
Next-Observation Prediction and Delayed Clue Memory: A Controlled MiniGrid Pilot

Neeraj Venna

Abstract

I test whether learning to predict the next observation helps a model remember a clue for a later choice. Two matched gated recurrent unit (GRU) models receive the same scripted MiniGrid Memory trajectories. A key or ball appears early, disappears from view, and determines the correct final choice. The task-only GRU succeeded on 100% of observed-clue episodes for all three training seeds. Adding action-conditioned next-view prediction gave 50%, 100%, and 100% success. Both models scored 50% when the clue was never visible, as expected for balanced forced choice. The extra loss improved prediction relative to the control’s untrained prediction head, but did not improve decisions. The control already solved the task, and three seeds cannot establish whether prediction helps or harms memory under other targets or training budgets. A post-hoc check found that 0 of 6,336 matched targets depended on clue identity after the clue disappeared. The prediction task therefore did not require retaining that identity. This explains a limitation of the target, not the cause of the failed run.

1 Introduction

A model can predict its next observation without remembering what matters for a later decision. BeliefSpec tests this distinction with a clue that disappears before it is needed. Both recurrent models learn to recover the clue from the observations and actions available so far. One also predicts the next observation given the current action. The comparison holds architecture, data, checkpoint selection, and evaluation episodes fixed.

2 Related Work

Prior work studies how prediction shapes an agent’s internal representation of its environment. Gregor et al. trained belief states using generative environment models for reinforcement learning; the prediction target and horizon determine which information the state must retain [4]. PlaNet and Dreamer learn latent dynamics for planning and imagined control [5, 6]. The present experiment instead tests a supervised final choice. AgentSpec compares combinations of perception, memory, reasoning, reflection, and action modules [1, 2]. BeliefSpec uses a matched-module comparison, but neither integrates AgentSpec nor reproduces its results. WorldEvolver and 3D-Belief study action-conditioned prediction and embodied belief modeling with representations and interfaces that differ from this fixed categorical task [7, 8].

3 Task

The task adapts MiniGrid Memory [3], using `MemoryEnv` with `size=13` and `random_length=False`. Each observation is a 7 by 7 grid of object IDs viewed from the agent’s position. The agent first sees a key or ball. A scripted controller follows the same corridor

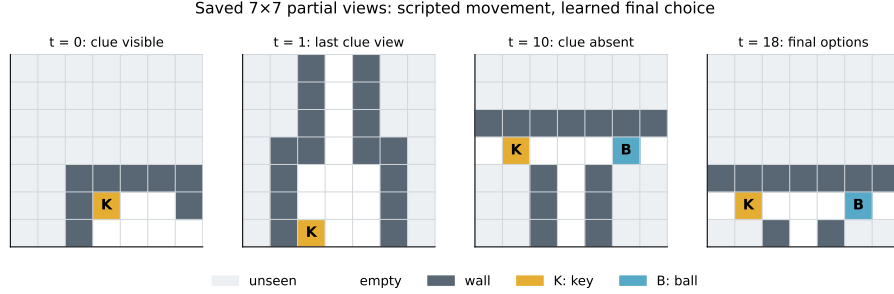


Figure 1: Views from episode `test:test-w8-r0:h0:top0:seen`. The clue appears at steps 0 and 1. Later keys and balls are choice options; the original clue has not returned. The choice at step 18 follows a delay of 17.

route regardless of the clue, then stops where two options are visible. The tested method chooses the top or bottom option. It does not control navigation.

Episodes vary clue identity, the object in the top option, whether the clue is visible, and delay. For visible clues, delay is the number of steps since the clue was last seen: 17, 29, or 53. In never-observed episodes, the clue is masked whenever it would be visible; the available observations cannot reveal the answer. Training, validation, and test sets contain 1,536, 384, and 768 episodes, respectively. Their visible histories do not overlap. All use the same layout, so this is not a test of generalization to new layouts.

The model receives object IDs, direction, previous action, and a route-phase indicator shared by all methods. Object IDs use categorical encoding; direction, action, and phase use one-hot encoding. Together these give an input dimension of 554. The prediction head also receives the action currently being executed. Inputs exclude mission text, filenames, seed identifiers, rewards, simulator hidden state, full-map images, and evaluator-only truth. Separate evaluator logs support scoring and audits.

4 Methods

Six methods feed the same final-choice controller. Current-observation memory uses the final view alone. Recent-history memory retains the latest 32 observation/action records. Structured memory stores a clue once observed; episodic retrieval returns an earlier clue-bearing frame if one exists. The learned control is a compact GRU trained to recover the clue. The predictive model has the same architecture and training setup, with an additional next-view prediction loss.

Each method returns probabilities for key, ball, and unknown. The controller converts them to a top-branch probability, $p_{\text{top}} = p_{\text{top clue}} + 0.5p_{\text{unknown}}$, and chooses top when $p_{\text{top}} \geq 0.5$. Clue-class accuracy and binary decision success are reported separately. Both learned models use clue labels derived only from visible history; a never-observed clue has label unknown. Cross-entropy (CE) measures error in these categorical predictions. The extra prediction loss uses state and action at time t to predict the next object grid at $t + 1$. Its training objective is task CE plus 0.1 times prediction CE. Object IDs are classes, not continuous numerical targets.

Both learned models use encoder width 48, GRU hidden size 32, batch size 64, Adam learning rate 0.003, gradient clipping 1.0, and 30 epochs (720 updates per run). Paired runs share initialization, minibatch order, optimizer settings, architecture, data, and checkpoint selection. Only the additional prediction loss differs. Task loss uses the actual final state of each sequence. Prediction loss excludes padding and averages over valid transitions and grid cells. The checkpoint with the lowest validation task cross-entropy is selected, with the earliest checkpoint breaking ties. Training seeds are 11, 22, and 33.

Each model allocates 56,171 parameters: 34,611 in the encoder, recurrence, and clue readout, plus 21,560 in the prediction head. The control allocates the same head for accounting, but trains only the shared 34,611 parameters. Its prediction metrics therefore describe an untrained head, not a trained predictive baseline.

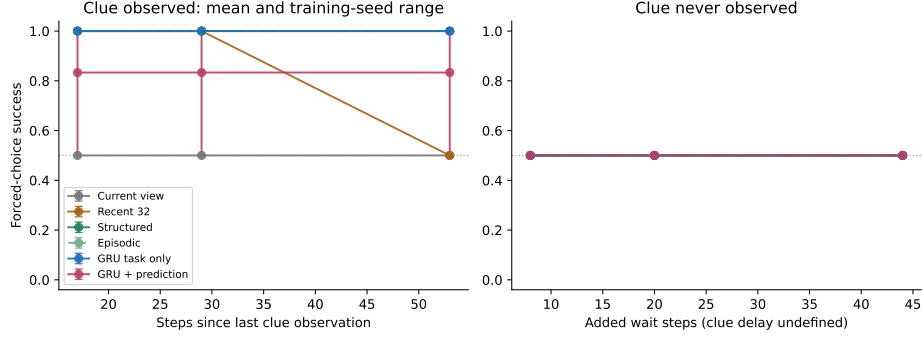


Figure 2: Decision success by delay. Points average three training seeds; bars span their minimum and maximum, not a confidence interval. Each deterministic baseline gives one value per condition. The right panel uses added wait steps because an unseen clue has no observation-based delay.

Table 1: The six learned runs. Observed success weights delays 17, 29, and 53 equally. Never-observed success uses balanced episodes with the clue masked.

Variant	Seed	Selected epoch	Observed success	Never success	Active params
Control	11	30	1.00	0.50	34,611
Predictive	11	10	0.50	0.50	56,171
Control	22	30	1.00	0.50	34,611
Predictive	22	30	1.00	0.50	56,171
Control	33	30	1.00	0.50	34,611
Predictive	33	30	1.00	0.50	56,171

The protocol preceded main training and final scoring. Checks covered observation/target alignment, padding masks, episode resets, gradient flow, equality after saving and loading, fitting a tiny subset, and solvability with structured memory. Further checks tested split overlap, hidden-state leakage, simulator replay, and interventions on decoded memory. No main run was excluded.

5 Results

Table 1 includes every learned run. For paired training seeds, predictive-minus-control success differences were -50 , 0 , and 0 percentage points. Averaging delays equally gives a mean difference of -16.667 percentage points and sample standard deviation 28.87 percentage points. A descriptive 95% t interval is $[-88.38, +55.04]$ percentage points ($df=2$). With so few seeds, this interval is unstable. Its replication unit is the training seed, not an individual episode; it is not an episode-binomial confidence interval.

The final view alone gave 50% success even when the clue had been visible earlier. Recent history solved delays 17 and 29, but scored 50% at delay 53 because the clue fell outside its 32-record window. Structured memory and episodic retrieval solved every observed-clue delay. Both scored 50% when the clue had never been seen.

For observed clues, the control prediction CE and accuracy were $2.4886/0.1180$, $2.5180/0.0815$, and $2.4858/0.0717$ for seeds 11, 22, and 33. The predictive model gave $0.5273/0.7633$, $0.4258/0.7963$, and $0.4306/0.7918$. Simply copying the current view gave accuracy 0.5999. The trained prediction head thus learned its target, but that improvement did not consistently yield better clue memory.

The failed run was predictive seed 11. All control seeds achieved readout accuracy 1.0 for both observed and never-observed clues; predictive seeds 22 and 33 did too. Predictive seed 11 instead had observed-clue accuracy 0.0 and never-observed accuracy 1.0. A saved failure trace shows predictive seed 11 assigning $p_{\text{key}} = 0.2507$, $p_{\text{ball}} = 0.2499$, and $p_{\text{unknown}} = 0.4994$. Its readout was unknown. The controller chose top, which was wrong because the hidden clue was ball and the top option was key.

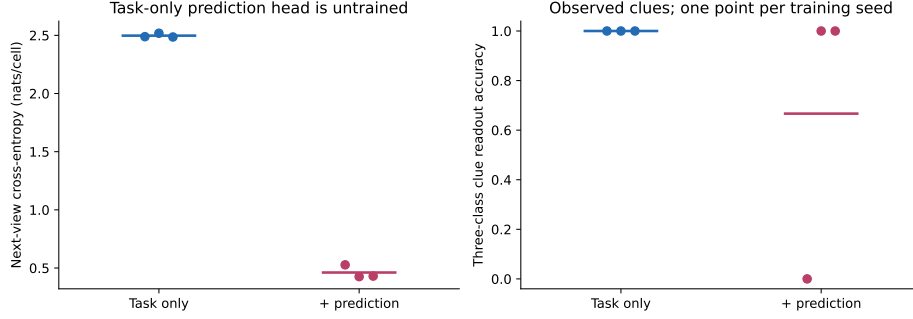


Figure 3: Prediction and clue readout on observed-clue episodes. Dots average delays within a seed; horizontal lines average seeds and do not show uncertainty. The task-only prediction head is untrained. Predictive seed 11 learned to predict the next view but did not recover the clue.

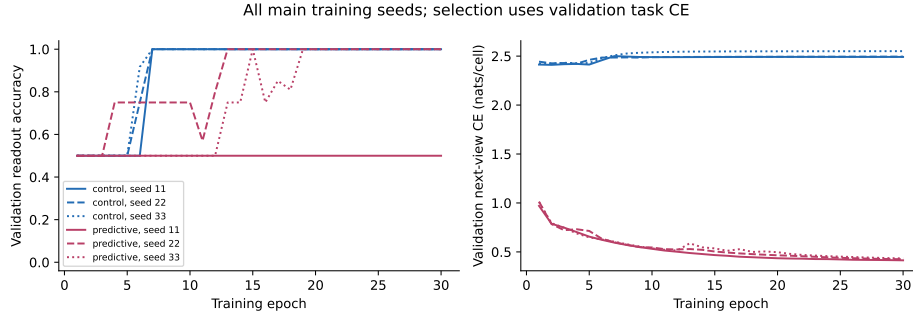


Figure 4: Validation performance for each training seed; lines do not represent uncertainty. All control seeds first reached at least 99% validation readout accuracy at epoch 7. Predictive seeds 22 and 33 reached it at epochs 13 and 15. Predictive seed 11 never reached that accuracy; the fixed validation task-CE rule selected epoch 10.

6 Prediction-target audit

The protocol noted that local prediction might not require remembering the clue. After the experiment, I compared targets in 192 matched key/ball episode pairs with observed clues. The only prediction input time with a clue-dependent next-view target was $t = 0$. Once the clue disappeared, 0 of 6,336 paired targets differed with clue identity. Retaining the clue was therefore unnecessary for this prediction objective over the measured delay. This post-hoc audit does not explain the seed-11 failure.

7 Limitations

The task-only GRU already solved the observed-clue task, leaving no room to measure a gain. Three training seeds are too few to characterize a failure that occurred once. A single layout also leaves transfer to other environments untested. The untrained control prediction head only checks whether the auxiliary head learned its target. No shuffled-target placebo, duplicate-clue-loss baseline, loss-weight sweep, or longer-budget rescue run was performed.

Predictive seed 11 could have failed because of optimization instability, competing losses, an unlucky seed, a target too local to require memory, or an insensitive task. These results do not show that prediction is generally harmful. They show that better local prediction did not reliably imply better delayed clue memory in this experiment.

8 Proposed experiments

Target relevance. Add a later panel whose content depends on the clue. Predict it before it becomes visible, and score the decision before the model receives it. A neutral panel in never-observed episodes avoids supplying hidden clue labels. Compare task-only and local-prediction GRUs with this model, a shuffled-panel-target placebo, and duplicate-clue CE supplying the same amount of direct clue supervision. The primary measure is paired seed-level decision success on a task calibrated below ceiling using development data; readout accuracy and panel CE are secondary. Better panel prediction without better decisions would weaken the hypothesis. Any gain could instead come from extra supervision, easier gradients, repeated clue labels, or regularization.

Information gathering. Before the final choice, allow *inspect*, which refreshes the clue reliably at a known cost, or *commit*. Compare matched task-only/predictive query policies with always-inspect, never-inspect, and observation-based confidence-threshold policies, charging each the same cost. Measure decision success minus inspection cost and query rates for sufficient versus insufficient histories. No gain over fixed policies, or querying regardless of need, would count against useful evidence gathering. The chosen cost might already favor a fixed policy; calibration or direct query supervision could also explain a gain.

Adaptation. Add a later event that confirms the clue, reveals its replacement, or invalidates it without revealing a replacement. Keep key/ball/unknown outputs; an unseen replacement must be labeled unknown. Compare matched task-only/predictive GRUs with initial-clue-only, latest-supported-fact, and recency memory. Measure success separately for unchanged, replacement-observed, and invalidated-only conditions, alongside unknown readouts and calibration. Persistent stale choices, discarded valid clues, or unjustified confidence after invalidation would count against useful adaptation. Recency, phase markers, direct update labels, or an easily read late cue could explain success without richer memory maintenance.

These experiments have not been run. Each needs a separate protocol, fresh training seeds, held-out trajectories, and difficulty/budget choices made on development data alone. The current model’s unknown output does not establish calibrated uncertainty, active evidence gathering, or continual adaptation.

9 Reproducibility

All six main runs used an arm64 Mac (macOS 15.7.7, 16 GiB RAM), CPU only, with two Torch threads. Logs record 109.224 seconds of total training time and peak RSS of 576,798,720 bytes. Task-only runs took 14.8–15.4 s each; predictive runs took 20.9–21.3 s. Inference over 768 episodes took 0.153–0.180 s per checkpoint, excluding simulator replay. These timings describe this execution, not a hardware benchmark.

Verification recomputed 7,680 episode rows, checked split separation and validation-based checkpoint selection, and exactly reproduced probabilities and prediction metrics from all six checkpoints. Retraining control seed 11 reproduced the original tensors and all 768 test outcomes. This checks reproducibility without adding an independent training seed.

Results are saved in `results/final/episodes.csv`, execution metadata in `artifacts/main_execution.json`, and verification in `review/2026-10-02/audit_after.json`. The original freeze manifest covers 50 files. Verification checks 49 current files and the original protocol at Git revision `0b0298e`. The edited publication copy of the protocol has a separate hash; `docs/evidence_preservation.md` records the distinction.

References

- [1] Jixuan Chen. AgentSpec: Understanding embodied agent scaffolds through controlled composition. <https://github.com/chenjix/AgentSpec>, 2026. Official repository, accessed 2026-10-01.
- [2] Jixuan Chen, Jianzhi Shen, Haoqiang Kang, Zhi Hong, Qingyi Jiang, Soham Bose, Yiming Zhang, Leon Leng, Amit Vyas, Lingjun Mao, Siru Ouyang, Kun Zhou, and Lianhui

- Qin. AgentSpec: Understanding embodied agent scaffolds through controlled composition. *arXiv preprint arXiv:2606.14674*, 2026. doi: 10.48550/arXiv.2606.14674. URL <https://arxiv.org/abs/2606.14674>. Version 1, submitted 2026-06-12.
- [3] Farama Foundation. MiniGrid memory environment documentation. <https://minigrid.farama.org/environments/minigrid/MemoryEnv/>, 2026. Accessed 2026-10-01.
 - [4] Karol Gregor, Danilo Jimenez Rezende, Frederic Besse, Yan Wu, Hamza Merzic, and Aaron van den Oord. Shaping belief states with generative environment models for RL. *arXiv preprint arXiv:1906.09237*, 2019. doi: 10.48550/arXiv.1906.09237. URL <https://arxiv.org/abs/1906.09237>. Version 2, revised 2019-06-24.
 - [5] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. *arXiv preprint arXiv:1811.04551*, 2019. doi: 10.48550/arXiv.1811.04551. URL <https://arxiv.org/abs/1811.04551>. Version 5, revised 2019-06-04.
 - [6] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2020. doi: 10.48550/arXiv.1912.01603. URL <https://arxiv.org/abs/1912.01603>. Version 3, revised 2020-03-17.
 - [7] Yifan Yin, Zehao Wen, Suyu Ye, Jieneng Chen, Zehan Zheng, Nanru Dai, Haojun Shi, Aydan Huang, Zheyuan Zhang, Alan Yuille, Jianwen Xie, Ayush Tewari, and Tianmin Shu. 3D-Belief: Embodied belief inference via generative 3d world modeling. *arXiv preprint arXiv:2605.11367*, 2026. doi: 10.48550/arXiv.2605.11367. URL <https://arxiv.org/abs/2605.11367>. Version 2, revised 2026-05-29.
 - [8] Xuan Zhang, Wenxuan Zhang, See-Kiong Ng, and Yang Deng. Self-Evolving World Models for LLM Agent Planning. *arXiv preprint arXiv:2606.30639*, 2026. doi: 10.48550/arXiv.2606.30639. URL <https://arxiv.org/abs/2606.30639>. Version 2, revised 2026-09-01; accepted at EMNLP 2026 Findings.