DEBFLOW: AUTOMATING AGENT CREATION VIA AGENT DEBATE

Jinwei Su[1], *Yinghui Xia*[2], *Yiqun Duan, Jun Du, Jianuo Huang, Tianyu Shi*[3], *Lewei He*[1†]

[1] School of Artificial Intelligence, South China Normal University
[2] AutoAgents.ai, [3]University of Toronto

ABSTRACT

Large Language Model (LLM)-based agentic systems have demonstrated strong capabilities across various tasks with a static workflow designed with expert knowledge. Recently, researchers have attempted to automate the generation and optimization of these workflows using code-based representations. However, most existing approaches rely on single-model reasoning combined with coarse-grained feedback—e.g., binary success/iterative failure long raw-logs, making workflow construction unstable and computationally expensive. To address these limitations, we introduce *DebFlow*, a framework that employs *multi-agent debate* to achieve more reliable workflow optimization. In DebFlow, multiple agents propose, critique, and refine candidate workflows through iterative debate, guided by *fine-grained feedback* formalized on a directed-graph representation. By revisiting and refining pilot trajectories under this debate-driven process—analogous to selective pressures in natural evolution—DebFlow converges toward more efficient workflows. Extensive experiments on six benchmarks show that DebFlow consistently surpasses prior automated frameworks in both accuracy and inference efficiency.

Index Terms— LLMs, Multi-agent debate, Workflow, Reflection

1. INTRODUCTION

Large Language Models (LLMs) have demonstrated strong capabilities across domains such as code generation [1], data analysis [2], and decision-making [3]. To extend these abilities into complex multi-step tasks, prior work has relied on manually crafted agentic workflows that orchestrate LLM actions. However, such human-designed solutions require extensive domain expertise, limit adaptability to new problem settings, and hinder scalability. As history in machine learning suggests, hand-designed heuristics are ultimately replaced by automated, learnable frameworks.

Current research aims to develop automated frameworks for discovering efficient agentic workflows, thus minimizing human intervention. ADAS [4] defines the entire agentic system in code. However, the efficiency limitations of the linear heuristic search algorithm of ADAS hinder its ability to generate effective workflows within a constrained number of iterations. AFlow [5] models the workflow as a series of interconnected LLM-invoking nodes, where each node corresponds to an LLM action, and the edges capture the logical structure, dependencies, and execution flow between these actions. AFLOW employs the Monte Carlo Tree Search (MCTS) algorithm to automatically optimize LLM agent designs. In the search process, Monte Carlo Tree Search (MCTS) often performs numerous redundant optimizations, leading to significant computational overhead. This inefficiency increases the overall cost of the search, as it spends excessive resources on exploring suboptimal or irrelevant branches in the decision tree. Consequently, the algorithm's performance can be hindered by this unnecessary expenditure of computational effort, impacting its scalability and effectiveness in large or complex problem spaces. This underscores the need for more cost-effective and efficient methods to automate the generation of agentic workflows.

Furthermore, previous works on ADAS [4] and AFlow [5] primarily comprised three core components: search space, search algorithm, and evaluation. In terms of search algorithms, these approaches predominantly relied on generating workflows through a single large language model (LLM), which significantly constrains the performance to the capabilities of the individual model.

In this work, we introduce **DebFlow**, a multi-agent framework that leverages role-playing debate to automate the generation and refinement of LLM-based workflows. Unlike prior debate-based methods [6, 7] that directly target query reasoning, DebFlow adapts debate to the domain of *Automated Agentic Optimization*. Workflows are represented as directed graphs of reusable operators (e.g., Ensemble, Review-and-Revise), enabling compositionality. A novel fine-grained feedback mechanism analyzes workflow failures at the node and edge level, producing error-correcting signals that guide subsequent debates. This debate–feedback loop enables efficient exploration of workflow space while avoiding redundant search. Across six diverse benchmarks, DebFlow discovers agentic designs that consistently outperform prior automated frameworks and even surpass human-crafted baselines. The key contributions of this work are as follows:

[†] Corresponding Author: helewei@m.scnu.edu.cn

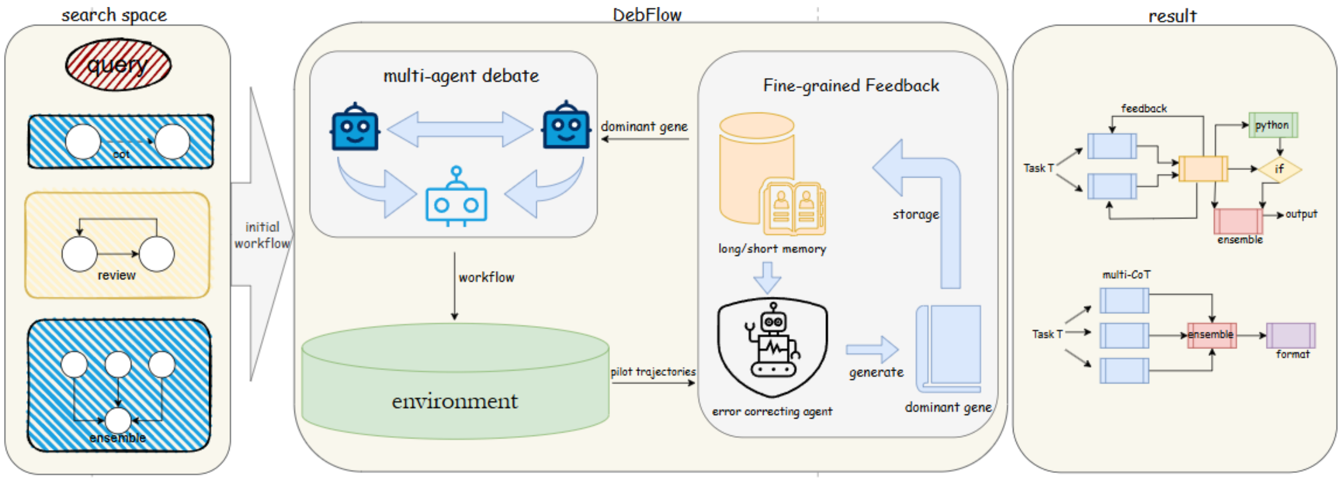

Fig. 1. The overall framework of **DebFlow.** The basic unit of framework invocation is the llm-invoking node, which can be combined to form different operators. The process starts with an initial workflow sampled from the search space. A multi-agent debate module generates candidate workflows, which interact with the environment and are evaluated. Fine-grained feedback analyzes failed workflows, stores key information, and guides subsequent debates by providing error-correcting signals. This iterative loop continues until the optimal workflow is obtained. Right: Case study and visualization. Tasks are from the MATH and HotpotQA benchmarks.

- We propose **DebFlow**, a framework that utilizes multi-agent debate to enhance the generation and optimization of workflows in large language model (LLM)-based agentic systems.
- **DebFlow** incorporates fine-grained feedback by revisiting pilot trajectories, enabling the system to extract informative signals that guide subsequent debates and workflow optimization.
- Experiments on six representative tasks demonstrate that **DebFlow** consistently discovers workflow variants that achieve superior performance compared to current baselines.

2. METHODOLOGY

In this section, we provide the technical details of **DebFlow**. The system overview is illustrated in Figure 1, where we utilize multi-agent debate and fine-grained feedback to facilitate automated workflow exploration.

2.1. Problem Formulation

We first define the basic unit of Debflow's search space, namely the agentic operator as follows:

Agentic Operator. An agentic operator $\mathcal{O}$ is a composite LLM-agent invocation process that involves multiple LLM calls:

$$\mathcal{O} = \{\mathcal{M}_i, \mathcal{P}_i \mathcal{T}_i\}, \quad \mathcal{M}_i \in \mathbb{M}, \quad \mathcal{P}_i \in \mathbb{P}, \quad \mathcal{T}_i \in \mathbb{T} \quad (1)$$

where $\mathcal{M}$ and $\mathbb{M}$ correspond to LLM backbones and the set of available LLMs, respectively. Similarly, $\mathcal{P}$ and $\mathcal{T}$ represent prompts and tools.

Workflow. We define an **agentic workflow** as a directed acyclic graph (DAG):

$$\mathcal{W} = \{\mathcal{O}, \mathcal{E}\}, \quad \mathcal{O} \subset \mathbb{O}, \quad \mathcal{E} \subset \mathcal{O} \times \mathcal{O}, \quad (2)$$

where $\mathcal{O}$ denotes the set of selected operators and $\mathcal{E}$ encodes their directed connectivity. The DAG structure enforces hierarchical operator execution.

With this formulation, the workflow optimization problem can be expressed as:

$$\mathcal{W}^* = \arg\max_{W \in S} G(W, T), \quad (3)$$

$\mathcal{W}^*$ is the optimal workflow configuration that maximizes the evaluation function G for the given task T. S is the search space.

2.2. Multi-agent Debate

Figure 1 illustrates the Multi-Agent Debate. We use the form of multi-agent debate to make the model more directional when generating workflow, avoiding the ”evolution” that failed in the past.

Given a task T, there are N **debaters**, denoted $D = \{D_i\}_{i=1}^N$. For each debater $D = \{D_i\}_{i=1}^N$ in the debate round r, a history record h_i will be maintained :

$$h_i = D_i(H_{r-1}, T, g, w) \quad (4)$$

where H represents the record of each history and g is the dominant gene, generated by fine-grained feedback. w is the workflow to be optimized. Update history: $H_r = H_{r-1} \cup \{h_1, h_2, \ldots, h_N\}$. To encourage thinking from more perspectives and improve the model's reasoning ability, we use a role-playing approach. In the multi-agent debate, we assign roles of the affirmative and the opposing sides. Similar to a debate competition, one side presents its argument, while the other side refutes it and offers its own viewpoint. Proponents propose a solution S_p:

$$S_p = f_p(\{h_i \mid D_i \in \text{Proponents}\}, H_r, T, g, w) \quad (5)$$

Opponents evaluate and propose a refined solution S_o:

$$S_o = f_o(S_p, \{h_i \mid D_i \in \text{Opponents}\}, H_r, T, g, w) \quad (6)$$

where f_p and f_o represent the synthesis processes of proponents and opponents, respectively.

To ensure that the arguments from both the Proponents and Opponents align with the task and goals, we introduce a **judge**. At the end of each debate round, the judge decides which side has the strongest argument. If both sides meet the requirements, the debate ends; otherwise, it continues.

$$(E_p, E_o) = J(S_p, S_o, H_r, T, g, P) \quad (7)$$

where E_p and E_o are evaluations of strengths and weaknesses. The judge determines:

$$W_r = \begin{cases} S_p & \text{if } J \text{ deems } S_p \text{ optimal} \\ S_o & \text{if } J \text{ deems } S_o \text{ optimal} \\ \emptyset & \text{if no solution is optimal} \end{cases} \quad (8)$$

If $W_r \neq \emptyset$, output W_r and terminate. Otherwise, proceed to round $r = r + 1$. If an optimal workflow is found:

$$W = W_r \quad (9)$$

If the maximum rounds R are reached:

$$W = J_{\text{final}}(\{S_p^{(r)}, S_o^{(r)} \mid r = 1, 2, \ldots, R\}, H_R, T, g) \quad (10)$$

where J_{final} selects the best solution based on historical outcomes.

Selection. The framework starts with an empty initial workflow, which follows an input-output (IO). As we continuously evaluate each workflow in the environment, every workflow receives a score. To optimize each workflow, we use multi-agent debate. To avoid getting stuck in local optima, we employ a soft mixed probability selection strategy to choose which workflow to optimize at each step. The formula for this selection strategy is as follows:

$$P_{\text{mixed}}(i) = \lambda \cdot \frac{1}{n} + (1-\lambda) \cdot \frac{\exp(\alpha \cdot (s_i - s_{\max}))}{\sum_{j=1}^{n} \exp(\alpha \cdot (s_j - s_{\max}))} \quad (11)$$

Where n is the number of candidate workflow, s_i is the score of workflow i, $s_{\max}$ is the maximum score, α controls the influence of scores, and λ balances uniform and weighted probabilities.

2.3. Fine-grained Feedback

Previous approaches predominantly utilized coarse-grained feedback, which often resulted in suboptimal optimization across each layer. To address this, we have introduced fine-grained feedback to enhance the precision and effectiveness of the optimization process.

The optimized workflow is executed within the environment, logging any failures for analysis. An error-correcting agent dissects the workflow to identify the failure-causing steps, generating an initial gene. This gene is refined using long/short memory, which stores past gene records, to produce a dominant gene. The dominant gene is then stored and logged in the failed workflow, providing a reference for the next multi-agent debate.

To formalize this process, let g_0 be the initial gene generated by the error-correcting agent, and let M represent the long / short memory of the past genes. The dominant gene g can be expressed as:

$$g = \mathcal{F}_{\text{LLM}}(g_0, M) \quad (12)$$

where $\mathcal{F}_{\text{LLM}}$ denotes the LLM-based gene refinement process that integrates g_0 with M to minimize failures.

3. EXPERIMENTS

3.1. Experiment Setup

Tasks and Benchmarks. We conduct experiments on six representative tasks covering four domains:(1)reading comprehension, HotpotQA [11], DROP [12]; (2)math reasoning, MATH [13]; (3)code generation, HumanEval [14] and MBPP [15]; (4)embodied, ALFWorld [16]. Following prior studies such as [4], we extracted 1,000 random samples each from the HotpotQA and DROP datasets. We also examined 617 problems from the MATH dataset, specifically choosing difficulty level 5 questions.

Baselines. We compare DebFlow with three series of agentic baselines:(1) single-agent execution methods, IO (direct LLM invocation), Chain-of-Thought [8], Self-Consistency (SC) [9], Self-Refine [10]; (2) Debate-optimized methods, MultiPersona [7], LLM-Debate [6]; (3) autonomous workflows, ADAS [4], AFlow [5].

LLM Backbones. In our experimental framework, we utilize GPT-4o-mini-0718 as the optimizer, with all models accessed through APIs. The temperature is set to 1. We configured 20 iteration rounds for AFLOW, 10 for DebFlow, and 30 for ADAS.

Evaluation Metrics. For quantitative assessment of model performance, we employ task-specific evaluation criteria across our experimental datasets. In mathematical reasoning tasks (GSM8K and MATHlv5 *), the accuracy of the solution is measured by the percentage metric of the solution rate. For programming proficiency evaluation (HumanEval and MBPP), we use the pass @1 metric, following

Method	**MATH**	**HotpotQA**	**HumanEval**	**MBPP**	**ALFWorld**	**DROP**	**Avg.**
GPT-4o-mini	47.8	68.1	87.0	71.8	38.7	68.3	63.6
CoT [8]	48.8	67.9	88.6	71.8	39.9	78.5	65.9
CoT SC [9]	47.9	68.9	88.6	73.6	40.5	78.8	66.4
Self Refine [10]	46.1	60.8	87.8	69.8	40.0	70.2	62.5
LLM-Debate [6]	48.5	66.4	88.7	70.3	44.6	75.2	65.6
MultiPersona [7]	50.8	69.2	88.3	73.1	39.1	74.4	68.8
ADAS [4]	43.1	64.5	82.4	53.4	47.7	76.6	61.3
AFlow [5]	53.8	73.5	90.9	81.4	59.2	80.3	73.2
DebFlow(Ours)	**56.5**	**75.4**	**92.0**	**82.6**	**62.3**	**82.7**	**75.3**

Table 1. Performance comparison of single-agent execution methods, Debate-optimized methods, and automated workflow optimization methods. Executed with GPT-4o-mini on divided test set, averaged over three trials.

Method	**MATH**	**HotpotQA**	**HumanEval**	**MBPP**	**DROP**	**Avg.**
AFlow$_{\text{gpt-4o-mini}}$	4.76	5.12	0.84	5.56	3.36	3.93
DebFlow$_{\text{gpt-4o-mini}}$	**4.23**	**3.34**	**0.61**	**1.78**	**1.24**	**2.24**
AFlow$_{\text{deepseek}}$	2.76	3.42	0.54	0.88	2.02	1.93
DebFlow$_{\text{deepseek}}$	**2.10**	**2.56**	**0.34**	**0.62**	**1.21**	**1.43**

Table 2. Training API costs. The baselines employ GPT-4o-mini as the optimizer, with subscripts indicating the model used as the executor.

Task	**w/o debate**	**w/o reflection**	**DebFlow**
Math	51.5	53.7	56.5
HotpotQA	71.7	72.8	75.4

Table 3. The ablation study of DebFlow.

the methodology established by [14]. The performance of the question answering (HotpotQA and DROP) is assessed through the calculation of the F1 score.

3.2. Experimental Results

Main Results. Table 1 shows that DebFlow achieves superior performance by outperforming all three categories of existing methods, including single-agent execution methods, Debate-optimized methods, and automated workflow optimization approaches. Specifically, On the embodied benchmark ALFWorld, DebFlow achieves the optimal 62.3%, outperforming the second-best AFLOW by 3.1%. On the MATH benchmark, it exceeds gpt-4o-mini by 8.7% and surpasses the SOTA baseline AFlow by 2.7%. Across six datasets in QA, Code, Embodied, and Math domains, Debflow surpasses all manually crafted workflows and demonstrates marginal improvements compared to automatically generated workflows.

Cost Analysis. DebFlow demonstrates a highly resource-efficient agentic automation system. Using GPT-4o-mini as the optimizer and testing with the GPT-4o-mini executors and DeepSeek (three trials averaged), Table 2 highlights DebFlow's superiority, reducing costs by 43% on average with GPT-4o-mini (e.g., 68% savings on MBPP: $1.78 vs. 5.56$ for AFlow) and 26% with DeepSeek. This efficiency underscores DebFlow's ability to deliver superior results in varied LLM environments without excessive resource demands, making it practical for real-world deployments.

Ablation Study. We perform an ablation study on two key components of DebFlow: (1) w/o Debate, removing the multi-agent debate mechanism and replacing it with a single LLM optimizer that generates candidate workflows randomly; and (2) w/o Reflection, removing the self-reflection module. We observe from Table 3 that removing the debate component causes the largest performance drop (-5.0% on MATH and -3.7% on HotpotQA), highlighting the importance of collaborative deliberation for reasoning. Removing the reflection module also degrades performance (-2.8% on MATH and -2.6% on HotpotQA), confirming its contribution to the overall effectiveness of DebFlow.

Case Study. As shown on the right of Figure 1, we present the workflows optimized by our framework on the MATH and HotpotQA benchmarks. HotpotQA tasks are often answerable directly from the questions, so optimal workflows are simple, requiring few operators. For the MATH dataset, we use level 5 difficulty questions. Simple workflows are often insufficient to solve these problems. Our framework can continuously optimize the workflows through multi-agent reasoning and fine-grained feedback, progressively adapting them to the tasks.

4. CONCLUSION

This study introduces DebFlow, a framework that uses multi-agent debate and fine-grained feedback to optimize workflows. Experiments show that DebFlow outperforms existing methods in performance and efficiency, while also reducing training resource consumption. In general, DebFlow provides an efficient, adaptable, and resource-friendly solution for automating agent creation.

5. REFERENCES

[1] Noah Shinn, Federico Cassano, Beck Labash, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao, "Reflexion: language agents with verbal reinforcement learning," in *Neural Information Processing Systems*, 2023.

[2] Sirui Hong, Yizhang Lin, Bangbang Liu, Binhao Wu, Danyang Li, Jiaqi Chen, Jiayi Zhang, Jinlin Wang, Lingyao Zhang, Mingchen Zhuge, Taicheng Guo, Tuo Zhou, Wei Tao, Wenyi Wang, Xiangru Tang, Xiangtao Lu, Xinbing Liang, Yaying Fei, Yuheng Cheng, Zhibin Gou, Zongze Xu, Chenglin Wu, Li Zhang, Min Yang, and Xiawu Zheng, "Data interpreter: An llm agent for data science," *arXiv preprint*, 2024.

[3] Chan Hee Song, Brian M. Sadler, Jiaman Wu, Wei-Lun Chao, Clayton Washington, and Yu Su, "Llm-planner: Few-shot grounded planning for embodied agents with large language models," in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 2986–2997.

[4] Shengran Hu, Cong Lu, and Jeff Clune, "Automated design of agentic systems," *ArXiv*, vol. abs/2408.08435, 2024.

[5] Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bangbang Liu, Yuyu Luo, and Chenglin Wu, "Aflow: Automating agentic workflow generation," *ArXiv*, vol. abs/2410.10762, 2024.

[6] Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch, "Improving factuality and reasoning in language models through multiagent debate," *ArXiv*, vol. abs/2305.14325, 2023.

[7] Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji, "Unleashing the emergent cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration," in *North American Chapter of the Association for Computational Linguistics*, 2023.

[8] 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*, vol. 35, pp. 24824–24837, 2022.

[9] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou, "Self-consistency improves chain of thought reasoning in language models," *arXiv preprint arXiv:2203.11171*, 2022.

[10] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Sean Welleck, Bodhisattwa Prasad Majumder, Shashank Gupta, Amir Yazdanbakhsh, and Peter Clark, "Self-refine: Iterative refinement with self-feedback," *ArXiv*, vol. abs/2303.17651, 2023.

[11] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning, "HotpotQA: A dataset for diverse, explainable multi-hop question answering," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, Eds., Brussels, Belgium, Oct.-Nov. 2018, pp. 2369–2380, Association for Computational Linguistics.

[12] Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner, "DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs," 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)*, Jill Burstein, Christy Doran, and Thamar Solorio, Eds., Minneapolis, Minnesota, June 2019, pp. 2368–2378, Association for Computational Linguistics.

[13] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Xiaodong Song, and Jacob Steinhardt, "Measuring mathematical problem solving with the math dataset," *ArXiv*, vol. abs/2103.03874, 2021.

[14] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé, Jared Kaplan, Harrison Edwards, Yura Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mo Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, David W. Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William H. Guss, Alex Nichol, Igor Babuschkin, Suchir Balaji, Shantanu Jain, Andrew Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew M. Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba, "Evaluating large language models trained on code," *ArXiv*, vol. abs/2107.03374, 2021.

[15] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie J. Cai, Michael Terry, Quoc V. Le, and Charles Sutton, "Program synthesis with large language models," *ArXiv*, vol. abs/2108.07732, 2021.

[16] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht, "Alfworld: Aligning text and embodied environments for interactive learning," *ArXiv*, vol. abs/2010.03768, 2020.