Additional Discussion to Reviewer Kark's Questions

Q2: Comparison with Existing Agentic Frameworks

We compare OctoTools with the related works as suggested below, with the detailed comparison shown in Table 1.

1. Training-free agentic framework. Unlike methods such as TPTU-v2 (Kong et al., 2023) and TaskMatrix.AI (Liang et al., 2024) that require fine-tuning or specialized training to incorporate new tools, our OctoTools framework is training-free. It does not mandate any large corpus or additional supervised data to guide tool usage. This design allows seamless deployment and immediate extension to novel domains or tasks without extra training overhead.

2. Dynamic planning. OctoTools adaptively revises its plan on-the-fly in response to intermediate results or new context. By contrast, frameworks like TPTU-v2 (Kong et al., 2023) or HuggingGPT (Shen et al., 2023) typically rely on a one-shot plan and do not iterate further, potentially missing opportunities for deeper error handling or refinement as the task unfolds.

3. Self-refinement. OctoTools integrates a stepwise mechanism to re-evaluate and correct earlier reasoning steps. This self-refinement capability distinguishes it from approaches (e.g., GPT-Functions, AutoAgents (Chen et al., 2023)) that lack explicit iteration over internal reasoning mistakes. The iterative correction process helps OctoTools address emerging inconsistencies or overlooked details at each step.

4. Toolset optimization. OctoTools explicitly optimizes which subset of available tools is most beneficial for a given domain or task. In contrast, methods such as EcoAct (Zhang et al., 2024), HuggingGPT, and LangChain do not incorporate a specialized algorithm for automatically selecting the best-performing tools. By focusing on a task-relevant subset, OctoTools mitigates the noise and overhead of calling superfluous tools.

5. Extensible tools. With its standardized *tool cards*, OctoTools can seamlessly integrate a wide array of functionalities—ranging from Python calculations to vision models—without altering the system's core logic. This architecture stands apart from TPTU-v2 (Kong et al., 2023) or TaskMatrix.AI (Liang et al., 2024), where adding new tools often entails significant re-engineering or retraining. In OctoTools, developers can add, remove, or update tools with minimal effort.

6. Publicly available code and demo. In contrast to frameworks like EcoAct or AutoAgents—where no readily deployable system is provided—OctoTools offers a user-friendly codebase and a live demo (links to be shared in the camera-ready version). This opens up direct experimentation, reproducibility, and further development opportunities for the research community.

7. Comprehensive evaluations. We conducted rigorous tests across 16 diverse tasks spanning vision, math, science, and more. Other agentic methods typically restrict their evaluations to narrower or fewer domains—e.g., HuggingGPT (Shen et al., 2023), GPT-Functions, and AutoAgents (Chen et al., 2023) have been validated on only a handful (or none) of tasks. OctoTools' broad coverage attests to its strong generalization and reliability.

8. In-depth study. OctoTools includes ablation experiments and deeper analyses examining how step-by-step planning, tool usage, and multi-step reasoning each contribute to success. Many existing frameworks (e.g., Magnetic-One (Fourney et al., 2024), AutoAgents (Chen et al., 2023)) present limited investigations into their systems' internal workflows. In contrast, OctoTools provides detailed discussions of successes and failure modes, thereby offering comprehensive insights into agent design and optimization.

	Training-Free	Dynamic Planning	Self-Refinement	Toolset Optimization	Extensible Tools	Runnable System	Comprehensive Evaluations	In-depth Comparison
OctoTools (ours)	✓	✓	✓	✓	✓	✓	✓, 16 tasks	✓
AutoGen	✓	✓	✗	✗	✓	✓	✗, 0 task	✗
GPT-Functions	✓	✓	✗	✗	✓	Limited	✗, 0 task	✗
LangChain	✓	✓	✓	✗	✓	✓	✗, 0 task	✗
EcoAct (Zhang et al., 2024)	✓	✓	✓	✗	✗	✗	✗, 1 task only	Limited
TPTU-v2 (Kong et al., 2023)	✗	✗	✗	✓	Limited	✗	✗, 1 task only	Limited
HuggingGPT (Shen et al., 2023)	✓	✗	✗	✗	✗	✗	✗, 0 task	✗
TaskMatrix.AI (Liang et al., 2024)	✗	✗	✗	✗	✗	✗	✗, 0 task	✗
Magnetic-One (Fourney et al., 2024)	✓	✓	✗	✗	✓	✓	✗, 2 tasks only	✗
AutoAgents (Chen et al., 2023)	✗	✗	✗	✗	✗	✗	✗, 2 tasks only	✗

Table 1. Comparison with more existing works.

Notations:

- **Training-free:** The framework can be deployed or extended with new tools without any additional training or fine-tuning of the language model.
- **Dynamic planning:** The system adaptively updates or refines its plan (including tool usage) based on intermediate observations or feedback during the reasoning process.
- **Self-refinement:** At each step, the agent can verify and refine its previous reasoning to address errors, inconsistencies, or missing information in earlier steps.
- **Toolset optimization:** There is an explicit mechanism (e.g., a lightweight selection algorithm) that identifies the most useful subset of tools for a given domain or task, with the guarantee of the performance gain based on the validation.
- **Extensible tools:** A wide range of tools (e.g., Python, web-search, vision models) can be added via standardized interfaces ("tool cards"). Introducing a new tool does not require changes to the core planner–executor logic.
- **Runnable system:** A publicly accessible or easily deployable agentic framework is provided so that others can run, test, and build upon it for research or practical applications.
- **Comprehensive evaluations:** The framework is rigorously tested on diverse and challenging benchmarks (OctoTools demonstrates results on **16 tasks**), showcasing consistent gains and broad generalization.
- **In-depth comparison:** Thorough analyses and ablations are presented (e.g., on multi-step reasoning, task planning, tool usage) that offer insights into the system's capabilities, limitations, and design trade-offs, along with the behavior difference over other frameworks.

Q3: Clarification on Key Technical Contributions and Design Choices

1. Training-free yet high-gain approach. OctoTools breaks from the requirement of additional training or fine-tuning, enabling direct deployment and rapid integration into new domains. Despite its training-free nature, our system consistently outperforms baselines across **16 diverse benchmarks**, demonstrating that careful design and effective coordination of pre-existing tools can drive significant improvements without supervised data or retraining.

2. Flexible "tool cards" for extensibility. At the heart of OctoTools is a suite of "tool cards," each encapsulating a tool's metadata (e.g., inputs/outputs, usage constraints, and best practices). This design streamlines the addition, removal, or updating of tools with minimal effort—developers simply attach the tool's metadata without modifying the framework. Notably, no model re-training or fine-tuning is required when introducing a new tool, enabling fast, iterative development.

3. Planner–executor paradigm with dynamic self-refinement. We adopt a two-tiered approach that separates high-level reasoning from low-level command generation (Zhang et al., 2024). The planner formulates both an overall strategy (i.e., which tools to use and in what order) and adapts its plan dynamically at each step, ensuring real-time self-refinement. Meanwhile, the executor focuses on converting planner instructions into executable code or API calls, improving transparency, maintainability, and accuracy by keeping strategic decisions and technical implementations distinct.

4. Explicit toolset optimization. OctoTools introduces a validation-driven mechanism to choose the most relevant subset of tools for a given task. By pruning redundant or less helpful utilities, we prevent the system from incurring additional overhead or confusion. This not only boosts efficiency but also ensures that the agent leverages the most effective tools for each problem domain.

5. Full trajectory storage for analysis. Every intermediate step—planner decision, executor command, and tool output—is recorded as part of a structured trajectory. This comprehensive logging allows developers and researchers to trace the system's decision-making process in detail, facilitating debugging, auditing, and iterative refinement of both tool usage and the planner–executor pipeline.

6. Large-scale quantitative benchmarking and in-depth comparison. Previous agentic frameworks often limit their scope to small or specialized sets of tasks (Kong et al., 2023; Fourney et al., 2024), overlooking the need for extensive evaluation and thorough analysis. In contrast, we provide large-scale benchmarking to pinpoint exactly where OctoTools gains its performance edge, and we systematically examine why popular frameworks may fail on more complex tasks. Our ablation studies and deep dives into multi-step reasoning yield valuable insights for future tool-based agent research, making this a pivotal study in the agentic tools literature.

References

Chen, G., Dong, S., Shu, Y., Zhang, G., Sesay, J., Karlsson, B. F., Fu, J., and Shi, Y. Autoagents: A framework for automatic agent generation. *arXiv preprint arXiv:2309.17288*, 2023.

Fourney, A., Bansal, G., Mozannar, H., Tan, C., Salinas, E., Niedtner, F., Proebsting, G., Bassman, G., Gerrits, J., Alber, J., et al. Magentic-one: A generalist multi-agent system for solving complex tasks. *arXiv preprint arXiv:2411.04468*, 2024.

Kong, Y., Ruan, J., Chen, Y., Zhang, B., Bao, T., Shi, S., Du, G., Hu, X., Mao, H., Li, Z., et al. Tptu-v2: Boosting task planning and tool usage of large language model-based agents in real-world systems. *arXiv preprint arXiv:2311.11315*, 2023.

Liang, Y., Wu, C., Song, T., Wu, W., Xia, Y., Liu, Y., Ou, Y., Lu, S., Ji, L., Mao, S., et al. Taskmatrix. ai: Completing tasks by connecting foundation models with millions of apis. *Intelligent Computing*, 3:0063, 2024.

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

Zhang, S., Zhang, J., Ding, D., Garcia, M. H., Mallick, A., Madrigal, D., Xia, M., Rühle, V., Wu, Q., and Wang, C. Ecoact: Economic agent determines when to register what action. *arXiv preprint arXiv:2411.01643*, 2024.