

Masked Diffusion Language Models are Strong and Steerable Text-Based World Models for Agentic RL

Darshan Deshpande

 Patronus AI

darshan@patronus.ai

Abstract

Recent growth in different reinforcement learning (RL) techniques have surfaced a need for a wide variety of specialized training environments. These environments are typically hand-curated, with task and reward difficulties that are fixed rather than adaptive, making them ineffective training signals once a model’s performance on the domain improves. As models continue to improve on these environments and reward signals grow increasingly sparse over longer horizons, the model encounters fewer diverse situations during rollouts, leaving it prone to overfitting on specific workflows or tool structures, also known as mode collapse. World models that simulate environment states have previously matched the performance of pure environment rollouts, making them a promising avenue for scaling diversity given that their outputs can be varied on-demand and at scale. However, autoregressive (AR) world models suffer from a fundamental left-to-right bias that prevents them from conditioning on globally interdependent state anchors such as tool schemas, prior turns, and expected outcomes. In this work, we (i) formalize text-based world modeling as a steerable transition-dynamics problem decomposed into initial environment state, task context, tool schemas, domain rules, and steering directives, and (ii) curate a dataset of 239,403 grounded state–action trajectories spanning nine open-source environments and twelve frontier model families. Using this dataset, we present a comparative study between AR LMs and masked diffusion language models (MDLMs), and show that MDLMs, by virtue of bidirectional anchor-aware denoising, produce better coherence, groundedness, and empirically validated rollout diversity than LLMs more than $4\times$ their total parameter size, with comparable inference latency. We introduce a plug-and-play GRPO training framework with deterministic state checks, and perform zero-shot transfer ablations on three out-of-distribution environments (ScienceWorld, ALFWorld, AppWorld) across three agent backbones from 1.2B–7B parameters (LFM2.5, Qwen3, Mistral), achieving absolute gains of up to 47% over raw baselines without environment-specific fine-tuning. Finally, we conduct a behavioral analysis of failure modes under adversarial scenarios and a human evaluation centered on realism, outcome correctness, and training utility to showcase their reliability. We open source our work to encourage research in this direction ¹.

1 Introduction

Reinforcement learning (RL) has transformed Large Language Models (LLMs) from passive sequence generators into decision-making agents capable of operating in complex and dynamic tool-use environments [84]. Such agents have demonstrated strong performance across diverse domains

¹Dataset: https://huggingface.co/PatronusAI/world_model_corpus
Training code: https://github.com/patronus-ai/mdlm_world_modeling

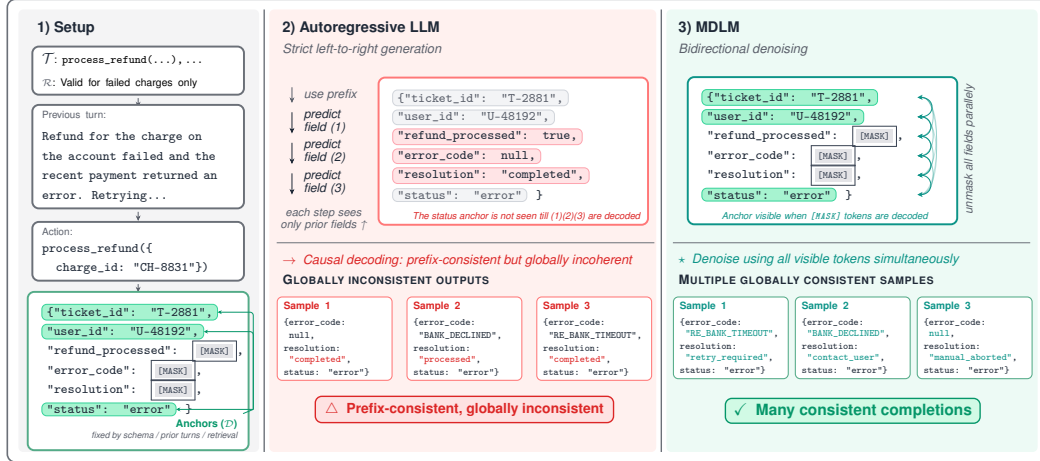


Figure 1: Anchor-aware structured generation: autoregressive vs. masked diffusion LMs. **(1)** A tool-call output has anchor fields $\mathcal{D}$ fixed by schema or context (ticket_id, user_id, trailing status: "error") interleaved with fields to be generated. **(2)** Left-to-right AR decoding cannot condition on the trailing anchor, yielding prefix-consistent but globally incoherent samples (e.g., refund_processed: true with status: "error"). **(3)** An MDLM denoises masked positions while attending to anchors on both sides, producing completions consistent with the error status.

including web navigation [44], code generation [29], and enterprise automation [16]. Unlike static reasoning tasks, agents deployed in multi-turn settings must maintain memory across dialogue rounds, perform sequential decision-making, and adapt to environmental feedback. Yet, training agents in multi-turn settings introduces compounding challenges: reward signal sparsity [22], distributional shift across turns [112], and limited generalization beyond the training environment [109].

A fundamental bottleneck constraining agent training is the availability, diversity, and scalability of environments [20, 108]. Standard RL environments are typically hand-engineered and static, causing agents to develop brittle policies that fail under distributional shift [74, 8]. Curriculum learning and unsupervised environment design have been proposed to address this [57], but constructing effective curricula still demands significant domain expertise. Domain randomization [64] and procedural content generation [47] have shown to improve inter-episode diversity, yet the environment remains fixed within each episode, leaving intra-episode adaptation largely unaddressed. Together, these limitations motivate learned environment simulators that can generate diverse, realistic dynamics at scale without hand-crafted engineering.

An alternative known as world models are generative models that predict future environment states conditioned on past states and actions. They offer a principled solution to this scalability and diversity problem [40]. Early works such as DreamerV3 [38] demonstrate that a world model can internalize environment dynamics and support imagined rollouts for planning, generalizing across over 150 diverse tasks with a single configuration. More recently, LLM-based text world models have emerged as a lightweight and scalable paradigm [55]. Li et al. (2025) show that well-trained world models maintain coherent latent state, scale predictably with data and model size, and consistently improve downstream agent performance. Works such as Wu et al. (2025) further demonstrate that reinforcement learning from verifiable rewards can improve world model fidelity in both language and video settings, making them a promising direction. However, a critical limitation of autoregressive models (ARs) as textual world models is their left-to-right generation bias: environment states are globally interdependent, yet causal models are architecturally constrained to condition only on preceding context, making reliable steering harder and susceptible to hallucinations [2]. While techniques like chain-of-thought [101] or Monte-Carlo-Tree-Search [111] have been used for exploration and reduce hallucinations [26], such techniques increase inference latency, making them unsuitable for very long, batched agent rollouts for GRPO-like [84] RL algorithms.

Masked diffusion language models (MDLMs) [70, 13, 18, 58] offer a structurally different alternative. By learning to iteratively denoise corrupted token sequences, MDLMs acquire bidirectional context over the full observation and naturally model global consistency which we show is essential for a faithful environment simulation. Unlike AR models, MDLMs are not subject to the causal masking constraint, allowing them to condition predicted states on both preceding and succeeding context within a state. Moreover, their denoising training objective encourages learning smooth, structured distributions over state spaces, which, in this paper, we hypothesize leads to more coherent and ecologically grounded environment rollouts. While diffusion models are standard for computer vision world models [95, 9, 27], their discrete, text counterparts, MDLMs have not been systematically evaluated as world models for textual environment simulations.

In this work, we address the following research questions:

1. How do masked diffusion language world models compare against causal large language world models on in-domain and out-of-domain environment simulations?
2. Does training RL agents on MDLM-generated rollouts yield measurable downstream performance improvements on held-out environments compared to AR-rollout baselines?
3. Do MDLM produced environment rollouts showcase a high degree of realism, correctness and training utility according to human experts? What are some common failures of such world models?

Through extensive experiments across different model sizes, we find that small MDLMs (8B parameters), including SDAR [18], LLaDA-2.1 [14], and WeDLM [58], consistently outperform finetuned causal models including Qwen-3.5 [93], Nemotron-3 [71], GLM-4.7-Flash [92], and GPT-OSS-20B [72] at scales up to 35B mixture-of-expert parameters and 27B dense AR models. This result suggests that the causal inductive bias, rather than model capacity, is the primary bottleneck in LLM-based world modeling. While previous works such as Li et al. (2025) require fine tuning of models for downstream applications, we show that GRPO training in a zero shot setting with MDLM generated trajectories leads to performance improvements of up to 47% across LiquidAI [1], Qwen [114] and Mistral [45] families ranging from 1.2-7B parameters. Finally, our human evaluation studies also show that these models achieve high human alignment scores across the training utility, realism and outcome correctness metrics with the four annotator averages reaching 4.75, 4.25 and 4.50 on a 1-5 Likert scale with high inter-annotator alignment ($\alpha \geq 0.89$).

2 Relevant Work

LLMs as synthetic data simulators LLMs have recently been applied to large scale synthetic data generation tasks for post-training [68, 60, 3]. Common techniques for diversification and scalable generation cover persona based diversification [33, 100], document or web-grounded techniques [63, 90, 122, 61], workflow-based grounding [66, 91, 80] and long horizon generation [41, 51, 31]. As RL techniques continue to mature and scale, several works have suggested techniques to scale tool simulations [34, 59] and agent-human interplay for trajectory diversification [105, 78]. Since the release of the GRPO algorithm [84], post-training focus has shifted towards specialized training environments with binary verifiable rewards [103, 104, 23]. The field has since seen a rapid surge of diverse training and evaluation environments covering coding [46, 65], computer use [110, 79, 121, 102], customer service [11] and agentic safety [25]. Works such as Song et al. (2026) have covered scaling of environments but since task diversity in isolation is not enough and rewards and tooling need to scale with model performance [42, 17], we have recently observed a rise of text-based world models [55, 99, 17] that aim to improve training signal by helping diversify model experiences in a controlled manner. In this work, we attempt to further showcase abilities of models at simulating realistic environment behaviors to improve downstream RL performance with the hope to move towards RL in imaginative worlds, similar to Li et al. (2024).

MDLMs as controllable models An ideal world model must be fast, high fidelity, generalizable and steerable [32, 10]. AR language models exhibit a strong left-to-right bias which restricts diversity and editability and the inference costs for such AR generation scales linearly with the sequence length, making these models difficult to setup for efficient and high performance inference [106]. The MDLM training procedure enables parallel token generation and bidirectional infilling which

further improve their applicability for arbitrary state prediction [81, 70]. Furthermore, works such as Nie et al., Sahoo et al., Shi et al. (2025, 2024, 2025) address the reversal curse and show potential for high quality generations. Since it is compute intensive to train MDLMs, recent works such as Fu et al., Gong et al. (2025, 2025) have shown that conversion of large pre-trained AR models to MDLMs is not only possible but also performant. Building on top of these works, optimization techniques such as block diffusion [7] and joint threshold for merging Token-to-Token editing with conventional Mask-to-Token decoding during inference [14] have been progressively studied and shown to be effective at scaling. Parallely, Nie et al. (2025) studied the scaling laws of such MDLMs and show that they are comparable to AR models which increasingly makes them suitable for world modeling applications. As MDLMs continue to scale across capabilities and domains [18, 58], their applicability to world modeling and environment state simulation is still understudied. We hope to address this gap through this work.

World Modeling Since domain specific RL requires highly specialized and robustly curated environments [24, 19, 119], the field has started exploring ways to improve experience-based learning [17] to reduce their dependencies on non-scalable and hard to maintain environments. While world modeling is well-studied in the computer vision domain, Li et al. (2025) were one of the first to exhaustively benchmark and ablate AR model performance for this task. Other parallel works have since explored diverse task and tooling simulations [62, 39] for scaling environment generations. Wang et al. (2025) applied augmentation of retrieved information to ground the content of their predictions whereas Wang et al. (2026) were the first to propose an end to end pipeline for task, tool and deterministic reward generation. But such a system has three drawbacks. First, the pipeline heavily relies on correctness of each agent created component which makes it susceptible towards cascading errors from previous components or poor design of environment components. Secondly, the downstream scalability of such a pipeline is not exhaustively studied which makes generalizability questionable. Lastly, steering for specific model behaviors is hard and infrastructural stability issues could potentially worsen if agent is not strong at coding or loses context. To address these challenges, we first add steering objectives to ensure grounding in real tool schemas, expected behaviors and desired outcomes to provide finer grained guidance. Secondly, we propose a plug-n-play framework for world model-based training by utilizing deterministic state checks which makes observability easier. Finally, we show generalization on six tasks spanning enterprise tool use tasks, coding environments, open ended APIs and text-based games.

3 Masked Diffusion Language Models for Steerable World Modeling

Formalizing the World Model Objective. Extending the definition of Li et al. (2025), we formalize a world model as a parametric approximation of the environment’s transition dynamics. For any textual environment, given a trajectory $\tau = (\{s_0, a_0\}, \dots, \{s_T, a_T\})$ with states $s_t \in \mathcal{S}$ and actions $a_t \in \mathcal{A}$, we decompose each state as $s_t = (e_t, c_t, \mathcal{T}, \mathcal{R}, \mathcal{D})$, where e_0 denotes the **initial environment state** (observable world configuration, e.g., files, databases, UI available at the beginning of the rollout), c_t the **task context** (user goal and conversational history), $\mathcal{T}$ the **tool schemas** (available tools and their signatures), $\mathcal{R}$ the **domain rules** (constraints on valid transitions), and $\mathcal{D}$ the **steering directives** (high-level instructions and safety constraints). The world model then learns

$$p_{\theta}(c_{t+1} \mid e_0, c_{\leq t}, a_{\leq t}; \mathcal{T}, \mathcal{R}, \mathcal{D}), \quad (1)$$

where $\mathcal{T}$, $\mathcal{R}$, and $\mathcal{D}$ remain fixed across the trajectory. In text-based environments, each s_t is serialized as a token sequence $\mathbf{x}^{(t)} = (x_1^{(t)}, \dots, x_L^{(t)})$, so the world model learns a distribution over structured token sequences conditioned on the history. Note that the world model represents $e_{\leq t}$ internally rather than emitting it as output, encouraging internal state tracking while avoiding context pollution. The fidelity of this learned distribution determines the quality of imagined rollouts for downstream agent training, and grounding generated states in environment variables and steering directives yields a natural and adaptable learning curriculum for the agent.

Masked Diffusion Language Models for World Modeling. Masked diffusion language models [70, 14, 18, 58] learn to predict a full next-state token sequence by iteratively denoising masked tokens, conditioned on an unmasked context. Given a clean next-state sequence during training $\mathbf{x}^{(t+1)} = (x_1^{(t+1)}, \dots, x_L^{(t+1)})$, the forward process samples a diffusion time $\tau \sim \mathcal{U}(0, 1]$ and independently replaces each token with a mask token [M] with probability τ , yielding the corrupted

sequence $\mathbf{x}_\tau^{(t+1)}$. A mask predictor p_θ is trained to recover the masked tokens conditioned on the history of past states $\mathbf{x}^{(\leq t)}$ and actions $\mathbf{a}^{(\leq t)}$. The training objective is:

$$\mathcal{L}_{\text{MDLM}}(\theta) = -\mathbb{E}_{\tau, \mathbf{x}^{(t+1)}, \mathbf{x}_\tau^{(t+1)}} \left[\frac{1}{\tau} \sum_{k=1}^L \mathbf{1}[x_{k,\tau}^{(t+1)} = [\text{M}]] \log p_\theta(x_k^{(t+1)} | \mathbf{x}_\tau^{(t+1)}, \mathbf{x}^{(\leq t)}, \mathbf{a}^{(\leq t)}) \right], \quad (2)$$

where the indicator selects positions that were masked at diffusion time τ . This objective is an upper bound on the negative log-likelihood of $\mathbf{x}^{(t+1)}$ under the model [70].

Steering via Conditional Masked Generation. Standard steering for LLMs is achieved by techniques such as speculative decoding [49] or prefix conditioning [83]. While this is effective for open-ended steering, a tokenized environment state has symmetric dependencies among its fields and entity attributes, tool schemas, and reward signals constrain one another without respecting any particular order. An AR factorization $\prod_i p(x_i | x_{<i})$ is forced to commit to one ordering, whereas the MDLM objective is equivalent to an any-order AR likelihood bound [73] and therefore learns every conditional direction, including conditioning on arbitrary known positions at inference time, from the same training signal. A further advantage of MDLMs is that predicting masked tokens in parallel from a shared context eliminates the within-state error accumulation that occurs when an AR model conditions later tokens on an already-hallucinated prefix [28]. This, in addition to selective mask filling as shown in Figure 1 strongly motivates the use of MDLMs for this task.

Diversity of MDLM Rollouts. Beyond coherence, the iterative denoising process in MDLMs induces greater structural diversity in rollout samples than causal sampling from an AR model. This effect is not due to temperature alone. It arises because MDLMs sample both *what* token to generate at each position and *which* position to commit to at each step. Gong et al. (2025) show that increasing the sampling temperature in a masked diffusion language model diversifies not only token choices but also generation order, a degree of freedom that AR decoders do not possess, since they must always generate the leftmost unfilled position. AR decoders are prone to *prefix mode collapse* [12], in which high-probability opening tokens are committed early and subsequently constrain all later decisions. By contrast, MDLMs can fill in the middle or end of a state before its beginning, which helps avoid concentrating samples around a narrow set of prefixes [96]. For state prediction, this is important because the rollout distribution should capture the true grounding in plausible next states given $(\mathbf{x}^{(\leq t)}, \mathbf{a}^{(\leq t)})$, rather than collapsing onto a mode induced by an arbitrary serialization order. We observe the empirical advantage of this in section 5.

4 Experimental Setup

Environment Selection and Dataset Curation. To study our research questions, we specially curate a total of 239,403 trajectories with tasks spanning software engineering (SWE-SMITH [116], CODERFORGE [6]), research automation (DEEPRESEARCHQA [37], OPENRESEARCHER [56]), customer service (TAUBENCH [11]), and general-purpose tool use (GORILLA/BFCLV4 [75], TOOLATHLON [53], PANDORA², WEBSHOP [117]). Following the findings of Li et al. (2025), we generate these trajectories with the following set of models to encourage diversity: QWEN-3.5-4B, QWEN-3.5-9B, QWEN3.5-397B-A17B-FP8, GPT-5.4-MINI, GPT-5.4, GPT-OSS-120B, CLAUDE-4.6-SONNET, CLAUDE-4.5-HAIKU, GEMINI-3-PRO, MINIMAX-M2.1, GLM-5, and DEEPSEEK-V3.2³. To evaluate fairly on out-of-domain environments, we select the fully deterministic SQL-only split from INTERCODE [115], along with OCCUBENCH [43] and API BANK [54], to assess industry-relevant specialized tools, tool schemas, and a wide range of general tool-use capabilities, respectively. After filtering, post-processing, and length diversification, the final training dataset contains **239,403** unique trajectory instances in the training split, with **19,454** validation trajectories and **20,847** held-out, stratified in-domain test trajectories reserved for evaluation. We provide full details of our dataset curation pipeline including trajectory generation, length diversification, environment state grounding, and dataset statistics in Appendix C. To show downstream effectiveness of MDLMs as world models, we use SCIENCEWORLD [97], ALFWORLD [87] and APPWORLD [94] as completely out of distribution environments covering entirely new tool schemas and open ended

²<https://huggingface.co/datasets/danilopeixoto/pandora-tool-calling>

³All open-source model trajectories were generated using the Fireworks AI API. All closed models were queried via their respective official API endpoints.

Table 1: Generation quality across MDLMs and AR models on in-domain and out-of-domain test sets. Best zero-shot result in each column is **bolded** and the best three-shot result is bold highlighted in **red**.

Model	In-domain			API-Bank			OccuBench			Intercode (SQL)		
	B-1	R-L	MAUVE	B-1	R-L	MAUVE	B-1	R-L	MAUVE	B-1	R-L	MAUVE
<i>Baseline Models</i>												
Qwen-3-8B	.136	.286	.148	.453	.421	.301	.524	.523	.705	.406	.397	.699
Qwen-3.5-27B	.152	.221	.187	.536	.611	.505	.746	.755	.810	.516	.614	.799
Qwen-3.5-35B-A3B	.125	.233	.189	.488	.585	.499	.667	.680	.811	.522	.598	.780
Nemotron-3-Nano-30B	.120	.152	.223	.412	.316	.510	.675	.667	.822	.486	.465	.785
GLM-4.7-Flash	.130	.266	.232	.486	.289	.532	.741	.739	.815	.537	.516	.790
GPT-OSS-20B	.125	.191	.231	.436	.384	.400	.716	.717	.798	.544	.540	.788
LLaDA-2.1-mini	.146	.158	.168	.401	.444	.452	.648	.655	.811	.412	.424	.701
WeDLM-8B	.139	.157	.241	.488	.522	.405	.670	.748	.821	.491	.555	.781
SDAR-8B	.145	.240	.252	.503	.581	.605	.637	.727	.828	.328	.466	.731
SDAR-30B-A3B	.121	.295	.261	.677	.649	.697	.709	.775	.831	.373	.482	.749
<i>3-shot Baseline Models</i>												
Qwen-3-8B	.379	.506	.348	.578	.509	.365	.559	.544	.788	.532	.523	.833
Qwen-3.5-27B	.509	.578	.588	.611	.660	.600	.731	.772	.834	.585	.623	.849
Qwen-3.5-35B-A3B	.334	.507	.590	.621	.663	.634	.744	.781	.832	.589	.662	.851
Nemotron-3-Nano-30B	.346	.555	.618	.675	.667	.700	.738	.724	.845	.584	.551	.825
GLM-4.7-Flash	.401	.565	.629	.686	.689	.707	.746	.742	.838	.612	.597	.822
GPT-OSS-20B	.339	.555	.580	.662	.670	.699	.731	.720	.822	.644	.640	.880
LLaDA-2.1-mini	.390	.521	.523	.567	.589	.555	.655	.613	.819	.425	.462	.821
WeDLM-8B	.382	.512	.400	.582	.511	.451	.692	.755	.826	.568	.555	.866
SDAR-8B	.349	.451	.500	.588	.619	.668	.621	.729	.945	.611	.622	.870
SDAR-30B-A3B	.401	.601	.666	.601	.662	.702	.700	.713	.849	.621	.641	.886
<i>Finetuned Autoregressive Models</i>												
Qwen-3-8B	.524	.572	.699	.712	.781	.840	.588	.590	.897	.610	.677	.884
Qwen-3.5-27B	.601	.732	.804	.700	.702	.765	.678	.725	.911	.659	.622	.917
GPT-OSS-20B	.462	.461	.810	.697	.730	.762	.669	.730	.942	.643	.695	.897
GLM-4.7-Flash	.570	.588	.891	.672	.705	.766	.723	.792	.950	.629	.670	.949
Nemotron-3-Nano-30B	.545	.506	.731	.685	.681	.789	.671	.788	.922	.612	.645	.956
Qwen-3.5-35B-A3B	.630	.730	.932	.691	.689	.791	.766	.772	.960	.646	.653	.956
<i>Finetuned Diffusion Models</i>												
LLaDA-2.1-mini	.623	.713	.968	.677	.682	.789	.698	.677	.891	.602	.621	.901
WeDLM-8B	.728	.758	.963	.748	.785	.838	.715	.788	.971	.726	.711	.921
SDAR-8B	.805	.813	.982	.701	.682	.772	.795	.797	.979	.751	.741	.955
SDAR-30B-A3B	.848	.853	.992	.711	.728	.831	.712	.782	.971	.783	.792	.965
SDAR-8B (3 shot)	.891	.899	.990	.946	.889	.882	.831	.810	.981	.822	.824	.967
SDAR-30B-A3B (3 shot)	.905	.910	.995	.951	.954	.900	.904	.862	.983	.831	.834	.970

game navigation and action tasks that were excluded from the training set. More details about the environment setup are available in Appendix D.

Baselines. We compare against a broad suite of state-of-the-art open-source autoregressive (AR) LLMs spanning 8B–35B parameters, covering both dense and mixture-of-expert (MoE) configurations: QWEN3.5-27B [93], GPT-OSS-20B [72], GLM-4.7-FLASH [92], NEMOTRON-3-NANO-30B-A3B [71], and QWEN3.5-35B-A3B [93]. For masked diffusion language models (MDLMs), we select the three strongest publicly available block-diffusion checkpoints: WEDLM-8B [58] and SDAR-8B/30B-A3B [18] (dense), and LLaDA-2.1-MINI (16B-1AB, MoE) [14]. Since SDAR-8B and WEDLM-8B adapt the QWEN3-8B [114] weights and change its objective to masked diffusion, we additionally include QWEN3-8B as a baseline to isolate the contribution of the diffusion training and inference procedure. For each model we report zero-shot, three-shot, and fine-tuned numbers to separate the effect of in-context conditioning from supervised adaptation.

Training. All models are fine-tuned with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95/0.99$), gradient clipping (clip=1.0), and a cosine schedule with warmup, at a maximum sequence length of 16,384 tokens. We perform an automated sweep over learning rates $\{1 \times 10^{-6}, 5 \times 10^{-6}, 1 \times 10^{-5}, 5 \times 10^{-5}\}$ and

Table 2: Aggregated Self-BLEU, Distinct-N ($N=\{1, 2, 3, 4\}$) and MAUVE scores for fine tuned world models across test splits.

Model	Self-BLEU ($\downarrow$)	Distinct-N ($\uparrow$)	MAUVE ($\uparrow$)
QWEN3.5-35B-A3B	.690	.253	.889
QWEN3.5-27B	.670	.321	.864
GPT-OSS-20B	.659	.228	.867
SDAR-8B	.601	.385	.902

batch sizes $\{1, 2, 4, 8\}$ and report only the best scores. For MDLMs, we retain the block size used during the model’s original instruction tuning to preserve coherence as per [18]. Each experiment is repeated with three seeds $\{42, 7000, 9000\}$, and we report the mean in Table 1 and Table 2. All training runs use ms-swift [120] for LLM training, dFactory ⁴ for SDAR and LLADA-2.1-MINI training and WEDLM’s official training code ⁵ with MagiAttention [118] for WEDLM-8B. Further training and evaluation details are available in Appendix E.

Metrics. We evaluate world models along three orthogonal dimensions corresponding to our research questions: generation fidelity (RQ1), downstream training utility (RQ2), and semantic correctness as judged by humans (RQ3). For generation fidelity, we report BLEU-1 and ROUGE-L to measure n -gram and longest-common-subsequence overlap with reference next-states, providing surface-form comparisons across the full suite of in-domain and out-of-domain test sets. We additionally report MAUVE [76], which measures distributional alignment between generated and reference states in embedding space and is less sensitive to surface variation than n -gram metrics. This setup directly tests the central prediction of our anchor-aware analysis (Figure 1): if causal models produce prefix-consistent but globally incoherent completions while MDLMs condition bidirectionally on anchors, then MDLMs should exhibit stronger distributional alignment with reference states even when n -gram overlap is comparable. We deliberately avoid exact-match accuracy because tool responses can produce valid versions of the same underlying state, making exact match a weak metric. We acknowledge that no surface-form metric directly verifies semantic equivalence and hence, we complement these scores with diversity metrics Self-BLEU [67] and Distinct-N [52] to test the mode-collapse claim in section 3. To measure downstream effectiveness of world models on held out environments, we use task success rate metric as proposed by each environment’s goal state. Finally, we structure an independent human evaluation for studying the realism, training utility and objective correctness of world model predictions which we delineate in Appendix A.

5 Results and Discussion

RQ1: How do MDLMs compare against AR LMs for World Modeling Tasks? Table 1 reports generation fidelity across in- and out-of-domain evaluation suites. We observe that in the zero-shot setting, MDLMs of comparable size outperform their AR counterparts on the distributional alignment metric (MAUVE), with SDAR-30B-A3B achieving the best zero-shot MAUVE on three of four splits. The gap is most pronounced on API-BANK (.697 vs. .532 for GLM-4.7-FLASH) and on the in-domain split (.261 vs. .232). We analyzed the outputs carefully to observe that in zero-shot settings, AR models tend to produce more verbosity which hurts distributional closeness to concise API responses. Since LLMs are good few shot learners [15], we tested three-shot prompting which narrows but does not close this gap. We notice that few shot examples help LLMs better understand the level of verbosity required and hence, models such as GLM-4.7-FLASH and NEMOTRON-3-NANO-30B end up outperforming or matching the SDAR-30B-A3B on API Bank like deterministic datasets. However, since the SDAR model builds on the representations of the base QWEN3-30B-A3B model, its ability to improve with in-context examples is complementary to the MDLM objective as seen for the WEDLM and LLADA-2.1-MINI models as well. After fine-tuning, we observe that MDLMs dominate across all four splits and all three metrics. SDAR-8B, despite being more than $4\times$ smaller than QWEN3.5-35B-A3B, surpasses every AR baseline on in-domain MAUVE (.982 vs. .932) and on OCCUBENCH MAUVE (.979 vs. .960). On INTERCODE, SDAR-8B is competitive on MAUVE (.955 vs. .956 for the best AR baselines, NEMOTRON-3-NANO-30B and QWEN3.5-35B-A3B) while achieving substantially higher BLEU/ROUGE (.751/.741 vs. .646/.653).

⁴<https://github.com/inclusionAI/dFactory>

⁵<https://github.com/tencent/WeDLM>

Table 3: Task success rate across different training methods and benchmarks. RL with QWEN-WM refers to QWEN3.5-27B, selected due to its strong performance and SDAR-WM refers to the SDAR-8B model. Environments marked with * are forced to be partially observable to study long context behaviors of the world model.

Model	Method	Appworld	SciWorld*	ALFworld*
LFM2.5-1.2B	Base	33.3%	1.7%	5.7%
	SFT only	51.2%	3.3%	42.9%
	SFT + RL w/ Qwen-WM	60.1%	11.7%	48.6%
	SFT + RL w/ SDAR-WM	62.0%	13.3%	53.6%
Qwen3-4B	Base	56.4%	16.7%	27.1%
	SFT only	62.5%	22.8%	39.3%
	SFT + RL w/ Qwen-WM	63.5%	33.3%	42.3%
	SFT + RL w/ SDAR-WM	66.9%	40.0%	46.1%
Mistral-7B-v0.3	Base	51.3%	3.3%	1.7%
	SFT only	63.8%	23.3%	11.7%
	SFT + RL w/ Qwen-WM	69.2%	33.3%	24.4%
	SFT + RL w/ SDAR-WM	71.2%	48.4%	34.3%

Three-shot SDAR-30B-A3B further pushes in-domain MAUVE to .995. Importantly, the QWEN3-8B baseline allows us to isolate the diffusion contribution as SDAR-8B and WEDLM-8B outperform fine-tuned QWEN3-8B across the board (e.g., +.283 in-domain MAUVE for SDAR-8B), demonstrating that gains arise from the masked diffusion objective rather than from base-model capacity or training data. Together, these results indicate that the causal inductive bias, rather than parameter count, is the primary bottleneck in LLM-based world modeling.

Diversity of MDLM rollouts. Table 2 reports rollout-level diversity for the strongest fine-tuned models across the out-of-domain sets. SDAR-8B achieves the lowest Self-BLEU (.601) and the highest Distinct-N (.385) and MAUVE (.900), confirming the hypothesis in section 3 that MDLM samples cover more of the plausible next-state distribution rather than collapsing onto prefix-induced modes. This is consistent with the observation of [35] that MDLM temperature jointly controls token choice and generation order, providing an extra axis of stochasticity unavailable to AR decoders.

RQ2: Does training agents on MDLM-generated rollouts improve downstream performance?

Table 3 reports task success rates on APPWORLD, SCIENCEWORLD, and ALFWORLD, across three agent backbones spanning 1.2B to 7B parameters. We select the QWEN3.5-27B and SDAR-8B due to their strong performance and ease of efficient dense model deployment (prompts in Appendix G). We also found that QWEN3.5-27B performance is equivalent to QWEN3.5-35B-A3B for these tasks in independent evaluations ($\pm 0.5\%$ overall performance across two runs). Across all nine model-environment pairs, GRPO training with SDAR-derived world model rollouts (SDAR-WM) outperforms both SFT-only baselines and GRPO with Qwen-derived rollouts (QWEN-WM). The improvements are largest on partially observable, long-horizon environments like on ALFWORLD, where LFM2.5-1.2B improves from 5.7% (base) to 53.6% with SDAR-WM (+47.9 absolute), and MISTRAL-7B-v0.3 improves from 1.7% to 34.3% (+32.6 absolute). On SCIENCEWORLD, MISTRAL-7B-v0.3 jumps from 3.3% to 48.4% (+45.1 absolute), surpassing the QWEN-WM variant by 15.1 points. The consistent gap between QWEN-WM and SDAR-WM (averaging +5.3 points across all configurations) cannot be attributed to capacity, since both world models are fine-tuned on identical data but it instead reflects the higher fidelity and diversity of MDLM rollouts established in RQ1. Notably, these improvements occur in a zero-shot setting with respect to the held-out environments, contrasting with prior work from Li et al. (2025) that required environment-specific fine-tuning and did not cover any environment formalizations.

RQ3: Human evaluation of realism, correctness, and training utility. We complement automatic metrics with a human study conducted by four industry experts recruited from Upwork⁶, each with at least two years of experience with LLM-based agentic harnesses. Annotators independently rated 100 SDAR-generated next states on a 1–5 Likert scale across three dimensions: *realism*, *outcome correctness*, and *training utility* (rubric and guidelines in Appendix A). As we can observe in Table 4,

⁶<https://upwork.com>

Table 4: Human evaluation results across three metrics. Mean is computed over annotator ratings (1-5 Likert scale) and Krippendorff’s α reports inter-annotator agreement. Definitions for metrics are in Appendix A

Metric	Mean	Krippendorff’s α
Realism	4.75	.932
Outcome Correctness	4.25	.891
Training Utility	4.50	.901

SDAR achieves means of 4.75 on realism, 4.25 on outcome correctness, and 4.50 on training utility, with Krippendorff’s α [48] above 0.89 on every dimension, indicating significant inter-annotator agreement. Annotators noted strong adherence to steering directives, particularly for adversarial scenarios such as forced tool failures, suggesting that bidirectional conditioning leads to reliable steerability without loss of realism. Common failure cases flagged by annotators included incorrect numeric key type coercions (string vs. integer), corrupted API keys, and a tendency for the model to collapse into repeated tool errors once any earlier turn returned an error, similar to Anil et al. (2024)’s observations. Next, we study the finer grained world model behaviors in MDLMs and AR models.

Behavioral and steerability analysis. We probe SDAR’s behavioral patterns under a curated set of adversarial scenarios, contrasting them with AR baselines. The scenarios cover (i) reasoning over database states that lack a direct answer but contain adjacent information, (ii) cases that force the model to act under information insufficiency, (iii) verbose tool outputs known to challenge MDLMs, (iv) varying trajectory length constraints, and (v) infeasible tasks given the available toolset and (vi) complex navigation scenarios that require strong state recollections like pagination of tool outputs. We observe four consistent patterns. First, MDLMs are largely agnostic to element ordering in steering instructions, but degrade into repetitions without a repetition penalty when tools demand emitting the entire database state (e.g., `get_all_queries`). While AR models are stronger at producing coherent texts as a result of good instruction tuning, we observed that AR models share this limitation but they instead devolve into producing ungrounded and hallucinated text instead of repetitions. Second, MDLMs are capable of performing basic database joins in parameter space and maintain strong consistency with prior trajectory events, making them well-suited for state simulation. Third, on infeasible tasks the model continues to produce non-degenerate environment states rather than terminating early, which is desirable behavior for exploratory rollouts. Fourth, at high temperatures MDLMs occasionally fall into block-level repetition patterns. These can be post-processed away and diminish considerably with larger MDLMs. Out of these patterns, two limitations persist across our study: (1) MDLMs struggle at producing well-formed and consistent API keys which we believe is due to a combination of post-training safety alignment of the base Qwen models and repetition due to small block size during diffusion, and (2) they exhibit pagination drift when tools require enumerated multi-page outputs. While these can be handled by optimizing steering, we expect both to improve as MDLM scaling and tool-use centric post-training continues to mature. We provide a few failure mode example snippets in Appendix H for reference.

6 Conclusion

In this work, we present the first systematic study of MDLMs as text-based world simulators for agentic RL. By formalizing world modeling as a steerable transition-dynamics problem with five grounded components and curating a 239,403-trajectory dataset across nine environments and twelve frontier model families, we showed that MDLMs, by virtue of their bidirectional, anchor-aware denoising, produce more coherent, grounded, and diverse environment rollouts than autoregressive LLMs more than $4\times$ their size, while remaining competitive in inference latency. Our zero-shot transfer experiments on SCIENCEWORLD, ALFWORLD, and APPWORLD further demonstrate that GRPO training with MDLM-generated rollouts yields absolute performance improvements of up to 47% across agents spanning 1.2B–7B parameters, without any environment-specific fine-tuning. Human evaluation (Krippendorff’s $\alpha > 0.89$ across realism, outcome correctness, and training utility) supports the reliability of these rollouts. Despite these gains, we observe failure modes including pagination drift, block-level repetition under verbose tool outputs, and brittle handling of structured fields such as API keys that limit the immediate use of small MDLMs as drop-in environment substitutes. We encourage future work on steering, robustness and reward hacking tendencies of these world models along with the potentially induced biases in the downstream model.

References

- [1] Liquid AI. Lfm2 technical report. *arXiv preprint arXiv:2511.23404*, 2025.
- [2] Aisha Alansari and Hamzah Luqman. Large language models hallucination: A comprehensive survey, 2026. URL <https://arxiv.org/abs/2510.06265>.
- [3] Ahmad Alismail and Carsten Lanquillon. A survey of llm-based methods for synthetic data generation and the rise of agentic workflows. In *International Conference on Human-Computer Interaction*, pages 119–135. Springer, 2025.
- [4] Chenxin An, Zhihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and Lingpeng Kong. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models, 2025. URL <https://hkunlp.github.io/blog/2025/Polaris>.
- [5] Cem Anil, Esin Durmus, Nina Panickssery, Mrinank Sharma, Joe Benton, Sandipan Kundu, Joshua Batson, Meg Tong, Jesse Mu, Daniel Ford, et al. Many-shot jailbreaking. *Advances in Neural Information Processing Systems*, 37:129696–129742, 2024.
- [6] Alpay Ariyak, Junda Zhang, Junxiong Wang, Shang Zhu, Federico Bianchi, Sanjana Srivastava, Ashwinee Panda, Siddhant Bharti, Chenfeng Xu, John Heo, Xiaoxia Shirley Wu, James Zhou, Percy Liang, Leon Song, Ce Zhang, Ben Athiwaratkun, Zhongzhu Zhou, and Qingyang Wu. Coderforge-preview: Sota open dataset for training efficient agents, February 2026. URL <https://www.together.ai/blog/coderforge-preview>. Project core leads: Alpay Ariyak; Zhongzhu Zhou; Qingyang Wu.
- [7] Marianne Arriola, Aaron Gokaslan, Justin T Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://arxiv.org/abs/2503.09573>.
- [8] Abi Aryan, Zac Liu, and Aaron Childress. Abidegym: Turning static rl worlds into adaptive challenges. *arXiv preprint arXiv:2509.21234*, 2025.
- [9] Jinbin Bai, Yu Lei, Hecong Wu, Yuchen Zhu, Shufan Li, Yi Xin, Xiangtai Li, Molei Tao, Aditya Grover, and Ming-Hsuan Yang. From masks to worlds: A hitchhiker’s guide to world models. *arXiv preprint arXiv:2510.20668*, 2025.
- [10] Philip J. Ball, Jakob Bauer, Frank Belletti, Bethanie Brownfield, Ariel Ephrat, Shlomi Fruchter, Agrim Gupta, Kristian Holsheimer, Aleksander Holynski, Jiri Hron, Christos Kaplanis, Marjorie Limont, Matt McGill, Yanko Oliveira, Jack Parker-Holder, Frank Perbet, Guy Scully, Jeremy Shar, Stephen Spencer, Omer Tov, Ruben Villegas, Emma Wang, Jessica Yung, Cip Baetu, Jordi Berbel, David Bridson, Jake Bruce, Gavin Buttimore, Sarah Chakera, Bilva Chandra, Paul Collins, Alex Cullum, Bogdan Damoc, Vibha Dasagi, Maxime Gazeau, Charles Gbadamosi, Woohyun Han, Ed Hirst, Ashyana Kachra, Lucie Kerley, Kristian Kjems, Eva Knoepfel, Vika Koriakin, Jessica Lo, Cong Lu, Zeb Mehring, Alex Moufarek, Henna Nandwani, Valeria Oliveira, Fabio Pardo, Jane Park, Andrew Pierson, Ben Poole, Helen Ran, Tim Salimans, Manuel Sanchez, Igor Saprykin, Amy Shen, Sailesh Sidhwani, Duncan Smith, Joe Stanton, Hamish Tomlinson, Dimple Vijaykumar, Luyu Wang, Piers Wingfield, Nat Wong, Keyang Xu, Christopher Yew, Nick Young, Vadim Zubov, Douglas Eck, Dumitru Erhan, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Raia Hadsell, Aäron van den Oord, Inbar Mosseri, Adrian Bolton, Satinder Singh, and Tim Rocktäschel. Genie 3: A new frontier for world models. 2025.
- [11] Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2 -bench: Evaluating conversational agents in a dual-control environment, 2025. URL <https://arxiv.org/abs/2506.07982>.
- [12] Lukas Berglund, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. The reversal curse: Llms trained on "a is b" fail to learn "b is a", 2024. URL <https://arxiv.org/abs/2309.12288>.

- [13] Tiwei Bie, Maosong Cao, Kun Chen, Lun Du, Mingliang Gong, Zhuochen Gong, Yanmei Gu, Jiaqi Hu, Zenan Huang, Zhenzhong Lan, et al. Llada2. 0: Scaling up diffusion language models to 100b. *arXiv preprint arXiv:2512.15745*, 2025.
- [14] Tiwei Bie, Maosong Cao, Xiang Cao, Bingsen Chen, Fuyuan Chen, Kun Chen, Lun Du, Daozhao Feng, Haibo Feng, Mingliang Gong, et al. Llada2. 1: Speeding up text diffusion via token editing. *arXiv preprint arXiv:2602.08676*, 2026.
- [15] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [16] Yibin Chen, Yifu Yuan, Zeyu Zhang, Yan Zheng, Jinyi Liu, Fei Ni, Jianye Hao, Hangyu Mao, and Fuzheng Zhang. Sheetagent: towards a generalist agent for spreadsheet reasoning and manipulation via large language models. In *Proceedings of the ACM on Web Conference 2025*, pages 158–177, 2025.
- [17] Zhaorun Chen, Zhuokai Zhao, Kai Zhang, Bo Liu, Qi Qi, Yifan Wu, Tarun Kalluri, Sara Cao, Yuanhao Xiong, Haibo Tong, Huaxiu Yao, Hengduo Li, Jiacheng Zhu, Xian Li, Dawn Song, Bo Li, Jason Weston, and Dat Huynh. Scaling agent learning via experience synthesis, 2025. URL <https://arxiv.org/abs/2511.03773>.
- [18] Shuang Cheng, Yihan Bian, Dawei Liu, Linfeng Zhang, Qian Yao, Zhongbo Tian, Wenhai Wang, Qipeng Guo, Kai Chen, Biqing Qi, et al. Sdar: A synergistic diffusion-autoregression paradigm for scalable sequence generation. *arXiv preprint arXiv:2510.06303*, 2025.
- [19] Zhoujun Cheng, Shibo Hao, Tianyang Liu, Fan Zhou, Yutao Xie, Feng Yao, Yuexin Bian, Yonghao Zhuang, Nilabjo Dey, Yuheng Zha, Yi Gu, Kun Zhou, Yuqi Wang, Yuan Li, Richard Fan, Jianshu She, Chengqian Gao, Abulhair Saparov, Haonan Li, Taylor W. Killian, Mikhail Yurochkin, Zhengzhong Liu, Eric P. Xing, and Zhiting Hu. Revisiting reinforcement learning for llm reasoning from a cross-domain perspective. *ArXiv*, abs/2506.14965, 2025. URL <https://api.semanticscholar.org/CorpusID:279447674>.
- [20] Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.
- [21] LMDeploy Contributors. Lmdeploy: A toolkit for compressing, deploying, and serving llm. <https://github.com/InternLM/lmdeploy>, 2023.
- [22] Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, Jiarui Yuan, Huayu Chen, Kaiyan Zhang, Xingtai Lv, Shuo Wang, Yuan Yao, Xu Han, Hao Peng, Yu Cheng, Zhiyuan Liu, Maosong Sun, Bowen Zhou, and Ning Ding. Process reinforcement through implicit rewards, 2025. URL <https://arxiv.org/abs/2502.01456>.
- [23] Jeff Da, Clinton Wang, Xiang Deng, Yuntao Ma, Nikhil Barhate, and Sean Hendryx. Agent-rlvr: Training software engineering agents via guidance and environment rewards. *arXiv preprint arXiv:2506.11425*, 2025.
- [24] Jeff Da, Clinton J. Wang, Xiang Deng, Yuntao Ma, Nikhil Barhate, and Sean M. Hendryx. Agent-rlvr: Training software engineering agents via guidance and environment rewards. *ArXiv*, abs/2506.11425, 2025. URL <https://api.semanticscholar.org/CorpusID:279391657>.
- [25] Edoardo DeBenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL <https://openreview.net/forum?id=m1YYAQj03w>.

- [26] Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-verification reduces hallucination in large language models. In *Findings of the association for computational linguistics: ACL 2024*, pages 3563–3578, 2024.
- [27] Jingtao Ding, Yunke Zhang, Yu Shang, Yuheng Zhang, Zefang Zong, Jie Feng, Yuan Yuan, Hongyuan Su, Nian Li, Nicholas Sukiennik, et al. Understanding world or predicting future? a comprehensive survey of world models. *ACM Computing Surveys*, 58(3):1–38, 2025.
- [28] Zihan Ding, Amy Zhang, Yuandong Tian, and Qinqing Zheng. Diffusion world model: Future modeling beyond step-by-step rollout for offline reinforcement learning. *arXiv preprint arXiv:2402.03570*, 2024.
- [29] Yihong Dong, Xue Jiang, Jiaru Qian, Tian Wang, Kechi Zhang, Zhi Jin, and Ge Li. A survey on code generation with llm-based agents. *arXiv preprint arXiv:2508.00083*, 2025.
- [30] Yonggan Fu, Lexington Whalen, Zhifan Ye, Xin Dong, Shizhe Diao, Jingyu Liu, Chengyue Wu, Hao Zhang, Enze Xie, Song Han, Maksim Khadkevich, Jan Kautz, Yingyan Celine Lin, and Pavlo Molchanov. Efficient-dlm: From autoregressive to diffusion language models, and beyond in speed, 2025. URL <https://arxiv.org/abs/2512.14067>.
- [31] Chaochen Gao, Xing Wu, Zijia Lin, Debing Zhang, and Songlin Hu. Nextlong: Toward effective long-context training without long documents. *arXiv preprint arXiv:2501.12766*, 2025.
- [32] Shenyuan Gao, Jiazhi Yang, Li Chen, Kashyap Chitta, Yihang Qiu, Andreas Geiger, Jun Zhang, and Hongyang Li. Vista: A generalizable driving world model with high fidelity and versatile controllability. *Advances in Neural Information Processing Systems*, 37:91560–91596, 2024.
- [33] Tao Ge, Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. Scaling synthetic data creation with 1,000,000,000 personas. *arXiv preprint arXiv:2406.20094*, 2024.
- [34] Anna Goldie, Azalia Mirhoseini, Hao Zhou, Irene Cai, and Christopher D Manning. Synthetic data generation & multi-step rl for reasoning & tool use. *arXiv preprint arXiv:2504.04736*, 2025.
- [35] Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, Hao Peng, and Lingpeng Kong. Scaling diffusion language models via adaptation from autoregressive models, 2025. URL <https://arxiv.org/abs/2410.17891>.
- [36] Shansan Gong, Ruixiang Zhang, Huangjie Zheng, Jiatao Gu, Navdeep Jaitly, Lingpeng Kong, and Yizhe Zhang. Diffucoder: Understanding and improving masked diffusion models for code generation. *arXiv preprint arXiv:2506.20639*, 2025.
- [37] Nikita Gupta, Riju Chatterjee, Lukas Haas, Connie Tao, Andrew Wang, Chang Liu, Hidekazu Oiwa, Elena Gribovskaya, Jan Ackermann, John Blitzer, et al. Deepsearchqa: Bridging the comprehensiveness gap for deep research agents. *arXiv preprint arXiv:2601.20975*, 2026.
- [38] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.
- [39] Andre He, Nathaniel Weir, Kaj Bostrom, Allen Nie, Darion Cassel, Sam Bayless, and Huzefa Rangwala. Resyn: Autonomously scaling synthetic environments for reasoning models, 2026. URL <https://arxiv.org/abs/2602.20117>.
- [40] Haoran He, Yang Zhang, Liang Lin, Zhongwen Xu, and Ling Pan. Pre-trained video generative models as world simulators. *arXiv preprint arXiv:2502.07825*, 2025.
- [41] Linda He, Jue Wang, Maurice Weber, Shang Zhu, Ben Athiwaratkun, and Ce Zhang. Scaling instruction-tuned llms to million-token contexts via hierarchical synthetic data generation. *arXiv preprint arXiv:2504.12637*, 2025.

- [42] Lukas Helff, Quentin Delfosse, David Steinmann, Ruben Härle, Hikaru Shindo, Patrick Schramowski, Wolfgang Stammer, Kristian Kersting, and Felix Friedrich. Llms gaming verifiers: Rlvr can lead to reward hacking, 2026. URL <https://arxiv.org/abs/2604.15149>.
- [43] Xiaomeng Hu, Yinger Zhang, Fei Huang, Jianhong Tu, Yang Su, Lianghao Deng, Yuxuan Liu, Yantao Liu, Dayiheng Liu, and Tsung-Yi Ho. Occubench: Evaluating ai agents on real-world professional tasks via language world models. *arXiv preprint arXiv:2604.10866*, 2026.
- [44] Xueyu Hu, Tao Xiong, Biao Yi, Zishu Wei, Ruixuan Xiao, Yurun Chen, Jiasheng Ye, Meiling Tao, Xiangxin Zhou, Ziyu Zhao, Yuhuai Li, Shengze Xu, Shenzhi Wang, Xinchun Xu, Shuofei Qiao, Zhaokai Wang, Kun Kuang, Tiejong Zeng, Liang Wang, Jiwei Li, Yuchen Eleanor Jiang, Wangchunshu Zhou, Guoyin Wang, Keting Yin, Zhou Zhao, Hongxia Yang, Fan Wu, Shengyu Zhang, and Fei Wu. OS agents: A survey on MLLM-based agents for computer, phone and browser use. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7436–7465, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.369. URL <https://aclanthology.org/2025.acl-long.369/>.
- [45] Albert Q Jiang, A Sablayrolles, A Mensch, C Bamford, D Singh Chaplot, Ddl Casas, F Bressand, G Lengyel, G Lample, L Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 10:3, 2023.
- [46] Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*, 2023.
- [47] Sung-Hyun Kim, In-Chang Baek, Seo-Young Lee, Geum-Hwan Hwang, and Kyung-Joong Kim. Multi-objective instruction-aware representation learning in procedural content generation rl. *arXiv preprint arXiv:2508.09193*, 2025.
- [48] Klaus Krippendorff. Computing krippendorff’s alpha-reliability. 2011.
- [49] Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding, 2023. URL <https://arxiv.org/abs/2211.17192>, 1(2), 2022.
- [50] Jiajian Li, Qi Wang, Yunbo Wang, Xin Jin, Yang Li, Wenjun Zeng, and Xiaokang Yang. Open-world reinforcement learning over long short-term imagination. *arXiv preprint arXiv:2410.03618*, 2024.
- [51] Jiayi Li, Xingxing Zhang, Xun Wang, Xiaolong Huang, Li Dong, Liang Wang, Si-Qing Chen, Wei Lu, and Furu Wei. Wildlong: Synthesizing realistic long-context instruction data at scale. *arXiv preprint arXiv:2502.16684*, 2025.
- [52] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. *corr abs/1510.03055. arXiv preprint arXiv:1510.03055*, 2015.
- [53] Junlong Li, Wenshuo Zhao, Jian Zhao, Weihao Zeng, Haoze Wu, Xiaochen Wang, Rui Ge, Yuxuan Cao, Yuzhen Huang, Wei Liu, Junteng Liu, Zhaochen Su, Yiyang Guo, Fan Zhou, Lueyang Zhang, Juan Michelini, Xingyao Wang, Xiang Yue, Shuyan Zhou, Graham Neubig, and Junxian He. The tool decathlon: Benchmarking language agents for diverse, realistic, and long-horizon task execution. 2025. URL <https://arxiv.org/abs/2510.25726>.
- [54] Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. Api-bank: A comprehensive benchmark for tool-augmented llms. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 3102–3116, 2023.
- [55] Yixia Li, Hongru Wang, Jiahao Qiu, Zhenfei Yin, Dongdong Zhang, Cheng Qian, Zeping Li, Pony Ma, Guanhua Chen, Heng Ji, et al. From word to world: Can large language models be implicit text-based world models? *arXiv preprint arXiv:2512.18832*, 2025.

- [56] Zhuofeng Li, Dongfu Jiang, Xueguang Ma, Haoxiang Zhang, Ping Nie, Yuyu Zhang, Kai Zou, Jianwen Xie, Yu Zhang, and Wenhua Chen. Openresearcher: A fully open pipeline for long-horizon deep research trajectory synthesis. *arXiv preprint arXiv:2603.20278*, 2026.
- [57] William Liang, Sam Wang, Hung-Ju Wang, Osbert Bastani, Dinesh Jayaraman, and Yecheng Jason Ma. EurekaVerse: Environment curriculum generation via large language models. *arXiv preprint arXiv:2411.01775*, 2024.
- [58] Aiwei Liu, Minghua He, Shaoxun Zeng, Sijun Zhang, Linhao Zhang, Chuhan Wu, Wei Jia, Yuan Liu, Xiao Zhou, and Jie Zhou. Wedlm: Reconciling diffusion language models with standard causal attention for fast inference. *arXiv preprint arXiv:2512.22737*, 2025.
- [59] Zuxin Liu, Thai Hoang, Jianguo Zhang, Ming Zhu, Tian Lan, Shirley Kokane, Juntao Tan, Weiran Yao, Zhiwei Liu, Yihao Feng, et al. Apigen: Automated pipeline for generating verifiable and diverse function-calling datasets. *Advances in Neural Information Processing Systems*, 37:54463–54482, 2024.
- [60] Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. On llms-driven synthetic data generation, curation, and evaluation: A survey. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11065–11082, 2024.
- [61] Alisia Lupidi, Carlos Gemmell, Nicola Cancedda, Jane Dwivedi-Yu, Jason Weston, Jakob Foerster, Roberta Raileanu, and Maria Lomeli. Source2synth: Synthetic data generation and curation grounded in real data sources. *arXiv preprint arXiv:2409.08239*, 2024.
- [62] Yuanjie Lyu, Chengyu Wang, Lei Shen, Jun Huang, and Tong Xu. Mock worlds, real skills: Building small agentic language models with synthetic tasks, simulated environments, and rubric-based rewards, 2026. URL <https://arxiv.org/abs/2601.22511>.
- [63] Pratyush Maini, Skyler Seto, Richard Bai, David Grangier, Yizhe Zhang, and Navdeep Jaitly. Rephrasing the web: A recipe for compute and data-efficient language modeling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14044–14072, 2024.
- [64] Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J Pal, and Liam Paull. Active domain randomization. In *Conference on Robot Learning*, pages 1162–1176. PMLR, 2020.
- [65] Mike A Merrill, Alexander G Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E Kelly Buchanan, et al. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. *arXiv preprint arXiv:2601.11868*, 2026.
- [66] Arindam Mitra, Luciano Del Corro, Guoqing Zheng, Shweti Mahajan, Dany Rouhana, Andres Coda, Yadong Lu, Wei-ge Chen, Olga Vrousos, Corby Rosset, et al. Agentinstruct: Toward generative teaching with agentic flows. *arXiv preprint arXiv:2407.03502*, 2024.
- [67] Ehsan Montahaei, Danial Alihosseini, and Mahdieh Soleymani Baghshah. Jointly measuring diversity and quality in text generation models, 2019. URL <https://arxiv.org/abs/1904.03971>.
- [68] M Nadas, L Diosan, and A Tomescu. Synthetic data generation using large language models: Advances in text and code. *arxiv* 2025. *arXiv preprint arXiv:2503.14023*, 2025.
- [69] Shen Nie, Fengqi Zhu, Chao Du, Tianyu Pang, Qian Liu, Guangtao Zeng, Min Lin, and Chongxuan Li. Scaling up masked diffusion models on text, 2025. URL <https://arxiv.org/abs/2410.18514>.
- [70] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models, 2025. URL <https://arxiv.org/abs/2502.09992>.
- [71] NVIDIA. Nemotron 3 Nano: Open, efficient mixture-of-experts hybrid Mamba-Transformer model for Agentic reasoning, 2025. URL <https://arxiv.org/abs/2512.20848>. Technical report.

- [72] OpenAI. gpt-oss-120b & gpt-oss-20b model card, 2025. URL <https://arxiv.org/abs/2508.10925>.
- [73] Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. *arXiv preprint arXiv:2406.03736*, 2024.
- [74] Sindhu Padakandla. A survey of reinforcement learning algorithms for dynamically varying environments. *ACM Computing Surveys (CSUR)*, 54(6):1–25, 2021.
- [75] Shishir G. Patil, Huanzhi Mao, Charlie Cheng-Jie Ji, Fanjia Yan, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*, 2025.
- [76] Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34: 4816–4828, 2021.
- [77] Krishna Pillutla, Lang Liu, John Thickstun, Sean Welleck, Swabha Swayamdipta, Rowan Zellers, Sewoong Oh, Yejin Choi, and Zaid Harchaoui. MAUVE Scores for Generative Models: Theory and Practice. *JMLR*, 2023.
- [78] Akshara Prabhakar, Zuxin Liu, Ming Zhu, Jianguo Zhang, Tulika Awalgaonkar, Shiyu Wang, Zhiwei Liu, Haolin Chen, Thai Hoang, Juan Carlos Niebles, et al. Apigen-mt: Agentic pipeline for multi-turn data generation via simulated agent-human interplay. *arXiv preprint arXiv:2504.03601*, 2025.
- [79] Christopher Rawles, Sarah Clinckemallie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyi Campbell-Ajala, Daniel Toyama, Robert Berry, Divya Tyamagundlu, Timothy Lillicrap, and Oriana Riva. Androidworld: A dynamic benchmarking environment for autonomous agents, 2024. URL <https://arxiv.org/abs/2405.14573>.
- [80] Haris Riaz, Sourav Sanjukta Bhabesh, Vinayak Arannil, Miguel Ballesteros, and Graham Horwood. Metasynth: Meta-prompting-driven agentic scaffolds for diverse synthetic data generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, page 18770–18803. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.962. URL <http://dx.doi.org/10.18653/v1/2025.findings-acl.962>.
- [81] Subham S Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.
- [82] Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models, 2024. URL <https://arxiv.org/abs/2406.07524>.
- [83] Kuniaki Saito, Kihyuk Sohn, Xiang Zhang, Chun-Liang Li, Chen-Yu Lee, Kate Saenko, and Tomas Pfister. Prefix conditioning unifies language and label supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2861–2870, 2023.
- [84] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- [85] Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data, 2025. URL <https://arxiv.org/abs/2406.04329>.

- [86] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. URL <https://arxiv.org/abs/1912.01734>.
- [87] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. URL <https://arxiv.org/abs/2010.03768>.
- [88] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.
- [89] Xiaoshuai Song, Haofei Chang, Guanting Dong, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Envscaler: Scaling tool-interactive environments for llm agent via programmatic synthesis. *arXiv preprint arXiv:2601.05808*, 2026.
- [90] Dan Su, Kezhi Kong, Ying Lin, Joseph Jennings, Brandon Norick, Markus Kliegl, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Nemotron-cc: Transforming common crawl into a refined long-horizon pretraining dataset. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2459–2475, 2025.
- [91] Shivchander Sudalairaj, Abhishek Bhandwaldar, Aldo Pareja, Kai Xu, David D Cox, and Akash Srivastava. Lab: Large-scale alignment for chatbots. *arXiv preprint arXiv:2403.01081*, 2024.
- [92] GLM Team, Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, Kedong Wang, Lucen Zhong, Mingdao Liu, Rui Lu, Shulin Cao, Xiaohan Zhang, Xuancheng Huang, Yao Wei, Yean Cheng, Yifan An, Yilin Niu, Yuanhao Wen, Yushi Bai, Zhengxiao Du, Zihan Wang, Zilin Zhu, Bohan Zhang, Bosi Wen, Bowen Wu, Bowen Xu, Can Huang, Casey Zhao, Changpeng Cai, Chao Yu, Chen Li, Chendi Ge, Chenghua Huang, Chenhui Zhang, Chenxi Xu, Chenzheng Zhu, Chuang Li, Congfeng Yin, Daoyan Lin, Dayong Yang, Dazhi Jiang, Ding Ai, Erle Zhu, Fei Wang, Gengzheng Pan, Guo Wang, Hailong Sun, Haitao Li, Haiyang Li, Haiyi Hu, Hanyu Zhang, Hao Peng, Hao Tai, Haoke Zhang, Haoran Wang, Haoyu Yang, He Liu, He Zhao, Hongwei Liu, Hongxi Yan, Huan Liu, Huilong Chen, Ji Li, Jiajing Zhao, Jiamin Ren, Jian Jiao, Jiani Zhao, Jianyang Yan, Jiaqi Wang, Jiayi Gui, Jiayue Zhao, Jie Liu, Jijie Li, Jing Li, Jing Lu, Jingsen Wang, Jingwei Yuan, Jingxuan Li, Jingzhao Du, Jinhua Du, Jinxin Liu, Junkai Zhi, Junli Gao, Ke Wang, Lekang Yang, Liang Xu, Lin Fan, Lindong Wu, Lintao Ding, Lu Wang, Man Zhang, Minghao Li, Minghuan Xu, Mingming Zhao, Mingshu Zhai, Pengfan Du, Qian Dong, Shangde Lei, Shangqing Tu, Shangtong Yang, Shaoyou Lu, Shijie Li, Shuang Li, Shuang-Li, Shuxun Yang, Sibao Yi, Tianshu Yu, Wei Tian, Weihang Wang, Wenbo Yu, Weng Lam Tam, Wenjie Liang, Wentao Liu, Xiao Wang, Xiaohan Jia, Xiaotao Gu, Xiaoying Ling, Xin Wang, Xing Fan, Xingru Pan, Xinyuan Zhang, Xinze Zhang, Xiuqing Fu, Xunkai Zhang, Yabo Xu, Yandong Wu, Yida Lu, Yidong Wang, Yilin Zhou, Yiming Pan, Ying Zhang, Yingli Wang, Yingru Li, Yinpei Su, Yipeng Geng, Yitong Zhu, Yongkun Yang, Yuhang Li, Yuhao Wu, Yujiang Li, Yunan Liu, Yunqing Wang, Yuntao Li, Yuxuan Zhang, Zezhen Liu, Zhen Yang, Zhengda Zhou, Zhongpei Qiao, Zhuoer Feng, Zhuorui Liu, Zichen Zhang, Zihan Wang, Zijun Yao, Zikang Wang, Ziqiang Liu, Ziwei Chai, Zixuan Li, Zuodong Zhao, Wenguang Chen, Jidong Zhai, Bin Xu, Minlie Huang, Hongning Wang, Juanzi Li, Yuxiao Dong, and Jie Tang. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models, 2025. URL <https://arxiv.org/abs/2508.06471>.
- [93] Qwen Team. Qwen3.5: Accelerating productivity with native multimodal agents, February 2026. URL <https://qwen.ai/blog?id=qwen3.5>.
- [94] Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In *ACL*, 2024.

- [95] Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. *arXiv preprint arXiv:2408.14837*, 2024.
- [96] Caio Vicentino. Autoregressive vs. masked diffusion language models: A controlled comparison. *arXiv preprint arXiv:2603.22075*, 2026.
- [97] Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. Science-world: Is your agent smarter than a 5th grader? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11279–11298, 2022.
- [98] Yiming Wang, Da Yin, Yuedong Cui, Ruichen Zheng, Zhiqian Li, Zongyu Lin, Di Wu, Xueqing Wu, Chenchen Ye, Yu Zhou, and Kai-Wei Chang. Llms as scalable, general-purpose simulators for evolving digital agent training, 2025. URL <https://arxiv.org/abs/2510.14969>.
- [99] Zhaoyang Wang, Canwen Xu, Boyi Liu, Yite Wang, Siwei Han, Zhewei Yao, Huaxiu Yao, and Yuxiong He. Agent world model: Infinity synthetic environments for agentic reinforcement learning, 2026. URL <https://arxiv.org/abs/2602.10090>.
- [100] Zhen Wang, Yufan Zhou, Zhongyan Luo, Lyumanshan Ye, Adam Wood, Man Yao, Saab Mansour, and Luoshang Pan. Deeppersona: A generative engine for scaling deep synthetic personas. *arXiv preprint arXiv:2511.07338*, 2025.
- [101] 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*, 35:24824–24837, 2022.
- [102] Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents, 2025. URL <https://arxiv.org/abs/2504.12516>.
- [103] Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye, Zhirong Wu, Yang Wang, Zhijian Xu, Xiao Liang, Junjie Li, Ziming Miao, et al. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms. *arXiv preprint arXiv:2506.14245*, 2025.
- [104] Jialong Wu, Shaofeng Yin, Ningya Feng, and Mingsheng Long. Rlvr-world: Training world models with reinforcement learning. *arXiv preprint arXiv:2505.13934*, 2025.
- [105] Jiangxu Wu, Cong Wang, TianHuang Su, Lin Haozhi, JunYang JunYang, Zhangchao Zhangchao, Binqiang Pan, SongpanYang SongpanYang, Mingpeng Mingpeng, Kai Shi, et al. Instruct: A review-driven multi-turn conversations generation method for large language models. *Findings of the Association for Computational Linguistics: ACL 2025*, pages 16578–16595, 2025.
- [106] Tong Wu, Zhihao Fan, Xiao Liu, Hai-Tao Zheng, Yeyun Gong, Jian Jiao, Juntao Li, Jian Guo, Nan Duan, Weizhu Chen, et al. Ar-diffusion: Auto-regressive diffusion model for text generation. *Advances in Neural Information Processing Systems*, 36:39957–39974, 2023.
- [107] Yihong Wu, Liheng Ma, Lei Ding, Muzhi Li, Xinyu Wang, Kejia Chen, Zhan Su, Zhanguang Zhang, Chenyang Huang, Yingxue Zhang, et al. It takes two: Your grpo is secretly dpo. *arXiv preprint arXiv:2510.00977*, 2025.
- [108] Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Xin Guo, Dingwen Yang, Chenyang Liao, Wei He, et al. Agentgym: Evaluating and training large language model-based agents across diverse environments. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 27914–27961, 2025.
- [109] Zhiheng Xi, Xin Guo, Jiaqi Liu, Jiazheng Zhang, Yutao Fan, Zhihao Zhang, Shichun Liu, Mingxu Chai, Xiaowei Shi, Yitao Zhai, et al. Can rl improve generalization of llm agents? an empirical study. *arXiv preprint arXiv:2603.12011*, 2026.

- [110] Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments, 2024.
- [111] Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P Lillicrap, Kenji Kawaguchi, and Michael Shieh. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*, 2024.
- [112] Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. *arXiv preprint arXiv:2509.02479*, 2025.
- [113] Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*, 2025.
- [114] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.
- [115] John Yang, Akshara Prabhakar, Karthik Narasimhan, and Shunyu Yao. Intercode: Standardizing and benchmarking interactive coding with execution feedback, 2023.
- [116] John Yang, Kilian Lieret, Carlos E. Jimenez, Alexander Wettig, Kabir Khandpur, Yanzhe Zhang, Binyuan Hui, Ofir Press, Ludwig Schmidt, and Diyi Yang. Swe-smith: Scaling data for software engineering agents, 2025. URL <https://arxiv.org/abs/2504.21798>.
- [117] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In *ArXiv*, preprint.
- [118] Tao Zewei and Huang Yunpeng. Magiattention: A distributed attention towards linear scalability for ultra-long context, heterogeneous mask training. <https://github.com/SandAI-org/MagiAttention/>, 2025.
- [119] Guibin Zhang, Hejia Geng, Xiaohan Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhong-Zhi Li, Xiangyuan Xue, Yijiang Li, Yifan Zhou, Yang Chen, Chen Zhang, Yutao Fan, Zihu Wang, Songtao Huang, Yue Liao, Hongru Wang, Meng Yang, Heng Ji, Michael Littman, Jun Wang, Shuicheng Yan, Philip Torr, and Lei Bai. The landscape of agentic reinforcement learning for llms: A survey. *ArXiv*, abs/2509.02547, 2025. URL <https://api.semanticscholar.org/CorpusID:281080233>.
- [120] Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. Swift:a scalable lightweight infrastructure for fine-tuning, 2024. URL <https://arxiv.org/abs/2408.05517>.
- [121] Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023.
- [122] Chiwei Zhu, Benfeng Xu, Xiaorui Wang, and Zhendong Mao. From real to synthetic: Synthesizing millions of diversified and complicated user instructions with attributed grounding. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10516–10543, 2025.

Appendix

A Human Evaluations

To verify that our gains come from the masked diffusion process itself rather than simply from access to more data, we conduct a thorough human evaluation of the MDLM world model’s outputs.

Table 5: Annotation rubrics for the three evaluation dimensions.

Score	Label	Description
<i>Training utility</i>		
5	Excellent	Correct and consistent outcome, realistic values, i.e. ideal training signal
4	Good	Minor value errors, but correct outcome type
3	Mediocre	Correct outcome type, but wrong schema or implausible values
2	Poor	Wrong outcome type or severely hallucinated structure
1	Useless	Completely wrong, nonsensical, or would teach incorrect behaviours, including incomplete outputs
<i>Realism</i>		
5	Indistinguishable	Looks exactly like a real API response
4	Mostly realistic	Minor inconsistencies (e.g. slightly wrong field names, but still valid)
3	Partially realistic	Notable hallucinations like invented fields that don't fit the domain
2	Implausible	Schema or values are implausible for this domain
1	Completely unrealistic	Random or nonsensical data
<i>Outcome correctness</i>		
5	Exactly correct	Correct outcome type and all key facts
4	Mostly correct	Correct outcome type with minor factual errors
3	Partially correct	Correct outcome type but significantly wrong facts
2	Wrong outcome	Wrong outcome type (e.g. predicted success when an error was expected)
1	Completely incorrect	Entirely wrong

Specifically, we study three independent metrics: realism, outcome correctness and training utility. We ensure that no harmful or sensitive content is present during the human evaluation process and that the 100 samples are thoroughly author validated for safety. Below are the complete human evaluation guidelines:

Annotation Guidelines

Annotator Requirements: Before beginning, please confirm you meet all three of the following requirements: Technical background: You are familiar with tool-calling LLMs, including tool schemas and how an environment responds to a given action.

Education and experience:

1. You hold a CS degree and have experience both using and building with AI agents.
2. Age: You are at least 18 years old.

Qualification Task (Complete First): Before accessing the full dataset, you must complete a short qualification task. This is used to evaluate annotation quality and ensure consistency across annotators. The qualification task consists of 5 sample items. These will be provided to you by the project coordinator before you receive the full dataset. Complete the qualification samples independently and without rushing. You must treat them exactly as you would the real task. Do not use any kind of AI for completing the task. AI model capabilities are not reliable for this task. Submit your qualification responses in the "Qualifying Tasks" tab of your assigned Google Sheet (see Submission Instructions below). The project coordinator will review your responses and confirm whether you are cleared to proceed.

Evaluating Predicted Next States (100 conversations):

What you'll see Each item presents a JSON formatted conversation. At the end of every conversation, there is an "action". You must rate the next state that is predicted after the

action. What you’ll do Rate the Predicted Next State on three separate dimensions. Each dimension is rated on a scale of 1–5. Assign scores independently. A response can score differently across dimensions. You must make your ratings independent of the expected output since tool outputs can produce different states (for example, sometimes a tool errors out, sometimes returns correct responses)

Dimension 1: *training_utility*

Would this prediction be useful for training a model? Think: "If I were an AI model and I saw this output from a tool I called, would the output be useful to me to proceed normally and correctly with the task?"

Dimension 2: *realism*

Does the predicted response look like a real API response for this domain?

Dimension 3: *outcome correctness*

Does the model correctly predict the outcome of the action?

[See [Table 5](#) for exact rubric breakdown for Likert scale.]

General Guidance: Be consistent. Apply the same standards across all items. Consider each dimension independently and don’t let one score influence another. Focus on validity of the tool output and its possibility to appear in a realistic situation If something looks hallucinated but happens to be plausible for the domain, use your domain judgment to calibrate.

As observed in [Table 4](#), the mean scores across four annotators remains high at 4.75, 4.25 and 4.5 for realism, outcome correctness and training utility respectively. We utilize Krippendorff’s alpha [48] to calculate interannotator agreement and reliability and we observe a high degree of consistency between the annotators. Common issues surfaced by annotators covered incorrect data type of numeric fields (e.g. string vs integer or float), corrupted API keys in tool calls or ungrounded and unrealistic websearch content. One instance flagged by two annotators was the MDLM’s tendencies to collapse into repeated tool errors once any arbitrary turn returns an error.

B Fairness of Evaluation

Since world model outputs are non-deterministic and API responses are generally in JSON format, the keys of such JSON mappings are seldom ordered which can lead to deflation of precision-based BLEU and ROUGE scores. To avoid this bias, we apply the following preprocessing steps to ensure fairness of evaluation.

1. We canonicalize JSON outputs to eliminate spurious mismatches: keys are sorted alphabetically in ascending order, all whitespace is stripped from both predicted and reference outputs, and stringified numeric entries are normalized.
2. For API-BANK, the schema specifies `input`, `output`, `exception`, and `api_name`. Since `input` is shared across models and does not reflect output quality, we omit it from evaluation.
3. We use the GPU implementation of MAUVE with $k = \max(2, \text{round}(\min(|p|, |q|)/10))$, following the auto-sizing recommendation from the original MAUVE paper. This ensures consistency with prior evaluations conducted under the implementation of [Pillutla et al. \(2023\)](#). We use the default GPT-2 implementation proposed by the authors for this task.

In addition to this, specifically for AR models, we find out higher temperatures lead to hallucination and more verbose outputs. To balance the verbosity-correctness ratio we evaluate across temperature = {0.5, 0.7, 0.9} and find that 0.5 achieves the best balance. With temperatures closer to 0, MAUVE scores drop by up to 0.2 points consistently which aligns with reduced variance in greedy decoding token selections.

C World Model Training Dataset Curation Details

Since no existing world modeling datasets exist as of the writing of this paper, to help compare causal and masked diffusion language models, we attempt to create a broad coverage dataset of real environment rollouts, capturing all unique state-actions pairs. We describe our complete dataset curation, filtering and analysis procedure below.

Trajectory Generation. For every environment selected, we setup the original environment as designed by the authors and generate complete rollouts with a variety of models. To ensure diversity of trajectories and broad behavioral coverage, we sample a trajectory-generating model uniformly at random from the following set: Qwen3.5-4B, Qwen3.5-9B, Qwen3.5-397B-A17B-FP8, GPT-5.4-mini, GPT-5.4, GPT-OSS-120B, Claude-4.6-Sonnet, Claude-4.5-Haiku, Gemini-3-Pro, MiniMax-M2.1, GLM-5, and DeepSeek-V3.2.⁷ Sampling across this diverse set of frontier models introduces natural variance in reasoning style, verbosity, tool-calling patterns, and error recovery behavior. We select the model with a starting seed of 42 for reproducibility.

Trajectory Length Diversification. Since steering of a world model relies heavily on the present state-action pair, trajectory lengths are important to diversify for natural rollout behaviors. To introduce trajectory length variance and prevent the model from overfitting to fixed-length contexts, we apply two complementary processing techniques.

Middle Truncation. For conversational histories exceeding 16,384 tokens (approximately 2% of the full dataset), we apply middle truncation to preserve realistic long-horizon workflows while respecting context length constraints. Specifically, we retain the original system prompt and user prompt alongside the final N exchanges, where $N \sim \mathcal{U}(0, N_{\max})$ and

$$N_{\max} = \left\lfloor \frac{T_{\max} - T_{\text{sys}} - T_{\text{user}}}{\bar{t}} \right\rfloor, \quad (3)$$

where $T_{\max} = 16,384$ is the context length threshold, T_{sys} and T_{user} are the token counts of the system and user prompts respectively, and $\bar{t}$ is the mean token length per exchange computed over the training corpus. When $N = 0$, only the system and user prompts are retained, representing the most aggressively truncated case and ensuring the model learns to predict environment states from minimal conversational context. This is a practically important capability for world models operating under constrained inference budgets. The removed portion of text is replaced with the placeholder token `[TRUNCATED {N_CHARS} characters...]`, which teaches the model to parse and process partially observable long trajectories that may require truncation at training or inference time.

Sub-trajectory Extraction. For the remaining trajectories, we randomly select a tool call boundary and truncate the trajectory at that point, treating the subsequent environment state as the ground truth prediction target. All utterances following the selected tool call are removed. We apply this transformation with a random 15% probability to prevent unnecessary truncation of the dataset which ensured better long horizon understanding in our experiments. This naturally exposes the model to a wide range of trajectory prefixes and reinforces robust prediction at varying horizon lengths.

Environment State Grounding. To enable better world model steering, we use Claude-4.6-Sonnet to synthesize additional ground truth environment context for each trajectory. This context includes extra database states, expected task reward structure, behavior patterns, and tool schemas and definitions. Concretely, we prompt Claude-4.6-Sonnet in high reasoning mode to analyze the full trajectory and generate instructional hindsight in the form of a structured description of what environmental information would have been sufficient to predict the final state, conditioned on the observed intermediate states. To promote realism and prevent overfitting to templated outputs, we deliberately include contextually adjacent but non-essential information in the synthesized context, such as neighboring database columns and irrelevant rows. This trains the model to selectively attend to relevant portions of large environment state dumps, reflecting real rollout conditions where custom prompt engineering is infeasible and ground truth database states are too large. We apply instruction augmentation to approximately 80% of the dataset to preserve the model’s general approximative capacity and reduce overfitting to hindsight instructions. After filtering, post-processing, and truncation, unaugmented

⁷All open-source model trajectories were generated using the Fireworks AI API. All closed models were queried via their respective official API endpoints.

instances constitute 17.2% of the final training dataset. To understand if the augmentations are necessary and do not produce unnecessary noise, we perform a minimalistic human evaluation. We select two independent annotators and task them to evaluate the trajectories in isolation and again with the grounding and steering objectives and noticed a high approval rate of 87% for instances with grounding and steering instructions. Most disapproval cases flagged by the annotators belonged to deterministic prediction such as re-executing a terminal command without changes or formulaic tool calls. On the other hand, most agreements were seen in non-deterministic environments such as deep research or customer service tasks. The prompt used for this grounding is given below([section C](#)).

HINDSIGHT INSTRUCTION SYNTHESIS PROMPT

ROLE AND OBJECTIVE

You are an expert environment annotator for training world-model agents on tool-use trajectories. Given a trajectory state and the next agent action, synthesize a "hindsight instruction": structured ground-truth environment context, tool schemas, and behavioral rules that would have been sufficient to predict the next environment observation, conditioned on the observed intermediate states. These instructions teach the model to ground predictions in environment mechanisms rather than templated heuristics, and to selectively attend to relevant portions of large environment state dumps.

Reason carefully and silently before writing the final instruction.

DOMAIN TAXONOMY

Decide which of the following four domains best matches the trajectory, then apply the corresponding section template. If ambiguous, default to "Tool Use".

- SWE : code/repo work, terminal/editor tools, tests, builds.
Examples: str_replace_editor, bash, pytest contexts.
- Tool Use : (near-)stateless single-tool API calls.
Examples: calculator, calculate_age, format converters, weather lookups with no persistent backing store.
- Customer Service : multi-turn agent operating under a written policy with user/account/profile state and back-office tools.
Examples: telecom support, retail returns, banking.
- Deep Research : open-web research with search and browsing tools; the environment is the unbounded web rather than a fixed DB.
Examples: web_search + read_webpage trajectories.

INTERNAL REASONING CHECKLIST

Before writing, work through:

1. Domain classification (one of the four above).
2. Which environment(s) and toolset(s) are active.
3. What state is observable from the conversation; what is implied but hidden.
4. Which tool the next action invokes; expected parameters and return shape.
5. The causal mechanism that determines the response (DB lookup, parameter validation, file slice, web fetch, policy gating).
6. Distractor / non-essential context to inject (irrelevant rows, sibling tools, unused credentials, neighboring file paths).
7. Minimal context that closes the prediction gap.

INPUT

You will receive a record with the following fields:

- 'state' : full conversation up to this point: system prompt, tool catalog, prior assistant/tool turns.
- 'action' : the next agent action (one or more tool calls).
- 'outcome' : (optional) the actual environment/tool response, used only to determine relevance, never quoted verbatim.

SECTION INVENTORY (canonical order)

Include sections in this exact order, using Markdown level-2 headers ('##'). Never include preamble or postscript.

1. ## ENVIRONMENT STATE -- CONDITIONAL
2. ## TASK CONTEXT -- REQUIRED
3. ## TOOL SCHEMAS -- REQUIRED
4. ## DOMAIN RULES -- CONDITIONAL
5. ## STEERING DIRECTIVES -- CONDITIONAL
6. PREDICTION TARGET: <...> -- REQUIRED (single final line, no '##')

Conditional sections are included only when informative. The PREDICTION TARGET line is ALWAYS the final line of the output.

CONDITIONAL-SECTION INCLUSION RULES

ENVIRONMENT STATE

- INCLUDE when stateful backing data exists: DB rows, files, customer profiles, repo state, prior tool returns, session/auth flags.
- OMIT for Deep Research (open web; no fixed state).
- For Tool Use: include only the input record / derived state, kept short.

DOMAIN RULES

- INCLUDE when there are policies, conventions, mechanisms, or constraints beyond the bare schema: telecom policy, repo conventions, research methodology, account/line/bill semantics, validation rules.
- OMIT for trivial single-tool calls where the schema is self-explanatory.

STEERING DIRECTIVES

- INCLUDE when reasoning hazards exist: tool-name confusion, parameter validation, mechanism-vs-outcome distinction, evidence quality.
- OMIT for trivial cases.

SECTION PROFILE BY DOMAIN

Section	SWE	Tool Use	Customer Service	Deep Research
ENVIRONMENT STATE	yes	yes (mini)	yes (data dump)	no
TASK CONTEXT	yes	yes	yes	yes
TOOL SCHEMAS	yes	yes	yes	yes
DOMAIN RULES	yes	usually no	yes	yes
STEERING DIRECTIVES	yes	usually no	yes	yes
PREDICTION TARGET	yes	yes	yes	yes

DOMAIN-SPECIFIC GUIDANCE

SWE

- ENVIRONMENT STATE: repository root path, visible file paths from prior turns, note that the rest of the tree is discoverable.
- TASK CONTEXT: high-level repo/PR/feature objective; tests-vs-source modification policy.
- TOOL SCHEMAS: editor/shell commands with sub-commands and parameter validity (e.g., 'view_range' only valid on files, not directories).
- DOMAIN RULES: language, test runner, "do not modify tests", project conventions, what the relevant module does.
- STEERING DIRECTIVES: file-slice semantics; navigation patterns; tool errors are determined by command/path validity, not task intent.

Tool Use

- ENVIRONMENT STATE: keep to a small block: input record + any parsed/derived state. No global DB dumps.
- TASK CONTEXT: one or two lines describing the call's intent.
- TOOL SCHEMAS: full schema for the called tool: parameters, types, format constraints, return shape, error conditions.
- DOMAIN RULES / STEERING DIRECTIVES: usually omitted.

Customer Service

- ENVIRONMENT STATE: data-dump style. Customer profile (id, name, DOB, email, phone, address, account_status, created_at, last_extension_date, goodwill credit), payment methods, associated lines/bills/IDs, retrievable fields, current timestamp. Include adjacent records that the next call won't read (distractors).
- TASK CONTEXT: supported task categories, identification status, current step in the workflow.
- TOOL SCHEMAS: lookup tools, action tools, transfer protocol with the exact handoff message string.
- DOMAIN RULES: policy excerpts, status semantics for accounts/lines/bills, scope boundaries, transfer protocol order.
- STEERING DIRECTIVES: surface behavior from tool outputs and policy, not assumed resolutions; status-driven branching; when to transfer.

Deep Research

- ENVIRONMENT STATE: OMIT.
- TASK CONTEXT: research question verbatim, domain/topic area, expected answer type, known constraints (thresholds, exclusions, reference points).
- TOOL SCHEMAS: web_search / read_webpage (or equivalents) with full response and error shapes (e.g., '{url, error}' for failures).
- DOMAIN RULES: synthesis methodology; what evidence must establish for the question; how partial evidence combines.
- STEERING DIRECTIVES: source preferences (authoritative > aggregator), treat constraints as route/property checks rather than assumptions, incremental retrieval semantics.

FORMAT AND STYLE

- Default to bullet/structured lists with sub-bullets; avoid prose paragraphs unless the content is genuinely narrative.
- Use data-dump format (key: value, indented sub-rows) when ENVIRONMENT STATE is rich (customer profiles, DB rows, repo state).
- Use exact tool, parameter, and field names from the trajectory.
- Third person, present tense, declarative voice.
- Vary phrasing across instances; do not produce templated boilerplate.
- Never quote the literal outcome; describe its shape and what determines it.
- The PREDICTION TARGET line is always the final line of the output, plain text, no '##' header, no trailing content.

DELIBERATE NOISE INJECTION

For ENVIRONMENT STATE (when present):

- Include neighboring/irrelevant rows, columns, fields, sibling files, or alternative records.
- For multi-app contexts, list ALL account credentials, not only the active app's.
- Include extra DB rows beyond the strictly necessary ones.

For TOOL SCHEMAS:

- Include compact one-line schemas of sibling tools that could be confused with the active one.

This noise is intentional and trains selective attention.

VALIDATION-FAILURE SEMANTICS

When the action omits a required parameter or violates a typed constraint (missing required IDs, wrong sort_by prefix, non-integer values, invalid enum, malformed dates), DOMAIN RULES must specify:

- which parameters are required and their types/format constraints,
- the validation-error response shape (e.g., HTTP 422 with message "Validation error. Reason: \n<param>: <reason>").

The predicted response is the validation error, not a successful payload.

```

## COMMON PITFALLS TO AVOID
- Using all 5 sections when the domain calls for fewer (Tool Use,
  Deep Research).
- Including ENVIRONMENT STATE for Deep Research.
- Skipping or moving the PREDICTION TARGET line.
- Inventing tool fields not present in the schema.
- Predicting the outcome value instead of what determines it.
- Producing identical phrasing across different trajectories.
- Forgetting noise injection in rich ENVIRONMENT STATE blocks.
- Using prose paragraphs where bullets/data-dumps are conventional.

## WORKED EXAMPLES (one per domain, abbreviated)

### Example A -- Tool Use (calculate_age)

## ENVIRONMENT STATE
- Tool backing data / query state:
  - Input record:
    - date_of_birth: 1990-05-15
  - Computed / derived state:
    - date_of_birth parsed as 1990-05-15

## TASK CONTEXT
- The agent is answering the user's question about their age by calling
  the 'calculate_age' tool with the provided date of birth.

## TOOL SCHEMAS
- 'calculate_age'
  - Description: Calculate the age based on date of birth.
  - Parameters:
    - 'date_of_birth' (string, format: date)
  - Returns:
    - '{ "age": integer }'
  - Error conditions:
    - Invalid or improperly formatted 'date_of_birth' may return an error.

PREDICTION TARGET: The tool response for the 'calculate_age' call in the
user turn.

### Example B -- Deep Research

## TASK CONTEXT
Research Question: <verbatim user question>
Domain/Topic Area: <e.g., New Zealand urban geography>
Answer Type Expected: <e.g., a list of city names>
Known Constraints:
- <constraint 1>
- <constraint 2>

## TOOL SCHEMAS
- web_search(query: string, max_results: integer) -> {
  query, provider, results: [ {title, url, snippet} ]
}
- read_webpage(url: string, max_chars: integer) -> {
  url, title?, content?
}
- Failure shape: { url, error }

## DOMAIN RULES
- The agent must synthesize a defensible answer from retrieved web
  evidence.
- Relevant evidence must establish each constraint from retrieved
  sources rather than assumption.

```

```

## STEERING DIRECTIVES
- Prefer authoritative/primary sources over aggregators.
- Treat exclusion criteria as route/property checks established from evidence.
- Search and browsing tools return only the requested query/URL content.

PREDICTION TARGET: The tool response returned by read_webpage for the URL and parameters in the USER turn.

### Example C -- SWE

## ENVIRONMENT STATE
- Repository: '/testbed' (Python codebase).
- Visible paths: '/testbed/<module>/<file>.py'.
- Repository structure beyond the shown path is discoverable via filesystem tools.

## TASK CONTEXT
- Active objective: inspect/modify the repository to address the task.
- Success is evaluated against existing tests; do not modify tests.

## TOOL SCHEMAS
- 'str_replace_editor'
  - 'view'
    - Parameters: path: string, view_range: optional [int, int]
    - Behavior: file -> numbered slice; directory -> listing.
    - Errors: invalid path, view_range on directory, invalid line range.
  - (sibling sub-commands: 'create', 'str_replace', 'insert')

## DOMAIN RULES
- Language: Python; test runner: pytest.
- Tests under '/testbed/tests/'; do not modify them.

## STEERING DIRECTIVES
- File slice semantics: only the requested range is returned with line numbers.
- Tool errors depend on command/path validity, not task intent.

PREDICTION TARGET: The observation returned by 'str_replace_editor' for the requested view command.

### Example D -- Customer Service

## ENVIRONMENT STATE
Current time: <timestamp>.

Customer profile:
- customer_id, full_name, date_of_birth, email, phone_number, address, account_status, created_at, last_extension_date, goodwill_credit_used_this_year

Payment methods: <list>
Associated lines: <ids>
Associated bills: <ids>

Retrievable fields via get_customer_info(): <fields>
Other domain records (lines/bills/plans/devices) accessible through respective tools when present in the episode.

## TASK CONTEXT
- Supported task categories: technical support, overdue bill payment, line suspension, plan options.
- Customer identification status: <status>.

```

```

## TOOL SCHEMAS
- get_customer_by_phone / by_id / by_name
- get_details_by_id, get_bills_for_customer, get_data_usage
- suspend_line, resume_line, refuel_data, enable_roaming, disable_roaming
- send_payment_request, transfer_to_human_agents
- Transfer protocol: call transfer_to_human_agents, then send exactly
  "YOU ARE BEING TRANSFERRED TO A HUMAN AGENT. PLEASE HOLD ON."

## DOMAIN RULES
- Policy scope, status semantics (Active/Suspended/Pending/Closed for
  account/line/bill), one-bill-awaiting-payment rule, suspension lift
  rules, refuel cap (2GB), etc.

## STEERING DIRECTIVES
- Surface behavior from tool outputs and policy, not assumed resolutions.
- Status-driven branching governs eligibility for subsequent actions.
- Follow transfer protocol only when within-scope resolution is impossible.

PREDICTION TARGET: The tool response or environment response produced by
the action in the user turn.

## NOW PROCESS

[state]
<<<state>>>

[action]
<<<action>>>

[outcome (optional)]
<<<outcome>>>

Procedure:
  Step 1: Classify the domain (SWE / Tool Use / Customer Service /
    Deep Research).
  Step 2: Decide which of the conditional sections to include based on
    the inclusion rules and the per-domain profile.
  Step 3: Generate the hindsight instruction in canonical section order,
    using bullets/data-dumps as appropriate to the domain.
  Step 4: Ensure the final line is 'PREDICTION TARGET: <...>'.

Output ONLY the hindsight instruction. No preamble. No postscript.

```

D Environment Setup for Downstream RL Agent Training with World Model Backend

We describe the dataset construction and training infrastructure for each of the three interactive environments used in our experiments. In all cases, training uses a World Model (WM) for rollout simulation during GRPO, while evaluation is performed against the real environment. Reward is computed from ground-truth environment state.

D.1 ScienceWorld

Environment and Data Splits. ScienceWorld [97] is a text-based science benchmark with 30 task types across 10 interconnected rooms, spanning state-change experiments, biological observations, and physics measurements. The agent navigates via free-text commands (`teleport to`, `pick up`, `focus on`) and submits answers via `focus on <object>`. We use 2 variations per task type for training and 2 for evaluation (60 each), drawn from disjoint variation pools. A per-task *room map* produced by an environment walk visiting all 10 rooms and opening containers is baked into the WM system prompt as ground-truth object locations.

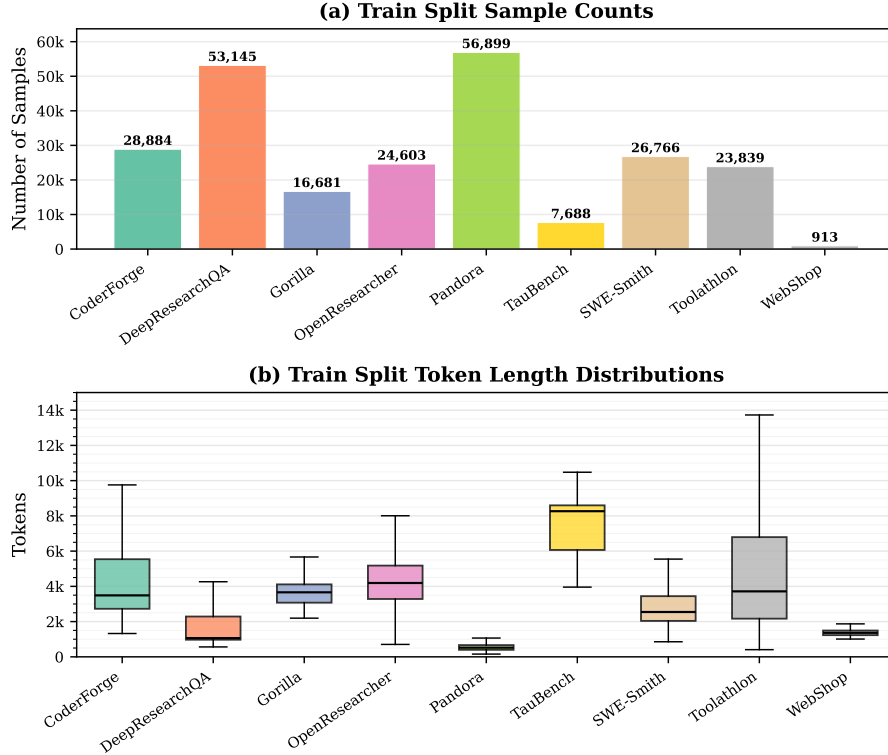


Figure 2: Training split counts and token length distributions per dataset. The sampled number of data points are proportional to openly available training tasks.

SFT Data and Formatting. Expert demonstrations are generated by GPT-5.5 (reasoning_effort=none) against the real environment, capped at 50 turns. Filtering to perfect-score (100) trajectories yields 37 demonstrations across diverse tasks. The agent prompt includes task-specific instructions (substance/container distinction, lifespan rankings, exploration strategy). For Mistral-7B, the system prompt is merged into the first user message, since Mistral v0.3’s chat template places system messages before the *last* user turn; LFM-2.5 (ChatML) and Qwen3 use their native templates.

Reward Structure The reward replays agent actions against a fresh ScienceWorld instance per generation, normalized to $[-0.2, 1.0]$: wrong-focus penalties (≤ -50) clamp to -0.2 , no-answer episodes receive -0.15 , and mild negatives clamp to -0.1 .

D.2 ALFWorld

Environment and Data Splits. ALFWorld [87] is a TextWorld-based household environment derived from ALFRED [86], where the agent operates in a single room populated with receptacles (cabinets, drawers, countertops) and completes picking, placing, cleaning, heating, cooling, or examining tasks. Actions follow a fixed grammar (go to, take, move, open, clean, heat, cool, use). We use the standard splits (3,553 training, 140 in-distribution evaluation), sampling 300 games for RL (shuffled, seed=7) and up to 200 per task type (balanced across 6 types) for SFT.

SFT Data. Expert demonstrations are generated by replaying ALFWorld’s built-in hardcoded expert plans against the real environment, parallelized across 8 workers and filtered to winning trajectories. This yields $\sim 1,000$ demonstrations averaging 17 actions each.

Reward Structure Reward is 1.0 for goal completion (checked via trajectory predicate matching without environment access), with partial credit for pipeline steps: 0.25 for take + place, 0.35 for take + transform + place, and 0.03 for take only. Episodes with 10+ actions but no take or move

receive -0.05 , and loop/repetition penalties scale reward by up to $0.3\times$. This action-oriented shaping was necessary for small models like LFM-2.5 and Qwen3 to break their instruction tuning tendencies of only interacting and exploring the environment space. These models completed approximately 50 steps before consistently giving up on exploitation heavy tasks.

D.3 AppWorld

Environment and Data Splits. AppWorld [94] is an interactive environment where agents complete tasks by making API calls to simulated applications (venmo, spotify, email, calendar, notes), receiving a natural language task description and invoking the correct sequence of tool calls. We use AppWorld’s standard train/test splits, with per-task WM system prompts encoding the available API schemas and task context. The agent system prompt is shared across tasks and loaded dynamically to align SFT and RL prompts.

SFT Data. Expert demonstrations are generated by running GPT-5.4 as the agent against all real AppWorld training tasks, with tool calls emitted in JSON and structured responses returned by the environment. We retain only trajectories that successfully complete the task (verified via `complete_task`).

World Model and Reward. The world model predicts API response text during GRPO rollouts, with a local proxy translating between the training plugin’s payload format and the world model server’s OpenAI-compatible chat completion API. The world model prompt includes the full API schema, current conversation state, and active tool call, following the same sectioned format as the other environments. Reward is computed by checking task completion predicates against the trajectory and the evaluation is done with ground truth Appworld server responses.

D.4 Common Infrastructure

Training Framework. All models are trained using ms-swift [120] with GRPO and vLLM colocate mode for on-policy rollout generation. DeepSpeed ZeRO-2 was used for Mistral-7B-v0.3’s multi-GPU training. We utilize 8xH100s for all training experiments (RL and world model) in this paper.

Hyperparameters. Across all environments, we perform sweeps on learning rate, KL penalty, temperatures and rollouts. We found that the generally optimal configuration for SFT required 2×10^{-6} to 8×10^{-6} with LFM-2.5 and Qwen-4B models requiring a slightly higher learning rate, closer to 6×10^{-6} . For RL, we use a lower learning rate in the range of 1×10^{-7} to 5×10^{-7} to ensure generalization after training. We consistently use gradient accumulation of 8 steps. Furthermore, we sweep across $\{0.5, 0.7, 0.9, 1.0\}$ temperature scales following [107, 113, 4]. We use cosine schedule for 1 epoch or convergence, whichever comes first based on the held out evaluation set. Checkpoints are saved every 20 steps and evaluated.

E Inference Harness and Optimizations for MDLMs

We use `lmdeploy` [21] to serve MDLMs tested in this paper. Specifically, we use a repetition penalty of 1.3 and diffusion steps $\{1, 25, 50, 100\}$. Since the world modeling task is deterministic and produces shorter outputs, we did not observe any output quality difference between steps 25 and 100 with the SDAR models. To maintain a balance between quality and speed, we select 50 denoising steps for our evaluations presented in Table 1.

While increasing block size for inference helps improve latency, we noticed a considerable performance drop when switching from a blocksize of $4 \rightarrow 8$ for the SDAR models. Since LLaDA-2.1-mini models were trained with a block size of 32, increasing their inference block size up to 64 had minimal impact on performance and accuracy in our tests when using the author recommended Joint-Threshold decoding ($\leq 5\%$ performance degradation on the test split while leading to speed-ups up to $2\times$). To ensure best performance for our results, we use the recommended block sizes per model (SDAR: 4, LLaDA-2.1-mini: 32). For WeDLM, which recommends window-based denoising instead of block-based denoising, we utilize the default window size of $W = 6$ and a distance-based penalty coefficient $\lambda = 0.10$ as recommended by the authors. While this varies across different GPU architectures, we were successfully able to reproduce the inference speed figures of WeDLM of $2.5\times$

Table 6: Out-of-domain evaluation of Qwen3.5-35B-A3B world model with and without thinking on OCCUBENCH, API-BANK, and INTERCODE test splits. Reported latency is for 4xH200 GPUs with LLM baselines using vLLM + FlashInfer, SDAR using the author recommended JetEngine and WeDLM using the official, vLLM-modified inference code.

Configuration	BLEU-1	ROUGE-L	MAUVE	Latency ($N = 4567$)
Qwen3.5-35B-A3B (no thinking)	.621	.664	.899	211s (1.00 $\times$)
+ Thinking (EFFORT = 5k)	.713	.768	.940	793s (3.75 $\times$)
+ Thinking (EFFORT = 10k)	.710	.758	.934	1028s (4.87 $\times$)
SDAR-8B (no thinking)	.749	.740	.902	380s (1.80 $\times$)
WeDLM-8B (no thinking)	.729	.761	.910	145s (0.68 $\times$)

as compared to Qwen3-8B for structured and constrained tool outputs with input lengths up to 8,192 tokens on average on a single H100.

F Impact of thinking on AR model performance

As discussed in [section 3](#), enabling thinking mode for these models has previously been shown [88] to improve diversity. We present a small ablation in [Table 6](#) to show that test time thinking helps improve AR performance which can be attributed to the diversity produced by the chain of thought. This further strengthens the argument for using MDLMs for world modeling because despite being much smaller, they offer much better diversity and hence performance to latency ratios than LLMs with similar performance.

G Prompts used for World Model for Downstream RL experiments

Template variables are shown in {curly_braces}. These are filled at runtime from the per-task WM system prompt, replayed state, and current action.

ALFWorld World Model Prompt

ENVIRONMENT STATE

ALFWorld is a TextWorld household environment derived from ALFRED. The agent operates in a single room (kitchen / bathroom / bedroom / living-room / office) populated with receptacles (cabinet, drawer, fridge, microwave, countertop, shelf, sidetable, diningtable, desk, dresser, bed, sinkbasin, stoveburner, toilet, bathtub, garbagecan, etc.) each carrying a 1-indexed instance id (e.g. 'cabinet 8'). Objects are also 1-indexed within their class (e.g. 'apple 1', 'mug 2'). All names are lowercase.

Task: {task}

Task type: {task_type}

Goal predicates: {goal_pred}

Current location: {current_loc}

Inventory: {inventory}

Receptacles in this room (all 1-indexed, lowercase):

- {recep_1}: {open_closed_status}
- {recep_2}: {open_closed_status}
- ...

Known receptacle contents (pre-baked from initial scene walk + updates from this rollout's take/put events):

[{recep_1}] {obj_1}, {obj_2}
 [{recep_2}] (empty)

...

Object state modifiers acquired earlier in this rollout:

- {obj}: {clean, hot, cool, sliced, on}

TASK CONTEXT

Predict ONLY the next textworld feedback string for the action below. The agent is interacting with the env one action at a time; you are predicting what textworld would print as the immediate response to the action — exactly one short paragraph of plain English. Do NOT plan, do NOT solve the task, do NOT echo the action, do NOT list admissible commands, do NOT output JSON or markdown.

Active action (verbatim, lowercase): {action_text}

DOMAIN RULES

Response must match the template VERBATIM. Always include instance numbers (e.g. 'stoveburner 1' not 'stoveburner', 'pot 1' not 'pot').

Example: go to stoveburner 1 → “You arrive at stoveburner 1. On the stoveburner 1, you see a pot 1, and a pan 2.”

go to <recep>

- CLOSED: “You arrive at <recep>. The <recep> is closed.”
- OPEN+items: “You arrive at <recep>. On the <recep>, you see a X 1, a Y 2, and a Z 1.”
- OPEN+empty: “You arrive at <recep>. On the <recep>, you see nothing.”
- Unknown: “Nothing happens.”

open <recep>

- Only valid when <recep> is currently CLOSED *and* agent is at <recep>.
- If valid AND <recep> contains items: “You open the <recep>. The <recep> is open. In it, you see <listed_items>.”
- If valid AND <recep> is empty: “You open the <recep>. The <recep> is open. In it, you see nothing.”
- Otherwise (already open, not openable, or not at it): “Nothing happens.”

close <recep>

- Only valid when <recep> is currently OPEN (and was openable in the first place) and agent is at <recep>.
- Success: “You close the <recep>.”
- Otherwise: “Nothing happens.”

take <obj> from <src>

- Valid iff agent is at <src>, <src> is OPEN, and <obj> is currently in <src>.
- Success: “You pick up the <obj> from the <src>.”
- <obj> MUST include the instance number: “You pick up the pot 1” not “You pick up the pot”.
- Otherwise: “Nothing happens.”
- After this, <obj> moves from <src> to inventory.

put <obj> in/on <dst> and move <obj> to <dst> (interchangeable)

- Valid iff <obj> is in inventory, agent is at <dst>, <dst> is OPEN.
- Success: “You move the <obj> to the <dst>.” (always ‘move ... to’, never ‘put ... in/on’, regardless of which form the agent typed.)
- Otherwise: “Nothing happens.”

clean <obj> with <recep> (recep must be a sinkbasin)

- Valid iff <obj> is in inventory, agent is at <recep>, <recep> is a sinkbasin.

- Success: “You clean the <obj> using the <recep>.” Object gains ‘clean’.
- Otherwise: “Nothing happens.”

heat <obj> with <recep> (recep must be a microwave)

- Valid iff <obj> is in inventory, agent is at <recep>, <recep> is a microwave.
- Microwaves do NOT need to be opened to heat — heat is a permitted action even when the microwave is closed.
- Success: “You heat the <obj> using the <recep>.” Object gains ‘hot’.
- If the agent then heats again or cools, the latest action wins (sticky).
- Otherwise: “Nothing happens.”

cool <obj> with <recep> (recep must be a fridge)

- Valid iff <obj> is in inventory, agent is at <recep>, <recep> is a fridge.
- Fridges do NOT need to be opened to cool.
- Success: “You cool the <obj> using the <recep>.” Object gains ‘cool’.
- Otherwise: “Nothing happens.”

slice <obj> with <tool> (tool must be a knife)

- Valid iff <tool> is in inventory and <obj> is at the agent’s location or in inventory.
- Success: “You slice the <obj> using the <tool>.” Object gains ‘sliced’.
- Otherwise: “Nothing happens.”

use <obj> (typically use desk lamp N)

- Valid iff <obj> is at the agent’s location or in inventory. <obj> is an object (not a receptacle) — desk lamps, floor lamps, etc. are not listed in the receptacle table.
- Success: “You turn on the <obj>.” Object gains ‘on’.
- Otherwise: “Nothing happens.”

examine <thing> — read-only; never changes the world.

- If <thing> is a receptacle at the agent’s location: “On the <thing>, you see <listed_items>.” (or ‘...you see nothing.’)
- If <thing> is an object in inventory or visible at current location: “This is a <state_words> <thing>.” where <state_words> reflects any modifiers (cold, hot, clean, sliced) — e.g. “This is a cold tomato 1.” If no modifier applies: “There’s nothing special about <thing>.”
- Otherwise: “Nothing happens.”

look

- If agent is in the middle of the room (not at any receptacle): “You are in the middle of a room. Looking quickly around you, you see nothing.” (Note: textworld emits the receptacle list ONLY in the reset welcome message, never on a subsequent look.)
- If agent is facing a receptacle: “You are facing the <recep>. Next to it, you see <adjacent_items_or_nothing>.”
- Adjacent receptacles in the same fixture group may also be listed: “You are facing the shelf 2, and shelf 1. Next to it, you see ...”.

inventory

- If carrying nothing: “You are not carrying anything.”
- Otherwise: “You are carrying: <listed_items>.” using the same Oxford-comma format as receptacle contents.

help — emits the verbatim TextWorld command-list block (rare; agent should not use this).

Universal fallback: any action whose preconditions are not satisfied — wrong location, missing object, closed destination for put/move, unknown object id, unknown receptacle id, malformed verb — produces exactly:

“Nothing happens.”

(Two words, capital N, period at end. No additional context.)

Closedness rules (which classes default to closed):

- Default-closed: cabinet, drawer, fridge, microwave, safe, box. Must be opened before take/put/move using their interior. Heat/cool/clean do NOT require open.
- Default-open / not-openable: countertop, shelf, sidetable, diningtable, desk, dresser, bed, sinkbasin, stoveburner, toaster, coffeemachine, toilet, bathtub, sofa, armchair, garbagecan, ottoman, tvstand, hand-/towel-holder, toiletpaperhanger.
- Use the receptacle table above as the authoritative open/closed source for this specific game; it overrides the class defaults when stated.

STEERING DIRECTIVES

- Copy receptacle and object names EXACTLY from Environment State (e.g. “desk 1” not “table”, “sinkbasin 1” not “sink”).
- Every name must include its instance number.
- Only use objects/receptacles from the Environment State. Do not invent.
- If uncertain, output “Nothing happens.”
- One short paragraph. No markdown, JSON, or admissible-commands lists.
- Prefix each item with “a ” in listings (e.g. “a apple 1, a mug 2, and a pan 1”).

PREDICTION TARGET: The single textworld feedback paragraph that the engine would print in response to the action {action_text} from the environment state above.

Appworld World Model Prompt

ENVIRONMENT STATE

The environment is AppWorld tool-response prediction.

Active task:

- {active_task}

Current date:

- {current_date}

Account credentials:

- {app}: username={username}, password={password}
- ...

Active tool call:

- tool_name: {tool_name}
- arguments: {tool_args_json}
- validation_status: {valid|invalid}
- validation_error: {error_message|null}

TASK CONTEXT

Predict only the JSON tool response for this one AppWorld action. Do not solve the whole user task. Do not echo the action.

TOOL SCHEMAS

- {tool_name}({param1}, {param2}, ...)
- Return shape for list/search tools: {"total": int, "{return_key}": array}
- Relevant source section: {section}
- For invalid calls, return exactly one JSON object with an error key.

DOMAIN RULES

- Validate the tool name and argument value types before generating a response.
- If validation_status is valid, the active tool is available and the arguments passed validation.

If invalid : Unknown tools, bad credentials, schema/type placeholder arguments, and missing target records return an error JSON object.

If valid : This is not an unknown-tool case. Do not return an error for tool availability.

- Returned records and IDs must come only from the record lines shown below.
- Return strict valid JSON only. Use JSON true/false/null, not Python True/False/None.
- No markdown, no commentary, no tracebacks, and do not double-quote or escape the whole JSON object.
- Do not include <think> text. Wrap the final JSON in <tool_response> only if the model requires tags.

COMPUTED QUERY STATE

- matched_count_after_filters: {matched_count}
- page_index: {page_index}
- page_limit: {page_limit}
- page_offset: {page_offset}
- records_on_this_page: {num_records}
- The response total field must equal matched_count_after_filters.
- The response array must contain only the records_on_this_page listed below.

Strict JSON records for this response page:

{json_records}

STEERING DIRECTIVES

- Expected validation status: {valid|invalid}.

If invalid : The exact error response body is: {"error": "{error_message}"}

If valid : The tool call is valid; do not return an error.

- For list/search tools, apply filters before pagination; never repeat page 0 for later pages.
- For login success, return a short access token only when credentials match Account credentials.
- If this is a list/search success, return {"total": {total}, "{return_key}": [...] } using only Strict JSON records for this response page.

PREDICTION TARGET: The strict JSON tool response for the action in the user turn.

Notes

The AppWorld WM prompt is substantially more structured than the other two environments because:

1. **Tool call validation** is performed locally before the WM is called. Invalid tool names, bad credentials, and schema-as-argument errors are caught and returned as deterministic error JSON without WM involvement.
2. **Query pre-computation:** For list/search tools (`spotify__search_songs`, `venmo__show_transactions`, etc.), the local responder filters records from the baked system prompt, applies pagination, and provides the exact JSON records the WM should return. This reduces the WM's job to formatting — it just needs to wrap the pre-computed records in the correct response shape.
3. **Per-tool routing:** The `SECTION_BY_TOOL` mapping determines which section of the system prompt to query (songs, albums, transactions, etc.), and `RETURN_KEY_BY_SECTION` determines the JSON key for the response array.
4. **Mutation tools** (create, delete, like, unlike, etc.) are handled deterministically with `{"status": "success", "message": "Action completed"}` — the WM is not called for these.

ScienceWorld World Model Prompt

ENVIRONMENT STATE

ScienceWorld is a text-based interactive science experiment environment with 10 rooms.

The agent is currently in: `{current_room}`.

Task: `{task_desc}`

Visible objects in `{current_room}`:

`{room_contents}`

Agent's inventory: `{inventory_items}`

Room adjacency (for 'go to' validation):

`{room_connections}`

TASK CONTEXT

The agent navigates rooms, picks up objects, and performs science experiments. Actions are plain text commands like 'teleport to kitchen', 'pick up glass cup', 'focus on water'.

The active action is: `{action_text}`

DOMAIN RULES

- `teleport to X` always succeeds. Response: "You teleport to the X."
- `go to X` only works if X is adjacent to the current room. If not adjacent: "No known action matches that input."
- `look around` returns a plain text list of objects in the current room. Format: "This room is called X. In it, you see: obj1, obj2, obj3. You also see: door to Y, door to Z."
- `pick up X` works only if X is visible in the current room AND has not already been picked up. Response: "You move the X to the inventory." If X is not present or already picked up: "No known action matches that input."
- `focus on X` works ONLY if X is explicitly listed in the current room description OR was previously picked up (in inventory). If X does NOT appear in the visible objects or inventory, respond: "No known action matches that input." NEVER confirm focus on an object that is not visibly present or in inventory — this is critical.
- `activate X / deactivate X` works if X is in the current room. Response: "The X is now activated/deactivated."
- `put X in/on Y` works if X is in inventory. Response: "You move the X to the Y."
- `open X / close X` works on doors and containers in the current room.
- `inventory` lists items the agent has picked up.

- `wait / wait1` advances time. Response: “(1 tick passes)”
- Invalid actions or objects not present: “No known action matches that input.”
- Items picked up earlier in the conversation are in inventory and no longer in their original room.

STEERING DIRECTIVES

GROUNDING RULE: Your response **MUST** use **ONLY** objects, rooms, and substances that appear in the **ENVIRONMENT STATE** section above. Do **NOT** invent, hallucinate, or elaborate beyond what is listed. If the environment state lists ‘a brick, a workbench’, respond with **EXACTLY** those items — do not add tools, equipment, or descriptions that are not in the state.

- For `look around`: copy the room description from the Visible objects section above **VERBATIM**. Do not add objects, do not elaborate, do not write prose. Just the object list.
- For `pick up X`: check if X appears in the Visible objects. If yes: “You move the X to the inventory.” If no: “No known action matches that input.”
- For `focus on X`: check Visible objects **AND** inventory. If X not found: “No known action matches that input.”
- Keep responses **SHORT** — one or two sentences maximum. Never write paragraphs, stories, or elaborate descriptions.
- Output plain text only. No JSON, no markdown, no code, no arrays, no role tags.

PREDICTION TARGET: The ScienceWorld observation text produced by the action ‘{action_text}’ given the current environment state.

H World Model Failure Modes

[Table 7](#) catalogs recurring world model (WM) failure modes observed during GRPO training for which prompts were later optimized. We present trajectory snippets from actual training runs to help explainability. These failures led to degradation of RL training signal and caused agent policy collapse for specific categories of tasks on the held out test set. We observed similar issues with both MDLMs and AR models but the hallucination and state inconsistency patterns for AR models were much more evident and harmful to agent behavior generalization.

Table 7: World model failure modes observed during GRPO training. Red highlights the erroneous portion. AW = AppWorld, ALF = ALFWorld, SW = ScienceWorld. These failures were addressed via grounding rules in the WM prompt, local deterministic responders, and JSON unwrapping in the training plugin.

Failure Mode	Env	Trajectory Snippet
(i) Incorrect pagination	AW	Agent requests page 1 of recommendations; WM returns the same records as page 0. Downstream, the agent double-counts songs and computes incorrect aggregates. T4: show_recommendations({page_index:0}) → {total:162, songs:[{id:40},{id:41},...]} T5: show_recommendations({page_index:1}) → {total:162, songs:[{id:40},{id:41},...]} ← same IDs
(ii) Object hallucination	SW	WM invents code-execution errors and objects absent from the environment state. Agent receives fabricated feedback that does not correspond to any valid ScienceWorld response. Agent: open cupboard WM: "Error during execution: name 'cupboard_is_opened' is not defined" ← fabricated [2pt] Agent: activate solar panel WM: "Solar panels cannot be used directly." ← not a ScienceWorld response
(iii) JSON-wrapped responses	SW	WM outputs structured JSON instead of plain-text environment observations. Downstream parser receives role/content wrappers that corrupt the conversation history. Agent: look around WM: [{"role":"tool", "content":"This room is called the kitchen. In it, you see: the agent, a substance called air, a chair..."}] ← JSON wrapper
(iv) Dropped instance numbers	ALF	WM substitutes generic class names for specific receptacle instances, breaking downstream state tracking and action validation. Agent: go to toaster 1 WM: "You arrive at toaster 1. On the table, you see a bread 2." Expected: "On the toaster 1, you see a bread 2." [2pt] Agent: go to microwave 1 WM: "You arrive at microwave 1. On the table, nothing..." Expected: "On the microwave 1, you see nothing."
(v) State inconsistency	SW	WM confirms picking up an object that is already in inventory, creating phantom duplicates. Agent "picks up" the thermometer three times across 17 turns. T6: pick up thermometer → "You move the thermometer to the inventory." T15: pick up thermometer → "You move the thermometer to the inventory." ← already held T17: pick up thermometer → "You move the thermometer to the inventory." ← third time
(vi) Verbose / garbled output	ALF, SW	WM generates excessively long descriptions with hallucinated items, or garbles object names by dropping instance numbers from object listings. ALFWorld: Agent: go to fridge 1 WM: "...On the fridge, you see a 2, and a mug." (object class dropped) [2pt] ScienceWorld (Qwen WM): Agent: look around (greenhouse) WM: "...a decorative stone gazing ball resting atop one corner... a rustic lantern holding dried lavender bundles... finally tucked away quietly within deepest recesses thereof lies forgotten treasure trove comprising ancient maps..." (886 hallucinated tokens)