report)
Resource-Aware Allocation
Task Decomposer
Reinforcement Learning Scheduler
Sub-task 1 Data Extraction
Sub-task 2 Trend Analysis
Resource Allocator CPU for ETL GPU for NLP
Dynamic Scheduler Priority Queue
Reinforcement Signal
Execution Monitor • Success Rate • Time Cost
Fault-Tolerant Monitoring
(1) Task Planning Architecture
(2) Natural Language Processing (NLP)
Raw Input (text/voice/image)
Multimodal Preprocessor • Text: SentencePiece • Voice: Whisper ASR • Image: CLIP encoder
Embedding Layer • Positional Encoding • Token Type IDs
Transformer Blocks (12-layer GPT-4)
Intent Classification • Few-shot Learning • Confidence Threshold
Entity Recognition • BIO Tags • Domain Dictionary
Sentiment Analysis • VADER++ • Aspect-based
Action Engine • API Schema Mapping • Parameter Validation
Structured Output ['intent':'complaint', 'product':'XG-200', 'urgency':0.87]
(3) Cognitive Data Processing
Raw Data Sources • Databases (SQL/NoSQL) • IoT Streams • PDF/Excel Files
Smart Connectors • Schema Autodiscovery • Adaptive Parsers • Data Profiling
Automated ETL Engine 1. Data Cleansing - Missing value imputation - Outlier correction 2. Feature Engineering - Temporal feature extraction - Embedding generation
Feature Store • Versioned Features • Statistical Metadata • Lineage Tracking
Anomaly Detection • Isolation Forest • LSTM Autoencoders • Business Rule Checks
AutoML • Automated Model Selection • Hyperparameter Tuning • Drift Detection
Semantic Tagging • LLM-based Metadata • Ontology Alignment • GDPR Compliance Tags
Self-documenting Reports • Dynamic Visualization • Explainability Scores • Actionable Insights
(4) Repetitive Task Automation
Vision-enabled RPA with contextual adaptation
Repetitive Task Cycle
Task Trigger (Schedule/Event)
Input Fetch • Game intex
Input Fetch • Same sources • Same format
Uniform Processing • Fixed rules • CV/NLP adapt
Execute Actions • Same format • Same location
Performance Review • Time tracking • Auto-tuning
(5) Cross-Platform Integration
Source Systems • SaaS Apps • Legacy Databases • IoT Devices
Neural API Mapper ←→
Protocol Decodor HTTP/S \ SAML
Schema Matcher Graph Neural Network
Security Bridge Bridge JSON↔XML Data Masking Protobufuf
Adaptive Transformer
Format Converter JSON↔XML ↔Protobuf
QoS Controller Retry Logic Rate Limiting
Monitoring Hub + API Call Tracing + Schema Drift Alerts + Latency Metrics

Figure 5: Representative architectures of AI-driven automation systems across functional categories

As depicted in Fig. 5 (2), **NLP Processing-based Automation** enables multimodal automation through layered linguistic analysis [79,149,169,170]. The raw inputs (text/voice/image) undergo unified preprocessing via *SentencePiece* tokenization [171], *Whisper* speech recognition [160,161], or *CLIP* visual encoding [162]. The system then processes embeddings through 12-layer GPT-4 transformer blocks [156,43], generating parallel outputs for: (1) *Intent Classification* using few-shot learning [66,172], (2) *Entity Recognition* with domain-enhanced BIO tagging [170], and (3) *Aspect-based Sentiment Analysis* via VADER++ [169,173]. These outputs converge at the *Action Engine*, which validates parameters against API schemas (e.g., mapping product mentions to CRM IDs) and triggers downstream workflows [91,92,174].

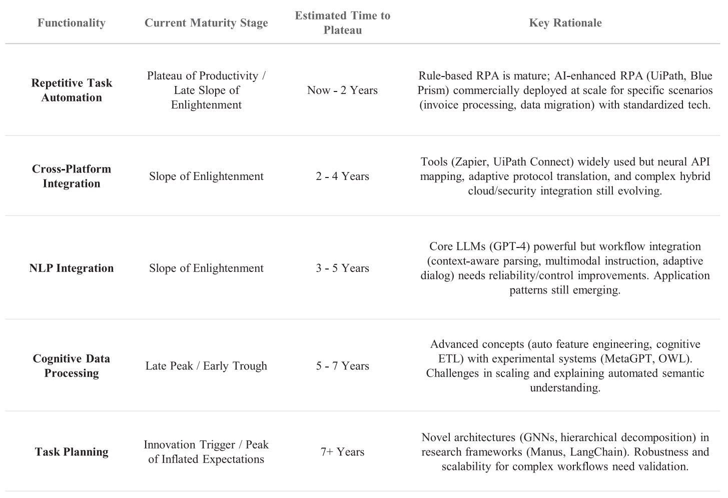
As illustrated in Fig. 5 (3), **Cognitive Data Processing**-based automation system demonstrates a comprehensive architecture for intelligent data automation. The pipeline initiates with *Raw Data Sources* spanning structured databases (SQL/NoSQL), semi-structured documents (PDF/Excel), and real-time IoT streams [14,34,55,66]. The cognitive data processing pipeline features three core processing stages: (1) *Smart Connectors* [80–82] that perform schema autodiscovery and adaptive parsing of diverse data sources, (2) *Automated ETL* [17,69,75] with dual-phase processing including data cleansing (missing value imputation, outlier correction) and feature engineering (temporal patterns, embeddings), and (3) *Feature Store* [14,34,76] that maintains versioned features with complete lineage tracking.

As illustrated in Fig. 5 (4), **Repetitive Task Automation** systems combine rule-based execution with adaptive computer vision (CV) [49,50] and natural language processing (NLP) [46,148,175,176] to handle high-volume, predictable workflows. The architecture follows a cyclical pattern where tasks are: (1) triggered by scheduled events or system alerts, (2) processed through standardized rules with contextual adaptations, and (3) executed with consistent output formats [33,177,178]. Key innovations include *vision-enabled element recognition* that maintains action precision across interface changes, and *performance auto-tuning* that optimizes processing time while preserving output consistency. These systems excel in scenarios like invoice processing (extracting fixed fields from varying layouts) or data migration (repetitive database operations).

Fig. 5 (5) illustrates a generalized architecture for **Cross-Platform Automation Systems**, designed to bridge heterogeneous data environments through intelligent integration layers [65,130,179]. The system begins with diverse *Source Systems* such as SaaS applications, legacy databases, and IoT devices [70,76,180]. A *Neural API Mapper* decodes protocol formats, aligns schemas using graph neural networks, and securely translates authentication protocols (e.g., OAuth2 [181,182] to SAML [143,183]). The *Adaptive Transformer* module converts between data formats (e.g., JSON, XML, Protobuf) and ensures quality-of-service via retry logic and rate limiting. Processed data flows into *Target Systems*, including cloud platforms and mobile endpoints, while a *Monitoring Hub* tracks API calls, schema drift, and latency metrics [184–186]. This architecture emphasizes modularity, protocol adaptability, and real-time observability, making it particularly suitable for hybrid cloud integration and dynamic enterprise workflows.

Beyond the functional taxonomy, understanding the relative maturity of these approaches is crucial for researchers alike. Fig. 6 presents a technology maturity curve analysis for the five functional categories, synthesizing insights from current implementations, representative systems, and ongoing research challenges. This indicates that while AI-enhanced repetitive task automation is nearing mainstream adoption, and cross-platform integration tools are demonstrating increasing robustness, the more cognitive capabilities, particularly NLP integration, cognitive data processing, and advanced task planning, exhibit significant potential but face longer development paths towards stable, widespread deployment. NLP integration shows rapid progress but requires refinement in operational reliability within workflows. Cognitive data processing and task planning represent the frontier of AI-driven automation, currently characterized by innovative prototypes and active research, with substantial work needed to overcome challenges in scalability, robustness, and generalization for complex, real-world environments.

Overall, the functional evolution of AI-driven automation tools reveals three significant technical trends: the convergence of symbolic and connectionist AI approaches [78,96,142], the emergence of self-configuring automation ecosystems [92,94,187], and the growing importance of human-AI collaboration frameworks [188–190]. Current research directions focus on developing more robust architectures capable of handling increasingly complex, multi-domain workflows while maintaining operational transparency.



Functionality	Current Maturity Stage	Estimated Time to Plateau	Key Rationale
Repetitive Task Automation	Plateau of Productivity / Late Slope of Enlightenment	Now - 2 Years	Rule-based RPA is mature; AI-enhanced RPA (UiPath, Blue Prism) commercially deployed at scale for specific scenarios (invoice processing, data migration) with standardized tech.
Cross-Platform Integration	Slope of Enlightenment	2 - 4 Years	Tools (Zapier, UiPath Connect) widely used but neural API mapping, adaptive protocol translation, and complex hybrid cloud/security integration still evolving.
NLP Integration	Slope of Enlightenment	3 - 5 Years	Core LLMs (GPT-4) powerful but workflow integration (context-aware parsing, multimodal instruction, adaptive dialog) needs reliability/control improvements. Application patterns still emerging.
Cognitive Data Processing	Late Peak / Early Trough	5 - 7 Years	Advanced concepts (auto feature engineering, cognitive ETL) with experimental systems (MetaGPT, OWL). Challenges in scaling and explaining automated semantic understanding.
Task Planning	Innovation Trigger / Peak of Inflated Expectations	7+ Years	Novel architectures (GNNs, hierarchical decomposition) in research frameworks (Manus, LangChain). Robustness and scalability for complex workflows need validation.

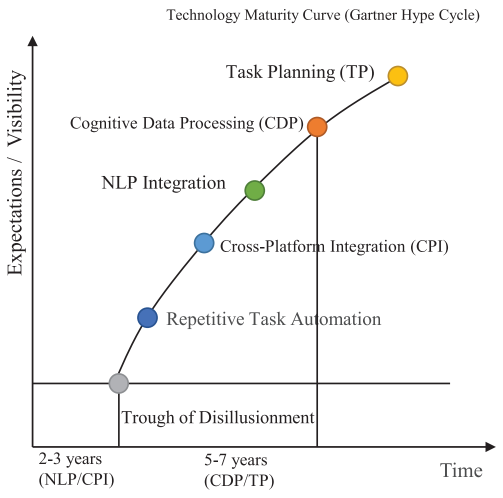


Figure 6: Technology maturity assessment of AI-driven automation functional categories

2.3 Taxonomy on Task Flow Management and Execution Mechanisms

The architectural design of task flow management [6,21,191] in AI-driven automation systems reflects fundamental trade-offs between predictability, efficiency, and adaptability. As detailed in Table 3, contemporary tools employ four distinct execution paradigms—sequential, parallel, iterative, and task decomposition-based approaches [31,86,192]—each demonstrating unique technical characteristics shaped by their underlying coordination models and error recovery strategies [39,95,99].

Table 3: Task flow management architectures

Management type	Technical characteristics	Representative systems
Sequential execution	Linear state transitions with static dependency resolution	UiPath [104,106], Zapier [30], AirSlate [122], HeyTAP [108], RecRules [110], ChatIoT [109], SmartFlow [55], LMRPA [33], AutoAgent [123]
Parallel execution	Concurrent processing with graph-based dependency tracking	OWL [120], MetaGPT [60], AutoGen [91], AgentVerse [65], CrewAI [121], TradingGPT [16], GameGPT [48], MAgIC [193], LLMArena [134], ChatDev [47]
Iterative execution	Feedback-aware workflow versioning with rollback mechanisms	LangChain [111,112], OpenManus [114], TaskMatrix.AI [23], LLM-Planner [155], Reflexion [83], ToRA [154], ReAct [78], Self-RAG [139], Tree-of-Thought [96], AlphaCodium [194], LoopGPT [195]

(Continued)

Table 3 (continued)

Management type	Technical characteristics	Representative systems
Task decomposition	Hybrid neuro-symbolic goal reduction architectures	Manus [117,118], OWL [120], AutoGPT [113], ToRA [154], LLM-Planner [155], AgentVerse [65], ToolLLM [59], Mixtral of Experts [131], ExpeL [196], EconAgent [32], TidyBot [197]

As shown in Fig. 7 (1), *Sequential Execution Automation System* [14,66,69,75] adopts a linear task execution model, where each task proceeds only after the previous one completes. This approach ensures simplicity, predictability, and ease of debugging, making it well-suited for workflows with fixed logic and minimal variability, such as form approvals or structured back-office operations [15,34]. However, the strict step-by-step coordination limits scalability and responsiveness, making the system less adaptable to dynamic environments or exception-heavy scenarios [33,177].

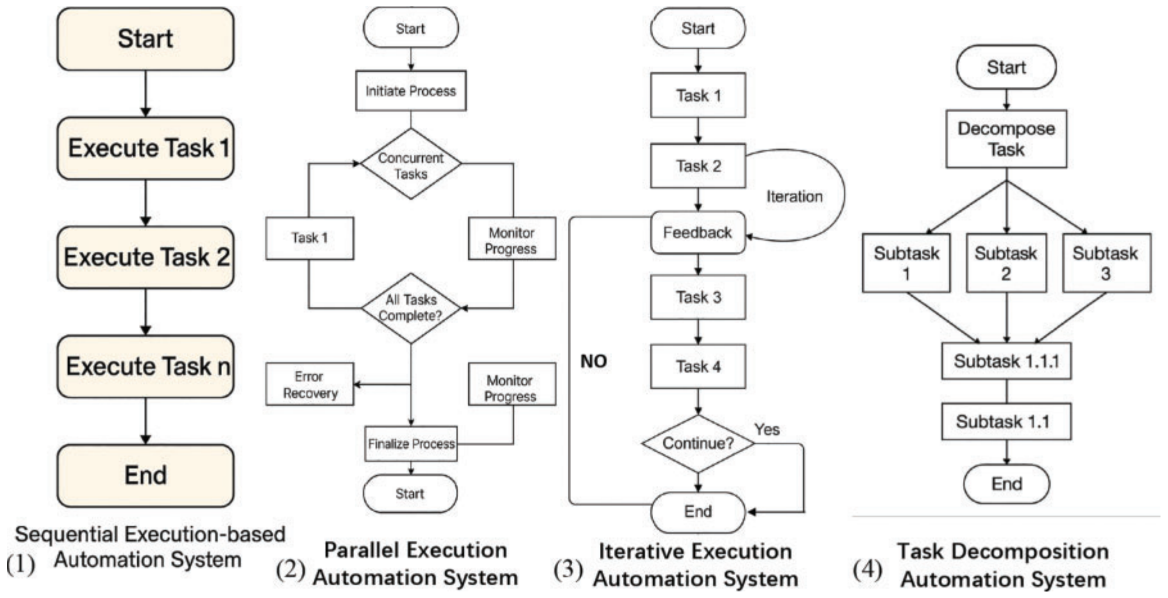


Figure 7: Representative architectures of AI-driven automation systems across functional categories

As illustrated in Fig. 7 (2), *Parallel Execution Automation System* enables concurrent processing of multiple tasks using a graph-based dependency resolution framework [65,130,179]. By distributing workload across agents and allowing simultaneous execution, it significantly improves throughput and performance in multi-tasking or data-intensive applications [49,50]. Nonetheless, the architecture introduces challenges in synchronization, consistency maintenance, and error tracking, which require robust monitoring and coordination mechanisms [184,185].

As depicted in Fig. 7 (3), *Iterative Execution Automation System* focuses on adaptive task flows by incorporating feedback loops and incremental workflow optimization [78,96,142]. This model is highly effective in environments where requirements evolve over time, such as conversational agents or adaptive planning systems, as it allows ongoing refinement of tasks based on observed outcomes. However, its complexity arises from the need for rollback mechanisms, state versioning, and audit trails to ensure compliance and traceability during iterative updates [92,94].

As shown in Fig. 7 (4), *Task Decomposition Automation System* leverages goal-oriented reasoning to break down complex objectives into smaller executable sub-tasks [71,72,198]. By combining neural and symbolic techniques, it supports scalable task planning across diverse domains and agent specializations [31,86]. This architecture excels in modularity and reuse, making it ideal for open-ended or cross-domain problems, though it demands sophisticated control strategies to manage coordination and task interdependencies effectively [39,95].

2.3.1 In-Depth Analysis of Each Execution Mechanisms

Sequential Task Execution: [66,69,75] Exemplified by UiPath [104,106,107] and Zapier [30], sequential models enforce strict linear execution through deterministic scheduling algorithms and finite-state machines. These systems prioritize operational reliability over flexibility, utilizing static priority queues and rule-based triggers to maintain process integrity. Their architecture proves particularly effective in legacy system integration and routine office workflows (e.g., employee onboarding approvals), where predictable patterns dominate. However, this rigid temporal coordination inherently limits throughput and dynamic exception handling, as task latency accumulates linearly and environmental changes require manual workflow redesign [33,177,178].

Parallel Task Execution: [65,130,179] Systems like OWL [120] and MetaGPT [60] implement concurrent processing through distributed task queues with graph-based dependency resolution. These architectures employ lock-free data structures and probabilistic conflict resolution to coordinate simultaneous execution across multi-agent environments. The technical innovation lies in real-time resource monitoring systems that balance computational load across agents while maintaining data consistency through semantic versioning protocols.

Iterative Task Execution: [78,96,142] LangChain [111,140,153] and OpenManus [114,117] exemplify feedback-driven architectures that combine versioned workflow states with reinforcement learning policies. These systems implement semantic differencing algorithms to identify optimization opportunities between iterations, enabling adaptive adjustment of execution paths. The technical challenge centers on designing efficient rollback mechanisms and audit trails that preserve compliance while supporting dynamic workflow evolution.

Task Decomposition Strategies: [71,72,198] Advanced frameworks like Manus [117,119] and AutoGPT [113,159] employ hybrid neuro-symbolic reasoning to decompose abstract goals into executable sub-tasks. Through multi-resolution analysis modules, these systems alternate between symbolic planning and neural subtask generation, enabling collaborative execution across specialized agents [31,86]. The architectural divergence manifests in centralized vs. distributed control philosophies—AutoGPT utilizes monolithic decomposition controllers, while OWL [120] implements decentralized negotiation protocols.

2.3.2 Comparative Analysis and Selection Guidelines

The choice of task execution paradigm requires careful consideration of operational requirements and system constraints [39,95]. As shown in Table 4, each mechanism presents distinct trade-offs across five critical dimensions: implementation complexity, throughput efficiency, dynamic adaptability, modularity, and observability [184,185].

Table 4: Comparative analysis of task execution paradigms

Characteristic dimension	Sequential	Parallel	Iterative	Decomposition
Implementation complexity	Low	Medium-High	High	Medium-High
Throughput performance	Low	High	Medium	Medium-High
Dynamic adaptability	Low	Medium	High	High
Modularity/Reusability	Low	Medium	Medium	High
Debugging & monitoring difficulty	Low	High	High	Medium-High

Sequential execution offers minimal implementation complexity and straightforward debugging [15,34], making it ideal for small-scale workflows with fixed logic patterns, though it sacrifices throughput and adaptability. Parallel architectures [49,50] significantly enhance processing capacity for batch operations through concurrent resource utilization, but introduce synchronization challenges and increase monitoring overhead [186]. Iterative models excel in dynamic environments requiring continuous optimization, employing feedback loops to adapt workflows, albeit with elevated implementation and maintenance costs [92,94]. Task decomposition strategies provide superior modularity for complex multi-domain tasks through specialized agent collaboration [115,192], though demanding sophisticated coordination frameworks.

In conclusion, the evolution of these mechanisms reveals an industry-wide shift from rigid workflow automation to adaptive process orchestration [39,71,95]. Modern automation systems increasingly combine multiple paradigms, while facing persistent challenges in maintaining operational coherence across hybrid coordination models [65,86,130]. For instance, employing decomposition for task planning follows by parallel execution [31,50,49]. In future research, greater emphasis can be placed on self-organizing architectures capable of dynamically selecting execution strategies based on real-time system states and environmental constraints [92,94,187].

2.4 Application Taxonomy of AI-Driven Automation

2.4.1 Primary Benefited Production Aspects of AI-Driven Automation

The application landscape of AI-driven automation tools spans multiple functional domains, each demonstrating unique technical implementations tailored to specific operational requirements [71,86,95]. As detailed in Table 5, these solutions exhibit specialized architectural configurations that address distinct challenges across enterprise workflows.

Customer Support Automation: Modern implementations leverage transformer-based architectures like GPT-4 [156,89,199] and specialized systems such as Cogito [200] to establish multi-modal communication channels. These systems combine natural language understanding with dialog management engines to process customer inquiries through hybrid symbolic-neural approaches. The technical differentiation emerges in their ability to handle context switching between structured FAQs and unstructured conversational flows while maintaining compliance with industry-specific communication protocols [78,92,96].

Table 5: Application domains of AI-driven automation tools

Application area	Technical implementation	Representative tools
Customer support automation	Transformer-based dialog systems with compliance-aware response generation.	GPT-4 [199,201], Cogito [200], MetaGPT [60], Med-Bot [57], IntellBot [52], EduChat [202], NOVI [58], CyberMentor [203]
Data analysis & reporting	Neuro-symbolic data fusion architectures with automated insight validation.	OWL [120], MetaGPT [60], UiPath AI Fabric [106], CypherBench [124], Docling [53], S3 [204], RecMind [205], mAIstro [206]
Document processing & content generation	Multi-stage neural pipelines with style-content decoupling.	DeepL [157,207], GPT-4 [199], LangChain [111,152], Docling [53], MiniRAG [138], Self-RAG [139], Rag-Fusion [208], MM-REACT [79]
Operational task automation	Vision-enhanced RPA with adaptive exception handling.	UiPath [106,107], OpenManus [114], AirSlate [122], SmartFlow [55], LMRPA [33], BreachSeek [163], AutoAgent [123], TidyBot [197]
Cross-platform workflow automation	Neural API mapping with protocol translation engines.	Zapier [30], OWL [120], UiPath AI Fabric [106], ToolLLM [59], AutoGen [91], AgentVerse [65], CrewAI [121], TaskMatrix.AI [23]

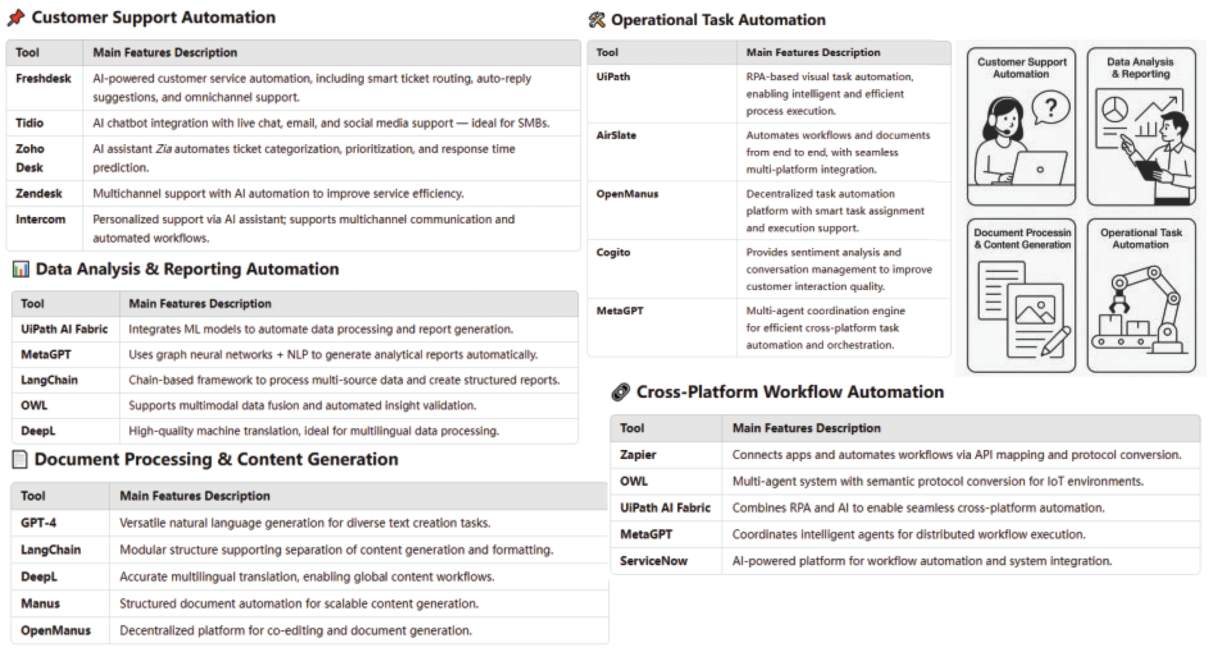
Data Analysis & Reporting [14,34,75]: Tools like OWL [120] and UiPath AI Fabric [106,107] demonstrate advanced data fusion capabilities, integrating structured database queries with unstructured text analysis through neuro-symbolic architectures. LangChain [111] introduces a novel chaining mechanism that connects disparate data sources through semantic linking, while MetaGPT [60] employs graph neural networks to visualize complex data relationships. These systems differ fundamentally from traditional Business Intelligence (BI) tools [17,32,209,210] through their dynamic hypothesis generation capabilities and automated insight validation frameworks.

Document Processing & Content Generation [53,72,198]: This domain combines neural machine translation systems like DeepL [157,207] with generative architectures in GPT-4 [45,199] and structured document processors like Manus [117–119]. The technical innovation lies in multi-stage processing pipelines that first parse document semantics through attention mechanisms, then apply domain-specific templates, and finally generate context-aware outputs. LangChain's contribution emerges through its modular architecture that separates content generation from style formatting, enabling adaptive reuse of document components across different output formats [10,112].

Operational Task Automation [33,66,177]: Systems such as UiPath [106,107] and AirSlate [122] represent the evolution of traditional RPA [15,66,81] through computer vision-enhanced workflow recognition and self-adjusting execution engines. OpenManus [114] introduces a decentralized approach to operational automation through its agent-based task allocation system, particularly effective in distributed procurement processes. The technical differentiation manifests in their exception handling mechanisms, ranging from rule-based fallback systems in UiPath to neural pattern recognition in AirSlate's document-centric workflows [70,178].

Cross-Platform Workflow Automation [65,130,179]: Zapier's [30] AI-enhanced integration framework and OWL's [120] multi-agent automation system exemplify two distinct technical approaches to platform interoperability [19,31,64,192]. While Zapier [30] employs neural API mapping to bridge commercial SaaS applications, OWL [120] utilizes semantic protocol translation for industrial IoT environments. UiPath AI Fabric [106,107] demonstrates a hybrid approach combining RPA connectors with machine learning-based workflow adaptation, particularly effective in legacy system integration scenarios. These systems share common challenges in maintaining data consistency across heterogeneous security models and authentication protocols [182,183].

As illustrates in Fig. 8, it presents the functional taxonomy and ecosystem architecture of prominent AI-driven automation tools. The tools are grouped into five distinct categories: *customer support automation, data analysis and reporting, document processing and content generation, operational task automation, and cross-platform workflow automation* [71,86,95]. Each category reflects a unique application domain and technological foundation, ranging from traditional RPA systems to advanced multi-agent automation systems and cognitive orchestration engines [60,65,111].


Customer Support Automation
Data Analysis & Reporting
Document Processin & Content Generation
Operational Task Automation

Customer Support Automation

Tool	Main Features Description
Freshdesk	AI-powered customer service automation, including smart ticket routing, auto-reply suggestions, and omnichannel support.
Tidio	AI chatbot integration with live chat, email, and social media support — ideal for SMBs.
Zoho Desk	AI assistant *Zia* automates ticket categorization, prioritization, and response time prediction.
Zendesk	Multichannel support with AI automation to improve service efficiency.
Intercom	Personalized support via AI assistant; supports multichannel communication and automated workflows.

Data Analysis & Reporting Automation

Tool	Main Features Description
UiPath AI Fabric	Integrates ML models to automate data processing and report generation.
MetaGPT	Uses graph neural networks + NLP to generate analytical reports automatically.
LangChain	Chain-based framework to process multi-source data and create structured reports.
OWL	Supports multimodal data fusion and automated insight validation.
DeepL	High-quality machine translation, ideal for multilingual data processing.

Document Processing & Content Generation

Tool	Main Features Description
GPT-4	Versatile natural language generation for diverse text creation tasks.
LangChain	Modular structure supporting separation of content generation and formatting.
DeepL	Accurate multilingual translation, enabling global content workflows.
Manus	Structured document automation for scalable content generation.
OpenManus	Decentralized platform for co-editing and document generation.

Operational Task Automation

Tool	Main Features Description
UiPath	RPA-based visual task automation, enabling intelligent and efficient process execution.
AirSlate	Automates workflows and documents from end to end, with seamless multi-platform integration.
OpenManus	Decentralized task automation platform with smart task assignment and execution support.
Cogito	Provides sentiment analysis and conversation management to improve customer interaction quality.
MetaGPT	Multi-agent coordination engine for efficient cross-platform task automation and orchestration.

Cross-Platform Workflow Automation

Tool	Main Features Description
Zapier	Connects apps and automates workflows via API mapping and protocol conversion.
OWL	Multi-agent system with semantic protocol conversion for IoT environments.
UiPath AI Fabric	Combines RPA and AI to enable seamless cross-platform automation.
MetaGPT	Coordinates intelligent agents for distributed workflow execution.
ServiceNow	AI-powered platform for workflow automation and system integration.

Figure 8: Primary functional taxonomy and system architecture of contemporary AI-driven automation tools

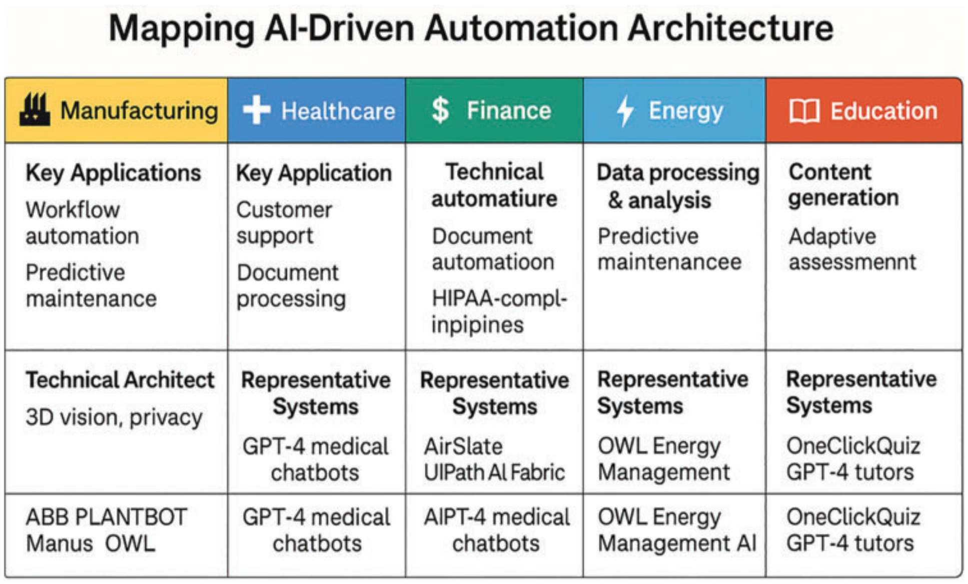
The visual layout highlights the vertical stack of automation complexity, starting from foundational tools like UiPath and Zapier [30,106] at the infrastructure layer [66,75], progressing through mid-layer platforms such as LangChain and MetaGPT [60,111] that facilitate reasoning and task decomposition, and culminating in agent-based ecosystems like Manus [117] and OWL [120] that enable decentralized coordination and emergent behavior [86,92]. By mapping tools according to both functionality and architectural sophistication, the figure provides a comparative perspective on how modern AI systems address varying degrees of autonomy, interoperability, and domain specialization within automation workflows [71,95].

Overall, Fig. 8 explains the role of each tool in automation while revealing new trends like LLM-based reasoning working with multi-agent automation systems. It also points toward the future development of fully adaptive, cognitive automation stacks [94,187].

In conclusion, the application-specific implementations reveal three fundamental technical trends: 1) Increasing specialization of neural architectures for domain-specific constraints [72,198], 2) Growing emphasis on hybrid symbolic-connectionist systems for enterprise compliance requirements [78,96], and 3) Emergence of multi-paradigm integration frameworks to handle heterogeneous automation environments [65,130]. Researchers are currently tackling the challenge of enhancing cross-domain generalization capabilities, while preserving application-specific optimization [85,111,211,212].

2.4.2 Industry-Specific Implementation and Production Scenarios

The convergence of industry-specific requirements and application-driven architectures has shaped the evolution of AI-driven automation technologies [14,34]. As shown in Fig. 9, we provide a structured classification of AI-driven automation architectures across five major industry/human living scenarios: *Manufacturing* [20,81], *Healthcare* [57,14,206], *Finance* [16,213], *Energy* [40,167], and *Education* [202,203]. Each column outlines the key applications, underlying technical frameworks, and representative systems currently deployed in practice. This layered visual categorization captures both the breadth and specificity of AI implementations across domains, ranging from predictive maintenance in manufacturing to adaptive assessment in education [68,202,203,214]. For instance, manufacturing leverages edge-cloud hybrid computing and 3D vision systems [25,211] to support physical automation, while healthcare systems prioritize privacy-preserving AI and compliance frameworks [13,215].



Mapping AI-Driven Automation Architecture

Manufacturing	Healthcare	Finance	Energy	Education
Key Applications Workflow automation Predictive maintenance	**Key Application** Customer support Document processing	**Technical automatiure** Document automatioon HIPAA-compl-inpipines	**Data processing & analysis** Predictive maintenancee	**Content generation** Adaptive assessmennt
Technical Architect 3D vision, privacy	**Representative Systems** GPT-4 medical chatbots	**Representative Systems** AirSlate UIPath AI Fabric	**Representative Systems** OWL Energy Management	**Representative Systems** OneClickQuiz GPT-4 tutors
ABB PLANTBOT Manus OWL	GPT-4 medical chatbots	AIPT-4 medical chatbots	OWL Energy Management AI	OneClickQuiz GPT-4 tutors

Figure 9: Primary functional taxonomy and system architecture of contemporary AI-driven automation tools

Meanwhile, Table 6 presents a unified taxonomy that cross-references industrial domains with their characteristic applications, technical implementations, and representative systems [17,35,209].

Table 6: Integrated Taxonomy of AI-driven automation by industry and application

Industry	**Key applications**	**Technical architecture**	**Representative systems**
Manufacturing	Complex workflow management, Predictive maintenance	Multi-modal perception (3D vision, force sensing), Real-time adaptive control, Edge-cloud hybrid computing	GPT Researcher [44], BOLAA [56], TPTU-v2 [68], Manus [117,119], AutoGPT [113], SuperAGI [116], OWL [120], ABB PLANTBOT [216], WorkGPT [217]
Healthcare	Customer support automation, Document processing	High-precision sensor fusion, Privacy-preserving AI, HIPAA-compliant data pipelines	MedAgents [12], GPT-4 medical chatbots [13,105], Med-Bot [57], mAIstro [206], D-Bot [215], *BD HemoSphere Alta* [218], LLM-TA [219]
Finance	Document automation, Cross-platform integration	Neural-symbolic compliance engines, Dynamic audit trails, SOX/GDPR embedded architectures	TradingGPT [16], Financial RPA [17], EconAgent [32], AirSlate [122], UiPath AI Fabric [106,107], CompeteAI [210], SmartFlow [55]
Energy	Data processing & analysis, Predictive maintenance	Distributed edge AI, Multi-agent grid optimization, Ruggedized hardware integration	OWL Energy Management [120], DeChen Edge AI [220], LLM-OptiRA [167], Hyperautomation [27], Intelligent Automation [211]
Education	Content generation, Adaptive assessment	Cognitive framework alignment, Multi-modal interaction, Pedagogical model integration	NOVI [58], MiniRAG [138], EduChat [202], GPT-4 tutors [199,201], CyberMentor [203], OneClickQuiz [221,222]

The integrated analysis reveals several critical architectural patterns [65,71,86]. Manufacturing systems exemplify the tight coupling between physical automation and AI decision-making, where the mainstream generative AI engine coordinates with industrial robots through sub-millisecond control loops [25,68,211]. This contrasts with healthcare implementations that prioritize ethical constraints, as seen in *BD HemoSphere Alta* [218]'s differential privacy mechanisms that anonymize patient data while maintaining diagnostic accuracy [98,215].

Financial sector tools demonstrate unique hybrid architectures, where AirSlate [122] combines NLP-driven document processing with regulatory compliance checks through neural-symbolic reasoning. The system dynamically generates audit trails by tracing data transformations across workflow steps, achieving both operational efficiency and compliance transparency. Similarly, education technologies like OneClick-Quiz [221,222] implement Bloom's Taxonomy through distilled language models, enabling rapid question generation while maintaining pedagogical alignment [202,203].

Energy management solutions [40,167,216] highlight the growing importance of distributed intelligence, with OWL's [120] multi-agent automation system optimizing grid operations through edge-based predictive models [223,224]. These architectures employ specialized hardware-software codesign to withstand harsh environmental conditions while maintaining real-time processing capabilities [18,19]. Across all domains, the convergence of application-specific requirements and industrial constraints continues to drive architectural innovation in AI-driven automation systems [94,95,142].

2.4.3 Lightweight AI-Driven Automation for Edge Devices, Mobile Systems, or Low-Resource Environments

The surge in Internet of Things (IoT) devices and the increasing deployment of mobile systems and embedded technologies have created a significant need for lightweight AI-driven automation solutions [223,224]. As a result, traditional cloud-based AI solutions, which typically rely on robust computational infrastructures, are not suitable for deployment in these low-resource environments. This section discusses how lightweight AI-driven automation systems are addressing these challenges and enabling effective automation for edge devices, mobile systems, and other low-resource environments.

1. Edge Device Automation: Edge devices, including sensors, cameras, and gateways, play a pivotal role in modern automation by collecting and processing data locally to minimize latency and reduce dependence on cloud infrastructure [220]. These devices often operate in environments where bandwidth is limited or unreliable, making cloud communication less efficient. To address this, edge AI systems are designed with compact models that execute inference tasks with minimal resources. Techniques such as model quantization, pruning, and knowledge distillation are frequently employed to compress large, resource-intensive models into smaller, more efficient versions without sacrificing accuracy [88,89]. For instance, lightweight convolutional neural networks (CNNs) and decision tree-based models have gained popularity for tasks like object detection, and predictive maintenance on embedded systems [21].

2. Mobile Systems and IoT Automation: Mobile systems and IoT devices often have limited power budgets and processing capabilities, yet they need to perform tasks like speech recognition, image processing, and sensor data analysis [160,162]. To enable AI automation on such devices, specialized AI chips and microcontrollers are integrated to offload computationally heavy tasks from the main processor, thus optimizing power usage [223]. This empowers edge devices and mobile systems to perform tasks like image recognition, natural language processing, and personalized recommendations locally [225].

3. Embedded Systems in Low-Resource Environments: Embedded systems in domains like agriculture, healthcare, and industrial automation are increasingly adopting lightweight AI-driven automation solutions [68,211,214]. These systems often rely on custom-designed hardware to meet performance and power constraints, which enable efficient AI processing with minimal power usage [224]. In these contexts, techniques like model compression, low-precision arithmetic, and hardware acceleration help make the automation feasible in low-resource environments [21].

4. The Future of Lightweight AI-Driven Automation: Looking ahead, several trends will further enhance the deployment of AI-driven automation on edge and low-resource devices. Federated learning, for instance, allows devices to collaboratively learn from data without transmitting sensitive information to the cloud, making it ideal for environments where privacy is paramount [223]. Furthermore, advances in AI chip design, including the integration of AI accelerators into microcontrollers, will continue to push the boundaries of what is possible in edge and mobile AI [224].

In conclusion, lightweight AI-driven automation is a rapidly evolving field that is crucial for enabling intelligent systems in edge devices, mobile systems, and low-resource environments [223,224]. These advancements will help bridge the gap between cloud-based automation and real-time, on-device processing, bringing AI-powered automation to a wide range of applications with minimal resource requirements.

2.5 Summary

This section systematically established a novel taxonomy for classifying AI-driven automation technologies, addressing the critical need for a unified framework in this rapidly evolving field. Our analysis delineates four foundational architectural paradigms:

1. Rule-Based Systems: Characterized by deterministic workflows, suitable for structured environments but limited in adaptability.

2. LLM-Driven Approaches: Leveraging natural language understanding for dynamic task orchestration, though challenged by interpretability and data dependence.

3. Multi-Agent Cooperative Models: Enabling complex, distributed problem-solving through agent collaboration, with trade-offs in coordination overhead.

4. Hybrid Architectures: Combining strengths of the above paradigms to balance flexibility and robustness, as exemplified by tools like Manus [117] and OWL [120].

Further, we categorize these technologies by functionality (e.g., NLP-centric vs. RPA-integrated), execution mechanisms (sequential vs. parallel), and human-AI collaboration modes (assistive vs. autonomous). This not only clarifies the current landscape but also highlights gaps in interoperability and scalability, which is the key challenges illustrated in Section 4.

3 Recent Mainstream Applications of AI-Driven Automation

3.1 Overview of Prominent AI-Driven Automation Tools

Recent years have witnessed the emergence of diverse AI-driven automation tools, each addressing distinct challenges across industries. The evolution of AI-driven automation has entered a transformative phase characterized by three fundamental shifts [6,71,95]: (1) from static rule-based systems to dynamic cognitive architectures [78,96,97], (2) from single-agent operations to collaborative multi-agent ecosystems [16,31,65,115], and (3) from domain-specific solutions to cross-platform cognitive frameworks [9,59,111].

As detailed in Table 7, we list several major representative AI-driven automation tools, accompanied by concise descriptions of their core functionalities and technological underpinnings. This progression manifests through distinct technological generations that address emerging challenges in social production automation [25,27,211].

Table 7: Latest statistics of several prominent AI-driven automation tools

AI automatic tools	Release date	Development team	Characteristic and capability description
GPT-4 Series [199,201]	March 2023	OpenAI	Versatile NLP models integrated via APIs to automate content generation, conversational interfaces, and semantic analysis in broader workflows.

(Continued)

Table 7 (continued)

AI automatic tools	Release date	Development team	Characteristic and capability description
LangChain [111,112]	October 2022	Harrison chase	An open-source framework for building language model-powered applications, enabling chained task execution, text generation, and seamless integration with external APIs.
Manus [117,119]	March 2025	Butterfly effect (Monica.im)	A multi-agent automation system with autonomous task planning, execution tracking, and adaptive memory modules for iterative workflow optimization.
OpenManus [114]	March 2025	MetaGPT	A modular extension of Manus, designed for customizable agent deployment in domain-specific automation scenarios.
OWL [120]	March 2025	CAMEL-AI	A scalable, open-source multi-agent automation system supporting cross-platform operations and heterogeneous data source integration for complex industrial workflows.
AutoGPT [113]	April 2023	Toran BruceRichards	A self-directed AI agent combining GPT architectures with reinforcement learning to autonomously generate and execute task sequences.
MetaGPT [60]	July 2023	DeepWisdom & KAUST	A multi-agent automation system optimized for multimodal task allocation, coordination, and dynamic resource management.
UiPath AI Fabric [106,107]	October 2020	UiPath	Enhances robotic process automation (RPA) with AI-driven decision-making, enabling hybrid rule-based and machine learning workflows for enterprise tasks.
Zapier [30]	May 2023	Zapier Inc.	A low-code platform enabling cross-application workflow automation through AI-augmented rule engines and API orchestration.
AirSlate [122]	December 2020	AirSlate Inc.	Combines RPA with AI for automating document-centric processes such as contract management and form processing.
AI-enhanced Tray.io [226]	February 2024	Tray.io Inc.	An enterprise-grade integration platform utilizing AI to streamline API-driven workflow automation across distributed systems.

3.1.1 Architectural Paradigm Shifts

The evolution of AI-driven automation tools demonstrates a clear architectural progression from rule-based systems to cognitive architectures [75,77,105]. Early hybrid RPA-AI solutions like UiPath AI Fabric [106,107] have given way to sophisticated neuro-symbolic systems exemplified by post-2023 developments such as LangChain [10,111,152]. These modern frameworks introduce cognitive chaining capabilities, where LLMs dynamically orchestrate sequential reasoning processes through API composition [59,128]. The transition reflects a broader industry movement from deterministic automation to adaptive, knowledge-driven workflow systems that demonstrate contextual awareness and learning capabilities.

3.1.2 Emerging Design Patterns

Analysis of development methodologies reveals two significant innovation patterns in the field. The academic-industrial fusion model, exemplified by MetaGPT [60]'s joint development between DeepWisdom and KAUST [227], successfully combines industrial-grade scalability [68,211] with academic research rigor in multi-agent coordination.

Concurrently, the open-source ecosystem approach demonstrates remarkable efficacy, as evidenced by LangChain's rapid community adoption, showcasing how community-driven development can accelerate toolchain maturation [111,228]. These AI-driven automation tools collectively establish a new *cognitive automation stack* comprising four layers: (1) Sensorimotor interfaces [4,108,229], (2) Neural reasoning cores [5,141,142], (3) Multi-agent coordination planes [127,129], and (4) Evolutionary adaptation mechanisms [39].

Overall, these above introduced tools exemplify the convergence of AI methodologies, such as LLMs, multi-agent automation systems, and hybrid rule-AI architectures, to address scalability, adaptability, and interoperability challenges in modern automation applications.

3.2 Detailed Discussion of Several Representative AI-Driven Automation Tools

3.2.1 Manus (From Butterfly Effect)

As a representative implementation of a multi-agent automation system, Manus [117,119] operates within an isolated virtual environment and is composed of three primary functional modules: the Planning Module, the Memory Module, and the Tool Use Module. The left part of Fig. 10 illustrates the core architectural design of the *Manus* multi-agent automation system. These components work collaboratively to enable Manus to interpret, plan, and execute complex tasks in an automated and adaptive manner.

The Planning Module serves as the "brain" of the system, responsible for understanding user intentions, decomposing abstract tasks into concrete actions, and generating executable plans. It supports semantic understanding (NLU) [78,96,173], priority sorting [71,86], resource allocation [25,167], DAG-based task decomposition [142,230], and exception handling. The Memory Module empowers the system to maintain coherence and personalization across sessions by storing user preferences, interaction history, and intermediate results. A typical implementation may include components such as user profiling vectors, interaction history databases (e.g., ChromaDB [231]), and short-term memory caches [83,232]. Finally, the Tool Use Module acts as the system's "hands", integrating capabilities such as web search, data processing, code execution [130,233,234], and content generation. This multi-tool orchestration allows Manus to effectively complete diverse and sophisticated tasks.

As illustrated on the right side of Fig. 10, Manus operates based on a structured task execution flow under a *Multiple Agent Architecture* [65,92,94]. The system runs within an independent virtual environment and is capable of handling both simple queries and complex project-level requests. Upon receiving a task input from the user, the system initiates a multi-stage process to understand, decompose, and complete the task effectively.

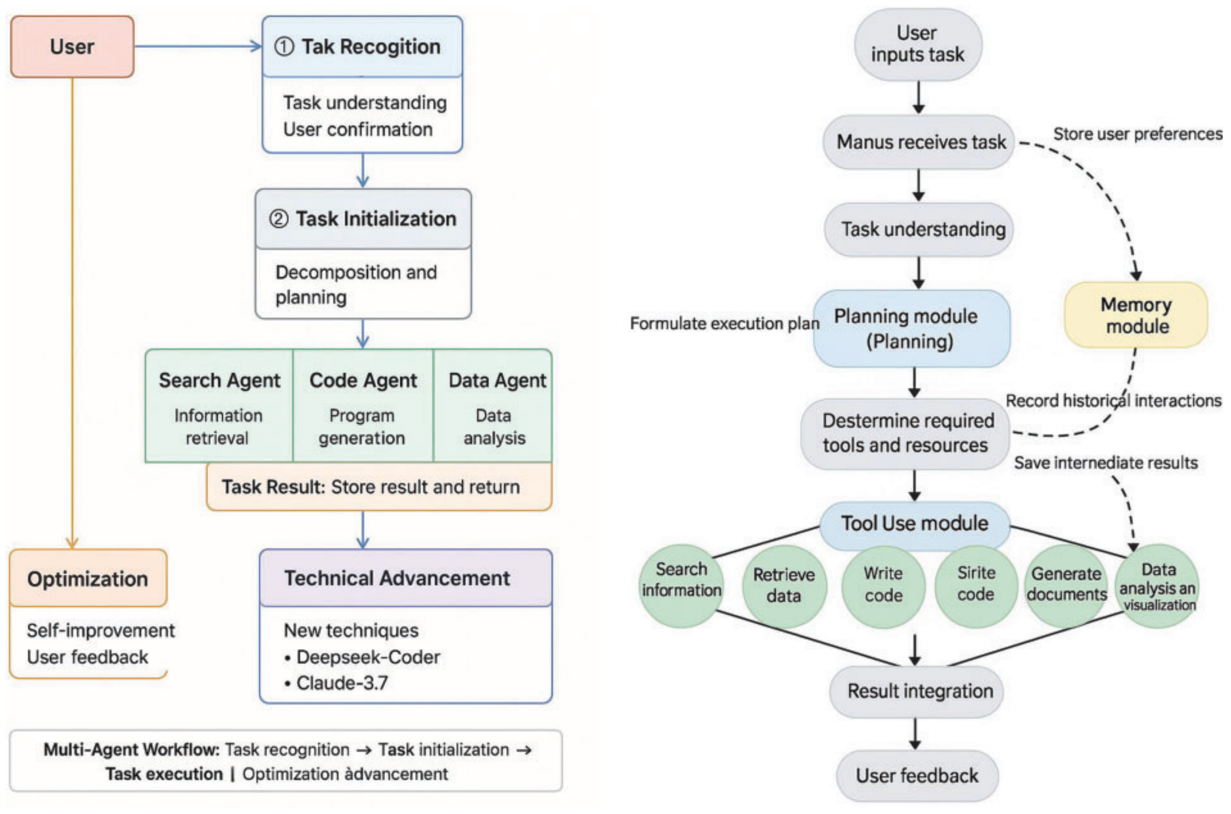


Figure 10: (1) Left: the core module design and composition of the manus multi-agent automation system; (2) Right: manus task processing flow under multiple agent architecture

The first step in this pipeline is **task reception**, where Manus accepts user input in various formats such as text, images, and documents [79,235]. This is followed by the **task understanding** phase, in which Manus leverages NLP techniques to extract key intents and semantic elements from the input [78,96]. During this phase, the Memory Module provides contextual support by supplying previously stored user preferences and interaction history, thereby enhancing personalization and interpretability [64,67,83]. When user goals are ambiguous, the system engages in interactive dialogue to help refine task objectives [92,187].

Once the task intent is clarified, the **task decomposition** stage begins. Here, the Planning Module breaks down complex tasks into a series of executable subtasks, establishes dependencies, and determines execution order [142,230]. This decomposition process typically results in a directed acyclic graph (DAG) [71,198] structure that organizes task execution in a logical and efficient manner. Through this layered and modular architecture, Manus achieves robust adaptability and automation in diverse real-world applications [86,114,117].

As shown in Table 8, the architecture of Manus is organized into six distinct layers, each responsible for a critical aspect of task execution within the multi-agent automation system. The user interface layer enables interaction through a command line or REST API [23,59,128,217], serving as the system's entry point. The control layer, powered by the flow engine, orchestrates task requests and initiates execution processes. At the core layer, planning and invocation flows are designed and instantiated to coordinate agent operations.

The agent layer includes specialized agents-planning, execution, and validation-that work collaboratively to decompose tasks, execute actions, and verify outcomes. The tool layer provides essential capabilities such as web search, code execution, and result validation. Finally, the infrastructure layer ensures system stability and extensibility through configuration management, logging, tool registration, and reusable task templates.

Table 8: Layered architecture of the **Manus**'s task execution system [117]

Layer	Component	Function
User interface layer	Command line interface/REST API	Users can send instructions or requests via the CLI or REST API to enter the system's flow engine for processing.
Control layer	Flow engine	As the core control center, it receives user requests, parses them, and forwards the demand to the planning engine, invoking agents and creating execution flow instances.
Core layer	Planning flow/Invocation flow/Flow instance	Planning Flow: Responsible for execution plan design. Invocation Flow: Responsible for execution path and agent invocation. Flow Instance: Executes specific instance flows, connecting and coordinating agents to perform tasks.
Agent Layer	Planning agent/Execution agent/Validation agent	Planning Agent: Responsible for task decomposition, tool invocation, and task orchestration. Execution Agent: Executes specific tasks (e.g., running code). Validation Agent: Validates whether task results meet expectations.
Tool layer	Search tool/Code tool/Validation tool	Search Tool: Provides information retrieval capabilities. Code Tool: Executes scripts or performs code generation. Validation Tool: Verifies if task results are valid.
Infrastructure layer	Configuration system/Logging system/Tool registration center/Template repository	Configuration System: Provides configuration parameters to support planning agents. Logging System: Records runtime logs for agent auditing and traceability. Tool Registration Center: Manages tools and supports agent execution. Template Repository: Stores reusable task templates for planning agents.

Technical Features of Manus: Manus [117] stands out in the field of AI agents due to its advanced architecture and a range of technical innovations that enable highly autonomous and efficient task execution. One of its most distinguishing features is its **autonomous planning capability** [114,118]. Unlike conventional tools that rely heavily on user input or predefined workflows, Manus can independently reason, plan, and execute tasks. In the General AI Assistant Benchmark (GAIA) [236], which evaluates real-world problem-solving capabilities of general-purpose AI assistants, Manus [117,119] achieved state-of-the-art (SOTA) performance with a 94% success rate in completing complex tasks automatically.

Another key strength of Manus lies in its **context understanding** [117]. The system excels at interpreting vague or abstract user inputs, allowing it to identify user needs with minimal instruction. For instance, a user can describe the content of a video, and Manus will locate the corresponding link across platforms. This context-awareness supports sustained multi-turn interactions [83,92,187], with the system capable of maintaining coherent dialogues over ten or more turns, greatly enhancing the user experience.

Manus is built upon a **multi-agent architecture**, enabling collaborative execution of tasks across different functional agents. Similar to Anthropic's Computer Use paradigm [117,119], Manus runs within a secure and isolated virtual machine, allowing specialized agents to handle planning, memory, and tool usage in parallel. This modular design ensures both scalability and fault tolerance when dealing with complex workflows.

Furthermore, **tool integration** is a hallmark of Manus's operational strength [59,123,237]. It can automatically invoke and coordinate various tools for web search, data analysis, code generation, document creation, and more. This flexible integration allows Manus to handle a wide spectrum of tasks, from information retrieval to sophisticated analytical and creative work. The architecture also supports custom tool plugin development, making it extensible to specific application domains [65,94,114].

Finally, Manus emphasizes **security and system robustness**. By executing tasks within a gVisor-based sandboxed environment [51,134,223,224], the system ensures process isolation, resource control, and execution safety. Combined with intelligent task scheduling mechanisms and modular agent design, Manus maximizes both efficiency and adaptability while maintaining high standards of stability and security [114,118].

3.2.2 OpenManus (From MetaGPT Team)

The innovative design of OpenManus [114] centers around building a highly lightweight agent framework that emphasizes modularity and extensibility. It defines the functionality and behavior of agents through a combination of pluggable tools and prompts, significantly lowering the barrier for developing and customizing agents [21,94,147]. Prompts determine the agent's reasoning logic and behavioral patterns, while tools provide actionable capabilities such as system operations, code execution, and information retrieval. By freely combining different prompts and tools, new agents can be rapidly assembled and empowered to handle a wide range of task types [31,113,115,238].

As depicted in Fig. 11, the OpenManus Automation Framework consists of four key components: *a modular multi-agent automation system, tool integration, configuration and model support, and a bottom-up execution workflow*. At its core, the framework includes specialized agents such as the Manus Main Agent [117,119], Planning Agent [148,175], ToolCall Agent [59,123], and customizable xAgents [92,94], each responsible for planning, task execution, and tool invocation. These agents interact seamlessly with integrated tools like vector databases [144,150], Python/NLP libraries [157,158], web APIs [59,237], and data visualization modules. The framework is further supported by flexible configuration files and multi-model integration capabilities, while the execution process is managed through a MetaGPT-based workflow [60,227] and a dynamic task queue system [8,56], ensuring scalable and efficient agent collaboration.

Essentially, OpenManus [114] is a multi-agent automation system. Unlike the one-shot [46,72], all-in-one response style of a single large model, a multi-agent automation system tackles complex real-world problems through an iterative cycle of planning, execution, and feedback. In the design of OpenManus, the core concept is illustrated on the right side of Fig. 11.

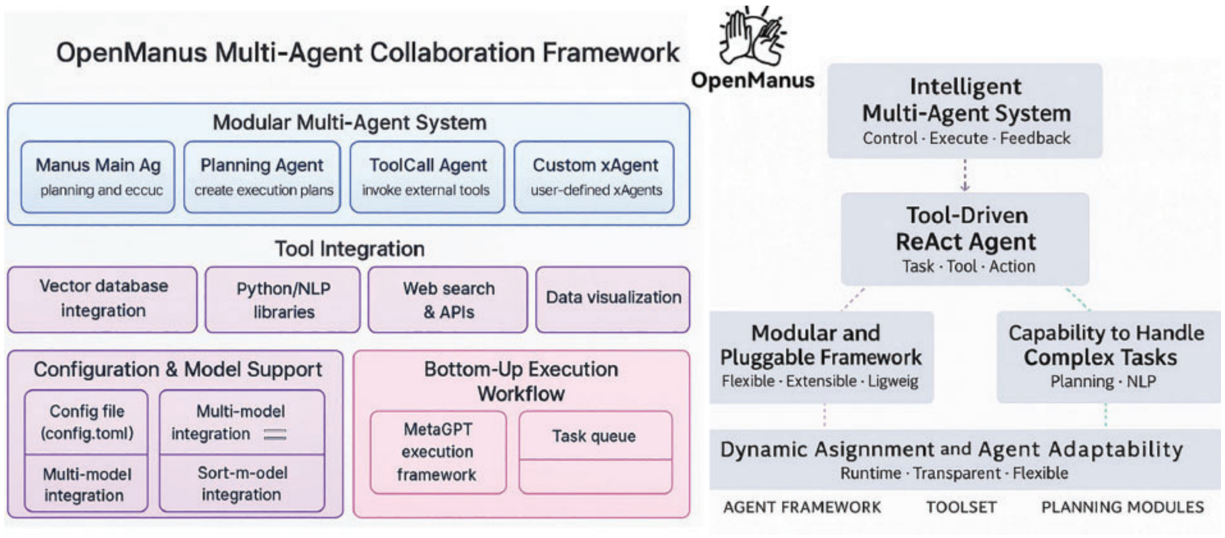

OpenManus Multi-Agent Collaboration Framework
Modular Multi-Agent System
Manus Main Ag
planning and eccuc
Planning Agent
create execution plans
ToolCall Agent
invoke external tools
Custom xAgent
user-defined xAgents
Tool Integration
Vector database integration
Python/NLP libraries
Web search & APIs
Data visualization
Configuration & Model Support
Config file (config.toml)
Multi-model integration
Multi-model integration
Sort-m-odel integration
Bottom-Up Execution Workflow
MetaGPT execution framework
Task queue
OpenManus
Intelligent Multi-Agent System
Control · Execute · Feedback
Tool-Driven ReAct Agent
Task · Tool · Action
Modular and Pluggable Framework
Flexible · Extensible · Ligweig
Capability to Handle Complex Tasks
Planning · NLP
Dynamic Asignnment and Agent Adaptability
Runtime · Transparent · Flexible
AGENT FRAMEWORK
TOOLSET
PLANNING MODULES

Figure 11: (1) Left: The roles and relationships among the agents in OpenManus; (2) Right: The architectural design philosophy of OpenManus

3.2.3 Optimized Workforce Learning for General Multi-Agent Assistance (OWL)

Optimized Workforce Learning for General Multi-Agent Assistance (OWL) [120] from Camel-AI[6] is a cutting-edge framework for multi-agent collaboration that pushes the boundaries of task automation, built on top of the CAMEL-AI Framework[7]. The left side of Fig. 12 shows the system architecture overview of OWL. As seen in this sub-figure, the architecture of OWL consists of three main components: User Query, Actor Agents, and Tools Pool [31,113,120]. The system starts with a User Query, where the user initiates a task request. Actor Agents form the core module responsible for intelligent coordination, while the Tools Pool offers external interfaces needed for operations like web browsing, document parsing, and code execution [59,237].

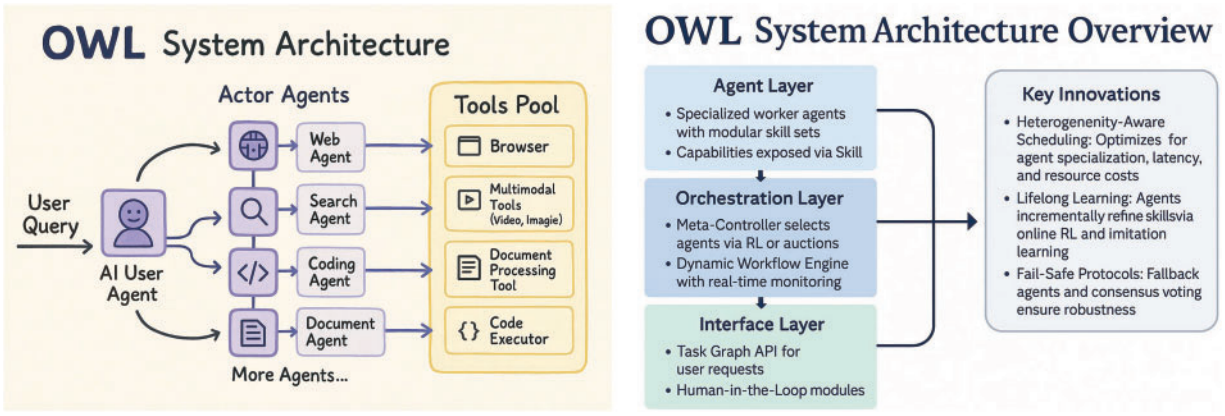

OWL System Architecture
Actor Agents
User Query
AI User Agent
Web Agent
Search Agent
Coding Agent
Document Agent
More Agents...
Tools Pool
Browser
Multimodal Tools (Video, Imagie)
Document Processing Tool
Code Executor
OWL System Architecture Overview
Agent Layer
• Specialized worker agents with modular skill sets
• Capabilities exposed via Skill
Orchestration Layer
• Meta-Controller selects agents via RL or auctions
• Dynamic Workflow Engine with real-time monitoring
Interface Layer
• Task Graph API for user requests
• Human-in-the-Loop modules
Key Innovations
• Heterogenenity-Aware Scheduling: Optimizes for agent specialization, latency, and resource costs
• Lifelong Learning: Agents incrementally refine skillsvia online RL and imitation learning
• Fail-Safe Protocols: Fallback agents and consensus voting ensure robustness

Figure 12: (1) Left: The system architecture overview of OWL; (2) Right: The system technical implementation overview of OWL

[6]Camel-AI: https://www.camel-ai.org/ (accessed on 09 July 2025).
[7]OWL Github: https://github.com/camel-ai/owl (accessed on 09 July 2025).

Among the Actor Agents, the AI User Agent and Assistant Agent work together to manage task decomposition and coordination [31,113,115]. The AI User Agent acts as the task initiator, and the Assistant Agent handles scheduling. Supporting them are functional agents-such as Web, Search, Coding, and Document Agents-each responsible for specific tasks like browser interaction, information retrieval, code execution, and document parsing [59,237].

The core functionalities of OWL include real-time information retrieval through online search engines such as *Wikipedia* [144] and *Google Search* [150], as well as multimodal processing capabilities that support videos, images, and audio from both internet sources and local storage. It enables automated browser interactions using the *Playwright* framework, allowing for actions like scrolling, clicking, typing, downloading, and navigating through web pages. Additionally, OWL supports document parsing, extracting information from Word, Excel, PDF, and PowerPoint files and converting the content into plain text or Markdown [53,229]. It also features code execution capabilities, allowing users to write and run Python code through an integrated interpreter.

The optimized workforce Learning for general multi-agent assistance (OWL) [120] framework is designed as a scalable, adaptive, and decentralized system for coordinating heterogeneous AI agents in dynamic environments. The right side of Fig. 12 shows a comprehensive feature and technical description of the OWL system. Its architecture consists of four interconnected layers: Agent Layer, Orchestration Layer, Learning Layer, and Interface Layer.

Specifically, (1). *Agent Layer* hosts a diverse pool of specialized worker agents (e.g., LLM-based, rule-based, or hybrid), each with modular skill sets. These agents expose their capabilities through a standardized `Skill API`, enabling dynamic task delegation and interoperability [31,113,115,238]. (2). *Orchestration Layer* employs a `Meta-Controller` to optimize agent selection via reinforcement learning (RL) [85,168] or auction-based mechanisms [16,65], ensuring efficiency, cost-effectiveness, and constraint satisfaction. Additionally, a `Dynamic Workflow Engine` constructs multi-agent automation pipelines (modeled as DAGs) for complex tasks, with real-time performance monitoring and adaptation [8,56]. (3). *Learning Layer* integrates `Federated Learning` to enable secure knowledge sharing among agents without raw data exchange, preserving privacy. It also leverages `Cross-Agent Transfer Learning` to generalize skills across domains, using shared embedding spaces or policy distillation techniques [135,175]. (4). *Interface Layer* provides a `Unified Task Graph API` to abstract high-level user requests into decomposable subtasks [31,113,115]. It also incorporates `Human-in-the-Loop (HITL)` modules [188,189], offering explainability dashboards and corrective feedback channels for human oversight.

Overall, OWL [120]'s architecture is agnostic to agent paradigms (LLMs, robots, etc.), making it suitable for enterprise automation, embodied AI, and large-scale assistive systems. With its flexible modular design and powerful tool integration capabilities, OWL is becoming a key foundation for open-source AI-driven automation applications.

3.2.4 LangChain (From Harrison Chase @LangChain Inc.)

LangChain [111,112] is a modular framework designed to facilitate the development of multi-language and multi-scenario AI applications. As a widely adopted open-source framework, LangChain is designed to assist developers in building artificial intelligence applications. By providing standardized interfaces for chains, agents, and memory modules, LangChain simplifies the development process of applications based on LLMs.

Specifically, as illustrated in Fig. 13, the architecture adopts a layered design that promotes extensibility, modularity, and cross-platform compatibility. Each layer in the stack is responsible for a specific aspect of system functionality, from foundational APIs [80,128] to high-level cognitive orchestration [74].

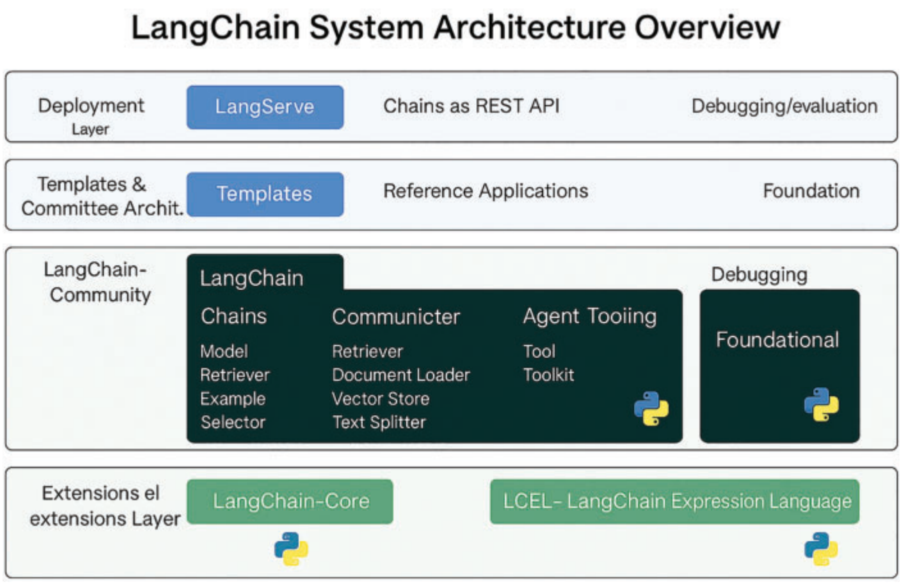


Figure 13: A comprehensive overview of the system architecture of LangChain [111]

At the top of Fig. 13, the **LangServe** module provides the capability to deploy LangChain [111] workflows as RESTful APIs [239]. This makes it easy to integrate AI-powered chains into production environments, particularly within microservice architectures. Built with Python[8], LangServe offers a standardized interface for external systems to call AI logic seamlessly. Beneath the deployment layer, LangChain offers a set of *Templates* and *Committee Architectures*, which serve as reference implementations for common tasks like question answering and document analysis. The heart of the system lies in the **LangChain** module itself [111]. This layer defines three primary capabilities: *Chains* for sequential task composition, *Agents* for dynamic decision-making, and *Retrieval Strategies* for data fetching based on similarity or keywords.

As shown in the center of Fig. 13, the **LangChain-Community** layer includes a rich ecosystem of pluggable modules for Model I/O, Prompting, Output Parsing, Document Loading, Vector Store integration, and Agent Toolkits. This layer enables developers to rapidly prototype and extend LangChain-based applications using components such as FAISS [240], Milvus [241], and external LLM APIs (e.g., ChatGPT [156,45,199] or LlaMa [88,89]).

At the bottom of Fig. 13, the **LangChain-Core** module defines abstract interfaces and reusable infrastructure for chains and agents. Alongside it, the *LangChain Expression Language (LCEL)* [242] provides a declarative syntax for chaining components. LCEL significantly reduces boilerplate code by allowing developers to describe "what to do" instead of "how to do it", improving focus on business logic while supporting streaming, async, and service composition.

LangChain's architecture supports a wide range of use cases, from retrieval-augmented generation (RAG) [138,139,208,229] to multi-agent collaboration workflows [126,175]. With built-in support for both

[8]Python.Org: https://www.python.org/ (accessed on 09 July 2025).

Python and JavaScript[9], LangChain is suitable for backend, frontend, and hybrid AI systems. Its modularity allows for rapid iteration, and its deployment tooling ensures a smooth transition from prototyping to production.

Moreover, LangGraph [228] is a key extension within the LangChain ecosystem, specifically designed for building stateful, multi-agent dynamic workflows [140,243,111]. It addresses the gaps in LangChain for complex cyclic processes and real-time interaction scenarios, offering developers a more flexible and efficient solution. Core features of LangGraph include cyclic graph support (loops and conditional branches between nodes), fine-grained state control (e.g., dynamic graph modification), and built-in persistence capabilities (e.g., task interruption recovery and human intervention). These features of LangGraph empower developers to effortlessly implement AI applications requiring iterative decision-making (such as adaptive RAG pipelines and multi-agent automation systems), while seamlessly integrating with the LangChain ecosystem.

3.2.5 LlamaIndex (From Jerry Liu @LlamaIndex)

LlamaIndex[10] [88,244] is a specialized framework designed to optimize the development of Retrieval-Augmented Generation (RAG) applications [46,139,208,229], excelling in multiple technical dimensions. Compared to well-known projects like Langchain [111,152], LlamaIndex distinguishes itself through its focused domain optimization and innovative design philosophy, offering users a more efficient and specialized development experience for RAG applications. LlamaIndex demonstrates exceptional versatility in handling diverse data formats.

As seen in the left side of Fig. 14, LlamaIndex's data processing pipeline demonstrates its capability to ingest diverse data sources including databases, structured documents, and unstructured APIs [144,150,229]. This framework transforms raw data through LLM processing and vectorization, creating optimized representations for retrieval tasks. This visualization highlights the system's robust data ingestion and preprocessing capabilities that form the foundation for effective RAG applications. The right side of Fig. 14 showcases LlamaIndex's supported use cases, ranging from Q&A systems and structured data extraction to semantic search and agent-based applications. This demonstrates the framework's versatility in addressing various NLP tasks while maintaining its specialized focus on retrieval-augmented generation scenarios. The separation of these components reflects LlamaIndex's modular architecture, allowing developers to implement specific functionalities while benefiting from the framework's optimized RAG core [138,229,244].

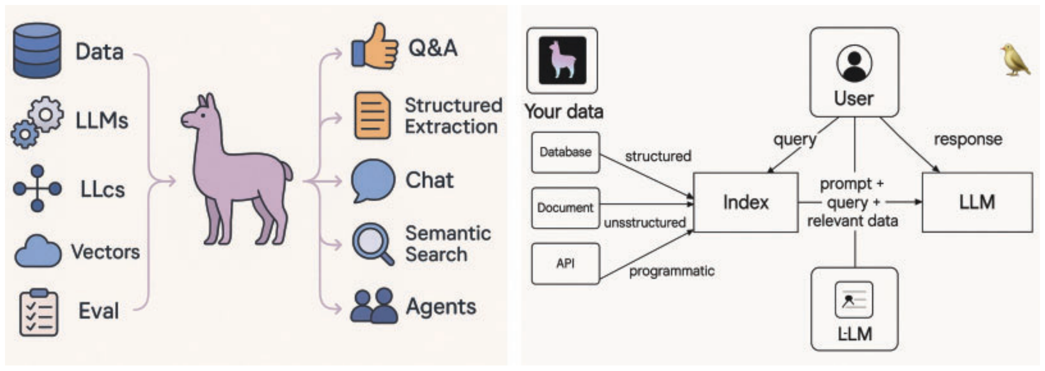


Figure 14: LlamaIndex architecture overview: (1) Left: Supported application scenarios of LlamaIndex; (2) Right: Data processing workflow of LlamaIndex

[9]JavaScript: https://web.developers.google.cn/javascript (accessed on 09 July 2025).
[10]LlamaIndex: https://docs.llamaindex.ai/en/stable/ (accessed on 09 July 2025).

And its key features and advantages-including Robust Data Ingestion & Preprocessing [244,144,150], Optimized Indexing & Query Architecture [46,138,244], Native Multimodal Support [79,162], Production-Grade Scalability [8,56], Modular & Extensible Design [94,111], Ecosystem Integration [157,158,123] can be concluded as follows:

First, LlamaIndex [244] excels in data ingestion and preprocessing. It not only supports a wide range of structured and unstructured data formats but, more importantly, ensures high-quality data encoding into LLM memory through flexible text splitting and vectorization mechanisms [144,150,229]. This lays a solid foundation for contextual understanding during the generation phase.

At the same time, LlamaIndex provides a rich selection of indexing data structures and query strategies, enabling developers to fully leverage scenario-specific efficiency advantages and achieve high-performance semantic retrieval. Such targeted optimizations are undoubtedly critical for RAG applications [46,138].

Another notable highlight is LlamaIndex's native support for multimodal data (such as images and videos) [79,162]. By integrating with leading vision-language models, it can introduce rich cross-modal context into the RAG generation process, adding new dimensions to the output. Undoubtedly, this will pave the way for numerous innovative applications.

Beyond core data management capabilities [245], LlamaIndex also emphasizes engineering best practices for RAG application development. It offers advanced features such as parallelized queries and Dask-based distributed computing support, significantly improving data processing efficiency and laying the groundwork for large-scale production deployment [8,56].

From an architectural perspective, LlamaIndex adheres to modular and extensible design principles. Its flexible plugin system allows developers to easily incorporate custom data loaders, text splitters, and vector indexing modules, catering to diverse and personalized requirements across different scenarios [94,111].

Additionally, seamless integration with the open-source ecosystem is an inherent strength of LlamaIndex. It provides out-of-the-box compatibility with popular tools and frameworks like Hugging Face[11] and FAISS[12] [158,159], enabling users to effortlessly leverage cutting-edge AI/ML capabilities and accelerate the development of innovative solutions.

As a professional-grade tool dedicated to RAG applications [46,139,208], LlamaIndex has become an ideal complement to general-purpose frameworks like Langchain. Developers can now freely choose between LlamaIndex's optimized, high-efficiency approach and Langchain's versatile, flexible paradigm based on their specific needs, maximizing both development efficiency and product quality.

As an evolving framework, LlamaIndex [244] continues to enhance its modeling capabilities for complex scenarios while integrating emerging advancements in LLM architectures and RAG methodologies. Future development focuses on intelligent auto-optimization features and expanded reference implementations to accelerate enterprise adoption.

3.2.6 AutoChain (From Forethought-AI)

AutoChain[13] [246] is a lightweight yet extensible framework proposed by Forethought-AI[14]. It builds upon the foundations of LangChain [111,140] and AutoGPT [113], offering developers a more efficient and flexible approach to building conversational AI agents. AutoChain's core philosophy revolves around three key principles: simplicity, customizability, and automation.

[11] Hugging Face: https://huggingface.co/ (accessed on 09 July 2025).
[12] FAISS: https://faiss.ai/ (accessed on 09 July 2025).
[13] AutoChain Github: https://github.com/Forethought-Technologies/AutoChain (accessed on 09 July 2025).
[14] AutoChain-AI: https://autochain.forethought.ai/ (accessed on 09 July 2025).

As seen in Fig. 15, the framework deliberately maintains a minimalist architecture to reduce cognitive overhead, abstracting fundamental LLM application workflows into intuitive building blocks. The workflow of AutoChain [246], that shown in Fig. 15, facilitates automated interactions between an intelligent agent and a user. The process begins by receiving and storing the user's query. The agent then evaluates whether to continue the interaction based on predefined conditions. If the maximum iteration limit is reached, it terminates with a final response; otherwise, the agent proceeds to the *Plan* phase to determine the next action (e.g., generating a response or requesting clarification). When external tools are required, the system checks if additional information is needed. If not, it executes the tool and appends the results to the input stream, storing them as a *Function Message*. The loop is controlled through binary decisions between *AgentAction* and *AgentFinish*, ultimately producing the agent's output [92,247,248].

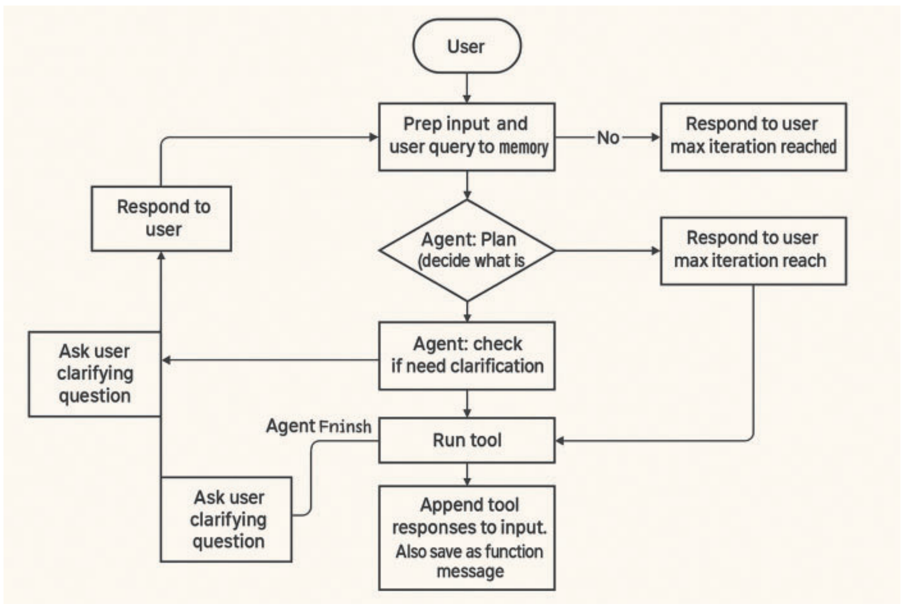

User
Prep input and user query to memory
No
Respond to user max iteration reached
Respond to user
Agent: Plan (decide what is
Respond to user max iteration reach
Ask user clarifying question
Agent: check if need clarification
Agent Fninsh
Run tool
Ask user clarifying question
Append tool responses to input. Also save as function message

Figure 15: The architecture framework of AutoChain [246]: Agent workflow with conditional branching, external tool calls, and termination control

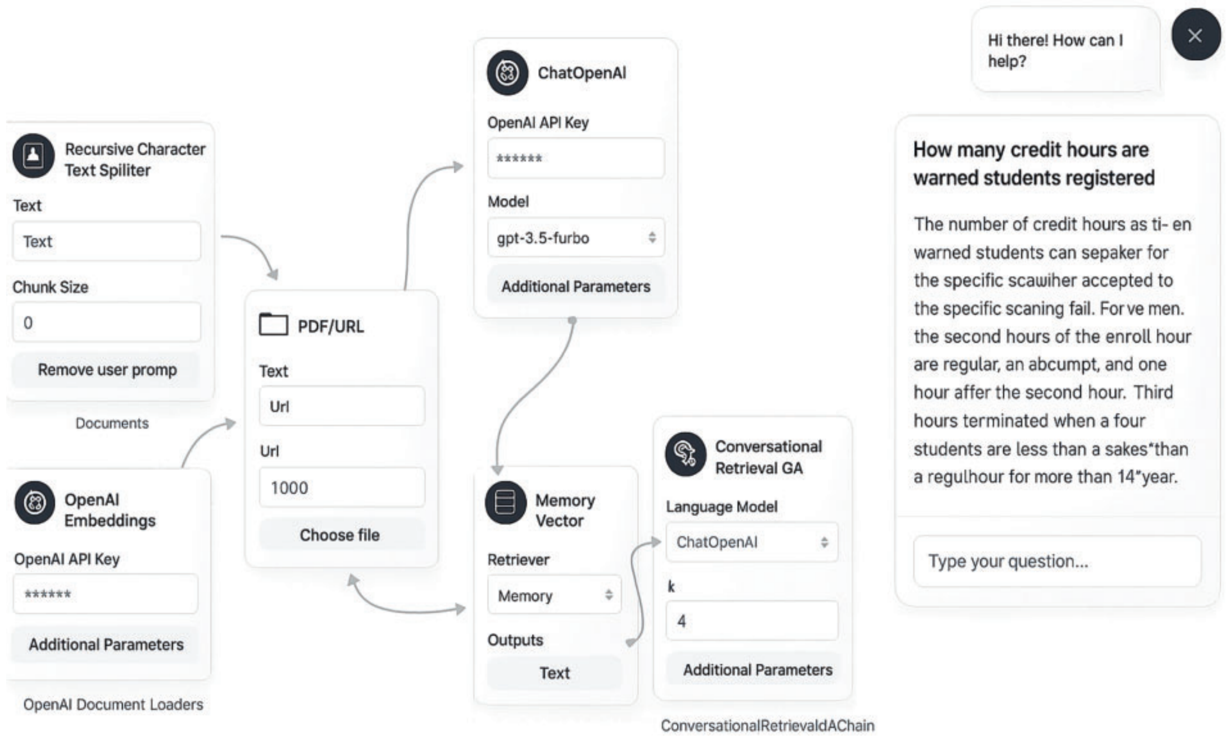
Unlike more complex alternatives, AutoChain [246] provides unparalleled customization through pluggable tools, data sources, and decision modules, enabling developers to create tailored solutions for unique use cases. This "embrace differentiation" approach is complemented by built-in conversation simulation capabilities that automate agent evaluation across diverse interaction scenarios, significantly accelerating development cycles.

AutoChain [246]'s balanced design caters to multiple user groups: beginners benefit from its gentle learning curve when creating basic dialogue agents [51,52,105], experienced LangChain users appreciate its familiar-but-simpler concepts for rapid prototyping, while AI researchers value its clean-slate extensibility for developing novel paradigms [31,113].

3.2.7 Flowise AI (From FlowiseAI Inc.)

Flowise AI[15] [249] emerges as an innovative open-source platform that significantly lowers the barriers to building LLM-based [37,250] applications through its no-code, drag-and-drop visual interface[16]. Unlike traditional coding-intensive frameworks, Flowise enables developers to construct sophisticated LLM workflows by simply connecting pre-built components, making AI application development accessible to non-programmers while maintaining professional-grade capabilities.

As shown in Fig. 16, this workflow demonstrates a typical conversational retrieval QA pipeline of Flowise-AI. The process begins with document ingestion using a PDF loader and text splitter, followed by generating vector embeddings through OpenAI's embedding API. These embeddings are stored in an in-memory vector store, enabling efficient semantic search. A user query is then processed by a language model (ChatOpenAI) [156,43], which interacts with the memory retriever to fetch relevant context and return accurate answers. Flowise-AI allows researchers and developers to build customized LLM applications without extensive programming knowledge, making it ideal for rapid prototyping and production-ready deployments [29,37,250].



Hi there! How can I help?

How many credit hours are warned students registered

The number of credit hours as ti- en warned students can sepaker for the specific scauwher accepted to the specific scaning fail. For ve men. the second hours of the enroll hour are regular, an abcumpt, and one hour affer the second hour. Third hours terminated when a four students are less than a sakes"than a regulhour for more than 14"year.

Figure 16: Automatic conversational retrieval architecture of Flowise AI

The most innovative aspect of Flowise lies in its deep integration with LangChain's powerful framework, exposing its full functionality-including LLM orchestration, chained operations, and data augmentation - through intuitive visual modules. The platform offers native compatibility with leading LLM providers (Anthropic, OpenAI, Cohere) [45,199,201] and data ecosystems (Pandas, SQL, Web APIs) [59,157,158], ensuring users can leverage cutting-edge AI capabilities without infrastructure complexities.

[15] Flowise AI Github: https://github.com/FlowiseAI/Flowise (accessed on 09 July 2025).
[16] Flowise AI: https://flowiseai.com/ (accessed on 09 July 2025).

Notably, Flowise combines this accessibility with enterprise-ready extensibility, featuring API support and embeddable deployment options for integration into diverse software environments. This unique combination of no-code simplicity, LangChain-powered sophistication, and ecosystem interoperability positions Flowise as a pivotal tool in the democratization of LLM technologies, enabling both individuals and organizations to rapidly develop and deploy AI solutions [37,250].

3.3 Comparisons of AI-Driven Automation Tools

The systematic comparison presented in Table 9 reveals fundamental architectural trade-offs among contemporary AI-driven automation tools, highlighting three critical dimensions of differentiation in their design philosophies.

Table 9: Comparative analysis of AI-driven automation tools

Aspects\ Tools	**Manus** [117,119]	**OpenManus** [114]	**OWL** [120]	**LangChain** [111]	**AI-Driven UiPath** [106]
Task complexity	High-Multi-agent, planning and execution	Moderate-Simplified modular design	High-Multi-agent collaboration	Moderate-Focus on NLP and workflows	Moderate-Focused on RPA with AI
Memory	Long-term with user history	Short-term task-specific memory	Incremental, integrates external data	External knowledge integration	Task-specific memory with reusable workflows
Architecture	Multi-agent automation system, task decomposition	Modular plug-and-play framework	Multi-agent with cross-platform support	Rule-based with AI-driven workflows	Hybrid-RPA with AI-powered tools
Execution flow	Sequential and parallel with feedback loops	Sequential, task division, execution	Parallel execution, multi-agent coordination	Sequential task execution	Sequential, linear task execution
Tool integration	Extensive custom tool integrations	High flexibility in tool choice	Cross-platform tool integration	Focused on NLP and AI workflows	Deep integration with enterprise tools
Memory manage-ment	Task-dependent and user-specific memory	Task-dependent, short-term memory	Adaptive, incremental learning	External, dynamic memory	Task-based, context-driven memory

First, the memory management spectrum demonstrates a clear progression from transient task-specific implementations (OpenManus) [114] to sophisticated hybrid approaches combining long-term user context (Manus) [117] with external knowledge integration (LangChain and LangGraph) [111,228]. This evolution reflects the growing industry demand for systems that balance operational efficiency with contextual awareness, particularly evident in OWL's adaptive incremental learning mechanism that bridges internal and external data sources [120,167].

Second, execution flow patterns diverge significantly between sequential-linear models (like UiPath AI Fabric [106,107]) and parallel-coordinated approaches (like OWL [120]), with Manus occupying a unique middle ground through its hybrid sequential-parallel execution with feedback integration. This architectural choice directly impacts each tool's suitability for different workflow complexities, where LangChain's sequential NLP-focused processing contrasts sharply with OWL's multi-agent parallel coordination for industrial-scale automation.

Third, the integration capabilities expose a fundamental tension between specialization and generalization. While LangChain [111] maintains tight focus on NLP and AI workflow integration, UiPath AI Fabric [106] prioritizes depth in enterprise tool connectivity, and OWL [120] emphasizes cross-platform interoperability. Manus and OpenManus [114,117] demonstrate opposing approaches to customization—the former through extensive custom tool integration and the latter via modular plug-and-play components—representing distinct solutions to the flexibility-scalability trade-off.

These comparative insights suggest that modern automation platforms are evolving along two orthogonal axes: (1) from single-agent to collaborative multi-agent automation architectures [91,129,136,251], and (2) from rigid workflow enforcement to adaptive process orchestration [102,122,217]. The emerging generation of tools (exemplified by Manus and OWL [120]) increasingly combines cognitive flexibility with industrial-grade reliability, though significant challenges remain in achieving seamless interoperability across these divergent architectural paradigms.

3.4 Scenario-Based Comparative Analysis of AI-Driven Automation Tools

To provide a more concrete understanding of the strengths and limitations of different AI-driven automation tools, we further conduct a scenario-based comparative analysis, compared with Section 3.3. This evaluation examines how each tool performs under three representative use cases: complex workflow automation, and enterprise-scale RPA integration.

The scenario-based comparison in Table 10 reveals critical trade-offs in tool selection across different automation contexts. For complex workflows requiring dynamic adaptation (Scenario Z1), systems like Manus [117] and OWL [120] demonstrate superior capabilities in handling iterative refinement and parallel execution, though they demand greater technical expertise for customization. In contrast, tools like UiPath AI [106] offer immediate productivity gains for standardized enterprise processes but sacrifice flexibility in dealing with unstructured exceptions.

Table 10: Scenario-based comparative analysis of AI-driven automation tools

Scenario	Tool	Strengths	Limitations
Z1: Complex Workflow Automation	Manus [117]	Advanced task decomposition; supports iterative refinement via feedback loops	High setup complexity; requires custom agent tuning
	OWL [120]	Multi-agent coordination excels at parallel subtask execution	Limited native document processing tools
	LangChain [111]	Strong NLP integration for text-heavy workflows	Weak handling of non-textual data (e.g., forms, tables)

(Continued)

Table 10 (continued)

Scenario	Tool	Strengths	Limitations
Z2: Enterprise—Scale RPA Integration	UiPath AI [106]	Native integration with legacy RPA systems and ERP platforms	Limited AI agent customization capabilities
	Flowise AI [249]	Visual no-code interface accelerates workflow design	Scalability challenges in high-volume production environments
	LlamaIndex [244]	Optimized for document-intensive RPA with RAG capabilities	Narrow focus on retrieval-augmented tasks limits generalizability
	LangChain [111]	Modular architecture supports hybrid RPA-AI pipelines	Requires significant development effort for enterprise integration

Enterprise integration scenarios (Scenario Z2) expose divergent approaches to bridging traditional RPA with modern AI capabilities. While specialized tools like LlamaIndex deliver exceptional performance for document-centric workflows, their narrow optimization comes at the expense of broader automation applicability. The comparison underscores an emerging industry need for platforms that can simultaneously deliver UiPath's [106] enterprise readiness, Flowise's [249] accessibility, and LangChain's [111] technical extensibility without compromising performance in production environments. These insights provide practitioners with a structured framework for evaluating tools against their specific operational requirements and technical constraints.

3.5 Summary

In this section, we present a systematic exploration of cutting-edge AI-driven automation technologies and their real-world applications. Our analysis begins by mapping the field's progression from basic rule-based systems to advanced cognitive architectures capable of dynamic reasoning and multi-agent collaboration. We examine how modern automation frameworks achieve unprecedented flexibility through innovative approaches to task planning, memory handling, and tool integration. The discussion covers various system implementations that demonstrate different strategies for workflow execution, from linear processing to sophisticated agent coordination models.

We also provide detailed comparisons of leading automation tools, revealing fundamental design choices and their practical implications. Our architectural analysis uncovers how contemporary systems balance performance with adaptability, while maintaining robust operation across diverse application scenarios. Through this comprehensive review, we illuminate the remarkable progress in AI-driven automation while establishing a clear framework for understanding current technological capabilities. The findings demonstrate how these systems are evolving from simple task automation to comprehensive problem-solving platforms.

4 Challenges in AI-Driven Automation

4.1 Existing Limitations

However, despite the remarkable progress in AI-driven automation, several challenges continue to hinder the widespread deployment of these technologies [24,64,252]. From data quality and governance to model interpretability [64,67], and from security [73,183] to robustness [84,136], the technical barriers of AI-driven automation significantly impact its progress. As shown in Fig. 17, we systematically categorize the key technical barriers to AI-driven automation into four interdependent dimensions: (1) *data-related challenges*, (2) *model limitations*, (3) *implementation barriers*, and (4) *security concerns*. This taxonomy reveals how fundamental issues in data quality and governance propagate through model development to system deployment [66,70,75]. The following discussion analyzes each category through both technical and operational lenses.

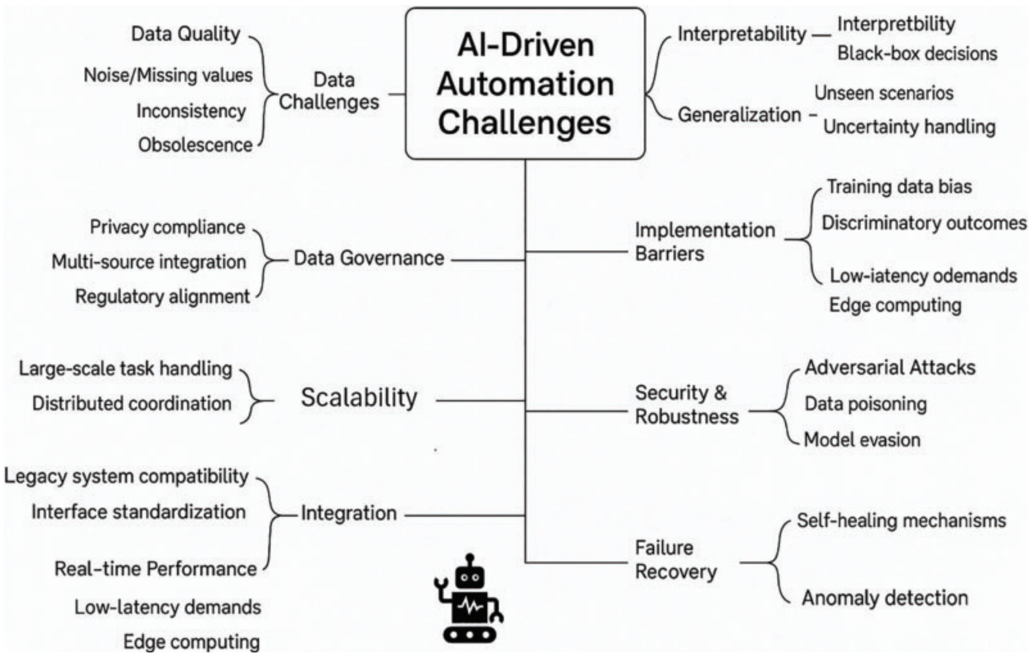


Figure 17: The primary categories of challenges in the development and deployment of AI-driven automation technologies

First and foremost, data quality and governance represent one of the primary challenges faced by AI-driven automation [33,66,70]. The performance of AI models is highly dependent on the quality and accuracy of input data. However, in the real world, data often suffers from noise, missing values, inconsistency, or obsolescence. This is particularly challenging in complex application scenarios, where ensuring the accuracy of data cleaning, integration, and processing becomes a critical issue. Additionally, in the realm of data governance [26,182,183], ensuring data privacy and security is a pressing concern. Automation systems need to process vast amounts of complex and diverse data sources while ensuring compliance with data privacy regulations, which places higher demands on technical infrastructure.

In addition to data governance, model interpretability and transparency are major challenges in current AI-driven automation technologies [7,64,67]. While deep learning and other complex algorithms have achieved significant success in specific tasks, their "black-box" nature makes it difficult to explain the task execution process. In high-risk fields, such as finance [3,17,24] and healthcare [13,57,14,219], the lack of transparency in AI systems' decisions can lead to traceability issues, thereby reducing trust in these systems. Therefore, improving the interpretability of AI systems, making their automatic production processes understandable to human users and developers, becomes crucial to enhancing transparency and fairness in automation systems [64,67].

Apart from interpretability, the generalization ability of models is also an important factor influencing AI-driven automation [39,84,136]. Many AI systems perform well in specific environments but may exhibit poor performance when faced with new, unseen situations. The actual applications of automation systems are often filled with uncertainties, and enhancing the adaptability of models to handle diverse human living needs is an urgent challenge that AI technology must address. To achieve this, AI models must be capable of processing various data patterns and reasoning effectively in complex and uncertain environments [39,84].

Another technical difficulty in AI-driven automation is system scalability. As automation applications expand, ensuring that systems can handle larger-scale tasks while maintaining efficiency and low latency becomes a significant challenge [8,56,65]. Especially in cloud computing and distributed environments, coordinating the collaboration of multiple AI models and managing resources effectively under dynamic loads remains a key issue in system design. This requires engineers to optimize resource management, load balancing, and fault tolerance within the system architecture [8,56].

Moreover, issues related to algorithm fairness and bias are crucial in the development of AI technologies [24,64,252]. AI systems' decisions heavily rely on training data, which may contain societal, cultural, and historical biases. These biases may lead to unfair or discriminatory decisions when automation is applied. In sensitive areas such as recruitment, credit evaluation, and judicial decisions, algorithmic bias can have profound negative consequences for individuals and society. Therefore, eliminating algorithmic bias and ensuring fairness in AI decisions remain key issues that must be prioritized in current AI research [64,252].

In terms of technical implementation, integration of AI-driven automation systems is another critical challenge [55,80,106]. AI-driven automation systems often need to collaborate with existing enterprise systems, software tools, hardware devices, and other automation technologies. Ensuring seamless integration of these different technological stacks and smooth data flow between them is a major issue in system design. Challenges in this area include compatibility between different platforms, standardization of data interfaces, and other related issues, which directly affect the deployment and maintenance of AI-driven automation systems [55,80].

At the same time, the real-time performance and processing capabilities of AI systems [19,184,186] must be considered in automation applications. In many practical scenarios, automation decisions need to be made within milliseconds or seconds, placing high demands on the system's computational power. Optimizing the efficiency of algorithms, reducing latency, and balancing computational resources and response speed, especially in big data and edge computing environments, have become critical concerns.

Finally, security and robustness [26,73,183] are still key obstacles in the application of AI-driven automation technologies. Due to the complexity of AI systems, they often become targets for malicious attacks. For example, hackers might manipulate data or perform model attacks to disrupt the human producing process of automation systems. Therefore, enhancing the security and robustness of AI systems to ensure stable operation in the face of external attacks and internal anomalies is a prerequisite for the success of AI automation applications.

In conclusion, while AI-driven automation technologies hold immense potential to drive intelligent transformation across industries, they still face a series of challenges in technical implementation [66,70,75]. Issues such as data quality and governance, model interpretability, system scalability, and algorithm fairness require continuous technological innovation and interdisciplinary collaboration to overcome [64,252]. As these technical difficulties are gradually addressed, AI-driven automation is expected to further enhance productivity and efficiency across various sectors, bringing profound changes to society.

4.2 Privacy and Security Compliance in AI-Driven Automation Systems

As AI-driven automation systems are increasingly deployed across sensitive domains such as healthcare, finance, and energy, ensuring compliance with privacy and security regulations like GDPR[17] becomes essential. This section outlines key mechanisms adopted by representative tools to address data protection, encryption, and authentication challenges.

4.2.1 Compliance Standards and Data Protection Mechanisms

GDPR Compliance. Tools such as *Med-Bot* [57] and *TradingGPT* [16] incorporate GDPR-aligned features including dynamic consent tracking, and data anonymization. These systems enforce data minimization and transparency through anonymous processing and user-controlled data retention policies.

HIPAA-Compliant Architectures. In healthcare, platforms like *MedAgents* [12] and GPT-based medical chatbots [57] utilize HIPAA-compliant data pipelines. These systems support encrypted data storage and transmission (e.g., AES-256[18])

4.2.2 Data Isolation, Encryption, and Authentication

To address potential risks to data integrity and confidentiality, the tools discussed employ robust data isolation techniques. For instance, in multi-agent cooperative models like OWL's energy management system [120], distributed edge AI environments utilize hardware-based isolation to protect data at each edge node. This prevents unauthorized access to data generated by the agents, ensuring that only trusted systems have access to sensitive operational data. Meanwhile, the use of end-to-end encryption is standard across many AI-driven automation platforms. For example, systems like SuperAGI [116] and OWL [120] leverage advanced encryption protocols such as AES-256 to protect data during storage and transmission. This ensures that even if the data is intercepted during communication, it remains unreadable to unauthorized entities. Despite these robust compliance and security mechanisms, challenges remain in ensuring full interoperability and scalability across systems that span multiple industries. Tools like AutoGPT [113] and WorkGPT [217], which operate across different domains, need to account for varying compliance standards in each sector.

In summary, contemporary AI-driven automation systems increasingly embed regulatory compliance as a core architectural principle, integrating mechanisms such as auditability, encryption, access control, and federated data isolation. Addressing these evolving requirements will be pivotal for the sustained deployment of AI solutions across regulated environments, particularly as automation extends into multi-agent, edge-centric, and globalized operational contexts.

4.3 Summary

In this section, we systematically examine the critical challenges currently facing AI-driven automation technologies through a structured analysis framework. Our investigation reveals how fundamental issues in data quality and governance create cascading effects throughout the entire automation pipeline, from model

[17] GDPR: https://gdpr-info.eu/art-32-gdpr/ (accessed on 09 July 2025).
[18] AES-Standard: https://www.nist.gov/publications/advanced-encryption-standard-aes (accessed on 09 July 2025).

development to system deployment. The discussion highlights the complex interplay between technical limitations and operational constraints that collectively impact real-world implementation.

We also demonstrated that the path toward reliable and trustworthy automation systems requires addressing multiple interdependent factors. The challenges span across data management, model interpretability, system robustness, and security considerations. Each presenting unique technical hurdles while simultaneously influencing other aspects of system performance. The examination provides valuable insights into the current technological barriers while establishing clear directions for future research and development.

5 Future Directions and Opportunities

The rapid evolution of AI-driven automation technologies has unlocked unprecedented opportunities for transforming industries, yet significant gaps remain in achieving robust, ethical, and scalable implementations [1,3,75,252]. Building on the challenges outlined above, here we explore promising research directions and emerging opportunities that could redefine the trajectory of AI-driven automation.

Specifically, these directions span technical innovations [72,78,175], interdisciplinary synergies [16,32,63], regulatory frameworks [17,64,180], and novel application paradigms [74,235,251], aiming to address current limitations while amplifying the societal and economic impact of automation technologies. As shown in Fig. 18, we present a comprehensive taxonomy of future research directions in AI-driven automation, systematically organized across ten key dimensions ranging from foundational cognitive architectures to industry-specific implementations. In the following subsections, we provide detailed analysis of each direction, beginning with core advances in cognitive systems.

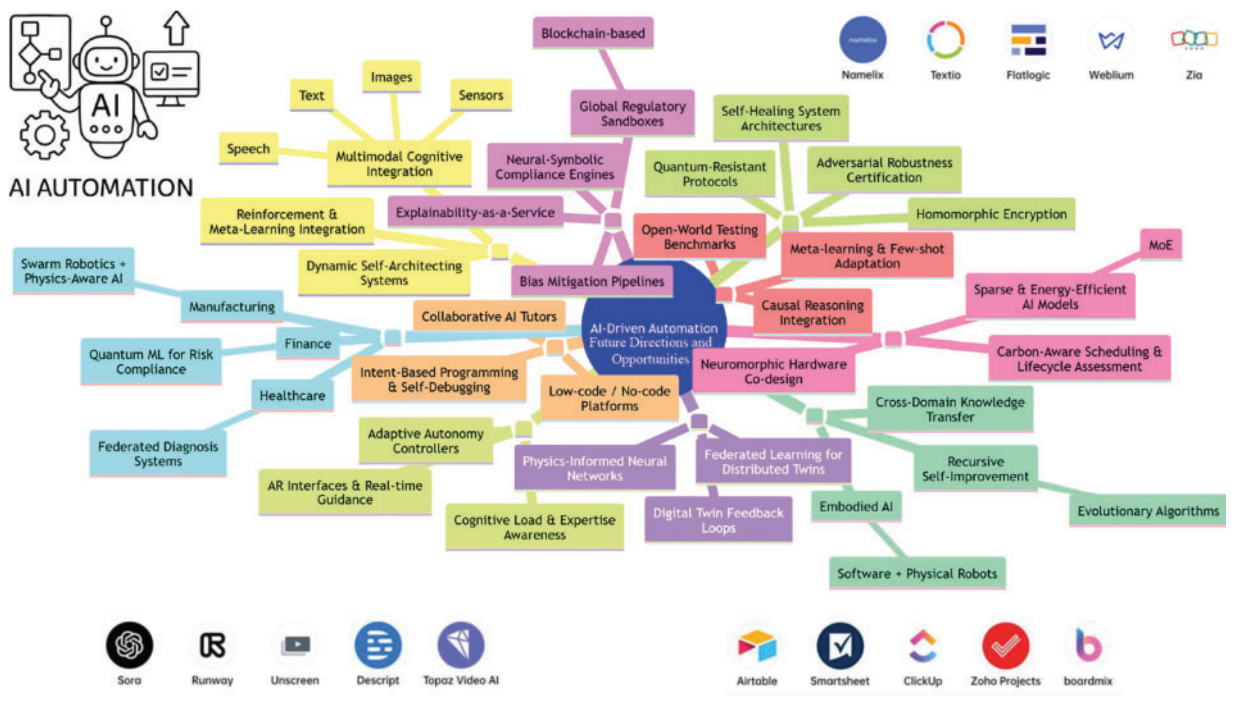

AI
AI AUTOMATION
AI-Driven Automation Future Directions and Opportunities
Blockchain-based
Global Regulatory Sandboxes
Neural-Symbolic Compliance Engines
Explainability-as-a-Service
Bias Mitigation Pipelines
Text
Images
Sensors
Speech
Multimodal Cognitive Integration
Reinforcement & Meta-Learning Integration
Dynamic Self-Architecting Systems
Swarm Robotics + Physics-Aware AI
Manufacturing
Finance
Quantum ML for Risk Compliance
Healthcare
Federated Diagnosis Systems
Collaborative AI Tutors
Intent-Based Programming & Self-Debugging
Low-code / No-code Platforms
Adaptive Autonomy Controllers
AR Interfaces & Real-time Guidance
Cognitive Load & Expertise Awareness
Physics-Informed Neural Networks
Federated Learning for Distributed Twins
Digital Twin Feedback Loops
Self-Healing System Architectures
Quantum-Resistant Protocols
Adversarial Robustness Certification
Homomorphic Encryption
Open-World Testing Benchmarks
Meta-learning & Few-shot Adaptation
Causal Reasoning Integration
MoE
Sparse & Energy-Efficient AI Models
Carbon-Aware Scheduling & Lifecycle Assessment
Neuromorphic Hardware Co-design
Cross-Domain Knowledge Transfer
Recursive Self-Improvement
Evolutionary Algorithms
Embodied AI
Software + Physical Robots
Namelix
Textio
Flatlogic
Weblium
Zia
Sora
Runway
Unscreen
Descript
Topaz Video AI
Airtable
Smartsheet
ClickUp
Zoho Projects
boardmix

Figure 18: An overview of future trends and opportunities in AI-driven automation tools and systems

5.1 Enhancing Cognitive Architectures for Dynamic Adaptation

A critical frontier lies in advancing AI-driven automation systems to dynamically adapt to evolving environments [97,142,253]. Current architectures, while increasingly sophisticated, remain constrained by rigid workflows or limited contextual awareness. Future research should prioritize self-architecting systems capable of autonomously selecting optimal execution paradigms (sequential, parallel, or hybrid) based on real-time workflow analysis. For instance, integrating reinforcement learning [85,168] with meta-learning [60,230] frameworks could enable systems to dynamically reconfigure task decomposition strategies or switch between rule-based and LLM-driven reasoning. Emerging tools like AutoGPT [113] hint at this potential but lack formal mechanisms for evaluating architectural trade-offs in real time.

Another promising direction involves multi-modal cognitive integration [79,174], where these systems seamlessly process and correlate diverse data types-text, speech, images, and sensor inputs. This demands advancements in cross-modal attention mechanisms and unified knowledge representation frameworks. For example, combining transformer-based language models [43,156] with graph neural networks [164–166] could enable systems to parse technical manuals, interpret equipment sensor data, and generate maintenance protocols within a single workflow. Such architectures would be transformative in manufacturing and healthcare, where decisions often hinge on heterogeneous data streams.

5.2 Advancing Human-AI Collaboration Frameworks

The next generation of AI-driven automation tools will transcend conventional "human-in-the-loop" models [254] to achieve context-aware symbiosis. Current AI-driven automation systems like MetaGPT [60] and OWL [120] demonstrate early-stage collaborative capabilities but lack nuanced understanding of human expertise levels or situational constraints. Future systems could employ adaptive autonomy controllers that modulate AI involvement based on:

- Operator skill profiling [146,175]: Leveraging user interaction histories to personalize assistance levels.
- Task criticality assessment [12,57]: Automatically escalating decisions to humans in high-risk scenarios (e.g., medical diagnostics).
- Cognitive load optimization [213,255]: Balancing automation depth with human oversight requirements to prevent decision fatigue.

Innovative interfaces will play a pivotal role here. Augmented reality (AR) [49,50] overlays, as seen in experimental platforms like DeChen's HMI solutions [220], could evolve into AI-guided procedural assistants. Imagine maintenance technicians receiving real-time, context-sensitive AR instructions generated by LLMs analyzing equipment manuals and historical repair data [128,196]. Such systems would require ultra-low-latency architectures (less than 10ms response times) and robust anomaly detection to prevent hazardous misinterpretations.

5.3 Bridging the Simulation-to-Reality Gap

A persistent challenge in industrial automation is the discrepancy between simulated environments and real-world deployment [22,176,256]. Emerging digital twin technologies offer a pathway to address this through:

- Physics-informed neural networks [100,230]: Embedding domain-specific constraints (e.g., mechanical stress limits) into AI models to improve simulation fidelity.
- Continuous twin updating [16,32]: Implementing bidirectional data flows where real-world sensor feedback refines digital models in real time.

- Federated learning architectures [65,130]: Enabling collaborative model training across distributed digital twins while preserving data privacy.

For example, in energy grid management, a federation of digital twins could simulate regional demand patterns while OWL-like multi-agent automation systems optimize power distribution [120]. This approach would demand breakthroughs in distributed optimization algorithms and secure data-sharing protocols to prevent adversarial attacks on interconnected twins.

5.4 Ethical AI and Regulatory Compliance by Design

As automation permeates high-stakes domains, embedding ethical guardrails into system architectures becomes imperative [17,64,67]. Current efforts, such as differential privacy in BD HemoSphere Alta [218,220], represent isolated solutions rather than systemic approaches. Future frameworks should integrate:

- Dynamic compliance engines [14,34]: Neural-symbolic systems that automatically map regulatory requirements (e.g., GDPR[19], HIPAA[20]) to workflow constraints.
- Bias mitigation pipelines [71,73]: Multi-stage architectures where data preprocessing, model training, and output validation layers each incorporate fairness checks.
- Explainability-as-a-service [64,67]: Modular interfaces that generate audit trails and rationale explanations tailored to different stakeholders (technicians, regulators, end-users).

A groundbreaking opportunity lies in developing global regulatory sandboxes-shared testing environments where governments and corporations collaboratively validate compliance mechanisms [35]. These sandboxes could utilize blockchain-based smart contracts to enforce transparency in automated producing processes, particularly in finance and healthcare.

5.5 Sustainable Automation through Energy-Efficient AI

The computational demands of modern AI systems, particularly LLMs, pose significant environmental challenges [73,198]. Future research must prioritize green automation strategies:

- Sparse model architectures [131,230]: Leveraging techniques like mixture-of-experts (MoE) [94,131] to reduce inference costs while maintaining performance.
- Hardware-algorithm co-design [11,216]: Developing specialized chips (e.g., neuromorphic processors [190,257]) optimized for common automation tasks like sensor fusion or NLP.
- Carbon-aware scheduling [73,198]: Dynamically shifting computational loads to regions/times with renewable energy surplus.

Experimental platforms like OWL [120]'s edge-computing modules demonstrate early progress, but broader adoption requires standardized metrics for measuring automation's carbon footprint [35]. Integrating lifecycle assessment tools into AI development pipelines could enable "sustainability-by-design" in next-generation systems [6,75].

5.6 Cross-Domain Generalization and Open-World Learning

Most current AI-driven automation tools excel in narrow domains but struggle with unseen scenarios-a critical limitation given the unpredictable nature of real-world environments [97,175]. Addressing these demands:

[19] General Data Protection Regulation (GDPR): https://gdpr-info.eu/ (accessed on 09 July 2025).
[20] Health Insurance Portability and Accountability Act (HIPAA): https://www.cloudflare-cn.com/learning/privacy/what-is-hipaa-compliance/ (accessed on 09 July 2025).

- Meta-learning frameworks [142,230]: Systems that rapidly adapt to new tasks by leveraging knowledge from related domains. For instance, a document processing pipeline trained on legal contracts could generalize to medical records through few-shot learning.
- Causal reasoning integration [175,97]: Moving beyond correlation-based models to systems that infer underlying causal relationships, enhancing robustness in dynamic settings.
- Open-world validation benchmarks [32,74]: Developing testing environments that simulate novel edge cases, such as equipment failures or unprecedented market shifts.

The fusion of LLMs with simulation engines presents a unique opportunity here. Imagine training AI agents in synthetic "automation gyms" that procedurally generate challenges ranging from supply chain disruptions to cyberattacks, fostering generalized problem-solving capabilities.

5.7 Democratization through Low-Code/No-Code Evolution

While platforms like Zapier have democratized automation development, significant barriers remain for non-technical users [29,30,38]. Future no-code systems could incorporate:

- Intent-based programming [37,30,250]: Allowing users to describe goals in natural language while AI handles implementation details.
- Self-debugging workflows [28,41]: Automated root-cause analysis and correction of faulty automation scripts.
- Collaborative AI tutors [36,42]: Context-aware assistants that guide users through complex workflow design via interactive dialogues.

A radical innovation would be self-assembling automation ecosystems, where AI agents negotiate API integrations and resolve compatibility issues autonomously [111]. This would require advances in semantic protocol translation and distributed consensus mechanisms, building upon current efforts in tools like LangChain [111,112].

5.8 Security and Resilience in Hyperconnected Systems

As AI-driven automation systems grow increasingly interconnected, novel attack vectors emerge. Future safeguards must adopt a multi-layered defense paradigm:

Homomorphic encryption [73,71]: Enabling secure data processing in untrusted environments (e.g., cloud-based RPA).

Adversarial robustness certification [136,258]: Formal methods to guarantee system behavior under worst-case scenarios.

Self-healing architectures [39]: Systems that automatically isolate compromised components and regenerate clean states using immutable ledger technologies.

The integration of quantum-resistant cryptography and AI-driven threat detection could create next-generation shields for critical infrastructure automation. For instance, power grid controllers might employ neural networks trained on synthetic attack data to preemptively neutralize threats.

5.9 Toward Artificial General Intelligence (AGI) in Automation

While current systems excel at specific tasks, the holy grail remains context-aware AGI capable of open-ended reasoning. Incremental steps toward this vision include:

- Recursive self-improvement [39]: Systems that autonomously refine their architectures through evolutionary algorithms.

- Embodied AI integration [155,176,235]: Combining software automation with robotic systems that learn through physical interaction.
- Cross-modal knowledge transfer [46,133,169]: Enabling insights gained in one domain (e.g., logistics optimization) to inform decisions in unrelated fields (e.g., patient scheduling).

Ethical considerations here are paramount. The development of AGI-powered automation demands international governance frameworks to prevent uncontrolled recursive self-enhancement scenarios.

5.10 Industry-Specific Revolution

Finally, domain-specific innovations will drive automation's next wave:

- Healthcare [13,14,57]: Federated learning systems that enable collaborative diagnosis across hospitals without sharing sensitive patient data.
- Manufacturing [25,27,68,211]: Swarm robotics coordinated through physics-aware AI models for adaptive production lines.
- Finance [3,103,17,32]: Real-time regulatory compliance engines using quantum machine learning to detect emerging market risks.

Each requires tailored solutions that balance technical capabilities with domain constraints-a challenge demanding close academia-industry collaboration.

Overall, the future of AI-driven automation lies not in isolated technological breakthroughs, but in the synergistic integration of cognitive architectures, ethical frameworks, and human-centric design. By addressing current limitations in adaptability, transparency, and sustainability, researchers can unlock the full potential of AI-driven automation as a force for equitable progress. Emerging paradigms-from self-built systems to AGI-assisted workflows-promise to redefine how organizations operate, but their success hinges on our ability to navigate technical complexities while upholding societal values. As the field advances, interdisciplinary collaboration will be crucial in transforming these visionary directions into tangible solutions that benefit industries and communities worldwide.

5.11 Summary

In this section, we explore the transformative potential and future directions of AI-driven automation across multiple dimensions. Our analysis reveals that the field is moving toward more adaptive, ethical, and human-centric systems, with key opportunities in cognitive architectures, human-AI collaboration, and industry-specific applications. We highlight how emerging technologies like digital twins, augmented reality interfaces, and quantum-resistant security protocols are creating new paradigms for intelligent automation. The discussion underscores the need for balanced progress that combines technical innovation with robust governance frameworks. As the field evolves, the most impactful advancements will likely emerge at the intersection of multiple disciplines, requiring close collaboration between researchers, practitioners, and policymakers to realize automation's full potential while addressing societal concerns.

6 Conclusion

The rapid evolution of AI-driven automation has ushered in a transformative era across industries, marked by unprecedented efficiency gains, enhanced task handling capabilities, and novel operational paradigms. Over the past decade, AI-driven automation has transitioned from rule-based systems to sophisticated cognitive architectures, yet the absence of a unified framework for categorizing methods, benchmarking performance, and addressing ethical risks has hindered both academic progress and industrial adoption. To bridge this gap, our work systematically reviews architectural innovations, functional paradigms, and real-world applications of AI-driven automation techniques across research areas and industries. Specifically, this survey has systematically examined the architectural, functional, and application-specific advancements in AI-driven automation, while systematically analyzing persistent challenges and emerging opportunities. Firstly, this survey begins by establishing several novel taxonomies to classify AI-driven automation tools into four architectural paradigms-rule-based, LLM-driven, multi-agent cooperative, and hybrid models-and further categorizes them by functionality, task execution mechanisms, and human-AI collaboration modes. Through comparative analysis of various AI-driven automation tools, emphasizing their applicability in domains such as healthcare, finance, and manufacturing. Finally, we provide a detailed summarization of challenges, including data governance gaps, model interpretability deficits, and scalability bottlenecks, provides a roadmap for addressing critical barriers to deployment. Additionally, we also discuss emerging trends, such as the integration of digital twins and neuro-symbolic reasoning, to forecast future research trajectories. By consolidating disparate advancements of AI-driven automation technique into a cohesive narrative, this survey not only clarifies the current advancements of AI-driven automation but also establishes foundational knowledge to guide responsible innovation, ensuring these transformative technologies align with societal needs and sustainable development goals.

Acknowledgement: This work is conducted by the first author, Weiqiang Jin, during his research at Xi'an Jiaotong University and his visiting period at The University of Hong Kong. The first author would like to express his deepest appreciation to Professor Hongyang Du from The University of Hong Kong for his expert and valuable insights, and to Professor Tagore from Xi'an Jiaotong University for his constant encouragement. The authors are particularly grateful to the editors and anonymous reviewers for their thorough evaluations, and insightful comments, which have significantly improved this work. The corresponding author of this paper is Biao Zhao.

Funding Statement: This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Author Contributions: The authors confirm contributions to this paper as follows: Weiqiang Jin: Conceptualization, Methodology, Formal analysis, Writing—original draft, Writing—review & editing, Project administration, Investigation, Validation; Ningwei Wang (Equal Contribution): Software, Data curation, Visualization, Writing—review & editing; Lei Zhang: Formal analysis, Writing—review & editing; Xingwu Tian: Investigation, Visualization, Conceptualization, Project administration; Bohang Shi: Resources; Biao Zhao: Writing—review & editing, Supervision. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: This work did not involve any associated code or datasets as it is primarily theoretical in nature. No data were generated or analyzed during this study.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest regarding the publication of this study.

References

1. Executive Office of the President. Artificial Intelligence, Automation, and the Economy. Washington, DC, USA: The White House; 2016 [Internet]. [cited 2023 Dec 24]. Available from: https://obamawhitehouse.archives.gov/blog/2016/12/20/artificial-intelligence-automation-and-economy.
2. Bhattacharyya S, Banerjee JS, De D, editors. Confluence of artificial intelligence and robotic process automation. 1st ed. Singapore: Springer; 2023. doi:10.1007/978-981-19-8296-5_16.
3. Shen Y, Zhang X. The impact of artificial intelligence on employment: the role of virtual agglomeration. Human Soc Sci Communicat. 2024;11(1):122. doi:10.1057/s41599-024-02647-9.

4. Liu G, Zhao P, Liu L, Guo Y, Xiao H, Lin W, et al. LLM-powered GUI agents in phone automation: surveying progress and prospects. arXiv:2504.19838. 2025.
5. Ke Z, Jiao F, Ming Y, Nguyen XP, Xu A, Long DX, et al. A survey of frontiers in LLM reasoning: inference scaling, learning to reason, and agentic systems. arXiv:2504.09037. 2025.
6. Wang L, Ma C, Feng X, Zhang Z, Yang H, Zhang J, et al. A survey on large language model based autonomous agents. Front Comput Sci. 2024;18:186345. doi:10.1007/s11704-024-40231-1.
7. Pan L, Saxon M, Xu W, Nathani D, Wang X, Wang WY. Automatically correcting large language models: surveying the landscape of diverse automated correction strategies. Transact Associat Computat Linguist. 2024;12(2):484–506. doi:10.1162/tacl_a_00660.
8. Zhou S, Xu FF, Zhu H, Zhou X, Lo R, Sridhar A, et al. WebArena: a realistic web environment for building autonomous agents. In: The Twelfth International Conference on Learning Representations; 2024 [Internet]; Vienna, Austria. [cited 2025 Jul 9]. Available from: https://openreview.net/forum?id=oKn9c6ytLx.
9. Chen G, Dong S, Shu Y, Zhang G, Sesay J, Karlsson B, et al. AutoAgents: a framework for automatic agent generation. In: Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24; 2024 Aug 3–9; Jeju, Republic of Korea. p. 22–30. doi:10.24963/ijcai.2024/3.
10. Pandya K, Holia M. Automating customer service using LangChain: building custom open-source GPT Chatbot for organizations. arXiv:2310.05421. 2023.
11. Zhang X, Chen H, He Y, Niu W, Li Q. Automatically generating rules of malicious software packages via large language model. arXiv:2504.17198. 2025.
12. Tang X, Zou A, Zhang Z, Li Z, Zhao Y, Zhang X, et al. MedAgents: large language models as collaborators for zero-shot medical reasoning. In: Findings of the Association for Computational Linguistics: ACL 2024. Bangkok, Thailand: Association for Computational Linguistics; 2024. p. 599–621 doi:10.1162/tacl_a_00648.
13. Chen X, Yi H, You M, Liu W, Wang L, Li H, et al. Enhancing diagnostic capability with multi-agents conversational large language models. npj Digital Medicine. 2025 03;8(1):159. doi:10.1038/s41746-025-01550-0.
14. Guarda T. Robotic process automation application in healthcare. In: Guarda T, Portela F, Gatica G, editors. Advanced research in technologies, information, innovation and sustainability. Cham, Switzerland: Springer Nature; 2025. p. 97–109. doi:10.1007/978-3-031-83210-9_8.
15. Huang F, Vasarhelyi MA. Applying robotic process automation (RPA) in auditing: a framework. Int J Account Inf Syst. 2019;35(2):100433. doi:10.1016/j.accinf.2019.100433.
16. Li Y, Yu Y, Li H, Chen Z, Khashanah K. TradingGPT: multi-agent system with layered memory and distinct characters for enhanced financial trading performance. arXiv:2309.03736. 2023.
17. Wang Y, Zhou W, Li Y, Sun J. Business optimization of financial centers in pharmaceutical enterprises based on robotic process automation technology. IEEE Access. 2025;13:51012–26. doi:10.1109/access.2025.3550962.
18. Kim Y, Iturrate I, Sloth C, Kim H. Safety-ensured control framework for robotic endoscopic task automation. arXiv:2503.08214. 2025.
19. Yao T, Ban M, Lu B, Pei Z, Qi P. Sim4EndoR: a reinforcement learning centered simulation platform for task automation of endovascular robotics. arXiv:2504.05330. 2025.
20. Ruan J, Chen Y, Zhang B, Xu Z, Bao T, du Q, et al. TPTU: task planning and tool usage of large language model-based AI agents. In: NeurIPS 2023 Foundation Models for Decision Making Workshop; 2023 [Internet]. [cited 2025 Jul 9].
21. Liu Z, Yao W, Zhang J, Yang L, Liu Z, Tan J, et al. AgentLite: a lightweight library for building and advancing task-oriented LLM agent system. arXiv:2402.15538. 2024.
22. Lin BY, Huang C, Liu Q, Gu W, Sommerer S, Ren X. On grounded planning for embodied tasks with language models. Proc AAAI Conf Artif Intell. 2023;37(11):13192–200. doi:10.1609/aaai.v37i11.26549.
23. Liang Y, Wu C, Song T, Wu W, Xia Y, Liu Y, et al. TaskMatrix.AI: completing tasks by connecting foundation models with millions of APIs. Intell Comput. 2024;3:0063. doi:10.34133/icomputing.0063.
24. Autor DH. Why are there still so many jobs? the history and future of workplace automation. J Econ Perspect. 2015;29(3):3–30. doi:10.1257/jep.29.3.3.

25. Badmus AD. Leveraging software automation to transform the manufacturing industry. J Knowl Learn Sci Technol. 2024;2(1):84–92. doi:10.60087/jklst.vol2.n1.p92.
26. Al-Slais Y, Ali M. Robotic process automation and intelligent automation security challenges: a review. In: 2023 International Conference On Cyber Management and Engineering (CyMaEn); 2023 Jan 26–27; Bangkok, Thailand. p. 71–7.
27. Kavitha R. Hyperautomation-beyond RPA: leveraging automation to transform the manufacturing industries. In: 2023 International Conference on Computer Communication and Informatics (ICCCI); 2023 Jan 23–25; Coimbatore, India. p. 1–5.
28. Airtable. Airtable: the modern platform for building AI-powered apps; 2025 [Internet]. [cited 2024 Nov 12]. Available from: https://www.airtable.com/.
29. Figma I. Figma: The collaborative interface design tool. Figma, Inc.; 2025 [Internet]. [cited 2025 Feb 24]. Available from: https://www.figma.com/.
30. Zapier. Zapier: the most connected AI orchestration platform; 2024 [Internet]. [cited 2024 Oct 24]. Available from: https://zapier.com/.
31. Team RA, Contributors. AgentGPT: autonomous AI agent platform. GitHub; 2023 [Internet]. [cited 2025 Feb 9]. Available from: https://github.com/reworkd/AgentGPT.
32. Li N, Gao C, Li M, Li Y, Liao Q. EconAgent: large language model-empowered agents for simulating macroeconomic activities. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. Bangkok, Thailand: Association for Computational Linguistics; 2024. p. 15523–36.
33. Abdellaif OH, Nader A, Hamdi A. LMRPA: large language model-driven efficient robotic process automation for OCR. arXiv:2412.18063. 2024.
34. JingXuan C, Tayyab M, Muzammal SM, Jhanjhi NZ, Ray SK, Ashfaq F. Integrating AI with robotic process automation (RPA): advancing intelligent automation systems. In: 2024 IEEE 29th Asia Pacific Conference on Communications (APCC); 2024 Nov 5–7; Bali, Indonesia. p. 259–65.
35. Erdmann S, Sandkuhl K. Robotic process automation in small enterprises: an investigation into application potential. Complex Syst Inform Model Quart. 2023;34(34):84–105. doi:10.7250/csimq.2023-34.04.
36. Botonic. Botonic: no-code chatbots; 2025 [Internet]. [cited 2025 Mar 19]. Available from: https://botonic.io/.
37. Voiceflow. Voiceflow: the fastest way to build advanced AI Agents; 2025 [Internet]. [cited 2025 Sep 17]. Available from: https://www.voiceflow.com/.
38. Quickbase. Quickbase: spend more time on work that matters; 2024. Quickbase helps you see, connect, and control your projects from day one to done-without the complexity [Internet]. [cited 2025 Jul 9]. Available from: https://www.quickbase.com/.
39. Yuan S, Song K, Chen J, Tan X, Li D, Yang D. EvoAgent: towards automatic multi-agent generation via evolutionary algorithms. In: Workshop on Open-World Agents: Synergizing Reasoning and Decision-Making in Open-World Environments (OWA-2024); 2024 [Internet]. [cited 2025 Jul 9]. Available from: https://openreview.net/forum?id=DdxAp2j3AP.
40. Riedesel J, Myers C, Ashworth B. Intelligent space power automation. In: Proceedings. IEEE International Symposium on Intelligent Control 1989; 1989 Sep 25–26; Albany, NY, USA. p. 168–73.
41. Google. Looker studio: no-code charts; 2025 [Internet]. [cited 2025 Apr 8]. Available from: https://lookerstudio.google.com/overview.
42. Webnode. Webnode: no-code website builder, Make your own website for free!. Webnode; 2024 [Internet]. [cited 2025 Mar 12]. Available from: https://www.webnode.com.
43. Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I. Language models are unsupervised multitask learners. OpenAI; 2019 [Internet]. [cited 2025 Jul 9]. Available from: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.
44. Contributors G. GPT Researcher: LLM based autonomous agent that conducts deep local and web research on any topic and generates a long report with citations. GitHub; 2024 [Internet]. [cited 2024 Aug 29]. Available from: https://github.com/assafelovic/gpt-researcher.

45. OpenAI, Jaech A, Kalai A, Lerer A, Richardson A, El-Kishky A, et al. OpenAI o1 system card. arXiv:2412.16720. 2024.
46. Lewis P, Perez E, Piktus A, Petroni F, Karpukhin V, Goyal N, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS '20. Red Hook, NY, USA: Curran Associates Inc.; 2020.
47. Qian C, Liu W, Liu H, Chen N, Dang Y, Li J, et al. ChatDev: communicative agents for software development. arXiv:2307.07924. 2024.
48. Chen D, Wang H, Huo Y, Li Y, Zhang H. GameGPT: multi-agent collaborative framework for game development. arXiv:2310.08067. 2023.
49. Wang S, Liu C, Zheng Z, Qi S, Chen S, Yang Q, et al. Avalon's game of thoughts: battle against deception through recursive contemplation. arXiv:2310.01320. 2023.
50. Xu Y, Wang S, Li P, Luo F, Wang X, Liu W, et al. Exploring large language models for communication games: an empirical study on werewolf. arXiv:2309.04658. 2024.
51. Ramasubbu D, Baskaran K, Yann G. Intrusive plug management system using chatbots in office environments. In: 2018 Asian Conference on Energy, Power and Transportation Electrification (ACEPT); 2018 Oct 30–Nov 2; Singapore. p. 1–4. doi:10.1109/acept.2018.8610869.
52. Dincy R. Arikkat, Abhinav M, Binu N, Parvathi M, Biju N, Arunima KS, et al. IntellBot: retrieval augmented LLM chatbot for cyber threat knowledge delivery. arXiv:2411.05442. 2024.
53. Livathinos N, Auer C, Lysak M, Nassar A, Dolfi M, Vagenas P, et al. Docling: an efficient open-source toolkit for AI-driven document conversion. arXiv:2501.17887. 2025.
54. Perdana A, Lee WE, Mui Kim C. Prototyping and implementing Robotic Process Automation in accounting firms: benefits, challenges and opportunities to audit automation. Int J Account Inf Syst. 2023;51(1):100641. doi:10.1016/j.accinf.2023.100641.
55. Jain A, Paliwal S, Sharma M, Vig L, Shroff G. SmartFlow: robotic process automation using LLMs. arXiv:2405.12842. 2024.
56. Liu Z, Yao W, Zhang J, Xue L, Heinecke S, Murthy R, et al. BOLAA: benchmarking and orchestrating LLM-augmented autonomous agents. arXiv:2308.05960. 2023.
57. Bhatt A, Vaghela N. Med-Bot: an AI-powered assistant to provide accurate and reliable medical information. arXiv:2411.09648. 2024.
58. Nam Y, Seo T, Shin G, Lee S, Im J. NOVI: chatbot system for university novice with BERT and LLMs. arXiv:2409.06192. 2024.
59. Qin Y, Liang S, Ye Y, Zhu K, Yan L, Lu Y, et al. ToolLLM: facilitating large language models to master 16000+ real-world APIs. arXiv:2307.16789. 2023.
60. Hong S, Zhuge M, Chen J, Zheng X, Cheng Y, Wang J, et al. MetaGPT: meta programing for a mmulti-agent collaborative framework. In: The Twelfth International Conference on Learning Representations; 2024 [Internet]; Vienna, Austria. [cited 2025 Jul 9]. Available from: https://openreview.net/forum?id=VtmBAGCN7o.
61. Duan J, Wang S, Diffenderfer J, Sun L, Chen T, Kailkhura B, et al. ReTA: recursively thinking ahead to improve the strategic reasoning of large language models. In: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Mexico City, Mexico: Association for Computational Linguistics; 2024. p. 2232–46.
62. Mao S, Cai Y, Xia Y, Wu W, Wang X, Wang F, et al. ALYMPICS: LLM agents meet game theory. In: Proceedings of the 31st International Conference on Computational Linguistics. Abu Dhabi, UAE: Association for Computational Linguistics; 2025. p. 2845–66.
63. Chen P, Zhang S, Han B. CoMM: collaborative multi-agent, multi-reasoning-path prompting for complex problem solving. In: Findings of the Association for Computational Linguistics: NAACL 2024. Mexico City, Mexico: Association for Computational Linguistics; 2024. p. 1720–38.
64. Saeed W, Omlin C. Explainable AI (XAI): a systematic meta-survey of current challenges and future opportunities. Knowl Based Syst. 2023;263(3):110273. doi:10.1016/j.knosys.2023.110273.

65. Chen W, Su Y, Zuo J, Yang C, Yuan C, Chan CM, et al. AgentVerse: facilitating multi-agent collaboration and exploring emergent behaviors. In: The Twelfth International Conference on Learning Representations; 2024 [Internet]; Vienna, Austria. [cited 2025 Jul 9]. Available from: https://openreview.net/forum?id=EHg5GDnyq1.
66. Wewerka J, Reichert M. Robotic process automation-a systematic mapping study and classification framework. Enterp Inform Syst. 2023;17(2):1986862. doi:10.1080/17517575.2021.1986862.
67. Samek W, Montavon G, Vedaldi A, Hansen LK, Muller KR, editors. Explainable AI: interpreting, explaining and visualizing deep learning. In: Lecture notes in computer science. Vol. 11700. 1st ed. Berlin/Heidelberg, Germany: Springer-Verlag; 2022.
68. Kong Y, Ruan J, Chen Y, Zhang B, Bao T, Shiwei S, et al. TPTU-v2: boosting task planning and tool usage of large language model-based agents in real-world industry systems. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track. Miami, FL, USA: Association for Computational Linguistics; 2024. p. 371–85.
69. Lee K, Zo H, Jeong M. Understanding user acceptance of robotic process automation: the user resistance perspective. Indust Manag Data Syst. 2025;125(3):1000–22. doi:10.1108/IMDS-07-2024-0628.
70. Schlegel D, Rosenberg B, Fundanovic O, Kraus P. How to conduct successful business process automation projects? An analysis of key factors in the context of robotic process automation. Business Process Manag J. 2024;30(8):99–119. doi:10.1108/bpmj-06-2023-0465.
71. Ferrag MA, Tihanyi N, Debbah M. From LLM reasoning to autonomous AI agents: a comprehensive review. arXiv:2504.19678. 2025.
72. Keskar NS, McCann B, Varshney LR, Xiong C, Socher R. CTRL: a conditional transformer language model for controllable generation. arXiv:1909.05858. 2019.
73. Li T, Yang YT, Pan Y, Zhu Q. From texts to shields: convergence of large language models and cybersecurity. arXiv:2505.00841. 2025.
74. Wang Z, Mao S, Wu W, Ge T, Wei F, Ji H. Unleashing the emergent cognitive synergy in large language models: a task-solving agent through multi-persona self-collaboration. In: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Mexico City, Mexico: Association for Computational Linguistics; 2024. p. 257–79.
75. Afrin S, Roksana S, Akram R. AI-enhanced robotic process automation: a review of intelligent automation innovations. IEEE Access. 2025;13(1):173–97. doi:10.1109/access.2024.3513279.
76. Lin CH, Chiu DKW, Lam KT. Hong Kong academic librarians' attitudes toward robotic process automation. Library Hi Tech. 2024;42:991–1014. doi:10.1108/LHT-03-2022-0141.
77. Feng X, Dou L, Li E, Wang Q, Wang H, Guo Y, et al. A survey on large language model-based social agents in game-theoretic scenarios. arXiv:2412.03920. 2024.
78. Yao S, Zhao J, Yu D, Du N, Shafran I, Narasimhan K, et al. ReAct: synergizing reasoning and acting in language models. arXiv:2210.03629. 2023.
79. Yang Z, Li L, Wang J, Lin K, Azarnasab E, Ahmed F, et al. MM-REACT: prompting ChatGPT for multimodal reasoning and action. arXiv:2303.11381. 2023.
80. Prrucha P, Skrbek J. API as method for improving robotic process automation. In: Business process management: blockchain, robotic process automation, and central and eastern europe forum. Cham, Switzerland: Springer International Publishing; 2022. p. 260–73. doi:10.1007/978-3-031-16168-1_17.
81. Asano Y, Okada K, Nakagawa S, Yoshie N, Shiomi J. Automation of polymer pressing by robotic handling with in-process parameter optimization. Robot Auton Syst. 2025;185(2):104868. doi:10.1016/j.robot.2024.104868.
82. Jacome-Leon JG, Zambrano-Vizuete M, Imbaquingo-Esparza DE, Botto-Tobar M. Blockchain and machine learning for intelligent automation in robotic process automation. SN Comput Sci. 2025;6(4):341. doi:10.1007/s42979-025-03852-2.
83. Shinn N, Cassano F, Berman E, Gopinath A, Narasimhan K, Yao S. Reflexion: language agents with verbal reinforcement learning. arXiv:2303.11366. 2023.

84. Vu T, Krishna K, Alzubi S, Tar C, Faruqui M, Sung YH. Foundational Autoraters: taming large language models for better automatic evaluation. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. Miami, FL, USA: Association for Computational Linguistics; 2024. p. 17086–105.
85. Nguyen TT, Nguyen ND, Nahavandi S. Deep reinforcement learning for multiagent systems: a review of challenges, solutions, and applications. IEEE Transact Cybernet. 2020;50(9):3826–39. doi:10.1109/tcyb.2020.2977374.
86. Masterman T, Besen S, Sawtell M, Chao A. The landscape of emerging AI agent architectures for reasoning, planning, and tool calling: a survey. arXiv:2404.11584. 2024.
87. Brown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, et al. Language models are few-shot learners. arXiv:2005.14165. 2020.
88. Touvron H, Lavril T, Izacard G, Martinet X, Lachaux MA, Lacroix T, et al. LLaMA: open and efficient foundation language models. arXiv:2302.13971. 2023.
89. Touvron H, Martin L, Stone K, Albert P, Almahairi A, Babaei Y, et al. Llama 2: open foundation and fine-tuned chat models. arXiv:2307.09288. 2023.
90. Zhou P, Pujara J, Ren X, Chen X, Cheng HT, Le QV, et al. Self-Discover: large language models self-compose reasoning structures. arXiv:2402.03620. 2024.
91. Wu Q, Bansal G, Zhang J, Wu Y, Li B, Zhu E, et al. AutoGen: enabling next-gen LLM applications via multi-agent conversation. arXiv:2308.08155. 2023.
92. Zhou X, Zhu H, Mathur L, Zhang R, Yu H, Qi Z, et al. SOTOPIA: interactive evaluation for social intelligence in language agents. arXiv:2310.11667. 2024.
93. Zhang J, Hou Y, Xie R, Sun W, McAuley J, Zhao WX, et al. AgentCF: collaborative learning with autonomous language agents for recommender systems. In: WWW '24. New York, NY, USA: Association for Computing Machinery; 2024. p. 3679–89. doi:10.1145/3589334.3645537.
94. Xie T, Zhou F, Cheng Z, Shi P, Weng L, Liu Y, et al. OpenAgents: an open platform for language agents in the wild. In: First Conference on Language Modeling; 2024 [Internet]; Philadelphia, PA, USA. [cited 2025 Jul 9]. Available from: https://openreview.net/forum?id=sKATR2O1Y0.
95. Weng L. LLM-powered autonomous agents. Lilian weng's blog; 2023 [Internet]. [cited 2025 Feb 9]. Available from: https://lilianweng.github.io/posts/2023-06-23-agent/.
96. Yao S, Yu D, Zhao J, Shafran I, Griffiths TL, Cao Y, et al. Tree of thoughts: deliberate problem solving with large language models. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. NIPS'23. Red Hook, NY, USA: Curran Associates Inc.; 2023.
97. Xin H, Guo D, Shao Z, Ren ZZ, Zhu Q, Liu B, et al. Advancing theorem proving in LLMs through large-scale synthetic data. In: The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24; 2024 [Internet]. [cited 2025 Feb 9]. Available from: https://openreview.net/forum?id=TPtXLihkny.
98. Ye H, Martinez M, Luo X, Zhang T, Monperrus M. SelfAPR: self-supervised program repair with test execution diagnostics. In: Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering. ASE'22. New York, NY, USA: Association for Computing Machinery; 2023. doi:10.1145/3551349.3556926.
99. Zhang J, Xiang J, Yu Z, Teng F, Chen XH, Chen J, et al. AFlow: automating agentic workflow generation. In: The Thirteenth International Conference on Learning Representations; 2025 [Internet]; Singapore EXPO. [cited 2025 Feb 9]. Available from: https://openreview.net/forum?id=z5uVAKwmjf.
100. Ma Y, Gou Z, Hao J, Xu R, Wang S, Pan L, et al. SciAgent: tool-augmented language models for scientific reasoning. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. Miami, FL, USA: Association for Computational Linguistics; 2024. p. 15701–36.
101. Luo M, Xu X, Liu Y, Pasupat P, Kazemi M. In-context learning with retrieved demonstrations for language models: a survey. Transactions on machine learning research; 2024 [Internet]. [cited 2025 Feb 9]. Available from: https://openreview.net/forum?id=NQPo8ZhQPa.
102. Montes N, Luck M, Osman N, Rodrigues O, Sierra C. Combining theory of mind and abductive reasoning in agent-oriented programming. In: Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems. AAMAS'24. Richland, SC, USA: International Foundation for Autonomous Agents and Multiagent Systems; 2024. p. 2839–41.

103. Tiron-Tudor A, Lacurezeanu R, Bresfelean VP, Dontu AN. Perspectives on how robotic process automation is transforming accounting and auditing services. Account Perspect. 2024;23(1):7–38. doi:10.1111/1911-3838.12351.
104. Axmann B, Dzhelil SA, Najeeb A. Practical comparison of uipath and power automate by creating an automation use case from logistics. In: 2024 14th International Conference on Advanced Computer Information Technologies (ACIT); 2024 Sep 19–21; Ceske Budejovice, Czech Republic. p. 224–9.
105. Nithuna S, Laseena CA. Review on implementation techniques of chatbot. In: 2020 International Conference on Communication and Signal Processing (ICCSP); 2020 Jul 28–30; Chennai, India. p. 0157–61.
106. Inc U. The UiPath platform: where robots and agents unite; 2023 [Internet]. [cited 2024 Aug 3]. Available from: https://www.uipath.com/.
107. Dobrica L. Robotic process automation platform UiPath. Commun ACM. 2022;65(4):42–3. doi:10.1145/3511667.
108. Corno F, De Russis L, Monge Roffarello A. HeyTAP: bridging the gaps between users' needs and technology in IF-THEN rules via conversation. In: Proceedings of the 2020 International Conference on Advanced Visual Interfaces. AVI '20. New York, NY, USA: Association for Computing Machinery; 2020.
109. Gao Y, Xiao K, Li F, Xu W, Huang J, Dong W. ChatIoT: zero-code generation of trigger-action based IoT programs. Proc ACM Interact Mob Wearable Ubiquitous Technol. 2024;8:103. doi:10.1145/3678585.
110. Corno F, De Russis L, Monge Roffarello A. RecRules: recommending IF-THEN rules for end-user development. ACM Transact Intell Syst Technol (TIST). 2019;10(5):58. doi:10.1145/3344211.
111. Chase H. LangChain: large language model application development framework. LangChain Inc.; 2023 [Internet]. [cited 2025 May 8]. Available from: https://github.com/langchain-ai/langchain.
112. MacManus R. LangChain: the trendiest LLM framework of 2023. Tencent cloud developer community; 2023 [Internet]. [cited 2025 May 8]. Available from: https://cloud.tencent.com/developer/article/2309826.
113. Torantulino. AutoGPT: an autonomous GPT-4 experiment; 2023 [Internet]. [cited 2025 May 13]. Available from: https://github.com/Significant-Gravitas/Auto-GPT.
114. Alps H. OpenManus: open-source replication of manus AI agent. OpenManus Project; 2025 [Internet]. [cited 2025 May 8]. Available from: https://github.com/henryalps/OpenManus.
115. Nakajima Y. BabyAGI: autonomous self-building agent framework. GitHub; 2024 [Internet]. [cited 2025 May 9]. Available from: https://github.com/yoheinakajima/babyagi.
116. Contributors S. SuperAGI: a dev-first open source autonomous AI agent framework. Enabling developers to build, manage & run useful autonomous agents quickly and reliably. GitHub; 2024 [Internet]. [cited 2024 Apr 24]. Available from: https://github.com/TransformerOptimus/SuperAGI.
117. Inc M. Manus AI: autonomous general-purpose AI agent. Monica Inc.; 2025 [Internet]. [cited 2025 May 8]. Available from: https://manus.im/.
118. Goyal M, Bhasin P. Beyond the model: key differentiators in large language models and multi-agent services. World J Adv Res Rev. 2025;26(1):2703–6. doi:10.30574/wjarr.2025.26.1.1295.
119. Shen M, Yang Q. From mind to machine: the rise of manus AI as a fully autonomous digital agent. arXiv:2505.02024. 2025.
120. CAMEL-AI. OWL: optimized workforce learning for general multi-agent assistance in real-world task automation; 2024 [Internet]. [cited 2024 Jul 15]. Available from: https://github.com/camel-ai/owl.
121. Venkadesh P, Divya SV, Kumar KS. Unlocking AI creativity: a multi-agent approach with Crew AI. J Trends Comput Sci Smart Technol. 2024;6(4):338–56. doi:10.36548/jtcsst.2024.4.002.
122. AirSlate. AirSlate: automate document workflow; 2024 [Internet]. [cited 2025 May 18]. Available from: https://www.airslate.com/.
123. Tang J, Fan T, Huang C. AutoAgent: a fully-automated and zero-code framework for LLM agents. arXiv:2502.05957. 2025.
124. Feng Y, Papicchio S, Rahman S. CypherBench: towards precise retrieval over full-scale modern knowledge graphs in the LLM era. arXiv:2502.05957. 2025.
125. Sukhbaatar S, Szlam A, Fergus R. Learning multiagent communication with backpropagation. In: Advances in neural information processing systems (NeurIPS). Red Hook, NY, USA: Curran Associates, Inc.; 2016. p. 2244–52.

126. Pan B, Lu J, Wang K, Zheng L, Wen Z, Feng Y, et al. AgentCoord: visually exploring coordination strategy for LLM-based multi-agent collaboration. arXiv:2404.11943. 2024.
127. Zhang S, Yin M, Zhang J, Liu J, Han Z, Zhang J, et al. Which agent causes task failures and when? on automated failure attribution of LLM multi-agent systems. arXiv:2505.00212. 2025.
128. Li M, Zhao Y, Yu B, Song F, Li H, Yu H, et al. API-Bank: a comprehensive benchmark for tool-augmented LLMs. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. Singapore: Association for Computational Linguistics; 2023. p. 3102–16.
129. Liu T, Wang X, Huang W, Xu W, Zeng Y, Jiang L, et al. GroupDebate: enhancing the efficiency of multi-agent debate using group discussion. arXiv:2409.14051. 2024.
130. Huang D, Zhang JM, Luck M, Bu Q, Qing Y, Cui H. AgentCoder: multi-agent-based code generation with iterative testing and optimisation. arXiv:2312.13010. 2024.
131. Jiang AQ, Sablayrolles A, Roux A, Mensch A, Savary B, Bamford C, et al. Mixtral of experts. arXiv:2401.04088. 2024.
132. Ge Y, Hua W, Mei K, Ji J, Tan J, Xu S, et al. OpenAGI: when LLM meets domain experts. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. NIPS'23. Red Hook, NY, USA: Curran Associates Inc.; 2023.
133. Ni B, Buehler MJ. MechAgents: large language model multi-agent collaborations can solve mechanics problems, generate new data, and integrate knowledge. Extreme Mech Lett. 2024;67(7):102131. doi:10.1016/j.eml.2024.102131.
134. Chen J, Hu X, Liu S, Huang S, Tu WW, He Z, et al. LLMArena: assessing capabilities of large language models in dynamic multi-agent environments. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. Bangkok, Thailand: Association for Computational Linguistics; 2024. p. 13055–77.
135. Cross L, Xiang V, Bhatia A, Yamins DL, Haber N. Hypothetical minds: scaffolding theory of mind for multi-agent tasks with large language models. arXiv:2407.07086. 2025.
136. Owotogbe J. Assessing and enhancing the robustness of LLM-based multi-agent systems through chaos engineering. arXiv:2505.03096. 2025.
137. Ur B, Pak Yong Ho M, Brawner S, Lee J, Mennicken S, Picard N, et al. Trigger-action programming in the wild: an analysis of 200,000 IFTTT Recipes. In: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems. CHI '16. New York, NY, USA: Association for Computing Machinery; 2016. p. 3227–31.
138. Fan T, Wang J, Ren X, Huang C. MiniRAG: towards extremely simple retrieval-augmented generation. arXiv:2501.06713. 2025.
139. Asai A, Wu Z, Wang Y, Sil A, Hajishirzi H. Self-RAG: learning to retrieve, generate, and critique through self-reflection. arXiv:2310.11511. 2023.
140. Nandakishor M. DeepRAG: building a custom hindi embedding model for retrieval augmented generation from scratch. arXiv:2503.08213. 2025.
141. Besta M, Barth J, Schreiber E, Kubicek A, Catarino A, Gerstenberger R, et al. Reasoning language models: a blueprint. arXiv:2503.08213. 2025.
142. Luo K, Ding Z, Weng Z, Qiao L, Zhao M, Li X, et al. Let's be self-generated via step by step: a curriculum learning approach to automated reasoning with large language models. arXiv:2410.21728. 2025.
143. Müller J, Oupický J. Post-quantum XML and SAML single sign-on. Pro Priv Enhanc Technol. 2024;2024(4):525–43. doi:10.56553/popets-2024-0128.
144. Bruch S, Gai S, Ingber A. An analysis of fusion functions for hybrid retrieval. ACM Transact Inform Syst. 2023;42(1):1–35. doi:10.1145/3596512.
145. Rajkumar N, Li R, Bahdanau D. Evaluating the text-to-SQL capabilities of large language models. arXiv:2204.00498. 2022.
146. Wang J, Wang J, Athiwaratkun B, Zhang C, Zou J. Mixture-of-agents enhances large language model capabilities. arXiv:2406.04692. 2024.
147. Zhou W, Jiang YE, Li L, Wu J, Wang T, Qiu S, et al. Agents: an open-source framework for autonomous language agents. arXiv:2309.07870. 2023.

148. Yao W, Heinecke S, Niebles JC, Liu Z, Feng Y, Xue L, et al. Retroformer: retrospective large language agents with policy gradient optimization. arXiv:2308.02151. 2024.
149. Jin W, Zhao B, Yu H, Tao X, Yin R, Liu G. Improving embedded knowledge graph multi-hop question answering by introducing relational chain reasoning. Data Min Knowl Disc. 2023;37(1):255–88. doi:10.1007/s10618-022-00891-8.
150. Santhanam K, Khattab O, Saad-Falcon J, Potts C, Zaharia M. ColBERTv2: effective and efficient retrieval via lightweight late interaction. arXiv:2112.01488. 2022.
151. Chen J, Prasad A, Saha S, Stengel-Eskin E, Bansal M. MAgICoRe: multi-agent, iterative, coarse-to-fine refinement for reasoning. arXiv:2409.12147. 2025.
152. Wang Z, Liu J, Zhang S, Yang Y. Poisoned langchain: jailbreak LLMs by LangChain. arXiv:2406.18122. 2024.
153. Lu P, Chen B, Liu S, Thapa R, Boen J, Zou J. OctoTools: an agentic framework with extensible tools for complex reasoning. arXiv:2502.11271. 2025.
154. Gou Z, Shao Z, Gong Y, Shen Y, Yang Y, Huang M, et al. ToRA: a tool-integrated reasoning agent for mathematical problem solving. arXiv:2309.17452. 2024.
155. Song CH, Wu J, Washington C, Sadler BM, Chao WL, Su Y. LLM-planner: few-shot grounded planning for embodied agents with large language models. arXiv:2212.04088. 2023.
156. Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I. Better language models and their implications; 2019 [Internet]. [cited 2025 Jul 9]. OpenAI Blog. Available from: https://openai.com/research/better-language-models.
157. DeepAI. DeepAI: artificial intelligence tools and services; 2025 [Internet]. [cited 2025 Apr 3]. Available from: https://deepai.org/.
158. Hugging Face I. Hugging Face: the AI community building the future [Internet]. [cited 2025 Mar 13]. Available from: https://huggingface.co.
159. Shen Y, Song K, Tan X, Li D, Lu W, Zhuang Y. HuggingGPT: Solving AI tasks with ChatGPT and its friends in hugging face. In: Oh A, Naumann T, Globerson A, Saenko K, Hardt M, Levine S, editors. Advances in neural information processing systems. Vol. 36. Red Hook, NY, USA: Curran Associates, Inc.; 2023. p. 38154–80.
160. Radford A, Kim JW, Xu T, Brockman G, McLeavey C, Sutskever I. Robust speech recognition via large-scale weak supervision. In: Proceedings of the 40th International Conference on Machine Learning. ICML'23; 2023 Jul 23–29; Honolulu, HI, USA.
161. OpenAI. Whisper: robust speech recognition via large-scale weak supervision. GitHub; 2022 [Internet]. [cited 2025 May 13]. Available from: https://github.com/openai/whisper.
162. Radford A, Kim JW, Hallacy C, Ramesh A, Goh G, Agarwal S, et al. Learning transferable visual models from natural language supervision. In: Proceedings of the 38th International Conference on Machine Learning (ICML). Vol. 139. Westminster, UK: PMLR; 2021. p. 8748–63.
163. Alshehri I, Alshehri A, Almalki A, Bamardouf M, Akbar A. BreachSeek: a multi-agent automated penetration tester. arXiv:2409.03789. 2024.
164. Khemani B, Patil S, Kotecha K, Tanwar S. A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions. J Big Data. 2024;11(1):18. doi:10.1186/s40537-023-00876-4.
165. Jiang M, Liu G, Zhao B, Su Y, Jin W. Relation-aware graph structure embedding with co-contrastive learning for drug-drug interaction prediction. Neurocomputing. 2024;572(1):127203. doi:10.1016/j.neucom.2023.127203.
166. Jiang M, Liu G, Su Y, Jin W, Zhao B. Hierarchical multi-relational graph representation learning for large-scale prediction of drug-drug interactions. IEEE Transact Big Data. 2025;11(3):961–75. doi:10.1109/tbdata.2025.3536924.
167. Peng X, Liu Y, Cang Y, Cao C, Chen M. LLM-OptiRA: LLM-driven optimization of resource allocation for non-convex problems in wireless communications. arXiv:2505.02091. 2025.
168. Marta D, Holk S, Vasco M, Lundell J, Homberger T, Busch F, et al. FLoRA: sample-efficient preference-based RL via low-rank style adaptation of reward functions. arXiv:2504.10002. 2025.
169. Jin W, Zhao B, Zhang L, Liu C, Yu H. Back to common sense: oxford dictionary descriptive knowledge augmentation for aspect-based sentiment analysis. Inform Process Manag. 2023;60(3):103260. doi:10.1016/j.ipm.2022.103260.

170. Yadav V, Bethard S. A survey on recent advances in named entity recognition from deep learning models. In: Bender EM, Derczynski L, Isabelle P, editors. Proceedings of the 27th International Conference on Computational Linguistics. Santa Fe, NM, USA: Association for Computational Linguistics; 2018. p. 2145–58. doi:10.1162/coli_a_00178.
171. Kudo T. Subword regularization: improving neural network translation models with multiple subword candidates. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. Melbourne, VIC, Australia: Association for Computational Linguistics; 2018. p. 66–75.
172. Wong B, Tanaka K. High-fidelity pseudo-label generation by large language models for training robust radiology report classifiers. arXiv:2505.01693. 2025.
173. Jin W, Zhao B, Zhang Y, Huang J, Yu H. WordTransABSA: enhancing Aspect-based Sentiment Analysis with masked language modeling for affective token prediction. Expert Syst Appl. 2024;238(12):122289. doi:10.1016/j.eswa.2023.122289.
174. Mandi Z, Jain S, Song S. RoCo: dialectic multi-robot collaboration with large language models. In: 2024 IEEE International Conference on Robotics and Automation (ICRA); 2024 May 13–17; Yokohama, Japan. p. 286–99.
175. Li H, Chong Y, Stepputtis S, Campbell J, Hughes D, Lewis C, et al. Theory of mind for multi-agent collaboration via large language models. In: Bouamor H, Pino J, Bali K, editors. Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. Singapore: Association for Computational Linguistics; 2023. p. 180–92.
176. Nottingham K, Ammanabrolu P, Suhr A, Choi Y, Hajishirzi H, Singh S, et al. Do embodied agents dream of pixelated sheep?: embodied decision making using language guided world modelling. arXiv:2301.12050. 2023.
177. Vidgof M, Bachhofner S, Mendling J. Large language models for business process management: opportunities and challenges. In: Di Francescomarino C, Burattin A, Janiesch C, Sadiq S, editors. Business process management forum. Cham, Switzerland: Springer Nature; 2023. p. 107–23. doi:10.1007/978-3-031-41623-1_7.
178. Haase J, Kremser W, Leopold H, Mendling J, Onnasch L, Plattfaut R. Interdisciplinary directions for researching the effects of robotic process automation and large language models on business processes. Communicat Associat Inform Syst. 2024;54(1):579–604. doi:10.17705/1CAIS.05421.
179. Kaiya Z, Naim M, Kondic J, Cortes M, Ge J, Luo S, et al. Lyfe agents: generative agents for low-cost real-time social interactions. arXiv:2310.02172. 2023.
180. Frick N. Barriers, facilitators and prerequisites for robotic process automation adoption in public administrations: a systematic literature review. In: Proceedings of the 32nd European Conference on Information Systems (ECIS 2024); 2024 Jun 13–19; Paphos, Cyprus.
181. Primbs J, Kern D, Menth M, KrauB C. Streamlining plug-and-charge authorization for electric vehicles with OAuth2 and OIDC. arXiv:2501.14397. 2025.
182. Lodderstedt T, Bradley J, Labunets A, Fett D. Best current practice for OAuth 2.0 security. RFC Editor; 2025 [Internet]. [cited 2025 Jul 9]. Available from: https://www.rfc-editor.org/info/rfc9700.
183. Hartl Z, Derek A. Towards automated formal security analysis of SAML V2.0 Web Browser SSO standard–the POST/Artifact use case. arXiv:2403.11859. 2024.
184. Zhang Y. New approaches to automated software testing based on artificial intelligence. In: 2024 5th International Conference on Artificial Intelligence and Computer Engineering (ICAICE); 2024 Nov 8–10; Wuhu, China. p. 806–10.
185. Pham P, Nguyen V, Nguyen T. A review of AI-augmented end-to-end test automation tools. In: Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering. ASE'22. New York, NY, USA: Association for Computing Machinery; 2023.
186. Ricca F, Marchetto A, Stocco A. AI-based test automation: a grey literature analysis. In: 2021 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW); 2021 Apr 12–16; Porto de Galinhas, Brazil. p. 263–70.
187. Xiao B, Yin Z, Shan Z. Simulating public administration crisis: a novel generative agent-based simulation system to lower technology barriers in social science research. arXiv:2311.06957. 2023.

188. Singh S, Heard J. Human-aware reinforcement learning for adaptive human robot teaming. In: 2022 17th ACM/IEEE International Conference on Human-Robot Interaction (HRI); 2022 Mar 7–10; Sapporo, Japan. p. 1049–52.
189. Amirshirzad N, Kumru A, Oztop E. Human adaptation to human-robot shared control. IEEE Transact Human-Mach Syst. 2019;49(2):126–36. doi:10.1109/thms.2018.2884719.
190. Lange DS, Gutzwiller RS. Human-autonomy teaming patterns in the command and control of teams of autonomous systems. In: Engineering psychology and cognitive ergonomics. Cham, Switzerland: Springer International Publishing; 2016. p. 179–88 doi:10.1007/978-3-319-40030-3_19.
191. Holt S, Luyten MR, van der Schaar M. L2MAC: large language model automatic computer for extensive code generation. arXiv:2310.02003. 2024.
192. Eumemic, Contributors. AI Legion: an LLM-powered autonomous agent platform. GitHub; 2023 [Internet]. [cited 2025 May 9]. Available from: https://github.com/eumemic/ai-legion.
193. Xu L, Hu Z, Zhou D, Ren H, Dong Z, Keutzer K, et al. MAgIC: investigation of large language model powered multi-agent in cognition, adaptability, rationality and collaboration. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. Miami, FL, USA: Association for Computational Linguistics; 2024. p. 7315–32.
194. Ridnik T, Kredo D, Friedman I. Code generation with AlphaCodium: from prompt engineering to flow engineering. arXiv:2401.08500. 2024.
195. Rahman F. LoopGPT: modular auto-GPT framework. PyPI; 2023 [Internet]. [cited 2025 Jan 2]. Available from: https://pypi.org/project/loopgpt/.
196. Zhao A, Huang D, Xu Q, Lin M, Liu YJ, Huang G. ExpeL: LLM agents are experiential learners. In: Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence; 2024 Feb 20–27; Vancouver, BC, Canada. p. 19632–42.
197. Wu J, Antonova R, Kan A, Lepert M, Zeng A, Song S, et al. TidyBot: personalized robot assistance with large language models. In: 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS); 2023 Oct 1–5; Detroit, MI, USA. p. 3546–53.
198. Mugnaini LG, Yamamoto BL, de Alcantara LL, Zacarias V, Bollis E, Pellicer L, et al. Efficient LLMs with AMP: attention heads and MLP pruning. arXiv:2504.21174. 2025.
199. OpenAI. GPT-4 technical report. arXiv:2303.08774. 2023.
200. Team CR. Cogito: a hybrid reasoning model with iterative distillation and amplification. Technical Report; 2023 [Internet]. [cited 2025 Jul 9]. Proprietary model combining symbolic reasoning and neural networks. Available from: https://www.cogitocorp.com.
201. OpenAI. Introducing GPT-4.5. OpenAI; 2025 [Internet]. [cited 2024 Dec 4]. Available from: https://openai.com/index/introducing-gpt-4-5/.
202. Dan Y, Lei Z, Gu Y, Li Y, Yin J, Lin J, et al. EduChat: a large language model-based conversational agent for intelligent education. In: China Conference on Knowledge Graph and Semantic Computing and International Joint Conference on Knowledge Graphs. Singapore: Springer Nature Singapore; 2025. p. 297–308.
203. Wang T, Zhou N, Chen Z. CyberMentor: AI powered learning tool platform to address diverse student needs in cybersecurity education. arXiv:2501.09709. 2025.
204. Gao C, Lan X, Lu Z, Mao J, Piao J, Wang H, et al. S3: social-network simulation system with large language model-empowered agents. arXiv:2307.14984. 2023.
205. Wang Y, Jiang Z, Chen Z, Yang F, Zhou Y, Cho E, et al. RecMind: large language model powered agent for recommendation. In: Findings of the association for computational linguistics: NAACL 2024. Mexico City, Mexico: Association for Computational Linguistics; 2024. p. 4351–64.
206. Tzanis E, Klontzas ME. mAIstro: an open-source multi-agentic system for automated end-to-end development of radiomics and deep learning models for medical imaging. arXiv:2505.03785. 2025.
207. GmbH D. DeepL translate: the world's most accurate translator; 2023 [Internet]. [cited 2025 Jul 9]. Available from: https://www.deepl.com/translator.

208. Rackauckas Z. Rag-fusion: a New take on retrieval augmented generation. Int J Nat Lang Comput. 2024;13(1):37–47. doi:10.5121/ijnlc.2024.13103.
209. Tsang YP, Tang V, Wu CH, Li F. Unlocking the potential of robotic process automation for digital transformation in logistics and supply chain management. J Glob Inf Manag. 2024;32(1):24. doi:10.4018/JGIM.361710.
210. Zhao Q, Wang J, Zhang Y, Jin Y, Zhu K, Chen H, et al. CompeteAI: understanding the competition dynamics of large language model-based agents. In: Proceedings of the 41st International Conference on Machine Learning. ICML'24; 2024 Jul 21–27; Vienna, Austria. p. 61092–6110.
211. Lievano-MartÃnez FA, FernÃndez-Ledesma JD, Burgos D, Branch-Bedoya JW, Jimenez-Builes JA. Intelligent process automation: an application in manufacturing industry. Sustainability. 2022;14(14):8804. doi:10.3390/su14148804.
212. Zhang Y, Hao S, Li S. Development and application of electronic tax declaration robot for individual income tax based on Uipath. In: Proceedings of the 2023 3rd International Conference on Big Data, Artificial Intelligence and Risk Management. ICBAR '23. New York, NY, USA: Association for Computing Machinery; 2024. p. 620–4.
213. Weiss M, Rahaman N, Wuthrich M, Bengio Y, Li LE, Schölkopf B, et al. Rethinking the buyer's inspection paradox in information markets with language agents; 2024 [Internet]. [cited 2025 Jul 9]. Available from: https://openreview.net/forum?id=6werMQy1uz.
214. Akintuyi OB. AI in agriculture: a comparative review of developments in the USA and Africa. Open Access Res J Sci Technol. 2024;10(2):060–70. doi:10.53022/oarjst.2024.10.2.0051.
215. Zhou X, Li G, Sun Z, Liu Z, Chen W, Wu J, et al. D-Bot: database diagnosis system using large language models. Proc VLDB Endow. 2024;17(10):2514–27. doi:10.14778/3675034.3675043.
216. Gill MS, Vyas J, Markaj A, Gehlhoff F, Mercangöz M. Leveraging LLM agents and digital twins for fault handling in process plants. arXiv:2505.02076. 2025.
217. Team O, Contributors. WorkGPT: autonomous agent framework for API orchestration. NPM; 2024 [Internet]. [cited 2025 Mar 9]. Available from: https://www.npmjs.com/package/workgpt.
218. BD (Becton, Dickinson and Company). HemoSphere Alta™ Monitor: hemodynamic monitoring system with predictive analytics and multi-sensor integration. Franklin Lakes, NJ, USA: BD; 2023 [Internet]. [cited 2025 Jul 9]. Available from: https://www.bd.com/en-us/products-and-solutions/products/product-families/hemosphere-alta-monitor.
219. Raza MZ, Xu J, Lim T, Boddy L, Mery CM, Well A, et al. LLM-TA: an LLM-enhanced thematic analysis pipeline for transcripts from parents of children with congenital heart disease. arXiv:2502.01620. 2025.
220. Cincoze Co, Ltd. Cincoze Edge AI. New Taipei City, Taiwan: Cincoze; 2024 [Internet]. [cited 2025 Feb 23]. Available from: https://www.cincoze.com/zh-tw/.
221. Yaacoub A, Da-Rugna J, Assaghir Z. Assessing AI-generated questions' alignment with cognitive frameworks in educational assessment. arXiv:2504.14232. 2025.
222. OneClickQuiz. Boost Moodle with AI-driven quizzes aligned to cognitive & linguistic standards; 2023 [Internet]. [cited 2024 Jun 1]. Available from: https://linktr.ee/oneclickquiz.
223. Kaiser S, Tosun A, Korkmaz T. Benchmarking container technologies on ARM-based edge devices. IEEE Access. 2023;11:107331–47. doi:10.1109/access.2023.3321274.
224. Kaiser S, Haq MS, Tosun A, Korkmaz T. Container technologies for ARM architecture: a comprehensive survey of the state-of-the-art. IEEE Access. 2022;10:84853–81. doi:10.1109/access.2022.3197151.
225. Fang Q, Zhou Y, Guo S, Zhang S, Feng Y. LLaMA-Omni2: LLM-based real-time spoken chatbot with autoregressive streaming speech synthesis. arXiv:2505.02625. 2025.
226. Ai T. Tray.AI: AI-ready integration & automation platform; 2023 [Internet]. [cited 2023 Nov 15]. Available from: https://tray.ai.
227. Li Y, Sun L, Zhang Y. MetaAgents: large language model based agents for decision-making on teaming. Proc ACM Hum-Comput Interact. 2025;9(2):CSCW134. doi:10.1145/3711032.
228. LangChain Inc. LangGraph: stateful, multi-agent LLM application framework; 2024 [Internet]. [cited 2025 May 8]. Available from: https://github.com/langchain-ai/langgraph.

229. da Silva LMV, Köcher A, König N, Gehlhoff F, Fay A. Capability-driven skill generation with LLMs: a RAG-based approach for reusing existing libraries and interfaces. arXiv:2505.03295. 2025.
230. Luo L, Liu Y, Liu R, Phatale S, Guo M, Lara H, et al. Improve mathematical reasoning in language models with automated process supervision; 2025 [Internet]. [cited 2025 Jul 9]. Available from: https://openreview.net/forum?id=KwPUQOQIKt.
231. Chroma. Chroma: the open-source AI application database; 2023 [Internet]. [cited 2023 Nov 15]. Available from: https://docs.trychroma.com/.
232. Fischer KA. Reflective linguistic programming (RLP): a stepping stone in socially-aware AGI (SocialAGI). arXiv:2305.12647. 2023.
233. Dong Y, Jiang X, Jin Z, Li G. Self-collaboration code generation via ChatGPT. ACM Trans Softw Eng Methodol. 2024;33(7):189. doi:10.1145/3672459.
234. Ashrafi N, Bouktif S, Mediani M. Enhancing LLM code generation: a systematic evaluation of multi-agent collaboration and runtime debugging for improved accuracy, reliability, and latency. arXiv:2505.02133. 2025.
235. Zhang H, Du W, Shan J, Zhou Q, Du Y, Tenenbaum JB, et al. Building cooperative embodied agents modularly with large language models. arXiv:2307.02485. 2024.
236. Mialon G, Fourrier C, Swift C, Wolf T, LeCun Y, Scialom T. GAIA: a benchmark for general AI assistants. arXiv:2311.12983. 2023.
237. Schick T, Dwivedi-Yu J, Dessi R, Raileanu R, Lomeli M, Hambro E, et al. Toolformer: language models can teach themselves to use tools. In: Oh A, Naumann T, Globerson A, Saenko K, Hardt M, Levine S, editors. Advances in neural information processing systems. Vol. 36. Red Hook, NY, USA: Curran Associates, Inc.; 2023. p. 68539–51.
238. Inc C. CrewAI: framework for orchestrating role-playing, autonomous AI agents. CrewAI Inc.; 2025 [Internet]. [cited 2025 May 8]. Available from: https://github.com/crewAIInc/crewAI.
239. aisuhua. RESTful API design references; 2023 [Internet]. [cited 2023 Nov 5]. Available from: https://github.com/aisuhua/restful-api-design-references.
240. Johnson J, Douze M, Jégou H. Billion-scale similarity search with GPUs. IEEE Transact Big Data. 2021;7(3):535–47. doi:10.1109/tbdata.2019.2921572.
241. Wang J, Yi X, Guo R, Jin H, Xu P, Li S, et al. Milvus: a purpose-built vector data management system. In: Proceedings of the 2021 International Conference on Management of Data; 2021 Jun 20–25; Online. p. 1–13.
242. LangChain. LangChain Expression Language (LCEL); 2023 [Internet]. [cited 2023 May 15]. Available from: https://python.langchain.com/docs/concepts/lcel/.
243. Wang J, Duan Z. Agent AI with LangGraph: a modular framework for enhancing machine translation using large language models. arXiv:2412.03801. 2024.
244. Liu J. LlamaIndex: data framework for large language model applications. LlamaIndex; 2022 [Internet]. [cited 2025 May 8]. Available from: https://github.com/run-llama/llama_index.
245. Radke AM, Dang MT, Tan A. Using robotic process automation (RPA) to enhance item master data maintenance process. Logforum. 2020;16(1):10.
246. Forethought. AutoChain: building generative agents with large language models; 2023 [Internet]. [cited 2023 Nov 15]. Available from: https://autochain.forethought.ai/.
247. Zhang C, Yang Z, Liu J, Li Y, Han Y, Chen X, et al. AppAgent: multimodal agents as smartphone users. In: Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems. CHI '25. New York, NY, USA: Association for Computing Machinery; 2025.
248. Lin BY, Fu Y, Yang K, Brahman F, Huang S, Bhagavatula C, et al. SwiftSage: a generative agent with fast and slow thinking for complex interactive tasks. In: Advances in neural information processing systems. Vol. 36. Red Hook, NY, USA: Curran Associates, Inc.; 2023. p. 23813–25.
249. FlowiseAI. Flowise: open source low-code tool for building LLM apps; 2023 [Internet]. [cited 2023 Nov 15]. Available from: https://flowiseai.com/.
250. Retool. Retool: the best way to build internal software; 2025 [Internet]. [cited 2025 Jan 9]. Available from: https://retool.com/.

251. Rasheed Z, Waseem M, Ahmad A, Kemell KK, Xiaofeng W, Duc AN, et al. Can large language models serve as data analysts? a multi-agent assisted approach for qualitative data analysis. arXiv:2402.01386. 2024.
252. Smith A, Anderson J. AI, robotics, and the future of jobs. Pew research center; 2014 [Internet]. [cited 2024 Jun 30]. Available from: https://www.pewresearch.org/internet/2014/08/06/future-of-jobs/.
253. Setlur A, Nagpal C, Fisch A, Geng X, Eisenstein J, Agarwal R, et al. Rewarding progress: scaling automated process verifiers for LLM reasoning. arXiv:2410.08146. 2025.
254. Nascimento N, Alencar P, Cowan D. GPT-in-the-loop: adaptive decision-making for multiagent systems. arXiv:2308.10435. 2023.
255. Liang T, He Z, Jiao W, Wang X, Wang Y, Wang R, et al. Encouraging divergent thinking in large language models through multi-agent debate. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. Miami, FL, USA: Association for Computational Linguistics; 2024. p. 17889–904.
256. Wu Y, Min SY, Bisk Y, Salakhutdinov R, Azaria A, Li Y, et al. Plan, eliminate, and track–language models are good teachers for embodied agents. arXiv:2305.02412. 2023.
257. Kovac G, Portelas R, Dominey PF, Oudeyer PY. The SocialAI school: a framework leveraging developmental psychology toward artificial socio-cultural agents. Front Neurorobot. 2024;18:1396359. doi:10.3389/fnbot.2024.1396359.
258. Zhuo TY, Li Z, Huang Y, Shiri F, Wang W, Haffari G, et al. On robustness of prompt-based semantic parsing with large pre-trained language model: an empirical study on codex. In: Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics. Dubrovnik, Croatia: Association for Computational Linguistics; 2023. p. 1090–102.