AI2Agent: An End-to-End Framework for Deploying AI Projects as Autonomous Agents

Jiaxiang Chen[1,2], Jingwei Shi[1], Lei Gan[1], Jiale Zhang[1], Qingyu Zhang[1], Dongqian Zhang[1], Xin Pang[1*], Zhucong Li[1,2*], Yinghui Xu[2]
[1] Continue-AI
[2] Artificial Intelligence Innovation and Incubation Institute, Fudan University, Shanghai, China
pxavdpro@gmail.com, {jiaxiangchen23,zcli22}@m.fudan.edu.cn
{xuyinghui}@fudan.edu.cn

Abstract

As AI technology advances, it is driving innovation across industries, increasing the demand for scalable AI project deployment. However, deployment remains a critical challenge due to complex environment configurations, dependency conflicts, cross-platform adaptation, and debugging difficulties, which hinder automation and adoption. This paper introduces AI2Agent, an end-to-end framework that automates AI project deployment through guideline-driven execution, self-adaptive debugging, and case & solution accumulation. AI2Agent dynamically analyzes deployment challenges, learns from past cases, and iteratively refines its approach, significantly reducing human intervention. To evaluate its effectiveness, we conducted experiments on 30 AI deployment cases, covering TTS, text-to-image generation, image editing, and other AI applications. Results show that AI2Agent significantly reduces deployment time and improves success rates. The code[1] and demo video[2] are now publicly accessible.

1 Introduction

AI is revolutionizing industries, from autonomous driving to healthcare and finance. However, to fully realize its potential, AI must be effectively deployed into diverse environments. Deployment remains a major bottleneck due to complex environment configurations, dependency conflicts, and cross-platform adaptation issues, which hinder scalability and adoption.By automating deployment and debugging, AI can be more efficiently integrated across diverse domains, reducing engineering barriers and accelerating innovation at scale.

Figure 1: **Left:** The AI2Agent user interface, illustrating the automated workflow where a user request (e.g., generating a talk show in a specific style) initiates a structured execution process. This includes searching for a suitable project, following the predefined guidelines for execution, auto-deployment and debug to ensure success. **Right:** A conceptual visualization of local auto-deployment and Agent auto-package, demonstrating how AI2Agent transforms text-to-speech(TTS) functionality into a fully autonomous agent.

Automated deployments in industry predominantly follow DevOps paradigm (Bass et al., 2015; Ebert et al., 2016), where execution environments, dependencies, and orchestration rules are defined using static configuration files (e.g., YAML). In this approach, developers manually configure environments, specifying software packages, version constraints, and computational resource allocations. CI/CD pipelines are then employed to automate building, testing, and deployment. To enhance automation, tools like AutoDevOps (Au-

*Corresponding author.
[1] https://github.com/continue-ai-company/AI2Agent
[2] https://youtu.be/seRTYtwgLrk

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), pages 535–541

toDevops, 2019) utilize predefined templates that streamline the configuration process.

However, these methods have significant limitations. First, they lack flexibility, as they cannot adjust deployment strategies based on runtime conditions. This often leads to manual fixes for dependency conflicts, environment mismatches, and hardware compatibility issues. Second, they fail to retain and reuse past solutions, meaning each deployment starts from scratch. Developers must repeatedly troubleshoot the same problems, making the process inefficient and time-consuming. Third, they do not support scalable AI workflows, as they focus on deploying isolated models rather than integrating multiple AI components. Real-world AI applications often require chaining models or services together, but without standardized interfaces, these tools cannot effectively automate such tasks. As a result, deployment remains fragmented, difficult to scale, and heavily dependent on manual intervention.

To overcome these limitations, we introduce AI2Agent, an end-to-end framework that transforms AI projects into Autonomous Agents, enabling adaptive and reusable deployments. Unlike static DevOps paradigm, AI2Agent leverages autonomous execution, real-time debug, and experience accumulation to stabilize deployment processes.It consists of three components: Guideline-driven Execution, Self-adaptive Debug, and Case & Solution Accumulation. Guideline-driven Execution ensures repeatable and structured deployments by following predefined guideline steps, such as searching for dependencies and executing system commands. Unlike static scripts, AI2Agent adapts dynamically to various environments. Self-adaptive Debug enhances deployment reliability by adjusting strategies based on real-time feedback, autonomously troubleshooting issues like search online or query the knowledge repository. Finally, Case & Solution Accumulation utilizes a Knowledge Repository and RAG (Edge et al., 2024; Guo et al., 2024; infiniflow, 2024) to store past experiences, continuously refining deployment strategies and reducing errors, leading to more efficient deployments over time.

To validate the capabilities of AI2Agent, we conducted a case study on the automated deployment of multiple AI applications, including TTS (Wang et al., 2025), text-to-image generation (Ramesh et al., 2021), and image editing (Brooks et al., 2023). As shown in Figure 1, AI2Agent autonomously searches for and executes deployments, following predefined guidelines, and successfully packages them as Agents through self-adaptive debug. Test results show that AI2Agent significantly shortened deployment time, improved success rates, and reduced errors, leading to faster and more reliable deployments. This demonstrates its potential in enabling standardized, modular, and reusable AI deployments, paving the way for a more autonomous and interoperable AI ecosystem.

The key contributions of this paper are as follows:

- We introduce **AI2Agent**, an end-to-end framework that automates AI project deployment by transforming them into autonomous Agents. It provides a standardized Agent interface, enabling modular management, seamless execution, and improved reusability, fostering a more interoperable AI ecosystem.

- Our approach comprises **Guideline-driven Execution, Self-adaptive Debug, and Case & Solution Accumulation**, forming a structured yet flexible framework. It follows predefined guidelines while dynamically adapting to deployment environments and continuously improving through accumulated experience.

- We evaluate AI2Agent across **30 AI projects**, covering areas such as TTS, text-to-image generation, and image editing. Experimental results show **significant reductions in deployment time and error rates**, demonstrating the effectiveness and reliability of our approach in streamlining AI deployment.

2 Related Work

2.1 DevOps Paradigm

Automated deployments in industry predominantly follow the DevOps paradigm (Bass et al., 2015; Ebert et al., 2016), where execution environments, dependencies, and orchestration rules are defined using static configuration files (e.g., YAML). Developers manually specify dependencies, version constraints, and resource allocations, while CI/CD pipelines automate the build, test, and deployment processes. While this improves

reproducibility, adapting to new environments, debugging failures, and handling model updates in AI projects still require significant manual effort.

To enhance automation, AutoDevOps (AutoDevops, 2019) offers predefined templates that simplify configuration and deployment. While effective for standardized workflows, its static nature limits adaptability, making it challenging to handle the complexity and variability of AI deployments. AI projects often require integrating multiple models, dynamically managing resource constraints, and resolving intricate dependency conflictstasks that demand greater flexibility. Although some approaches (Battina, 2019; Karamitsos et al., 2020; Bou Ghantous, 2024; Enemosah, 2025) incorporate machine learning or LLM to improve automation, they still lack the autonomy and intelligence needed to independently adapt to evolving deployment requirements.

As shown in Figure 2, AI2Agent addresses these challenges through three key components: Guideline-driven Execution, Self-adaptive Debug, and Case & Solution Accumulation. By following structured deployment guidelines, autonomously identifying and fixing errors, and continuously refining deployment strategies based on past cases, AI2Agent significantly reduces manual intervention. This enables more flexible, efficient, and scalable AI deployments across diverse environments.

2.2 LLM-Based Agents

Large Language Models (LLMs) (Openai, 2023) excel in reasoning and decision-making but struggle with executing actions and using external tools. AI agents address this by integrating LLMs with structured tool use, enhancing automation and adaptability.

ReAct (Yao et al., 2023) enables agents to iteratively reason, act, and observe but lacks mechanisms to draw on past experiences, thus limiting long-term planning. AutoGPT (Yang et al., 2023) improves autonomy through multi-step planning yet struggles with execution reliability in real-world scenarios. MetaGPT (Hong et al., 2023) enhances multi-agent collaboration but remains constrained by workflow adaptability. AutoGen (Wu et al., 2023) introduces agent collaboration but faces issues with scalability and consistency.

Existing approaches either focus on reasoning without leveraging past experiences or improve execution with tools but lack adaptive workflow management. AI2Agent addresses this gap by integrating experience-driven learning, structured tool use, and dynamic workflow orchestration, ensuring efficient and adaptable AI deployment.

3 Method

3.1 Overview of the AI2Agent Framework

As illustrated in Algorithm 1, AI2Agent is an end-to-end framework designed to transform AI projects into autonomous agents, enhancing both automation and reusability in deployment. Unlike traditional DevOps and AutoDevOps, which rely on manual configurations or static templates, AI2Agent integrates intelligent reasoning, automated debugging, and iterative experience accumulation to enable dynamic and adaptive deployment. The framework consists of three core modules: **Guideline-Driven Execution**, which standardizes deployment steps; **Self-Adaptive Debug**, which resolves issues through automated analysis; and **Case & Solution Accumulation**, which continuously refines deployment strategies based on past experiences.

3.2 Guideline-Driven Execution

AI2Agent adopts a guideline-driven execution strategy to ensure a structured, efficient, and reliable deployment process. Instead of relying solely on autonomous adjustments, it follows predefined, validated workflows that incorporate best practices and accumulated expertise, as illustrated in Figure 3.

By adhering to these step-by-step guidelines, AI2Agent minimizes uncertainty and maintains consistency across deployments. In complex or ambiguous scenarios, it proactively references its Knowledge Repository to retrieve relevant cases and proven solutions, thereby reducing unnecessary trial-and-error debugging.

This structured approach not only improves deployment stability and efficiency but also mitigates potential failures by grounding the execution process in historical insights and validated experience.

3.3 Self-Adaptive Debug

To address the dynamic nature of deployment environments, AI2Agent integrates a self-adaptive debugging mechanism that refines execution strategies based on real-time feedback. This mechanism consists of three key components:

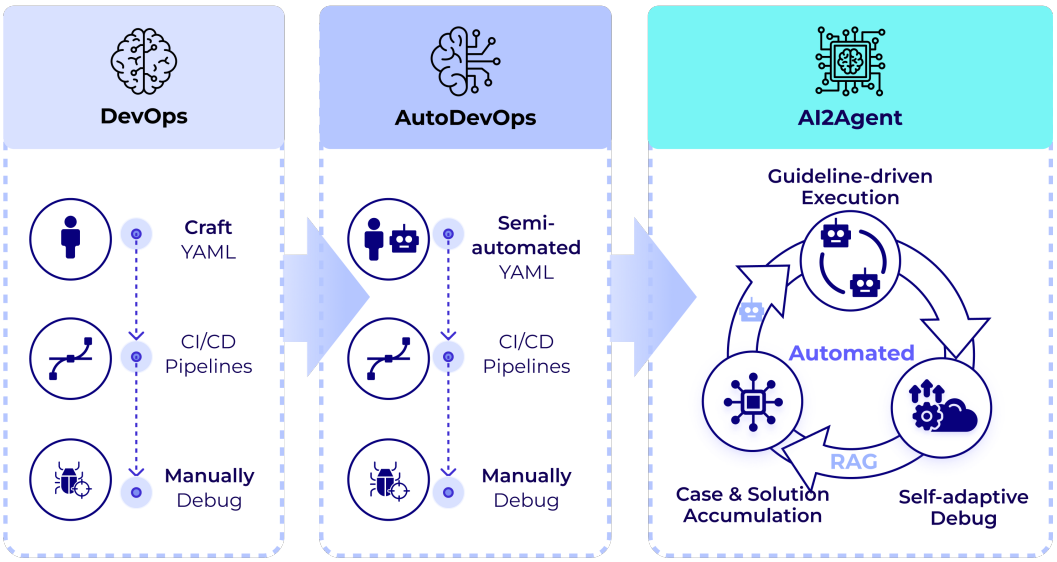

Figure 2: Comparison of Paradigms. DevOps relies on manual YAML configuration and CI/CD workflows with manual debug. AutoDevOps offers semi-automated configuration but still requires human intervention. AI2Agent achieves end-to-end automated performance, including guideline-driven execution, self-adaptive debug, and case & solution accumulation.

Step-by-Step Execution AI2Agent dynamically adjusts its execution flow in response to real-time environmental feedback. During deployment, it continuously monitors system parameters such as computational resources, dependency versions, and runtime errors. Based on this data, AI2Agent adjusts execution parameters to enhance performance and maintain system stability.

Environment-Aware Debug When deployment failures occur, AI2Agent automatically diagnoses issues by analyzing execution logs and system constraints. It identifies failure patterns, refines its execution strategy, and applies corrective actions to enhance robustness. If necessary, AI2Agent can perform an online search to gather additional information or solutions, further improving its problem-solving capabilities. This proactive approach reduces disruptions and ensures smoother execution.

Knowledge-Guided Refinement To improve debugging efficiency, AI2Agent queries its Knowledge Repository for relevant failure cases and proven solutions. By leveraging historical insights, AI2Agent accelerates problem resolution and refines debugging techniques, increasing deployment success rates while minimizing human intervention.

3.4 Case & Solution Accumulation

AI2Agent continuously refines its deployment strategies by accumulating experience. The Knowledge Repository serves as a structured database that records successful deployment cases, failure resolutions, and improvement strategies. This module enables two key processes:

Retrieval of Deployment Insights During deployment, AI2Agent retrieves relevant historical cases using Retrieval-Augmented Generation (RAG). These insights help in selecting optimal configurations, dependency management strategies, and execution plans tailored to the current task.

Continuous Learning and Refinement Every deployment contributes new insights to the repository. AI2Agent analyzes both successful and failed deployments, extracting lessons from error logs and corrective actions. These insights are integrated into future executions, ensuring a continuously evolving and increasingly autonomous deployment framework.

4 Case Study

To evaluate the effectiveness of AI2Agent in automating AI project deployments, we conducted a comprehensive study across 30 AI applications,

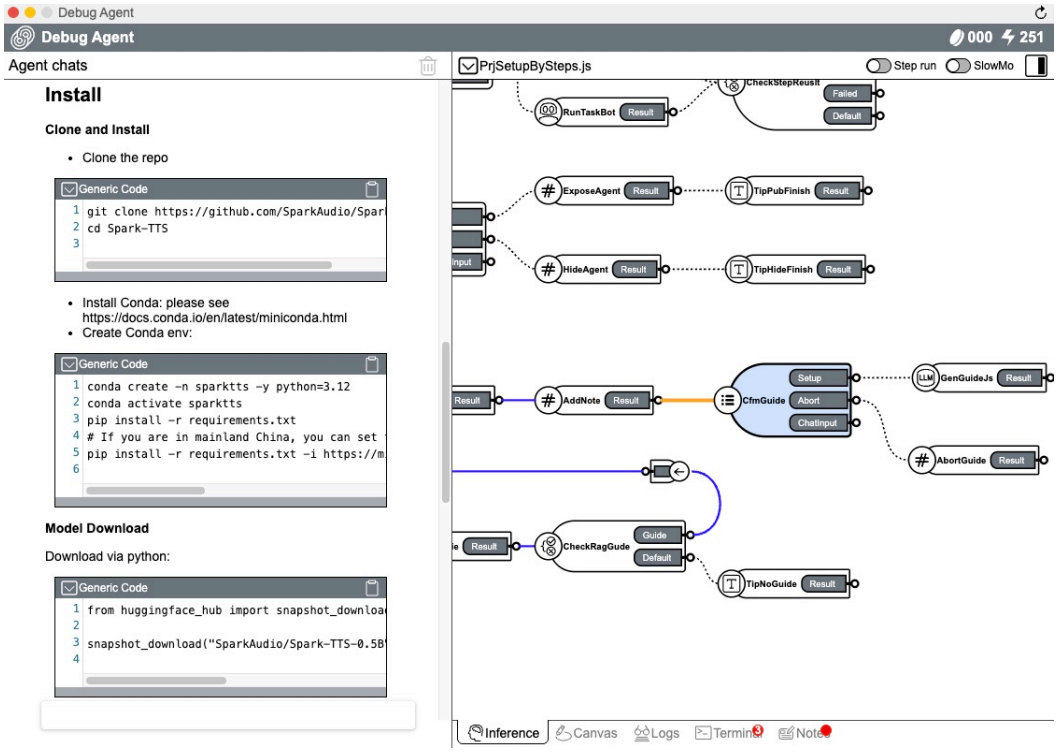

Figure 3: Screenshot of our local auto-deployment process. **Left:** Execution following predefined guidelines to ensure structured and reliable deployment. **Right:** The inference and planning interface for dynamically adapting to deployment conditions.

Algorithm 1 AI2Agent: Automated AI Deployment

Require: Repository R, Execution Environment $\mathcal{E}$ **return** Successfully deployed AI Agent $\mathcal{A}$

// Guideline-driven Execution
$G \leftarrow f_{\text{load}}(R)$
$S \leftarrow \emptyset$ // Store execution results
for each step $G_t \in G$ **do**
  // Execute step
  $S_t \leftarrow f_{\text{exec}}(\mathcal{E}, G_t)$
  $S \leftarrow S \cup \{S_t\}$
  // Self-adaptive Debug
  while $f_{\text{status}}(S_t) = 0$ **do**
    $G'_t \leftarrow f_{\text{search}}(S_t, R)$ //search solutions
    $S'_t \leftarrow f_{\text{exec}}(\mathcal{E}, G'_t)$
    $S \leftarrow S \cup \{S'_t\}$
    // Case & Solution Accumulation
    $R \leftarrow f_{\text{merge}}(G'_t, S'_t)$
  end while
end for
// Auto-Deployed AI Agent
$\mathcal{A} \leftarrow f_{\text{auto-deploy}}(S)$
return $\mathcal{A}$

including text-to-speech (TTS), text-to-image generation, and image editing. Figure 4 showcases a sample user interface from a successfully deployed project. This section further details details our deployment environment, evaluation setup, and a demo case of Spark-TTS.

4.1 Deployment Environment

We conducted all experiments using our self-developed **AI2Apps IDE (Pang et al., 2024)**, which is based on a web container framework. AI2Apps IDE is available in both web-based and locally deployed versions. To ensure security and full control over execution, all cases in this study were conducted using the locally deployed version. This approach guarantees that AI project installations and configurations are fully automated within a sandboxed environment, mitigating potential security risks associated with external dependencies and permissions.

4.2 Evaluation Setup

We assessed AI2Agent's performance across 30 AI applications spanning multiple domains, including text-to-speech (TTS), text-to-image generation, and image editing. To establish a fair com-

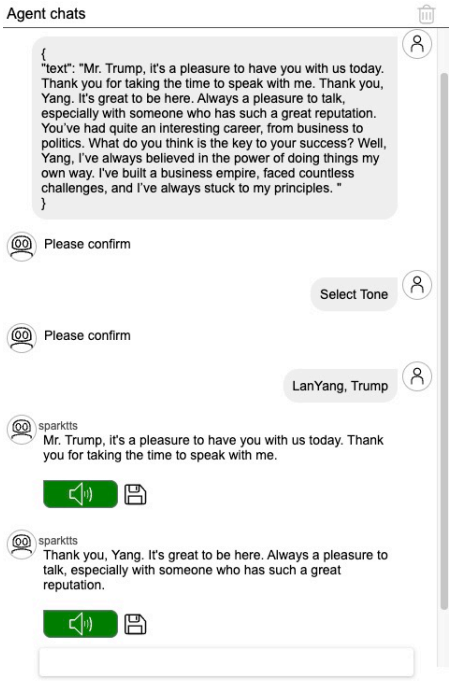

Figure 4: Screenshot of the user interface after auto-deployment.

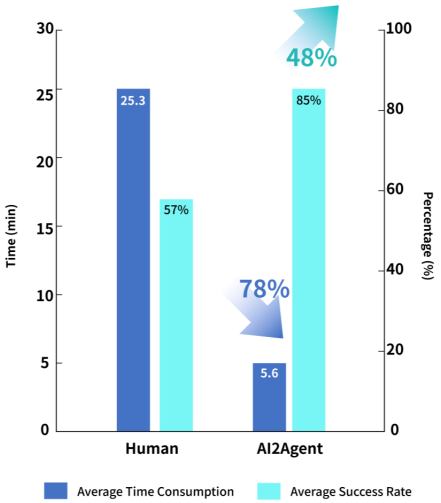

Figure 5: Comparison of manual vs. AI2Agent deployment: (1) **Time Consumption**: AI2Agent reduces deployment time by 78%. (2) **Success Rate**: AI2Agent improves success rate by 48%.

parison, we recruited participants with deep learning experience and provided them with guideline documents for manual deployment. Each participant independently attempted to deploy the applications without automation assistance.

To quantify AI2Agents effectiveness, we evaluated two key metrics: **Deployment Time Reduction**: The average time required for installation and configuration, excluding model download time, as it is influenced by network speed. **Success Rate Improvement**: The percentage of successful deployments completed without additional troubleshooting.

As shown in Figure 5, AI2Agent achieved a **78%** reduction in average deployment time and a **48%** increase in overall success rate, demonstrating its ability to streamline and enhance AI deployment efficiency.

4.3 Demonstration: Deploying Spark-TTS

To showcase AI2Agents automation capabilities, we present a case study on deploying **Spark-TTS**, a powerful text-to-speech (TTS) system. AI2Agent autonomously manages the entire setup process, including dependency management, model configuration, and environment preparation, ensuring a streamlined and error-free deployment. As illustrated in the user interface shown in Figure 4 and video, AI2Agent significantly reduces manual effort while maintaining a high success rate. The final speech output demonstrates excellent pronunciation accuracy, fluency, and overall quality, further validating the effectiveness of the framework.

5 Conclusion

Deploying AI projects is often hindered by complex environment configurations, dependency conflicts, and debugging challenges, limiting automation and scalability. While DevOps and AutoDevOps improve automation through YAML configurations and CI/CD pipelines, they still require manual intervention and lack adaptability in dynamic environments. This paper introduced AI2Agent, an end-to-end framework for automating AI deployment. By integrating guideline-driven execution, self-adaptive debugging, and case & solution accumulation, AI2Agent minimizes manual effort and dynamically adapts to deployment challenges. Over time, it learns from past deployments, enhancing efficiency and success rates. Experiments on 30 AI applications showed that AI2Agent reduces deployment time by 78% and increases success rates by 48%, demonstrating its potential to streamline AI deployment. This framework provides a scalable, automated solution for AI adoption across industries.

Limitations

AI2Agent enhances automation and adaptability in AI deployment, yet there is always room for further refinement. As AI projects grow increasingly complex, we look forward to exploring new ways to make deployments even more seamless and intelligent. We welcome researchers and practitioners to contribute to AI2Agents evolution, helping to expand its capabilities and applications.

Ethics Statement

(1)This work is the authors original research and has not been previously published elsewhere. (2)The paper is not under consideration for publication in any other venue. The research is conducted with integrity, ensuring truthful and complete reporting of methods, findings, and limitations. (3)AI2Agent does not involve the collection of personally identifiable information or sensitive user data. Any case study participants were volunteers who provided informed consent before participation, and all identifiers used in experiments were anonymized. (4)AI2Agent is designed to enhance the deployment of AI applications, promoting efficiency while ensuring responsible AI practices. (5)Our work does not involve training or fine-tuning large language models (LLMs); it strictly utilizes publicly available APIs permitted for research purposes, ensuring compliance with ethical and legal standards.

References

AutoDevops. 2019. Autodevops. https://docs.gitlab.com/topics/autodevops/.

Len Bass, Ingo Weber, and Liming Zhu. 2015. *DevOps: A software architect's perspective*. Addison-Wesley Professional.

Dhaya Sindhu Battina. 2019. An intelligent devops platform research and design based on machine learning. *training*, 6(3).

G Bou Ghantous. 2024. Enhancing devops using ai. In *CEUR Workshop Proceedings*.

Tim Brooks, Aleksander Holynski, and Alexei A Efros. 2023. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18392–18402.

Christof Ebert, Gorka Gallardo, Josune Hernantes, and Nicolas Serrano. 2016. Devops. *IEEE software*, 33(3):94–100.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Aliyu Enemosah. 2025. Enhancing devops efficiency through ai-driven predictive models for continuous integration and deployment pipelines. *International Journal of Research Publication and Reviews*, 6(1):871–887.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. Lightrag: Simple and fast retrieval-augmented generation.

Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, et al. 2023. Metagpt: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 3(4):6.

infiniflow. 2024. ragflow. https://github.com/infiniflow/ragflow.

Ioannis Karamitsos, Saeed Albarhami, and Charalampos Apostolopoulos. 2020. Applying devops practices of continuous automation for machine learning. *Information*, 11(7):363.

Openai. 2023. Explore gpts. https://chat.openai.com/gpts.

Xin Pang, Zhucong Li, Jiaxiang Chen, Yuan Cheng, Yinghui Xu, and Yuan Qi. 2024. Ai2apps: A visual ide for building llm-based ai agent applications. *arXiv preprint arXiv:2404.04902*.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. In *International conference on machine learning*, pages 8821–8831. Pmlr.

Xinsheng Wang, Mingqi Jiang, Ziyang Ma, Ziyu Zhang, Songxiang Liu, Linqin Li, Zheng Liang, Qixi Zheng, Rui Wang, Xiaoqin Feng, et al. 2025. Spark-tts: An efficient llm-based text-to-speech model with single-stream decoupled speech tokens. *arXiv preprint arXiv:2503.01710*.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*.

Hui Yang, Sifu Yue, and Yunzhong He. 2023. Auto-gpt for online decision making: Benchmarks and additional opinions. *arXiv preprint arXiv:2306.02224*.

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