Technical Appendix for HASHIRU: Hierarchical Agent System for Hybrid Intelligent Resource Utilization

Kunal Pai, Parth Shah, and Harshil Patel

May 31, 2025

Abstract

This technical appendix provides a detailed overview of the HASHIRU system, complementing the main paper. HASHIRU (Hierarchical Agent System for Hybrid Intelligent Resource Utilization) is a novel Multi-Agent System (MAS) framework designed to enhance flexibility, resource efficiency, and adaptability. It features a "CEO" agent dynamically managing specialized "employee" agents, prioritizing local LLMs and incorporating an economic model for resource allocation. This document elaborates on its architecture, components, operational mechanisms, dataset utilization, safety measures, and benchmark results.

1 Introduction and Motivation

Rapid advancements in Large Language Models (LLMs) [6, 12, 51] have spurred the development of autonomous Multi-Agent Systems (MAS) [13, 64]. These systems show promise in complex domains like scientific discovery [4] and software engineering [48]. However, contemporary agentic frameworks often suffer from several limitations. These include **Rigidity**, where predefined roles hinder adaptation [68]; **Resource Obliviousness**, characterized by a lack of mechanisms to optimize computational resources (API costs, memory, CPU), leading to inefficiency [44]; **Model Homogeneity**, the practice of defaulting to a single powerful LLM, thereby missing efficiency gains from diverse models [70]; and **Limited Autonomous Tool Creation**, which restricts dynamic self-improvement [43, 60, 67]. HASHIRU is introduced to address these limitations by providing a dynamic, resource-aware, and adaptable MAS framework.

2 Background and Relation to Prior Work

Intelligent agent concepts have evolved significantly from symbolic AI [53, 55] to current LLM-driven frameworks [61, 66]. While hierarchical MAS models offer clear control [13, 24], they can introduce bottlenecks [14, 15]. HASHIRU employs a CEO-Employee hierarchy but distinguishes itself through **dynamic team composition**, unlike systems with static hierarchies (e.g., CrewAI [10], ChatDev [48]). Its resource management is centralized and budget-constrained, contrasting with market-based mechanisms [8] or systems with implicit cost tracking like AutoGen [65] and LangGraph [29]. HASHIRU's hybrid intelligence prioritizes local models (via Ollama [37]), differing from systems reliant on large proprietary APIs (e.g., GPT-4 [39], Claude 3 [1]). Furthermore, its integrated autonomous API tool creation advances beyond predefined toolsets common in systems like ReAct [**?**] or those with basic function calling [38], aiming for greater autonomy akin to Voyager [60].

3 HASHIRU System Architecture

HASHIRU's architecture is designed to overcome rigidity, resource obliviousness, and limited adaptability through a hierarchical, dynamically managed MAS optimized for hybrid resource utilization.

3.1 Overview

The key tenets of HASHIRU are **Dynamic Hierarchical Coordination**, where the CEO manages strategy, task allocation, and dynamic team composition; **Dynamic Lifecycle Management**, meaning employees are hired/fired based on runtime needs and resource constraints, governed by an economic model; **Hybrid Intelligence**, involving a strategic preference for LLMs within a predefined budget, while flexibly accessing external APIs/models; **Explicit Resource Management**, which ensures continuous monitoring and control of costs (financial, memory) against budgets; and **Adaptive Tooling**, which utilizes predefined tools alongside autonomous creation of new API tools.

3.2 Hierarchical Structure: CEO and Employee Agents

3.2.1 CEO Agent

The CEO Agent serves as a **singleton, central coordinator, and system entry point**. Its **responsibilities** are comprehensive, including interpreting user queries/tasks, decomposing main tasks into subtasks, identifying required capabilities, managing the Employee agent pool (see Section 3.3), assigning subtasks, monitoring Employee progress/performance, synthesizing results into final output, managing the overall resource budget (see Section 3.5), and initiating new tool creation (see Section 3.6). For its **implementation**, it employs Gemini 2.0 Flash [20] as the core LLM. The agent's reasoning capabilities are **enhanced** as its system prompt is engineered to evoke inherent chain-of-thought (CoT) processes [63], complementing its baseline reasoning, tool use support, and cost-efficiency.

3.2.2 Employee Agents

Employee Agents are **specialized agents instantiated by the CEO for specific sub-tasks**, with each typically wrapping an LLM or providing dedicated tool access. Their **characteristics** include *Specialization*, meaning capabilities are tailored to task types (e.g., code generation, data analysis, information retrieval); *Dynamic Existence*, as they are created and destroyed by the CEO based on operational needs, performance, and resource constraints; *Task Execution*, where they receive task specifications from the CEO, execute them, and return results; and *Resource Consumption*, with associated costs (API fees, estimated hardware utilization for local models) being tracked by the system. For their **implementation models**, Employees are constructed using diverse base models. Examples include smaller local models like Mistral 7B [26] and Llama 3 [35], as well as more capable models like Gemini 1.5 [16], Qwen2.5 [50], and DeepSeek-R1 [11]. The CEO agent configures these employees via tailored system prompts generated based on specific task requirements. **Model Access Modalities** are varied. *Local Execution* involves models running locally, often facilitated by Ollama [37]. *External APIs* allow integration with external models such as Gemini 2.5 Flash [21], Qwen QwQ [59], Hermes3 [58], with future considerations for Llama 4 [34] and Mistral Saba [36]. Access is managed via platforms like Hugging Face [25], Groq [22], and Lambda.ai [28].

3.3 Dynamic Agent Lifecycle Management

A core innovation is the CEO's dynamic management (hiring/firing) of Employee agents. This process is driven by a cost-benefit analysis aimed at optimizing task performance within defined resource constraints. **Hiring Triggers** occur if a sub-task requires capabilities not currently available or if existing agents provide them inefficiently. Conversely, **Firing Triggers** may lead to an agent being fired if it underperforms, remains idle for extended periods, becomes too costly, or if resource limits (budget, memory) are being approached. The **Decision Factors** influencing these actions include Task Requirements (specific capabilities needed for pending subtasks), Agent Performance (historical success rates, quality of output, operational efficiency), and Operational Costs (API fees, estimated compute for local models, and other associated costs). HASHIRU incorporates an **economic model** to regulate this lifecycle. This model includes a **Hiring Cost ("Starting Bonus")**, a one-time cost incurred upon instantiation of local models, representing setup overhead, which can be quantitatively scaled based on the model's resource profile (e.g., higher for models requiring more VRAM). An **Invocation Cost ("Salary")** is a recurring cost applied each time a local model is used, reflecting the operational load (e.g., inferred compute, system resource engagement). Finally, an **Expense Cost** is a recurring cost for external API calls (e.g., OpenAI, Anthropic), typically calculated based on token usage as per the provider's pricing. These transaction costs discourage excessive agent churn, promoting team stability and efficient resource utilization.

3.4 Hybrid Intelligence and Model Management

HASHIRU is architected for **hybrid intelligence**, strategically leveraging a diverse set of cognitive resources. A key principle is **Local-First Prioritization**, giving preference to smaller (typically 3B–7B parameters), cost-effective local LLMs, often managed via Ollama [37]. This approach enhances efficiency, reduces reliance on external APIs, and offers potential benefits in terms of privacy and latency. The system also supports **Integrated External Resources**. This means it can flexibly integrate External LLM APIs, providing access to powerful foundation models (e.g., Gemini 2.5 Flash [21]) when task complexity demands it, subject to cost-benefit analysis by the CEO. It can also integrate External Tool APIs from third-party software and data sources, and Self-Created APIs, which are tools dynamically generated by HASHIRU itself (see Section 3.6). The **CEO-Led Resource Selection** process involves the CEO managing this heterogeneous pool of models and tools, selecting the most appropriate resource based on task difficulty, required capabilities, and budgetary constraints.

3.5 Resource Monitoring and Control

Explicit resource management is a central feature of HASHIRU. The system, coordinated by the CEO, monitors both **Financial Costs**, which include the accumulation of external API costs (based on documented pricing) and the internal "hiring" and "invocation" costs from the economic model for local agents, and **Memory Usage**, referring to the memory footprint (e.g., VRAM) of active Employee agents. For local models, this is estimated based on known requirements and tracked as a percentage of a predefined total available budget (e.g., a 16 GiB VRAM capacity might represent 100% of the local model memory budget).

3.6 Tool Utilization and Autonomous Creation

HASHIRU's CEO agent utilizes predefined tools (functions, APIs, databases) to interact with external environments and perform actions beyond text generation, aligning with established practices [38, 67]. A distinctive feature is **integrated, autonomous tool creation**. This process unfolds in three main stages. First, for **Specification**, the CEO defines the tool's specification, including inputs, outputs, and desired functionality, when it identifies a missing capability. Second, in **Commissioning Logic Generation**, it commissions the generation of the tool's logic (code), which may involve using external APIs (with provided credentials) or leveraging a specialized code-generating employee agent. Finally, during **Deployment**, the generated logic is deployed as a new, callable API endpoint within the HASHIRU ecosystem. This autonomous creation process employs a few-shot prompting approach, where HASHIRU analyzes existing tools within its system to learn how to specify and implement new ones [6]. The system can then iteratively refine the generated tool code by analyzing execution errors or suboptimal outputs, promoting self-correction.

3.7 Memory Function: Learning from Experience

HASHIRU incorporates a **Memory Function** enabling the CEO to learn from past interactions, particularly errors, and adapt its strategies. The **Mechanism** involves a historical log that stores significant past events, focusing on failed attempts or suboptimal outcomes. For **Retrieval**, when encountering new or recurring challenges, the system queries this memory. Retrieval relies on semantic similarity between the current context (task description, recent actions, error messages) and stored memory entries. Embeddings are generated by the **all-MiniLM-L6-v2** model [62], and **cosine similarity** determines relevance. In terms of **Application**, retrieved memories provide contextual information, helping agents understand past failures and adjust strategies to avoid repeating mistakes. This process aligns with Retrieval-Augmented Generation (RAG) concepts [30] and supports learning by reflecting on past actions, similar to ideas in self-reflective RAG [3] and frameworks like Reflexion [54].

4 Case Studies Demonstrating Self-Improvement

Case Study 1: Self-Generating Cost Model for Agent Specialization, details how HASHIRU automated the research and integration of local model performance data and cloud API costs into its internal economic model using web search capabilities (Commit: `https://github.com/HASHIRU-AI/HASHIRU/commit/70dc268b121cbd7c50c6691645d8a99912766965`).

Case Study 2: Autonomous Tool Integration for the CEO Agent, shows HASHIRU demonstrating autonomous integration of new tools by employing few-shot learning from existing tool templates and iterative bug fixing, directly committing new tools to its codebase (Commits: `https://github.com/HASHIRU-AI/HASHIRU/blob/main/src/tools/user_tools/python_sandbox_tool.py`, and tool example `https://github.com/HASHIRU-AI/HASHIRU/blob/main/src/tools/default_tools/get_website_tool.py`).

Case Study 3: Autonomous Budget Management, showcases a self-regulating mechanism to monitor budget allocation and prevent overspending, with HASHIRU refusing external API use when budget limits were exceeded. This addresses common issues with token-based billing [40, 41, 52].
Case Study 4: Learning from Experience via Error Analysis and Knowledge Retrieval, explains how HASHIRU learned from incorrect responses by generating linguistic critiques and actionable guidance (akin to verbal reinforcement learning [54]), then indexing this feedback into a RAG system [30] for future retrieval. This mirrors RLHF principles [42, 71].

5 Experimental Setup

5.1 Evaluation Objectives

Experiments were designed to evaluate HASHIRU's performance, efficiency, and adaptability. Further objectives included assessing the impact of dynamic management with economic constraints on resource utilization and task success, the effectiveness of the hybrid (local-first) intelligence strategy, and the system's ability to autonomously create and utilize tools.

5.2 Evaluation Tasks and Datasets

A diverse set of tasks was selected.

5.2.1 Academic Paper Review

For this task, HASHIRU was tasked to simulate peer review by generating a review summary and recommending acceptance/rejection for ICLR 2023 papers. The **Dataset** consisted of 50 papers from ICLR 2023. The **Rationale** for this task was to probe the system's ability in decomposition of complex criteria, delegation to specialized agents, and resource management for large documents.

5.2.2 Reasoning and Problem-Solving Tasks

To evaluate broad reasoning, knowledge retrieval, and problem-solving, several benchmarks were employed. These included **Humanity's Last Exam [47]**, a test of graduate-level technical knowledge and complex reasoning (using a subset of 40 questions). Another was the **ARC (AI2 Reasoning Challenge) [5]**, featuring challenging multiple-choice science questions that demand knowledge retrieval, logical inference, and multi-step problem-solving (using a mixed set of 100 questions). **StrategyQA [17]** provided yes/no questions requiring implicit multi-step reasoning with evidence from Wikipedia (subset of 100 questions). For mathematical and scientific problem-solving, **JEEBench [2]** was used, containing pre-engineering problems from IIT JEE-Advanced that require long-horizon reasoning (subset of 120 questions). Arithmetic and algebraic reasoning were further assessed using **GSM8K [9]**, which consists of grade school math word problems evaluating multi-step mathematical reasoning (subset of 100 questions); this benchmark is crucial for assessing arithmetic and algebraic reasoning. **SVAMP [45, 46]** provided math word problems designed to test question sensitivity and robust reasoning against structural alterations (subset of 100 questions). Finally, **MMLU [23]** evaluated pretrained knowledge across 57 diverse subjects (STEM, humanities, law, ethics) from elementary to professional levels, with subsets used for law (112 questions), math (110 questions), and psychology (127 questions).

5.2.3 Safety Evaluation

The safety evaluation **Task** was to assess whether the CEO's delegation mechanism compromises its inherent safety protocols. The **Dataset** used was a 50-prompt subset of JailbreakBench [7], which includes adversarial prompts designed to test LLM safety robustness [32, 33, 72]. This evaluation is **Critical** for ensuring responsible operation when task delegation is involved.

5.3 Baselines for Comparison

The primary baseline for reasoning tasks was Gemini 2.0 Flash [20] operating in isolation, with both HASHIRU and Gemini 2.0 Flash at a temperature of 0.2; this choice was made to isolate the architectural benefits of HASHIRU over a single competent agent. For paper reviews (which are multi-agent by design) and JailbreakBench (which evaluates HASHIRU's internal safety integrity), direct comparison to a single isolated agent was not applicable in the same way; HASHIRU's performance was evaluated directly in these cases. Statistical significance of differences was assessed using t-tests [57].

5.4 Evaluation Metrics

The evaluation metrics included **Task Success Rate / Quality**, measured as the percentage of tasks completed correctly (binary for most tasks) or the quality of output (e.g., coherence, relevance for paper reviews). **Resource Consumption** was another key metric, focusing on wall-clock time per task; financial costs and memory usage were also important considerations in HASHIRU's design and evaluation philosophy. Finally, **System Dynamics and Adaptability** were assessed qualitatively, including the number and utility of autonomously created tools and agents where applicable.

6 Results and Discussion

The results offer several insights. Regarding the **Academic Paper Review**, HASHIRU's 58% success rate highlighted its capability to manage complex, multi-perspective tasks by dynamically forming an appropriate team of agents (three Gemini 1.5 Flash [19] models). Despite leveraging capable models, the overall resource footprint remained low ($\approx 100\,\text{s}$ per review), illustrating how dynamic lifecycle management and hybrid intelligence can outperform a single, monolithic agent [69] or a static multi-agent system like AgentReview [27]. In terms of **Safety (JailbreakBench)**, a 100% success rate in safely handling adversarial prompts by the CEO agent (without harmful delegation) demonstrated that HASHIRU's delegation mechanism did not compromise the foundational model's safety. For **Reasoning Tasks**, statistically significant improvements were observed on JEEBench ($p < 0.05$), GSM8K ($p < 0.01$), SVAMP ($p < 0.05$), and MMLU Math ($p < 0.05$), indicating strong performance in mathematical and formal reasoning, often due to effective tool integration. Notably, for GSM8K, HASHIRU achieved 96% versus the baseline's 61%. On other tasks such as AI2 Reasoning Challenge, Humanity's Last Exam, StrategyQA, MMLU Law, and MMLU Psychology, HASHIRU showed comparable or slightly better performance than the baseline, though these differences were not always statistically significant. This suggests areas where more specialized agent configurations or advanced reasoning strategies could be beneficial. For instance, on Humanity's Last Exam, HASHIRU (5%) doubled the baseline's score (2.5%), indicating better handling of highly complex tasks by deploying more potent specialized agents (DeepSeek-R1 7B [11]). Overall, the results support HASHIRU's core contributions: dynamic resource-aware agent lifecycle management, a hybrid intelligence model, the potential for autonomous tool creation (demonstrated in case studies and implicit in benchmark tool use), and an effective economic model.

7 Limitations and Future Work

Current limitations include the CEO agent's restricted hierarchical communication. Employee agents cannot create sub-agents, which limits the depth and flexibility of delegation. The system also requires further progress in autonomous tool creation and alignment, calibration of the internal economic model, and optimizing memory for long-term histories and complex conversational states. Additionally, the cost-effectiveness of the system under varying combinations of local and external model usage remains underexplored, particularly regarding task-specific trade-offs in performance and latency.

Future work will address these limitations and extend HASHIRU's capabilities across multiple dimensions. Key priorities include improving the CEO's strategic reasoning, enabling deeper agent hierarchies, and exploring decentralized cognition for scalable agent collaboration. We also plan to design a full-spectrum tool management lifecycle encompassing creation, validation, reuse, and deprecation, supported by richer metadata and confidence scores. The economic model will be made adaptive, learning from historical interactions and resource utilization patterns to dynamically rebalance incentives and costs.

A critical enhancement will be the introduction of **calibrated tool invocation**: HASHIRU will compare its internal confidence with the expected utility and reliability of a tool before invocation, using uncertainty quantification to decide whether tool use is warranted. This draws on recent advances in calibration techniques for LLMs (e.g., [31, 56]) and is particularly relevant given the growing prevalence of tool-augmented agents (e.g., [49]). Other directions include improved system explainability through cost-benefit and ablation analyses, expanding the range of supported local models, specializing architectural components for domains like peer review and formal verification, and formalizing a principled ethical and safety framework to govern autonomous decisions and tool construction.

8 Conclusion

This technical appendix has detailed the architecture, mechanisms, and evaluation of HASHIRU. By integrating hierarchical control, dynamic resource-aware agent management, a hybrid local-first intelligence strategy, and autonomous tool creation, HASHIRU offers a significant step towards more robust, efficient, and adaptable multi-agent systems.

[1] Experiments were run on a MacBook M1 2020 edition.

Table 1: Summary of Experimental Results. SR denotes Success Rate.[1]

Task	HASHIRU SR (%)	Baseline SR (%)	p-value	Avg. Time (s)	Resource Use (Illustrative HASHIRU Config)
ICLR 2023 Paper Review	**58**	N/A	N/A	≈100	Low (3 Gemini 1.5 Flash [19] models)
JailbreakBench	**100**	N/A	N/A	≈1	Negligible (CEO model)
AI2 Reasoning Challenge	**96.5**	95	>0.05	≈2	Low (1 Gemini 1.5 8B [18])
Humanity's Last Exam	**5**	2.5	>0.05	≈15	Moderate to High (1 DeepSeek-R1 7B [11])
StrategyQA	**85**	82	>0.05	≈2	Negligible (Tools)
JEEBench	**80**	68.3	**<0.05**	≈9	Negligible (Tools)
GSM8K	**96**	61	**<0.01**	≈2	Low (Tools & 1 Gemini 1.5 8B [18])
SVAMP	**92**	84	**<0.05**	≈3	Negligible (Tools)
MMLU Law	58.4	**61.6**	>0.05	≈3	Low to Moderate (Tools & 1 Gemini 2.5 Flash [21])
MMLU Math	**91.8**	87.2	**<0.05**	≈4	Negligible (Tools)
MMLU Psychology	**78.7**	78.3	>0.05	≈3	Low to Moderate (Tools & 1 Gemini 2.5 Flash [21])

References

[1] Anthropic. The Claude 3 model family: Opus, Sonnet, Haiku. Model Card, March 2024. Accessed: 2025-05-01.

[2] Daman Arora, Himanshu Singh, and Mausam. Have LLMs advanced enough? a challenging problem solving benchmark for large language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7527–7543, Singapore, December 2023. Association for Computational Linguistics.

[3] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*, 2023.

[4] Daniil A Boiko, Robert MacKnight, and Gabe Gomes. Emergent autonomous scientific research capabilities of large language models. *arXiv preprint arXiv:2304.05332*, 2023.

[5] Michael Boratko, Harshit Padigela, Divyendra Mikkilineni, Pritish Yuvraj, Rajarshi Das, Andrew McCallum, Maria Chang, Achille Fokoue-Nkoutche, Pavan Kapanipathi, Nicholas Mattei, et al. A systematic classification of knowledge, reasoning, and context within the arc dataset. *arXiv preprint arXiv:1806.00358*, 2018.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[7] Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J. Pappas, Florian Tramèr, Hamed Hassani, and Eric Wong. Jailbreakbench: An open robustness benchmark for jailbreaking large language models. In *NeurIPS Datasets and Benchmarks Track*, 2024.

[8] Scott H. Clearwater, editor. *Market-Based Control: A Paradigm for Distributed Resource Allocation*. World Scientific, 1996.

[9] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. Dataset introduced: GSM8K (Grade School Math 8K).

[10] CrewAI Inc. Crewai. `https://www.crewai.com/`, 2025. Accessed: 2025-05-01.

[11] DeepSeek-AI and others. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. 2025. arXiv:2501.12948.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[13] Ali Dorri, Salil S Kanhere, and Raja Jurdak. Multi-agent systems: A survey. *Ieee Access*, 6:28573–28593, 2018.

[14] Matthew E Gaston and Marie DesJardins. Agent-organized networks for dynamic team formation. In *Proceedings of the fourth interna-*

tional joint conference on Autonomous agents and multiagent systems, pages 230–237, 2005.

[15] Matthew E Gaston and Marie DesJardins. Agent-organized networks for multi-agent production and exchange. In *Proceedings of the 20th national conference on Artificial intelligence-Volume 1*, pages 77–82, 2005.

[16] Gemini Team. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. 2024. arXiv:2403.05530.

[17] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics (TACL)*, 2021.

[18] Google DeepMind and Google AI. Gemini 1.5 flash-8b: Production-ready lightweight model. `https://developers.googleblog.com/en/gemini-15-flash-8b-is-now-generally-available-for-use/`, 2024. Accessed: 2025-05-24.

[19] Google DeepMind and Google AI. Gemini 1.5 flash: Lightweight multimodal model. `https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/1-5-flash`, 2024. Accessed: 2025-05-24.

[20] Google DeepMind and Google AI. Gemini 2.0 flash: Model card, api, and announcement. `https://developers.googleblog.com/en/start-building-with-the-gemini-2-0-flash-family/`, 2025. See also: `https://console.cloud.google.com/vertex-ai/publishers/google/model-garden/gemini-2.0-flash-001`, `https://ai.google.dev/gemini-api/docs/models`. Accessed: 2025-05-22.

[21] Google DeepMind and Google AI. Gemini 2.5 flash: Model card, api, and announcement. `https://developers.googleblog.com/en/start-building-with-gemini-25-flash/`, 2025. See also: `https://console.cloud.google.com/vertex-ai/publishers/google/model-garden/gemini-2.5-flash-preview-04-17?inv=1&invt=AbxICQ`, `https://ai.google.dev/gemini-api/docs/models`. Accessed: 2025-05-11.

[22] Groq, Inc. Groq: Fast ai inference, 2025. Accessed: 2025-05-22.

[23] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021.

[24] Bryan Horling and Victor Lesser. A survey of multi-agent organizational paradigms. *The Knowledge engineering review*, 19(4):281–316, 2004.

[25] Hugging Face, Inc. Hugging face: The ai community building the future, 2025. Accessed: 2025-05-22.

[26] Albert Q Jiang, Alexandre Xu, Arthur Mensch Guillaume Lample Nicolàs Lachaux, François Rozenberg, Timothée Lacroix, Thibaut Lavril, Teven Le Scao Eleonora Gaddipati, Lucile Saulnier Lixin Ortiz, Dieuwke Hiemstra Lélio Renard Tang, et al. Mistral 7B. 2023.

[27] Yiqiao Jin, Qinlin Zhao, Yiyang Wang, Hao Chen, Kaijie Zhu, Yijia Xiao, and Jindong Wang. Agentreview: Exploring peer review dynamics with llm agents, 2024.

[28] Lambda Labs. Lambda: Gpu cloud and deep learning workstations, 2025. Accessed: 2025-05-22.

[29] LangChain. Langgraph: A framework for agentic workflows. `https://www.langchain.com/langgraph`, 2024. Accessed: May 1, 2025.

[30] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021.

[31] Putra Manggala, Atalanti A Mastakouri, Elke Kirschbaum, Shiva Kasiviswanathan, and Aaditya Ramdas. Qa-calibration of language model confidence scores. In *The Thirteenth International Conference on Learning Representations.*

[32] Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, David Forsyth, and Dan Hendrycks. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024.

[33] Mantas Mazeika, Andy Zou, Norman Mu, Long Phan, Zifan Wang, Chunru Yu, Adam Khoja, Fengqing Jiang, Aidan O'Gara, Ellie Sakhaee, Zhen Xiang, Arezoo Rajabi, Dan Hendrycks, Radha Poovendran, Bo Li, and David Forsyth.

Tdc 2023 (llm edition): The trojan detection challenge. In *NeurIPS Competition Track*, 2023.

[34] Meta AI. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. Weblog post, April 2025. Accessed: May 30, 2025.

[35] Meta Llama Team. The Llama 3 Herd of Models. 2024. arXiv:2407.21783.

[36] Mistral AI. Mistral saba. Weblog post, February 2025. Accessed: May 30, 2025.

[37] Ollama Team. Ollama. `https://ollama.com/`, 2023. Accessed: 2025-05-01.

[38] OpenAI. Function calling. OpenAI API Documentation, 2023. Accessed: 2025-05-01.

[39] OpenAI. Gpt-4 technical report, 2023.

[40] OpenAI Community. Sos: Alarming situation of excessive billing, 2025.

[41] OpenAI Community. Sudden high costs for chatgpt api usage, 2025.

[42] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[43] Aaron Parisi, Yao Zhao, and Noah Fiedel. Talm: Tool augmented language models, 2022.

[44] Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *The 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*, UIST '23, page 1–22, New York, NY, USA, 2023. Association for Computing Machinery.

[45] Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are nlp models really able to solve simple math word problems? *arXiv preprint arXiv:2103.07191*, 2021.

[46] Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are NLP models really able to solve simple math word problems? In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2021. Introduces the SVAMP challenge dataset.

[47] Long Phan, Alice Gatti, Ziwen Han, et al. Humanity's last exam, 2025.

[48] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. Chatdev: Communicative agents for software development. *arXiv preprint arXiv:2307.07924*, 2023.

[49] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Shijie Wang, Zelin Lu, Siyu Xi, Xiao Liu, Yongyan Li, Zihan Wang, Zixuan Liu, Jian-Guang Lou, et al. Toolllm: Facilitating large language models to master 16000+ real-world APIs, 2023. Accessed: May 26, 2025.

[50] Qwen Team, An Yang, et al. Qwen2.5 Technical Report. 2024. arXiv:2412.15115.

[51] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[52] Reddit user. 0.56 to $343.15 in minutes – google gemini api, 2025.

[53] Stuart J. Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2010.

[54] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652, 2023.

[55] Yoav Shoham. Agent-oriented programming. *Artificial Intelligence*, 60(1):51–92, 1993.

[56] Claudio Spiess, David Gros, Kunal Suresh Pai, Michael Pradel, Md Rafiqul Islam Rabin, Amin Alipour, Susmit Jha, Prem Devanbu, and Toufique Ahmed. Calibration and correctness of language models for code. *arXiv preprint arXiv:2402.02047*, 2024.

[57] Student. The probable error of a mean. *Biometrika*, pages 1–25, 1908.

[58] Ryan Teknium, Jeffrey Quesnelle, and Chen Guang. Hermes 3 technical report. *arXiv preprint arXiv:2408.11857*, 2024.

[59] The Qwen Team. Qwq-32b: Embracing the power of reinforcement learning. Weblog post, March 2025. Accessed: May 30, 2025.

[60] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023.

[61] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. A survey on large language model based autonomous agents, 2023.

[62] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, 2020.

[63] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[64] Michael Wooldridge. *An introduction to multiagent systems*. John wiley & sons, 2009.

[65] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed H. Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

[66] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Linyi Yang, Ting Ruan, Yongquan Yang, Peng Li, Yitao Chang, and Yanlin Wang. The rise and potential of large language model based agents: A survey, 2023.

[67] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2210.03629.

[68] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B. Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models, 2023.

[69] Ruiyang Zhou, Lu Chen, and Kai Yu. Is llm a reliable reviewer? a comprehensive evaluation of llm on automatic paper reviewing tasks. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 9340–9351, 2024.

[70] Wangchunshu Zhou, Jianshu Chen, Jialong Wu, Yiheng Xu, Kexin Wang, Jintian Zhang, Yuan Gao, Zhiyong Wu, Kevin Tian, Yubo Feng, Linyi Yang, Bokai Quan, Cong Yu, Yuhang Wang, Shishen Lan, Yan Wang, Hong-Cheng Guo, Chaoyu Chen, Tianxiang Sun, Jin Xiong, Yi Lu, Peng Li, Lichao Sun, Lifan Yuan, Hang Li, and Xiangang Li. Agents: An open-source framework for large language model based autonomous agents, 2023.

[71] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

[72] Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023.