ABSTRACT

A popular approach to building agents using Language Models (LMs) involves iteratively prompting the LM, reflecting on its outputs, and updating the input prompts until the desired task is achieved. However, our analysis reveals two key shortcomings in the existing methods: (i) limited exploration of the decision space due to repetitive reflections, which result in redundant inputs, and (ii) an inability to leverage insights from previously solved tasks. To address these issues, we introduce `DoT`[1] (Diversity of Thoughts), a novel framework that a) explicitly reduces redundant reflections to enhance decision-space exploration, and b) incorporates a task-agnostic memory component to enable knowledge retrieval from previously solved tasks—unlike current approaches that operate in isolation for each task. Through extensive experiments on a suite of programming benchmarks (HumanEval, MBPP, and LeetCodeHardGym) using a variety of LMs, `DoT` demonstrates up to a **10%** improvement in Pass@1 while maintaining cost-effectiveness. Furthermore, `DoT` is modular by design. For instance, when the diverse reflection module of `DoT` is integrated with existing methods like Tree of Thoughts (ToT), we observe a significant **13%** improvement on Game of 24 (one of the main benchmarks of ToT), highlighting the broad applicability and impact of our contributions across various reasoning tasks.

1 INTRODUCTION

Developing autonomous AI-based frameworks leveraging Large Language Models (LLMs) to solve challenging reasoning and decision-making tasks is an active area of research. Recent research works have focused on utilizing different LLMs as the backbone to develop AI agents that can understand, reason, and reflect on their own actions when solving a task. An AI agent typically performs iterative inference to effectively reason and solve a task by leveraging LLMs and knowledge of tools if available. Examples of such agentic architectures are ReAct (Yao et al., 2023b), Tree of Thoughts (ToT) (Yao et al., 2023a), and Toolformer (Schick et al., 2023), which have explored the in-context learning abilities of LLMs, proposing iterative inference-level algorithms to mimic human-like learning. Such agent architectures promote generalistic approaches to reasoning and learning, which differs from prior reinforcement learning-based solutions introduced by Mnih et al. (2013); Le et al. (2022) that were tailored towards a specific task and demanded extensive compute and data for training.

Building upon ReAct, Reflexion (Shinn et al., 2023) improved the reasoning capabilities in the agent by introducing "self-reflections", a mechanism to steer the model away from previous failed trajectories. Similarly, LATS (Zhou et al., 2023) extended these ideas by enabling multiple reasoning paths and employing search algorithms like Monte Carlo Tree Search (MCTS) to identify optimal solutions. These methods represent significant strides in enhancing LLMs' reasoning capabilities through an iterative feedback.

Table 1: Self-reflection counts and output token usage on a HumanEval subset, highlighting redundancy across problems. "Ref" denotes reflections.

Problem Name	Total Ref	Unique Ref	#Tokens
145_order_by_points	36	4	5129
130_tri	111	8	14,484
129_minPath	84	10	12,227
132_is_nested	36	9	4292
84_solve	96	13	10,329

Table 2: Pass@1 and cost comparison for OpenAI o1 on the LeetCodeHardGym benchmark. DoT outperforms Reflexion.

Method	Pass@1	Cost
Base	45	$12.75
Reflexion	62	$24.24
DoT	**72.5**	$33.47

Despite these advancements, current frameworks suffer from few critical limitations: *(i) poor exploration of the decision space, primarily due to repetitive reflections, and (ii) an inadequate memory mechanism*. To depict these shortcomings, we revisited the self-reflections generated by Reflexion and LATS. Specifically, we analyzed the generated self-reflections by these two approaches for the HumanEval dataset (Chen et al., 2021b). Upon careful inspection, we observed that several self-reflections were repetitive. We employed an LLM (GPT-4o) to identify clusters of similar self-reflections and we manually inspected a random sample (see details of prompt used in Appendix A.3). Our analysis revealed that a significant portion of the self-reflections generated by LATS were repetitive and redundant, leading to poor decision-space exploration and excessive token usage. We present key statistics for a representative subset of the HumanEval dataset in Table 1. Similar patterns were observed across other datasets.

Reflexion and LATS incorporate a *local* memory component to store reflections and failed implementations, using it as additional context to improve subsequent generations. However, this memory is reset before tackling the next task. In contrast, Didolkar et al. (2024) demonstrated that recent LLMs can leverage metacognitive abilities to apply prior insights to new tasks, achieving significant performance gains. Their method involves curating in-context examples based on skill exemplars—insights from related tasks—but relies on a predefined set of exemplars, which necessitates additional training data and limits adaptability across diverse tasks.

To address these limitations, we propose Diversity of Thoughts (DoT), a novel framework that enhances decision space exploration by (a) reducing redundant reflections and (b) integrating a task-agnostic memory component. DoT promotes diverse reasoning attempts using a variety of thought/step sampling strategies, ensuring effective intermediate reflections to guide decision-making. The task-agnostic memory bank dynamically retrieves relevant in-context examples from previously solved tasks, boosting the model's ability to generate informed reasoning paths.

This approach not only reduces redundant computations but also significantly improves efficiency, resulting in a more cost-effective solution. For example, on the LeetCodeHardGym benchmark, DoT achieved a 10% gain in Pass@1 (Claude Sonnet 3.5) with a 4× cost reduction compared to LATS (see Section 3.2.1 and Table 6). Additionally, DoT demonstrated substantial improvements on OpenAI o1[2] in a preliminary study, outperforming both the base model and Reflexion (see Table 2).

In summary, our main contributions in this paper are mentioned below:

- We identify redundancy as a critical limitation in the existing reflection-based reasoning frameworks (e.g., Reflexion, LATS), as shown anecdotally in Figure 1 and quantitatively in Table 1. Additionally, current frameworks handle each task in isolation, missing the opportunity for cross-task knowledge transfer.
- We propose (i) DoT, a novel framework that promotes diversity in reasoning trajectories, and (ii) DoT-bank, which further extends DoT by incorporating similar trajectories from a task-agnostic memory bank to enhance decision space exploration.
- Our extensive experiments across multiple code generation datasets (HumanEval, MBPP, LeetCodeHardGym) and models demonstrate the effectiveness of DoT. We achieve state-of-the-art results, with up to a 10% improvement in Pass@1 on LeetCodeHardGym

[2] https://openai.com/index/introducing-openai-o1-preview/

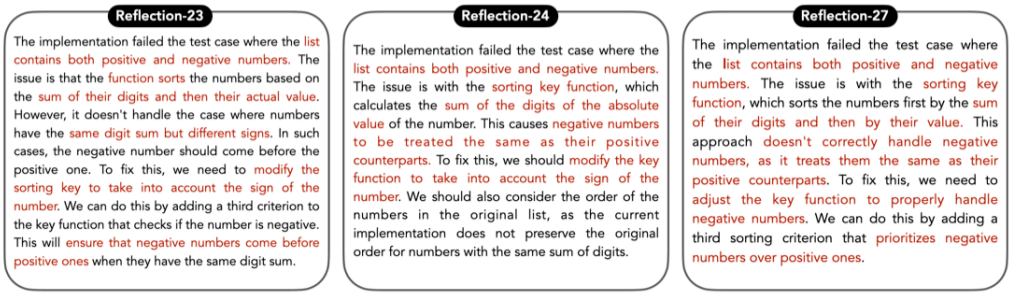

Figure 1: Examples of repetitions in generated self-reflections from LATS on problem `145_order_by_points` in the HumanEval dataset. The words highlighted in red are redundant across Reflection 23, 24 and 27.

for Claude Sonnet 3.5, while reducing token costs by 4× compared to LATS and 1.4× more expensive than Reflexion.

2 METHODOLOGY

Problem Setup. We briefly review the standard LLM-based reasoning and decision making setup. We are provided with an input x in natural language, along with a pretrained LLM $p_\theta(\cdot)$ parameterized by frozen learned parameters θ. The goal is to generate an output response y that corresponds to a solution for reasoning tasks or a set of actions for decision making. Traditional input-output prompting ($y \sim p_\theta(x)$) leads to sub-optimal performance. Following the observations made by Brown et al. (2020), a flurry of prompting techniques were proposed, which prepend additional input text with specific instructions or few-shot input-output examples to input query x. Incorporating such additional context helps to improve reasoning and decision making performance. $\texttt{prompt}_{IO}(x)$ denotes a generic stage in the process of transforming the given input prompt x into output y as: $y \sim p_\theta(\texttt{prompt}_{IO}(x))$. An illustration of this process is shown in Figure 2.

```
System Message:
    You are an AI that only responds with python code, NOT ENGLISH. You
↪   will be given a function signature and its docstring by the user.
↪   Write your full implementation (restate the function signature).

User Message:
def cube_Sum(n: int) -> int:
    """
    Write a python function to find the cube sum of first n even
    ↪  natural numbers.
    """
```

Figure 2: An example of $\texttt{prompt}_{IO}(x)$ on programming tasks. The User Message denotes x in the above example, and the $\texttt{prompt}_{IO}(x)$ prepended a System Message specific to a task.

Preliminaries. Our framework's design is inspired by Reflexion (Shinn et al., 2023), which enhances reasoning through iterative interaction with an external API or environment. Reflexion involves three agents—actor (M_a), evaluator (M_e), and self-reflection (M_{sr})—working cyclically until a termination condition is met. For code generation, the process is as follows:

(a) The actor M_a receives input and generates an output (e.g., a code snippet).
(b) The evaluator M_e scores this output (e.g., the number of unit tests passed).
(c) If the score is low (i.e., code fails), the self-reflection agent M_{sr} diagnoses the issue, suggests a fix, and adds it to the input context along with the failed action. This new input is given to the actor, and the cycle repeats until success or a trial limit is reached.

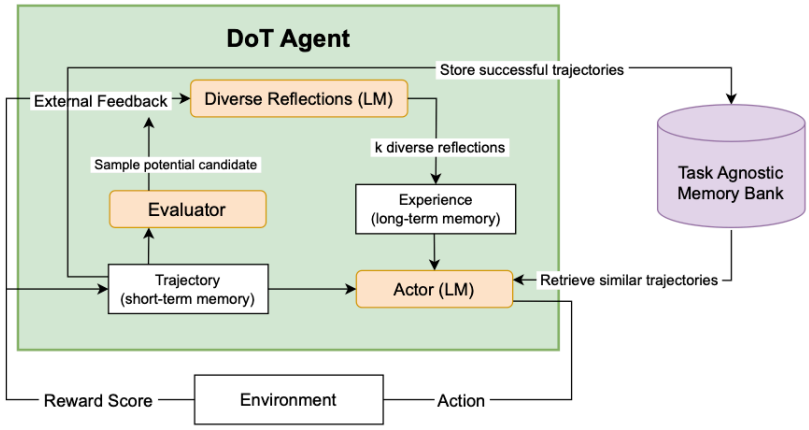

Figure 3: Overview of DoT and DoT-bank's architecture. We introduce a new diverse-reflections model to mitigate redundancy in reflections, and a task agnostic memory bank to enable knowledge transfer across tasks.

As noted in Table 1, in several cases, the generated self-reflections are repetitive and redundant, thereby hindering the effective exploration of the decision space.

2.1 DoT: Proposed Framework

DoT builds upon Reflexion by replacing the self-reflection model with a novel diverse-reflections model, M_{dr}. We extend this further with DoT-bank, which introduces a task-agnostic memory bank, MB. Thus, DoT comprises three models: the actor M_a and evaluator M_e (similar to Reflexion) and the diverse-reflections model M_{dr}. In DoT-bank, MB is additionally leveraged. We provide a detailed description of each component and the workflow that illustrates how these pieces are integrated. An illustration is provided in Figure 3.

Actor ($\mathbf{M_a}$): The actor is powered by an LLM of choice with specific instructions to generate actions conditioned on the state observations; these actions could be snippets of text, or other actions as per the setting. For example when the task is code generation, the input to the actor is a task description, associated history etc. as a text string, and the output action is a code snippet.

Evaluator ($\mathbf{M_e}$): Computes a reward score for the action generated by the actor. Depending on the task, M_e can be an external environment (e.g., Python interpreter), a Likert scale rating (Yao et al., 2023a; Zhou et al., 2023), or another LLM (Shinn et al., 2023; Zhou et al., 2023). For example, in the setting of code generation, the reward can be measured by the number of visible or synthetic unit tests passed.

Diverse-Reflections ($\mathbf{M_{dr}}$): To address redundancy, we propose the *Diverse-Reflections* module, which generates k diverse reflections in one shot using an explicit prompt: $z_i = p_\theta(z_{1\ldots k} \mid \text{DivIO}(x))$. The exact prompt structure ($\text{DivIO}(x)$) is detailed in Figure 5 (see Appendix A.4). Reflections from previous iterations are also included in the context. We investigate alternative approaches for generating k diverse reflections in Section 3.3.3 and find one-shot sampling to be the most effective in terms of both performance and cost. Recent work (Hayati et al., 2024) supports that one-shot sampling can produce semantically diverse outputs. Additionally, we introduce a diversity metric in Section 3.3.1 to quantitatively demonstrate how our $\mathbf{M_{dr}}$ module effectively reduces redundancy in generated reflections.

Task-Agnostic Memory Bank (MB): Our second contribution is a persistent, task-agnostic memory MB that stores successful trajectories of the DoT agent. A trajectory is deemed successful if it passes all visible or synthetic tests. Each entry in MB is indexed by a unique task-id and consists of a complete task trajectory. When the actor M_a is invoked, we retrieve j relevant trajectories from MB and include them in its input context. As more tasks are solved, MB grows, facilitating knowledge transfer between tasks. We evaluate our framework in two configurations—DoT (without MB) and DoT-bank (with MB)—and analyze the impact of the number of retrieved examples on per-

Algorithm 1 `DoT` and `DoT-bank` Framework

Require: Dataset $\mathcal{D}$, Retriever, Max Trials T_{max}
Initialize: Actor M_a, Evaluator M_e, Diverse Reflections Generator M_{dsr},
Memory Bank $MB = \emptyset$, Failed Tasks $F = \emptyset$

Phase 1: `DoT`: solving and constructing memory bank
for each task $t \in \mathcal{D}$ **do**
    Generate τ using M_a and evaluate using M_e
    if τ passes evaluation **then**
        Add τ to MB
    else
        set i = 0
        while t not solved or $i < T_{max}$ **do**
            Generate k diverse reflections using M_{dsr}
            Generate k new trajectories using by feeding these reflections and τ to M_a
            for each generated trajectory τ_i **do**
                Evaluate τ_i using M_e and record score
                if τ_i passes evaluation **then**
                    Add τ_i to MB, **break the while loop**
                end if
            end for
            if none of the k trajectories passed **then**
                Select a trajectory at random (weighted by the corresponding score)
            end if
            Increment i
        end while
        if no trajectory passes after T_{max} trials **then**
            Add t to F
        end if
    end if
end for

Phase 2: `DoT-bank`: reattempting failed tasks using memory bank
for each failed task $t \in F$ **do**
    Retrieve J similar trajectories from MB
    $M_{a'} \rightarrow$ Inject retrieved trajectories into context of actor
    Repeat Phase 1 with $M_{a'}$ and update MB if successful
end for

formance in Section 3.3.3. The orchestration of these agents is presented as `DoT` framework in Algorithm 1. Additional implementation details and a discussion on MB are provided in Appendix A.1.

2.2 Relation to Other Methods

Tree-of-Thought (ToT) (Yao et al., 2023a) extends Chain-of-Thought (CoT) by exploring multiple reasoning paths over intermediate states (also referred to as "thoughts"). It constructs a tree, where each node contains $[x, z_{1...i}]$ and represents a partial solution. Thoughts can either be sampled or proposed using CoT: $z_i \sim p_\theta^{CoT}(x, z_{1...i-1})$. To select the final solution, classic search tree algorithms such as BFS/DFS are employed, combined with a language model-based evaluator that assigns a value to each node.

LATS (Zhou et al., 2023) unifies reasoning, acting, and planning under one framework. It replaces the cyclic workflow in Reflexion with a Monte Carlo Tree Search (MCTS) for *proficient exploration* of the decision space. During tree exploration, each node is assigned a value, which is a convex combination of LM based evaluation and self-consistency. A noteworthy detail is that current action is agnostic at a given level of the search tree to previous actions: $a_t^{(i)} \sim p_\theta(s_t)$. We refer the reader to Algorithm 1 in (Zhou et al., 2023) for additional details.

3 EXPERIMENTS

4 Related Work

We categorize the literature on LLM reasoning capabilities into three main themes: (i) enhancing step-by-step reasoning abilities, (ii) promoting diversity in the reasoning process, and (iii) incorporating memory mechanisms and external memory to facilitate learning from past experiences.

Reasoning in LLMs. Since LLMs were identified as few-shot learners (Brown et al., 2020), several prompting techniques and inference strategies have been proposed to improve their reasoning abilities. Chain of Thought (CoT)(Wei et al., 2022) introduced step-by-step reasoning to enhance problem-solving. Self-Consistency(Wang et al., 2023) extended CoT by leveraging multiple independent CoT runs for improved outcomes. Tree of Thoughts (ToT)(Yao et al., 2023a) employed search algorithms like BFS/DFS with LLM-guided heuristics to further boost performance. ReAct (Yao et al., 2023b) integrates reasoning and action steps, while Reflexion (Shinn et al., 2023) extends this approach by incorporating a "self-reflection" component, building upon the Self-Refine framework (Madaan et al., 2023). More recently, LATS (Zhou et al., 2023) integrated Monte Carlo Tree Search (MCTS) into Reflexion, achieving gains at the expense of higher token usage and costs. However, our analysis revealed that Reflexion and LATS generate redundant self-reflections thereby hindering effective decision space exploration.

Diversity in Reasoning. Inducing diversity in LLM reasoning has been explored through methods like DIV-se (Naik et al., 2024), which uses varied prompts and personas (e.g., "Think like Alan Turing" or "Think like a Math Professor"). However, it requires manual persona selection, limiting flexibility. Flow of Reasoning (FoR)(Yu et al., 2024) uses GFlowNet to train LLMs to generate diverse reasoning without predefined personas but involves task-specific fine-tuning. In contrast, our framework, `DoT`, promotes diversity through structured prompts and one-shot sampling, leveraging a task-agnostic memory bank without personas, manual interventions, or task-specific training. This enables `DoT` to explore multiple decision branches and adapt across domains seamlessly.

LLM Memory. Memory mechanisms enable LLMs to retain and learn from past experiences, enhancing their ability to reason and adapt. MemoryBank (Zhong et al., 2024) enables LLMs to update and retrieve past interactions to align with user intent. MemoChat (Lu et al., 2023) trains LLMs to efficiently retrieve relevant dialogue history. Reflexion (Shinn et al., 2023) leverages memory in a reinforcement learning manner, with past action described in verbal format. CLIN Majumder et al. (2024) stores causal abstractions in an evolving memory. In contrast, our task agnostic memory bank stores entire agent trajectories retrieving them as in-context examples, enriching decision-making and reasoning.

External Memory for LLMs. External memory methods like Buffer of Thoughts (BoT) (Yang et al., 2024) introduce high-level "thought templates" to assist reasoning, while Needle in a Haystack Chaudhury et al. (2024) demonstrates the utility of external memory for tasks requiring long context lengths. Our approach differs by storing complete task trajectories in a dynamic, task-agnostic memory bank, injecting relevant examples to improve reasoning. The memory-bank grows as the agent solves more tasks.

5 Limitations & Conclusion

`DoT` and `DoT-bank` significantly reduce costs compared to LATS but remain $1.4\times$ more expensive than Reflexion and up to $8\times$ higher than the base LLM. Iterative reasoning frameworks also suffer from increased latency due to repeated interactions with external environments (e.g., tools, APIs), limiting scalability. While our one-shot sampling mitigates redundancies in generated reflections, it is not entirely foolproof, leaving room for future research on more efficient strategies to generate diverse reflections. This work addresses key limitations in self-reflection-based reasoning frameworks, namely redundant reflections and missed opportunities for cross-task knowledge transfer. Although focused on programming tasks, extending these methods to other reasoning domains presents an exciting avenue for future exploration.

REFERENCES

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Subhajit Chaudhury, Soham Dan, Payel Das, Georgios Kollias, and Elliot Nelson. Needle in the haystack for memory based large language models. *arXiv preprint arXiv:2407.01437*, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri 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, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew 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. 2021a.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021b.

Aniket Rajiv Didolkar, Anirudh Goyal, Nan Rosemary Ke, Siyuan Guo, Michal Valko, Timothy P Lillicrap, Danilo Jimenez Rezende, Yoshua Bengio, Michael Curtis Mozer, and Sanjeev Arora. Metacognitive capabilities of LLMs: An exploration in mathematical problem solving. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=D19UyP4HYk.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, and Akhil Mathur et. al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Shirley Anugrah Hayati, Minhwa Lee, Dheeraj Rajagopal, and Dongyeop Kang. How far can we extract diverse perspectives from large language models? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Hoi. CodeRL: Mastering code generation through pretrained models and deep reinforcement learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=WaGvb7OzySA.

Junru Lu, Siyu An, Mingbao Lin, Gabriele Pergola, Yulan He, Di Yin, Xing Sun, and Yunsheng Wu. Memochat: Tuning llms to use memos for consistent long-range open-domain conversation. *arXiv preprint arXiv:2308.08239*, 2023.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=S37hOerQLB.

Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Peter Jansen, Oyvind Tafjord, Niket Tandon, Li Zhang, Chris Callison-Burch, and Peter Clark. CLIN: A continually learning language agent for rapid task adaptation and generalization. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=xS6zx1aBI9.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013. URL https://arxiv.org/abs/1312.5602.

Ranjita Naik, Varun Chandrasekaran, Mert Yuksekgonul, Hamid Palangi, and Besmira Nushi. DIVERSITY OF THOUGHT IMPROVES REASONING ABILITIES OF LARGE LANGUAGE MODELS, 2024. URL https://openreview.net/forum?id=FvfhHucpLd.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 68539–68551. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/d842425e4bf79ba039352da0f658a906-Paper-Conference.pdf.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1b44b878bb782e6954cd888628510e90-Paper-Conference.pdf.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E. Gonzalez, and Bin CUI. Buffer of thoughts: Thought-augmented reasoning with large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=ANO1i9JPtb.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 11809–11822. Curran Associates, Inc., 2023a. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/271db9922b8d1f4dd7aaef84ed5ac703-Paper-Conference.pdf.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=WE_vluYUL-X.

Fangxu Yu, Lai Jiang, Haoqiang Kang, Shibo Hao, and Lianhui Qin. Flow of reasoning: Efficient training of llm policy with divergent thinking. *arXiv preprint arXiv:2406.05673*, 2024.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19724–19731, 2024.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning acting and planning in language models, 2023.