INVESTIGATING LATENT SPACE RECURRENT REASONING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Transformers have demonstrated remarkable performance in natural language processing and related domains, as they largely focus on sequential, autoregressive next-token prediction tasks. Yet, they struggle in logical reasoning, not necessarily because of a fundamental limitation of these models, but possibly due to the lack of exploration of more creative uses, such as latent space and recurrent reasoning. An emerging exploration in this direction is the Hierarchical Reasoning Model Wang et al. (2025), which introduces a novel type of recurrent reasoning in the latent space of transformers, achieving remarkable performance on a wide range of 2D reasoning tasks. Despite the promising results, this line of models is still at an early stage and calls for in-depth investigation. In this work, we perform a critical review on this class of models, examine key design choices and demonstrate interesting variants that significantly outperform what is reported on the Sudoku-Extreme and Maze-Hard tasks. Our results also raise surprising observations and intriguing directions for further research.

1 INTRODUCTION

The Transformer architecture Vaswani et al. (2017) has become the foundation of modern large language models (LLMs), powering systems such as BERT Devlin et al. (2019), PaLM Chowdhery et al. (2022) and GPT Brown et al. (2020); Achiam et al. (2023). Scaling these models has led to impressive progress across natural language understanding and generation tasks. Nevertheless, benchmarks specifically designed to probe reasoning—including GSM8K for arithmetic reasoning Cobbe et al. (2021), BIG-Bench for diverse challenging tasks Srivastava et al. (2022) and MATH for competition-level mathematics Hendrycks et al. (2021)—have revealed limitations in systematic multi-step reasoning, even for the largest models.

To address these gaps, recent work has proposed methods to elicit reasoning through prompting or architectural modifications. Chain-of-thought prompting Wei et al. (2022) and scratchpad supervision Nye et al. (2021) train or prompt models to generate explicit intermediate steps, while program-aided approaches such as PAL Gao et al. (2023) offload computation to external interpreters. At the same time, theoretical studies highlight structural limits of attention-only architectures: for example, Abbe et al. (2024) demonstrate a "locality barrier" that constrains global reasoning unless additional memory mechanisms are introduced. These insights echo findings from cognitive science, which emphasizes the hierarchical and compositional nature of human reasoning Dedhe et al. (2023). Together, this body of work motivates the exploration of architectures and training paradigms that go beyond standard token-level generation, such as latent-space reasoning and recurrent refinement, which we examine in this paper, with an emphasis on the latest popular embodiment of these two ideas — Hierarchical Reasoning Model (HRM) Wang et al. (2025).

2 LATENT SPACE REASONING AND HIERARCHICAL REASONING MODEL (HRM)

2.1 LATENT SPACE VS. TOKEN SPACE REASONING

Token-space reasoning relies on generating explicit intermediate steps as natural-language tokens, as in chain-of-thought prompting Wei et al. (2022). While such methods improve interpretability, they often introduce inefficiency: reasoning traces may be long, redundant, and brittle to errors in individual tokens Nye et al. (2021); Kojima et al. (2022). Moreover, many benchmarks reveal that token-based reasoning, despite improvements, remains slow and sometimes unreliable on harder problems Srivastava et al. (2022); Cobbe et al. (2021).

Latent-space reasoning takes a different approach, performing reasoning operations within the model's hidden representations rather than explicit text. Recent work on latent chain-of-thought and related methods Hao et al. (2024) shows that iterative refinement in latent space can yield more compact and flexible reasoning traces. These approaches reduce reliance on token generation, enabling richer intermediate states, faster inference, and in some cases improved accuracy Hao et al. (2024). Latent-space reasoning thus provides a promising complement to explicit token-space methods.

2.2 HRM VS. TRANSFORMERS

The Hierarchical Reasoning Model (HRM) Wang et al. (2025) is the latest popular model that performs latent-space reasoning. It introduces a dual-loop recurrent architecture inspired by cognitive science. It alternates between a fast low-level (L) module that executes candidate reasoning steps and a slower high-level (H) module that updates a more abstract plan. HRM iterates these cycles until a learned halting signal indicates completion, yielding effective depth far beyond its nominal parameter count. Notably, HRM achieves strong performance on reasoning-heavy benchmarks such as Sudoku and ARC-AGI-1 Chollet (2019) without pretraining or explicit chain-of-thought supervision Wang et al. (2025). Despite the promising ideas and concepts introduced by the paper, it is unclear how much benefit these L and H modules bring, compared to a plain transformer. In our experiments in Figure 2b, we show that even without H module, a plain 8-layer L-Module only HRM performs similarly to the original 4-layer L-module, 4-Layer H-module HRM. This is consistent to what is reported by ARC-Foundation (2025) , which says that plain transformer performs similarly to HRM.

3 RECURRENT REASONING AS A TYPE OF DIFFUSSION MODEL

3.1 HRM IS JUST A DIFFUSION MODEL

HRM's training algorithm — which replaces Backpropagation Through Time (BPTT) with a "one-step" gradient rule — turns out to be conceptually analogous to how diffusion models are trained. In a denoising diffusion process, one repeatedly applies a neural network to gradually denoise a sample, learning to map a noisy input back to the clean data (Ho et al., 2020; Song et al., 2023). Similarly, HRM repeatedly refines an internal latent state towards the solution.

Formally, diffusion training uses example pairs (x_t, x_0), teaching a network $f_\theta(x_t, t)$ to recover the original data x_0 from a noisy version x_t. HRM's one-step training can be viewed the same way: the model takes an *intermediate reasoning state* (analogous to x_t) and learns to map it directly to the final answer (analogous to x_0). The only substantive difference is the data: diffusion models add artificial noise to construct x_t, whereas HRM's latent state is already partway to the solution. This distinction is minor: in fact, HRM corresponds exactly to a special case of a "latent consistency model" (Song et al., 2024).

3.2 BPTT VS. DIFFUSION: A BIOLOGICAL PERSPECTIVE

A key motivation for HRM is biological plausibility: by avoiding BPTT, the authors claim a more brain-like learning rule. Once we recognize the parallel with diffusion models, it follows that any diffusion-like recurrent process could share this advantage. Diffusion models do not require back-propagation through multiple time steps; they train a feedforward mapping from one state to an-

other independently. In HRM's words, the one-step gradient has "constant memory" versus BPTT's growing footprint. Diffusion training is inherently of this form: the model learns locally at each step without unfolding the entire recurrence in memory.

Therefore, diffusion models offer an equally biologically plausible paradigm for deep sequential computation. Generating a sample with a diffusion model is effectively a recurrent process (iteratively refining a latent) but it is trained with a feedforward loss. This suggests that mechanisms inspired by diffusion could provide a basis for "deep reasoning" in the brain without requiring implausible credit assignment. In summary, HRM's replacement of BPTT is not unique: any latent consistency or diffusion model likewise avoids BPTT, pointing to a broader potential for biologically plausible recurrent learning. Further work should formalize this connection and explore how diffusion-like updates might operate in neural circuits.

4 Adaptive Recurrent Reasoning for Both Training and Inference

4.1 Key Finding

We identify a critical mismatch in the original HRM implementation: while its Adaptive Computation Time (ACT) module learns to decide when to halt during training, evaluation ignores this policy and always executes the maximum number of reasoning steps. To address this, we modify HRM to use its learned halting policy at evaluation as well. Our implementation preserves the inner carry state across evaluation steps to ensure reasoning continuity and a fair comparison with training dynamics.

Our adjustment allows HRM at evaluation time to more faithfully reflect the model's learned reasoning process. By accessing the learned ACT during inference, the model is able to decide earlier on that a puzzle is completed, preventing potential errors it could make if it were to complete its maximum number of reasoning steps. Indeed, as we observe in Figure 1 and Figure 2, our modified HRMs highly benefit from applying ACT halting during inference, greatly surpassing the original HRM performance. In Figure 3, we observe that the model flexibly allocates computation steps across different puzzles, with some puzzles being solved in very few reasoning steps. Our findings also raise interesting questions about how models reason, and how they differ from or align with human reasoning.

4.2 Methods

We build on the Hierarchical Reasoning Model (HRM) introduced by Wang et al. (2025), which integrates the ACT module into a multi-step recurrent reasoning architecture. The HRM architecture consists of two hierarchical reasoning modules: a high-level (H) module that processes global abstractions, updating less frequently, and a low-level (L) module that handles local reasoning, which updates more frequently. The model also maintains an inner carry state that preserves computational context across reasoning iterations at every step. The model's ACT module uses Q-learning to balance exploration (continuing reasoning) against exploitation (terminating with the current solution). At each step m, a Q-head outputs values for *halt* and *continue* as $Q^m = \big(\hat{Q}^m_{\text{halt}}, \hat{Q}^m_{\text{continue}}\big)$. Halting occurs if (i) the maximum horizon $M_{\max}$ is reached or (ii) $\hat{Q}^m_{\text{halt}} > \hat{Q}^m_{\text{continue}}$ once a stochastic minimum threshold $M_{\min}$ has been satisfied.

Choosing *halt* gives reward $\hat{G}^m_{\text{halt}} = \mathbf{1}\{\hat{y}_m = y\}$, while *continue* yields

$$\hat{G}^m_{\text{continue}} = \begin{cases} \hat{Q}^{m+1}_{\text{halt}}, & m \geq M_{\max}, \\ \max\big(\hat{Q}^{m+1}_{\text{halt}}, \hat{Q}^{m+1}_{\text{continue}}\big), & \text{otherwise.} \end{cases}$$

The loss at step m combines prediction accuracy and stopping decisions:

$$\mathcal{L}^m_{\text{ACT}} = \text{Loss}(\hat{y}_m, y) \;+\; \text{BCE}(\hat{Q}^m, \hat{G}^m).$$

In the released HRM source code, however, while the ACT module is used during training, evaluation almost always executes for the full M_{max} steps. This approach ignores the halting mechanism,

ensuring uniform batch shapes but preventing evaluation from using the model's learned stopping policy to its full potential. As a result, evaluation performance diverges from training performance, as the model under evaluation may take unnecessary extra steps even when a puzzle is already solved. By neglecting to use the ACT module to halt when the puzzle is already solved, potential computational efficiency gains and performance are compromised.

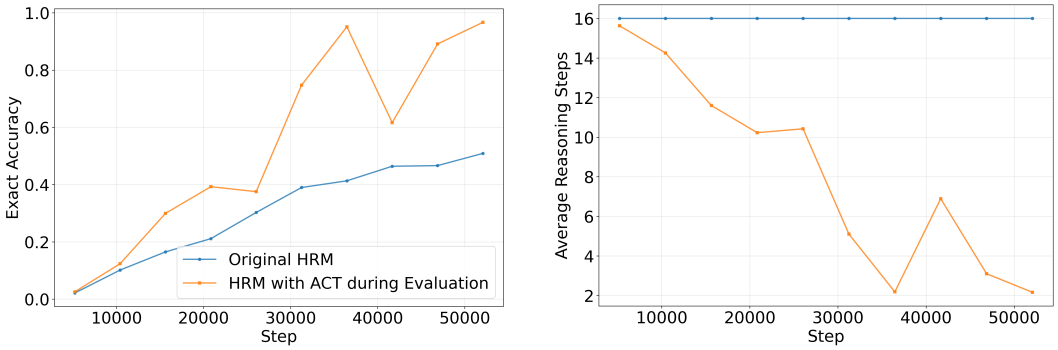
We modify HRM to apply halting learned during training to evaluation as well. Specifically, sequences are completed when it either halts naturally ($\sigma(\hat{Q}^m_{\text{halt}}) > 0.5$) or reaches the maximum step limit ($t \geq M_{max}$). We use ($\sigma(\hat{Q}^m_{\text{halt}}) > 0.5$) rather than the stochastic $\hat{Q}^m_{\text{halt}} > \hat{Q}^m_{\text{continue}}$ comparison in the original HRM training implementation to determine whether to halt or continue the sequence, as evaluation does not benefit from using the exploratory $\hat{Q}^m_{\text{continue}}$ logits.

Evaluation proceeds until all sequences in a batch have completed, whether through having used the ACT module to halt the sequence or evaluating the sequence up to the max step. Rather than running all the sequences up to the max step, this modification allows evaluation to measure both task accuracy and the model's ability to adaptively allocate computation.

In the original HRM implementation, evaluation resets the inner carry state at each step with fresh batch data, making evaluation deterministic but disconnected from the halting logic of the ACT module. In our implementation, we preserve the inner carry state across evaluation steps unless either evaluation begins or all sequences are complete, to ensure continuity of computation across the sequences and avoid premature resets that could distort halting behavior. In the evaluation while-loop, batches terminate once all sequences complete.

Our changes ensure that evaluation tests the learned halting policy, not just maximum-depth reasoning of the model. We compare our HRM, where ACT is used in both training and evaluation, against the original HRM with fixed-step evaluation to measure accuracy improvements. Specifically, we train this modified HRM, utilizing its halting capabilities in its evaluation mode, on the released Sudoku-Extreme dataset.

4.3 RESULTS



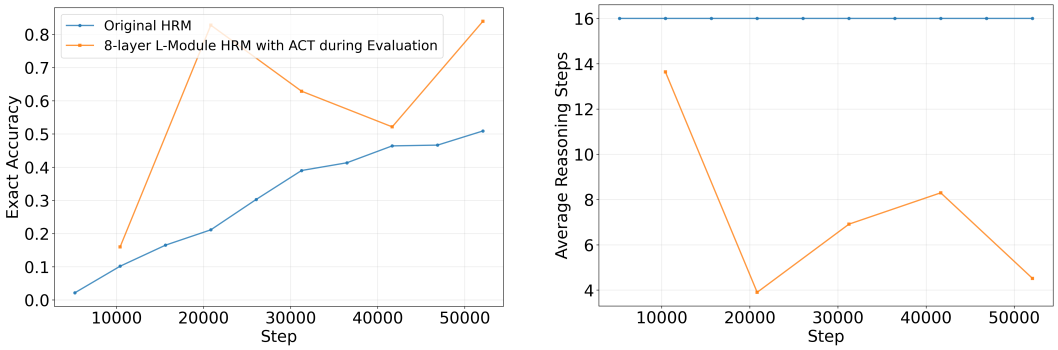
(a) Exact accuracy on test data (1 if the whole sequence is correct, 0 otherwise)

(b) Average number of reasoning steps during inference

Figure 1: Comparison of Sudoku performance of the original HRM and our modified HRM, which uses the ACT halting mechanism during evaluation in addition to training. Our modified HRM largely outperforms the original HRM (which is similar to the 55% performance reported in the original HRM paper), achieving nearly perfect performance on this difficult Sudoku-Extreme dataset. The modified HRM also takes less steps to reason during evaluation due to the ACT halting mechanism.

Remarkably, our modified HRM implementation, which uses ACT in both training and evaluation, largely outperforms the original HRM, achieving nearly perfect performance on the Sudoku-Extreme dataset compared to the original HRM's 55% performance (Figure 1). This shows that halting during evaluation is a crucial component for realizing the model's reasoning potential. When the model continues reasoning after it has already produced a correct solution, additional steps can

destabilize its internal carry state and introduce errors. Over-reasoning can harm performance, as the model's fixed-step evaluation can force it it to keep making edits and predictions even when those updates are unnecessary. By contrast, allowing the ACT policy to halt at inference can help prevent this drift. We can observe more of the model's flexible halting process, and its ability to reason in fewer steps, during evaluation in Figure 3.



(a) Exact accuracy on test data (1 if the whole sequence is correct, 0 otherwise)

(b) Average number of reasoning steps during inference

Figure 2: Comparison of Sudoku performance of the original HRM and our modified 8-layer L-Module HRM, which still uses ACT halting during evaluation. Our modified HRM also largely outperforms the original HRM. Our 8-layer L-Module HRM also takes less steps to reason during evaluation due to the ACT halting mechanism.

In Figure 2, we do away with the high-level H-module in the HRM and instead expand the low-level L-module to 8 layers, while continuing to enable ACT during evaluation. The performance advantage remains. The model with halting consistently achieves higher accuracy than the original HRM, while also requiring fewer steps on average to complete a puzzle.

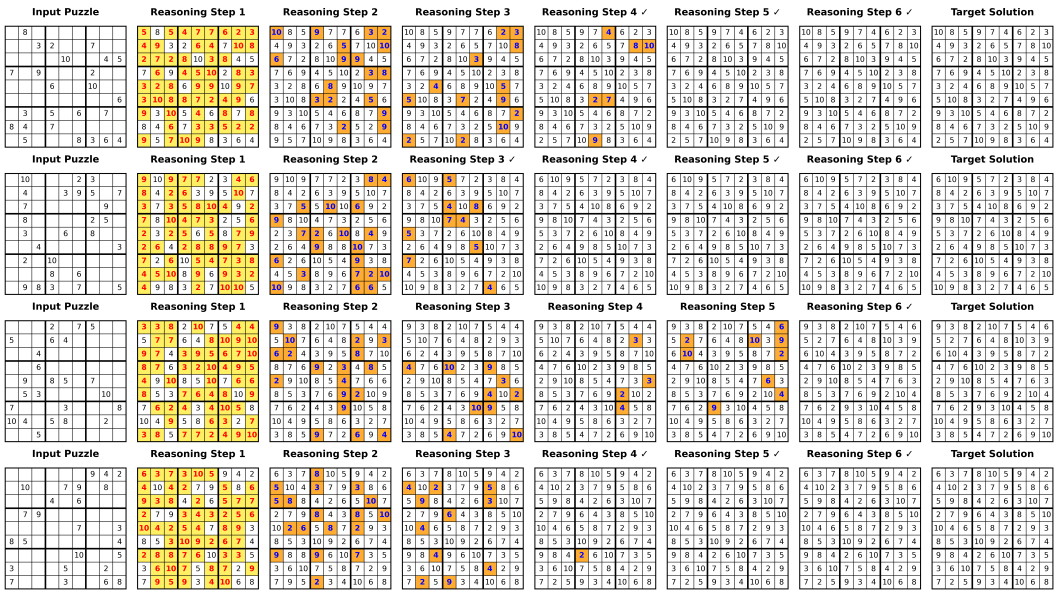


Figure 3: Reasoning with very few steps: Our modified HRM uses the ACT mechanism to decide to halt earlier during evaluation. A surprising observation is that the model learns to solve difficult Sudoku tasks with very few reasoning steps. This is in sharp contrast to how humans reason, posing an interesting puzzle on the nature of reasoning.

Figure 3 provides examples where the model halts after only a handful of reasoning steps, while still producing correct solutions. The model dynamically adjusts computation across puzzles and sometimes solves difficult puzzles with very short trajectories. This behavior contrasts with human reasoning, where solving a Sudoku puzzle typically involves step-by-step constraint propagation until the grid is fully filled. Adaptive halting allows the model not only to avoid unnecessary computation but also to achieve correct solutions through compressed reasoning pathways, raising interesting questions about how neural models represent and execute reasoning.

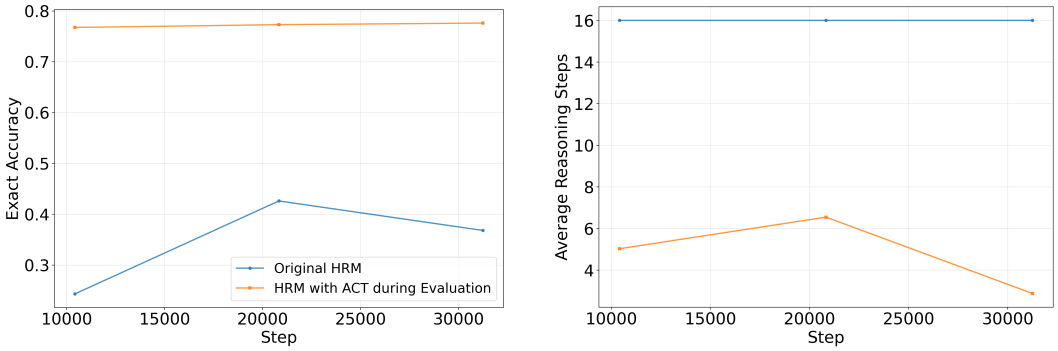
4.4 Reasoning with Few Steps

We demonstrate the trained model's behavior on some concrete examples of Sudoku puzzles in Figure 3. It is very surprising to see that the model is able to solve Sudoku solves with just a few steps of reasoning, which behaves very differently from humans.

5 Generalization to Other Datasets

5.1 Maze-Hard

To show that our approach generalize to other datasets, we tested its performance on the Maze-Hard dataset. The results are shown in 4. Our approach converges much faster and achieves double performance within 30,000 training iterations.



(a) Exact accuracy on test data (1 if the whole sequence is correct, 0 otherwise), for 6000 epochs

(b) Average number of reasoning steps during inference, for 6000 epochs

Figure 4: (a) and (b) compare the HRM models' performance on the Maze-Hard dataset. The modified HRM also outperformed the original HRM, while utilizing its halting ability from the ACT to take fewer reasoning steps. Additionally, c is the continued training run of the original HRM that achieves similar performance as reported.

5.2 ARC-AGI

ARC-AGI has large board size and requires significantly more computational resources to run, which can take a few days on a single GPU. At the time of writing, we are actively running experiments on ARC-AGI. We will report results on ARC-AGI in the next revision of the paper.

6 Discussion and Conclusion

In this paper, we perform a critical review on the recently popular reasoning model HRM. We discuss the similarity between HRM and difussion models. With some simple variants, we achieve drastically better performance than the original HRM model, raising the Sudoku-Extreme performance from previously reported 55% to above 90%. We highlight the interesting relationship between testing performance and testing reasoning steps, where the model learns to reason with fewer and fewer steps while achieving increasingly better performance. Visualization on Sudoku examples provides

an intuitive demonstration of model's surprising behavior, which differs greatly from human problem solving.

REFERENCES

Emmanuel Abbe, Sanjeev Arora Bengio, Ethan Dyer, et al. How far can transformers reason? the locality barrier and the power of attention. *arXiv preprint arXiv:2402.04625*, 2024.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

ARC-Foundation. The hidden drivers of hrm's performance on arc-agi. `https://arcprize.org/blog/hrm-analysis`, 2025. Accessed: 2025-09-24.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

François Chollet. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2022.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. In *arXiv preprint arXiv:2110.14168*, 2021.

Aditya Dedhe, Alexander Fabbri, Reuben Cohn-Gordon, and Michael C. Frank. Origins of hierarchical reasoning in human cognition. In *Proceedings of the Annual Meeting of the Cognitive Science Society (CogSci)*, 2023.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, 2019.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Kyunghyun Lee, Jamie Callan, and Yiming Yang. Pal: Program-aided language models. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2023.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *arXiv preprint arXiv:2103.03874*, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*, 2022.

Maxwell Nye, Anders Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. Show your work: Scratchpads for intermediate computation with language models. In *arXiv preprint arXiv:2112.00114*, 2021.

Bowen Song, Soo Min Kwon, Zecheng Zhang, Xinyu Hu, Qing Qu, and Liyue Shen. Solving inverse problems with latent diffusion models via hard data consistency. In *Proceedings of the International Conference on Learning Representations*, 2024.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML '23, pp. 32211–32252. PMLR, 2023.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. In *arXiv preprint arXiv:2206.04615*, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi Yadkori. Hierarchical reasoning model. *arXiv preprint arXiv:2506.21734*, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.