Dissecting Hierarchical Reasoning Models: A Mechanistic Study

Anonymous Authors[1]

Abstract

Machine reasoning at present is largely performed as a process of generating natural language reasoning. The Hierarchical Reasoning Model (HRM) offers an alternative solution for reasoning in the latent space. Despite strong reasoning performance, the internal mechanisms that enable HRM to reason remain underexplored. In this work, we aim to mechanistically understand how HRM reasons and what information it encodes via comparison against baseline models, causal interventions, linear probing with directed ablation, and sparse autoencoders for feature discovery. Our analyses reveal key findings regarding the internal iterative refinement in the latent space of HRM, how HRM encodes information in its hierarchical structure, and why standard interpretability tools fail to understand HRM. [1]

1. Introduction

Machine reasoning aims to equip a model with the capability to derive conclusions, construct plans, or solve problems whose solutions cannot be obtained through simple pattern matching (Wei et al., 2023; Bubeck et al., 2023). It has a wide range of applications in mathematical proofs (Trinh et al., 2024), software development (Jimenez et al., 2024), scientific discovery (Jumper et al., 2021), robotics (Zhou, 2025), and complex problem solving in domains like healthcare (Wang et al., 2025b), finance (Wu et al., 2023), and logistics (Li et al., 2023). Recent progress in machine reasoning has largely been driven by externalizing the reasoning process as natural language (Wei et al., 2023; Zelikman et al., 2022; Wang et al., 2023). Though effective, reasoning based on natural language could incur a high computational cost and generate an incorrect reasoning process due to generating an unnecessarily large number of tokens that do not truly reflect the internal reasoning (Wang et al., 2025c; Chen et al., 2025b).

Latent-space reasoning naturally emerges as an alternative. Instead of verbalizing each intermediate step, a reasoning model simply refines its continuous latent states directly (Hao et al., 2025). Within this landscape, the Hierarchical Reasoning Model (HRM) (Wang et al., 2025a) stands out. It is hierarchical in nature, coupling two recurrent modules at different timescales: a slow high-level state updated per cycle and a fast low-level state updated multiple times per cycle. Such a hierarchical nature is analogous to fast and slow cognitive thinking (Murray et al., 2014) and makes HRM empirically strong on reasoning tasks like Sudoku, Maze, and ARC-AGI, using a small number of parameters (27M parameters, trainable with modest compute).

Strong as it could be, the mechanistic understanding behind the success of HRM remains unexplored. Existing work partially addresses the gaps. Ren & Liu (2026) provides a mechanistic study of HRM and analyzes fixed-point validations, grokking-dynamics, and PCA-based reasoning modes, concluding that HRM makes structured guesses as opposed to step-by-step reasoning. However, it does not measure the contribution of either module, nor does it discuss the information encoded in the latent representations. (Hao et al., 2025; Geiping et al., 2025; Shen et al., 2025) performs studies in large transformer-based models without any hierarchical structure like HRM. To date, three key research questions still remain nascent:

- (RQ1) Which algorithmic strategy does HRM employ to coordinate high-level state and low-level state for solving reasoning tasks like Sudoku?
- (RQ2) What information do the latent representations encode over reasoning steps?
- (RQ3) Are the decodable features from the activations of HRM the same features that the model causally relies on to produce a solution?

In this paper, we consider Sudoku as our main task. Sudoku is well suited since it is easy to track the intermediate progress toward a solution with verifiable correctness. We conduct extensive analyses via (1) baseline comparison over

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

[1]Link To Code

four baseline models based on transformer and/or recurrent modules, (2) causal interventions on latent representations across puzzles and steps, and (3) representational analysis using probes and sparse autoencoders (Cunningham et al., 2023) for feature discovery. Based on the analyses, we have the following central findings:

- **HRM solves Sudoku by iteratively refining a solution state in the high-level activation, while the low-level activation acts as a working buffer** (Section 4). Across 16 recursive steps, the number of constraint violations decreases by 60%. Ablating high-level activations degrades solution accuracy (-6.8% at step 0 to -26.0% at step 15), whereas ablating the low-level activation has negligible effects. The high-level state on cross-puzzle patching shows that the activation carries puzzle-specific content. The two-timescale split is what makes a persistent, targetable state available for interpretability analysis.
- **Probe readout $\neq$ causally relevant information** (Section 5). Constraints in Sudoku tasks are linearly decodable from the high-level state at $\sim 90\%$ accuracy, yet directed ablation of probe directions produces effects indistinguishable from random controls.
- **HRMs reasoning cannot be localized to a small feature set in latent space** (Section 6). Ablation of SAE features trained on HRM activations produce significantly larger effects than linear probe directions, but the top 50 SAE features are no more impactful than 50 random features. We conclude that the constraint information is spread across the full 512-dimensional high-level state.

2. Related Work

Our work characterizes the internal mechanism of HRM, a recurrent hierarchical model that reasons in a latent space. Here we discuss related work in latent-space reasoning and mechanistic interpretability.

Reasoning in the latent space Recent works in latent-space reasoning can be categorize into latent chain-of-thought (CoT) and recurrence in reasoning. For latent CoT, Coconut (Hao et al., 2025) feeds the hidden state of a model back into itself as the next input embedding, mimicking the chain of thought in the discrete token space in large language models (Wei et al., 2023). Heima (Shen et al., 2025) further compresses the hidden representations into shorter latent sequences, while Zhou et al. (2026) optimizes the geometry or length of latent thinking. These methods treat the latent state as a direct substitute for CoT tokens and retain a left-to-right ordering of the token stream. To enable recurrence in reasoning, the key idea is to repeatedly passing through a single neural network block over multiple iterations. Adaptive Computation Time (ACT) (Graves, 2017) and Universal Transformers (Dehghani et al., 2019) are early examples of recurrent reasoning. Latent-space recurrent reasoning is later scaled up by Geiping et al. (2025). Hierarchical Reasoning Model (HRM) (Wang et al., 2025a) reasons in the latent space by introducing a high-level module and a low-level module and letting these two modules interact. Tiny Recursive Model (TRM) (Jolicoeur-Martineau, 2025) explores small, lightweight recursive models. Recursive language models (Zhang et al., 2026) take recursion as a primary driver of computational depth. These works aim to understand whether recursion helps improve reasoning capability and report their end-task accuracies. Our work primarily considers HRM and aims to understand how hierarchical design and recurrence enables stronger reasoning performance with less number of parameter.

Mechanistic interpretability Mechanistic interpretability aims to understand the internal mechanism of the learning model. One line of works study how to decode interpretable features from the latent representations. Linear probes isolate what is linearly decodable from a model representation (Belinkov, 2021); iterative null-space projection and amnesic probing extend probing with a causal experiment by ablating the probed direction and measuring downstream behavior (Ravfogel et al., 2020; Elazar et al., 2021). Sparse autoencoders (SAEs) decompose hidden states in an over-complete dictionary of sparsely activating features and have produced mono-semantic features in language models (Bricken et al., 2023; Cunningham et al., 2023; Templeton et al., 2024), with recent extensions to CoT reasoning traces (Chen et al., 2025a; Theodorus et al., 2025). Activation patching and trajectory analysis are alternative to understand how the model works internally. For example, Zhang et al. (2025) performs causal analysis of continuous thought representations, Vilas et al. (2025) characterizes latent temporal signals, Zhou et al. (2026) studies the geometry of reasoning trajectories, and Bogdan et al. (2025) identifies "thought anchors" in CoT. The work that is most relevant to ours is done by Ren & Liu (2026). It argues that HRM is a structured guesser through findings regarding fixed-point violations, latent dynamics across steps, and PCA-based reasoning modes, and proposes an ensemble-voting mechanism to improve Sudoku accuracy, but does not analyze what the hierarchical structure encodes nor test whether identified features are causally relevant during inference, which we address in this work.

3. Background

This section provides the context required to follow our

analysis.

3.1. Problem Statement and Notation

We consider a supervised sequence-to-sequence problem with input $\mathbf{x} \in \mathcal{V}^S$ and target $\mathbf{y} \in \mathcal{V}^S$ of equal length, where for Sudoku S=81 and $\mathcal{V}$={<blank>, $1, \ldots, 9$}. A model f_θ produces per-cell logits $\hat{\mathbf{y}} = f_\theta(\mathbf{x}) \in \mathbb{R}^{S\times|\mathcal{V}|}$ and predictions are taken as greedy $\arg\max$; intermediate hidden states have shape $S \times d$ and we use *cell* and *token position* interchangeably.

3.2. Sudoku Task and Evaluation

Sudoku and Sudoku-Extreme. Sudoku is a constraint-based puzzle defined on a 9×9 grid where blank cells must be filled with a digit from $1{-}9$ such that no digit repeats in any row, column, or sub-grid (*Sudoku constraints*); the input $\mathbf{x}$ is the flattened grid in row-major order with blanks encoded as a dedicated token and $\mathbf{y}$ is the flattened solution. We use the Sudoku-Extreme dataset released with (Wang et al., 2025a), filtered to "extreme" difficulty puzzles requiring multi-step constraint propagation, and use the publicly released HRM checkpoint without retraining; evaluations use the first 500 puzzles of the test loader (training corpus construction in App. A.2).

Metrics. We report four metrics. **Cell Accuracy** is the fraction of cells whose predicted digits match the ground-truth solution and satisfy the Sudoku constraints; **Puzzle Accuracy** is the fraction of puzzles for which all 81 cells are correct; **Hamming Distance** is the number of cells that differ between two solution states, which we use to quantify how much an intervention perturbs the model's output relative to its unperturbed prediction; and **Constraint Violations** is the number of row, column, and sub-grid digit uniqueness violations in a predicted board, providing a graded signal of validity.

3.3. Hierarchical Reasoning Model

The released HRM checkpoint of Wang et al. (2025a) is a 27M-parameter recurrent encoder with two interacting Transformer modules operating on tensors of shape $S\times d$ (d=512, 8 heads): a *low-level* state z_{L} updated T=2 times per cycle and a *high-level* state z_{H} updated once per cycle, both initialized from a fixed normal vector. Within a *segment* of N=2 such cycles, $z_{\mathrm{L}} \leftarrow f_L(z_{\mathrm{L}}, z_{\mathrm{H}} + \tilde{\mathbf{x}})$ and $z_{\mathrm{H}} \leftarrow f_H(z_{\mathrm{H}}, z_{\mathrm{L}})$, after which an output head f_{O} projects z_{H} to per-cell logits. An ACT halting head (Graves, 2017), trained with Q-learning, runs up to $M_{\max}$=16 segments while propagating the carry $(z_{\mathrm{H}}, z_{\mathrm{L}})$; we index segments by the *recursive step* $t \in \{0, \ldots, 15\}$. HRM is trained with deep supervision and a one-step gradient approximation (only the final $z_{\mathrm{L}}, z_{\mathrm{H}}$ updates are differentiated). Unless stated otherwise, our analyses run all 16 segments to obtain a full $z_{\mathrm{H}}^{(0)}, \ldots, z_{\mathrm{H}}^{(15)}$ trajectory. Full update equations, training details, and the carry mechanism are in App. A.1.

3.4. Baseline Architectures

To isolate the contributions of *recurrence* (reusing weights or hidden states across multiple refinement updates of the same input) and *hierarchy* (maintaining two states updated on different timescales), we compare HRM against four baselines (Table 1). All baselines share HRM's input embedding f_{I}, output head f_{O}, training corpus, and optimizer/learning-rate/batch-size schedule (Wang et al., 2025a), and differ only in the reasoning module; HRM itself is not retrained.

- **Recurrent Transformer.** A single hidden state updated by an 8-layer post-norm Transformer (no weight sharing) for 16 steps with input re-injection; isolates recurrence without hierarchy.
- **Universal Transformer** (Dehghani et al., 2019). The weight-shared variant of the above; tests whether weight tying changes the picture.
- **Plain Transformer.** An 8-layer feedforward Transformer with a single forward pass; isolates the Transformer backbone without test-time-compute scaling.
- **GRU.** A stacked `nn.GRU` (Cho et al., 2014) run as a single forward pass; tests whether the Transformer backbone itself is necessary.

Table 1. Architectures compared in this paper. "Recur." = refinement-step recurrence; "Hier." = two states on different timescales; "Steps" = max encoder updates per input. 'Recur." refers specifically to refinement-step recurrence, *not* sequence-axis recurrence; the GRU is recurrent over the sequence but performs a single refinement pass. All models are matched on parameter count and trained with the same recipe.

Model	Recur.	Hier.	Steps	Params
HRM	✓	✓	16	27M
Recurrent Trans.	✓		16	27M
Univ. Trans.	✓		16	29M
Plain Trans.			1	27M
GRU			1	27M

4. Finding 1: HRM Algorithm and Module Specific Tasks

This section addresses *what algorithm does HRM implement to solve Sudoku, and how is the work divided between its high and low modules?* We provide five lines of evidence that HRM iteratively refines a solution state carried in the

high level state (z_{H}), while the low level state (z_{L}) acts as a working buffer that iterates to find a temporary solution.

Setup. All experiments use the trained HRM checkpoint (Section 3) and the held out split of Sudoku-Extreme. We report cell accuracy, total constraint violations, and Hamming distance (to ground-truth). Each result lists its no intervention baselines so that Δacc values can be read directly. All baselines and HRM share the same training data and hyper-parameters.

4.1. Baseline Comparison

Table 2. Final-step metrics on 500 held-out puzzles. Recurrent models at step 15; non-recurrent at step 0. Viols = total row+col+box violations.

Model	Cell%	Puzzle%	Hamming	Viols
HRM	80.3	45.4	16.0	29.7
Recurrent Trans.	81.4	51.4	15.1	16.7
Univ. Trans.	80.7	48.4	15.6	19.1
Plain Trans.	61.9	0.2	30.9	61.3
GRU	49.6	0.0	40.8	132.2

The $20-30\%$ gap between recurrent and non-recurrent architectures demonstrates that iterative refinement through recurrence is essential for Sudoku-Extreme. We also see that HRM's constraint violations decrease from 73.6 (step 0) to 29.8 (step 15), a 60% reduction, while non-recurrent baselines remain at >30 mismatched cells (Figure 1).

Among the recurrent models, performance is comparable. The Recurrent Transformer slightly outperforms HRM in puzzle accuracy (51.4% vs. 45.4%). This means **hierarchy does not dominate in raw accuracy** at this scale, but as we show later, it produces a qualitatively different internal structure with a persistent high-level state (z_{H}) amenable to targeted intervention.

4.2. z_{H} is Causally Essential

If z_{H} carries the evolving solution, ablation at any step should hurt; more specifically, ablating the z_{H} activation should be more detrimental than it would be later in the ablation process. To test this, we replace either z_{H} or z_{L} with the zero vector at specific steps and measure the accuracy change (5,000 puzzles; baseline: 82.6% cell accuracy). Setting $z_{\mathrm{H}} = \mathbf{0}$ at *all* steps impacts accuracy by -20.5%. Per-step ablation reveals *increasing* importance over time; step 0 ablation causes -7.8%, while step 15 ablation causes -27.6% (Figure 2). To test whether z_{L} shares this pattern, we ablate z_{L} at scale (same 5,000 puzzles and baseline). Setting $z_{\mathrm{L}} = \mathbf{0}$ at *all* steps also reduces accuracy by -19.3%, which is similar in magnitude to z_{H}. However, the per-step profiles are strikingly different; **no single-step z_{L} ablation exceeds -1**, with most steps showing effects indistinguishable from zero (CI crossing zero). Only the final two steps (14 and 15) reach marginal significance (-0.5 and -1.0 respectively). This asymmetry confirms that solution-state information resides in z_{H}, while z_{L}'s content at any individual step is expendable but significant in aggregate.

4.3. Freezing Reveals Progressive Refinement

While the ablation study gives us information about the relevance of each activation, freezing activations tells us when relevant changes occur. We freeze z_{H} (prevent updates) while letting z_{L} continue, sweeping $k \in \{0, 1, 2, 3, 4, 5\}$ on 1,000 puzzles (baseline 81.8%) and extending to $k \in \{8, 12\}$ on a 200-puzzle subset (baseline 82.0%).

Table 3 shows a smooth decay, with freezing at the initial steps impacting accuracy more. This rules out the fact that z_{H} is a static, initialized plan; it evolves continuously and does most of the work in the first 5-8 steps.

Table 3. Freeze-after-step-k effect on cell accuracy. Top block: n=1,000 (baseline 81.8%). Bottom block: n=200 subset (baseline 82.0%) for the longer-horizon k.

Freeze step k	Δacc (%)	Accuracy (%)
0 (static z_{H})	-9.1	72.7
1	-6.2	75.6
2	-3.9	77.9
3	-3.0	78.8
4	-2.0	79.8
5	-1.5	80.3
8	-0.5	81.5
12	-0.1	81.8

4.4. Later z_{H} states can substitute for earlier ones

We inject z_{H} taken from a donor step into a target step for the same puzzle within a forward pass and measure Δacc (500 puzzles; baseline: 80.3%). Adjacent-step transplants (e.g., step 0$\rightarrow$2) produce negligible effects ($|\Delta\text{acc}| < 0.6$). Crucially, transplanting *later* steps into earlier positions (e.g., step 9$\rightarrow$2) slightly improves accuracy (+1.3); later z_{H} states contain more solution specific information that can be substituted in less-refined states. All effects are small (< 2), indicating that z_{H} evolves *continuously* rather than through discrete phase transitions.

4.5. z_{H} trajectories reveal convergence vs. wandering

Ren & Liu (2026) introduced PCA trajectory visualization for HRM, classifying per-sample paths into four reasoning modes. We extend their qualitative approach with a *quantitative population-level* analysis: we collect mean-pooled z_{H} at each ACT step for 200 puzzles, project onto the top-2 PCA components (explaining 57% of variance), and compute

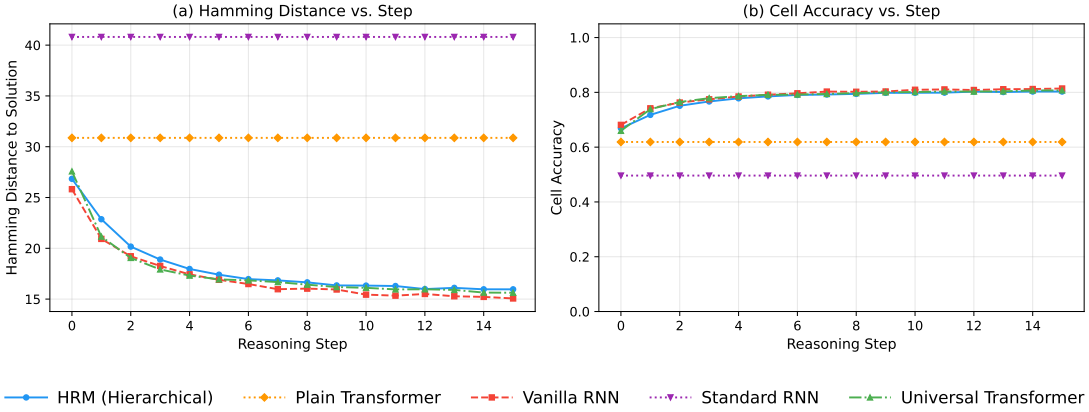

Figure 1. Across all five architectures, the three recurrent models (HRM, Recurrent Transformer, Universal Transformer) **(a)** steadily reduce Hamming distance to the ground-truth solution and **(b)** improve cell accuracy over 16 ACT steps, while the non-recurrent baselines (Plain Transformer, GRU) remain flat at >30 mismatches and substantially lower accuracy.

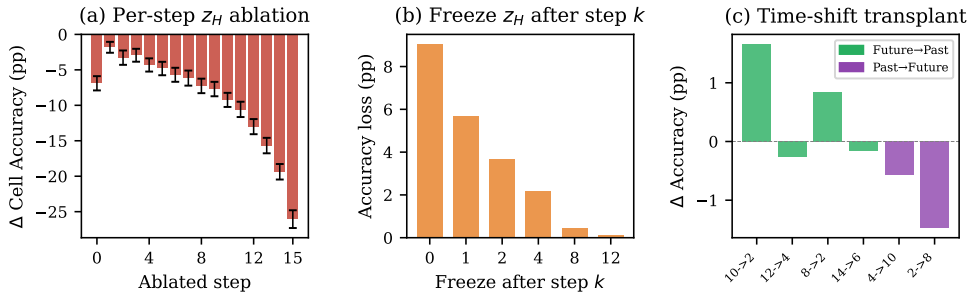

Figure 2. Causal evidence for z_{H} as a dynamic plan. **(a)** Per-step ablation: zeroing z_{H} is increasingly destructive at later steps (-6.8 at step 0 vs. -26.0 at step 15). **(b)** Freeze curve: preventing z_{H} updates after step k shows progressive refinement—freezing at step 0 loses -9.1; by step 8, the plan is nearly complete. **(c)** Time-shift: transplanting later z_{H} states into earlier positions slightly *improves* accuracy, confirming monotonic refinement.

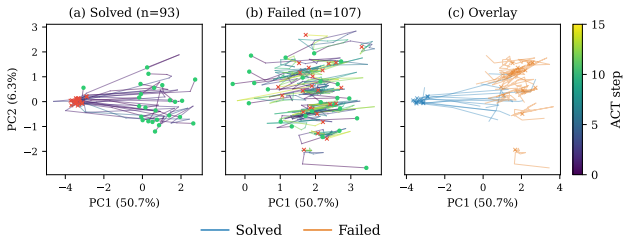

Figure 3. PCA trajectories of z_{H} across 16 ACT steps (200 puzzles). **(a)** Solved puzzles converge to a tight cluster. **(b)** Failed puzzles scatter widely, with endpoints dispersed. **(c)** Overlay: solved (blue) and failed (orange) occupy distinct regions. Green dots = step 0; red X = step 15.

directional convergence metrics.

Figure 3 reveals that solved puzzles (panel a) converge to a tight cluster, their trajectories start spread but rapidly collapse, consistent with convergence to a shared fixed-point region. Failed puzzles (panel b) scatter widely, with endpoints dispersed across PC space. The overlay (panel c) confirms that solved and failed trajectories occupy distinct regions. Quantitative directional metrics (App. A.5, Figure 8) confirm fixed-point convergence for solved puzzles ($\cos > 0.99$ by step 8, $\|\Delta z_{\mathrm{H}}\| \to 0$, norms $\sim$9.5) versus persistent oscillation for failed ones ($\|\Delta z_{\mathrm{H}}\| \approx 1.5$ at step 15, norms $\sim$6.8), quantitatively confirming the "spurious attractor" phenomenon identified qualitatively by Ren & Liu (2026).

Summary. HRM solves Sudoku by iteratively refining a candidate solution that resides in z_{H}. Recurrence is necessary (§4.1); z_{H} is causally relevant at all steps, while z_{L}'s individual content is expendable (§4.2). The refinement is progressive and is usually completed by steps 5 onward (§4.3, §4.4), z_{H} is puzzle-specific (§5.1), and its trajectory geometrically converges for solved puzzles while wandering fro failed ones (§4.5). The two timescale split is what makes a persistent, intervenable, high-level state available for the representation analysis in §5.

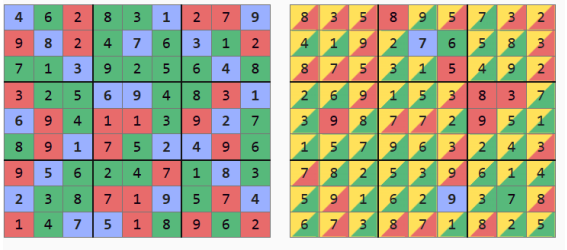

Figure 4. Cross-puzzle z_{H} patching: final-step predictions for a clean baseline run (left) vs. a run with z_{H} patched from a donor puzzle at step 5 (right). Cell colour: blue = given clue, green = constraint-satisfying, red = constraint-violation, yellow = changed from the previous step. Recipient puzzle in Figure 9.

5. Finding 2: Probe Readout ≠ Causally Relevant Information

This section addresses RQ2: *what information do HRM's latent representations encode, and how is it organized over time?* We establish that z_{H} carries an iteratively refined solution state. We now ask whether the activation carries puzzle specific information, sudoku constraint-specific information, or both. We also probe the z_{H} to find out how much of this information is decodable and how it's organized.

Setup. The cross puzzle patching analysis (§5.1) uses paired donor/target puzzles drawn from the test set. For §5.2 and §5.3 we collect z_{H} at five ACT steps ($t \in \{0, 4, 8, 12, 15\}$) on 500 puzzles and decode six per-cell targets: *cell digit* (10-class), *is-given*, *per-cell correctness*, and *row/col/box constraint violation* (binary).

5.1. z_{H} encodes puzzle-specific state

We run two puzzles through HRM in parallel, and at a chosen donor step k, replace the target's z_{H} with the donor's z_{H} at the same step (input embedding $\tilde{\mathbf{x}}$ stay's the target's). The forward pass is then continued while we measure the cell change pattern and the per-cell targets. The experiments show that patching results in a significant drop in accuracy : patching steps 1-3 reduces the target's accuracy by -53.1 (from 70.4% to 17.3%), and patching later steps (5–7) is even worse (-61.7). Beyond accuracy, we notice the post-patching solution *rewrites the given cells* replacing them with digits consistent with the donor puzzle's constraints (Figure 4). This is inconsistent with z_{H} acting as a generic reasoning module and confirms that z_{H} carries *puzzle-specific constraint states* that become increasingly specialized across ACT steps.

Table 4. Linear probe validation accuracy (%) on z_{H}. Violation probes gain ∼15% from step 0 to 15, mirroring the refinement dynamics of Finding 1.

Target	Step 0	4	8	12	15
Cell digit	98.5	98.9	99.0	99.2	99.0
Is given	99.6	95.5	96.4	96.5	97.3
Row violation	74.1	88.0	89.0	88.8	**89.9**
Col violation	74.5	87.7	88.3	88.9	**90.1**
Box violation	74.2	86.9	88.0	88.6	**88.8**
Correctness	71.1	80.7	83.2	83.1	**83.5**

5.2. Linear probes decode constraint information

We fit a linear probe (logistic/softmax regression with weight decay, 60 epochs SGD) on the z_{H} vector and report the best validation accuracy on held-out cells. Two categories emerge (Table 4; Figure 5a). *Structural features* (cell digit, is-given) are available from step 0 at $> 98\%$, encoded directly from the input embedding; later steps do not affect this. We attribute the small dip in "is-given" (between steps 0 and 4) to z_{H} storing the current solution state information and moving away from input features. *Dynamic features* (violations, correctness) emerge across steps; the largest jump occurs between steps 0 and 4 (∼14%), matching the rapid-refinement phase from Section 4.3.

A 2-layer MLP probe (512→256→1, ReLU) trained on the same data improves mean target accuracy at step 15 by < 1, confirming that constraint information is **linearly accessible** in z_{H} (see App. A.8).

5.3. Interpretable Geometry of Probe Weight-Vectors

We analyze the geometry of the probe weight vectors. We compute pairwise cosine similarities between the row, column, box, and correctness probes at step 15, and run PCA over those four weights to estimate the dimensionality of the subspace they span within z_{H}. The probe weight vectors reveal interpretable structure (Table 5). The three violation directions cluster together (cosines $0.67-0.73$), highlighting "violation" semantics, while correctness points in the opposing direction (-0.55 to -0.59). PCA shows these four directions span a ∼3-dimensional subspace (PC1: ∼77% variance), occupying only 0.6% of the 512-dimensional z_{H} space.

Table 5. Pairwise cosine similarity of probe weight vectors (step 15).

	Row	Col	Box	Corr.
Row	1.00	0.67	0.69	−0.55
Col	0.67	1.00	0.73	−0.59
Box	0.69	0.73	1.00	−0.55
Correct	−0.55	−0.59	−0.55	1.00

Summary. HRM's z_{H} activation is puzzle specific (§5.1) and linearly encodes Sudoku constraint information in a low-dimensional subspace.

6. Finding 3: Computation Is Deeply Distributed

Here we try to tackle the question: *are the features we can decode from HRM's activations the same features the model causally relies on to produce a solution?* We answer this using a directed ablation study on both linear probe directions (Section 5) and on features learned by a sparse autoencoder (SAE) trained on z_{H}. If HRM uses one specific probe direction in it's forward pass, projecting it out should impair behaviour relative to a random direction of the same rank.

Setup. We evaluate both ablations on held-out Sudoku extreme puzzles by comparing post-intervention cell accuracies to our baseline of the same puzzle. Random controls are constructed by sampling Gaussian directions (for probes) or random feature selection (for SAE ablation), to test whether the targeted interventions are significant.

6.1. Directed Ablation of Probe Directions

For every unit-norm probe direction $\hat{\mathbf{w}}$ (Section 5), we apply $z_{\mathrm{H}} \leftarrow z_{\mathrm{H}} - (z_{\mathrm{H}} \cdot \hat{\mathbf{w}})\hat{\mathbf{w}}$ at every ACT step (removing the readable direction from z_{H}). We then let the forward pass continue and measure the change in cell accuracy Δacc on 500 sample puzzles. We additionally ablate the full 3-D row/col/box subspace simultaneously

Table 6. Directed ablation of probe directions vs. random controls (500 puzzles). Probe directions produce effects indistinguishable from random.

Direction ablated	Δacc (%)	**p-value**
Row violation	-0.09	0.408
Col violation	-0.59	0.004
Box violation	-0.23	0.365
Correctness	$+0.72$	0.209
Is given	$+0.20$	—
Random control 1	-0.44	—
Random control 2	$+0.07$	—
Random control 3	$+0.95$	—

We report that **no probe direction produces a larger effect than random controls** (Table 6): all accuracy changes are within $\pm 1\%$, violation count changes are within ± 0.4 cells, even the marginally significant column-violation direction (p=0.004) has a tiny effect size (Cohen's d=-0.13, Δacc=-0.59%), and ablating the full 3D row+col+box subspace simultaneously yields $\Delta\text{acc} < 0.2\%$. The directions along which constraint information is most linearly readable ($\sim$90% accuracy) are not the directions along which the model uses that information; constraints are encoded in a *distributed, redundant representation* across many dimensions.

Takeaway. High probe accuracy should not be interpreted as evidence that a model encodes information in a computationally meaningful way without causal validation.

6.2. SAE Feature Analysis

Probes assume relevant features are linear in z_{H}. SAE learn an over-complete, sparse, non-linear feature dictionary, so SAE-feature ablation tests whether a sparser, model-discovered basis recovers the causal mass that probe directions miss. We train an SAE with a dictionary size of $d = 2048$ and an L1 coefficient of $\lambda = 0.01$ on z_{H} activations across all ACT steps. Out of 2048 features, 1344 (65.6%) are active. The average pairwise cosine between features is 0.048 (near-orthogonal), indicating that no feature is highly specialized. Following (Templeton et al., 2024), we ablate (i) the top-50 SAE features ranked by activation magnitude and (ii) 50 random SAE feature subsets at inference time by re-encoding z_{H}, zeroing the targeted coefficients, and substituting the decoded reconstruction back into the forward pass; we then compare the resulting Δacc on 300 puzzles to the probe and random direction conditions of Section 6.1.

Across the 14 most-correlated features, $r_{\text{best}} \in [0.31, 0.41]$ vs. $r_{\text{next}} \in [0.21, 0.40]$; no feature satisfies $|r_{\text{best}}| > 2|r_{\text{next}}|$, i.e. no SAE feature is concretely identifiable with a single symbolic target (full table in App. A.10, Table 8).

Table 7. Causal ablation comparison (300 puzzles). SAE-feature ablation is significantly more damaging than probe-direction ablation, but top SAE features are not more damaging than random ones.

Condition	Δacc (%)	**Std** (%)
Top-50 SAE features	-3.9	16.1
Random SAE features	-4.3	15.6
Probe directions	$+0.1$	9.3
Random directions	$+0.6$	8.0

Results. Table 7 shows that SAE-feature ablation produces significantly larger effects than probe-direction ablation. However, the top-50 SAE features are *not* more impactful than the random-50 SAE features, causal importance is spread across the dictionary rather than concentrated in any nominally important subset. Individual effects range from -2.4% to -7.0%.

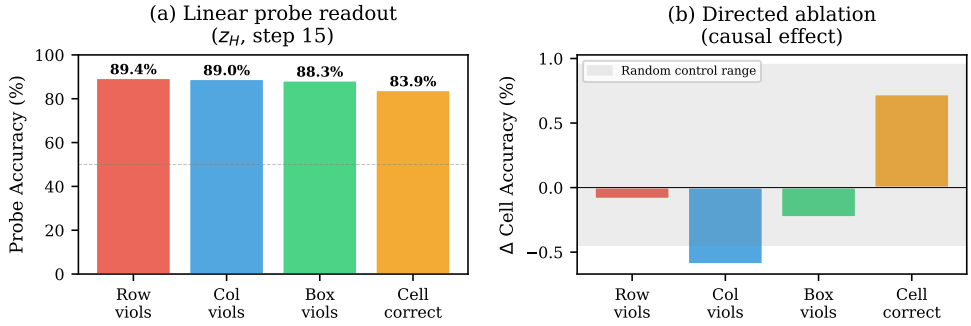

Figure 5. Readout ≠ Causally Relevant. **(a)** Linear probes decode constraint violations from z_{H} with $\sim$90% accuracy at step 15, improving over time. **(b)** Directed ablation: removing each probe direction produces effects indistinguishable from random controls ($|\Delta\text{acc}| < 1\%$). The directions that are most *readable* are not the directions the model *uses*.

Summary. Combining results across all three findings yields a clear ordering of causal importance: **full z_{H} ablation** (-19.3%) $\gg$ **SAE features** (-3.9%) $\gg$ **probe directions** ($\approx$ 0%), confirming that computation in z_{H} is **deeply distributed** (App. A.12, Figure 10). Linear probes reveal features that are not causally relevant (Section 6.1); SAE decomposition recovers more causal structure but no sparse subset of SAE features captures HRM's mechanism either, with top-ranked features (Table 8) no more impactful than random ones. High probe accuracy on a feature does not imply that the feature is relevant, even sparse-feature decompositions can be insignificant.

7. Discussion and Conclusion

From our findings, we characterize HRM as performing *constraint-aware iterative refinement on a puzzle specific solution state distributed across z_{H}*.

Hierarchy as an interpretability feature Our baseline comparison (Section 4.1) reproduces (Knoop & Kamradt, 2025): a flat recurrent transformer matches HRM on Sudoku-Extreme. The hierarchical split is not what makes HRM novel, but it provides an intervenable high-level state causally separable from the working buffer (Section 4.2); without it, none of our interventions would have an obvious target. Hierarchy is thus better understood as an *inductive bias for interpretability*.

Readout ≠ causally relevant information The most transferable result is the dissociation in Sections 6.1 and 6.2: easily decodable directions in z_{H} have no measurable causal effect when ablated, and an overcomplete SAE recovers no mono-semantic features for the symbolic targets tested (Table 8). Probing-only studies thus risk overstating feature relevance.

Why representations might be more distributed in HRM than in feed-forward LLMs We attribute this to HRM's training regime: the one-step gradient approximation back-propagates only through the final $z_{\mathrm{L}}, z_{\mathrm{H}}$ updates of each segment, so each segment must compress a full differential update to the solution state into a single 512-d vector with no per-feature pressure for sparsity. Distributed representations may thus be the default for recurrent latent-reasoning models with truncated objectives; we leave proof to future work.

Limitations Three limitations bound our claims: (i) all experiments use Sudoku-Extreme, and characterizations may differ on tasks like Maze or ARC-AGI; (ii) Section 6.2 uses a single SAE configuration (d=2048, λ=0.01) and a wider sweep may reveal mono-semantic features; (iii) we probe/ablate z_{L} only in Section 4.2, leaving its causal feature content open.

Conclusion and future work. Our three findings are: (i) recurrence drives HRM accuracy and z_{H} is the progressively refined solution state (Section 4); (ii) z_{H} is puzzle-specific and encodes Sudoku constraints linearly in a low-dimensional space (Section 5); (iii) the directions HRM actually uses are deeply distributed, with probe/SAE features showing no causal effect on inference. Together they characterize HRM as a constraint-aware iterative refiner whose causally relevant features are distributed across z_{H}. Natural follow-ups: replicate on Maze/ARC, sweep SAE configurations, extend probes/SAEs to z_{L}, and test whether sparsity-promoting regularization makes features more monosemantic.

References

Belinkov, Y. Probing classifiers: Promises, shortcomings, and advances, 2021. URL https://arxiv.org/abs/2102.12452.

Bogdan, P. C., Macar, U., Nanda, N., and Conmy, A. Thought anchors: Which llm reasoning steps mat-

ter?, 2025. URL https://arxiv.org/abs/2506.19143.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., Lasenby, R., Wu, Y., Kravec, S., Schiefer, N., Maxwell, T., Joseph, N., Hatfield-Dodds, Z., Tamkin, A., Nguyen, K., McLean, B., Burke, J. E., Hume, T., Carter, S., Henighan, T., and Olah, C. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y. T., Li, Y., Lundberg, S., Nori, H., Palangi, H., Ribeiro, M. T., and Zhang, Y. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023. URL https://arxiv.org/abs/2303.12712.

Chen, X., Plaat, A., and van Stein, N. How does chain of thought think? mechanistic interpretability of chain-of-thought reasoning with sparse autoencoding, 2025a. URL https://arxiv.org/abs/2507.22928.

Chen, Y., Benton, J., Radhakrishnan, A., Uesato, J., Denison, C., Schulman, J., Somani, A., Hase, P., Wagner, M., Roger, F., Mikulik, V., Bowman, S. R., Leike, J., Kaplan, J., and Perez, E. Reasoning models don't always say what they think, 2025b. URL https://arxiv.org/abs/2505.05410.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014. URL https://arxiv.org/abs/1406.1078.

Cunningham, H., Ewart, A., Riggs, L., Huben, R., and Sharkey, L. Sparse autoencoders find highly interpretable features in language models, 2023. URL https://arxiv.org/abs/2309.08600.

Dehghani, M., Gouws, S., Vinyals, O., Uszkoreit, J., and Łukasz Kaiser. Universal transformers, 2019. URL https://arxiv.org/abs/1807.03819.

Elazar, Y., Ravfogel, S., Jacovi, A., and Goldberg, Y. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175, 03 2021. ISSN 2307-387X. doi: 10.1162/tacl_a_00359. URL https://doi.org/10.1162/tacl_a_00359.

Geiping, J., McLeish, S., Jain, N., Kirchenbauer, J., Singh, S., Bartoldson, B. R., Kailkhura, B., Bhatele, A., and Goldstein, T. Scaling up test-time compute with latent reasoning: A recurrent depth approach, 2025. URL https://arxiv.org/abs/2502.05171.

Graves, A. Adaptive computation time for recurrent neural networks, 2017. URL https://arxiv.org/abs/1603.08983.

Hao, S., Sukhbaatar, S., Su, D., Li, X., Hu, Z., Weston, J., and Tian, Y. Training large language models to reason in a continuous latent space, 2025. URL https://arxiv.org/abs/2412.06769.

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. Swe-bench: Can language models resolve real-world github issues?, 2024. URL https://arxiv.org/abs/2310.06770.

Jolicoeur-Martineau, A. Less is more: Recursive reasoning with tiny networks, 2025. URL https://arxiv.org/abs/2510.04871.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S. A. A., Ballard, A. J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D., Clancy, E., Zielinski, M., Steinegger, M., Pacholska, M., Berghammer, T., Bodenstein, S., Silver, D., Vinyals, O., Senior, A. W., Kavukcuoglu, K., Kohli, P., and Hassabis, D. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, August 2021.

Knoop, M. and Kamradt, G. The hidden drivers of HRM's performance on ARC-AGI. https://arcprize.org/blog/hrm-analysis, 2025. ARC Prize blog post.

Li, B., Mellou, K., Zhang, B., Pathuri, J., and Menache, I. Large language models for supply chain optimization, 2023. URL https://arxiv.org/abs/2307.03875.

Murray, J. D., Bernacchia, A., Freedman, D. J., Romo, R., Wallis, J. D., Cai, X., Padoa-Schioppa, C., Pasternak, T., Seo, H., Lee, D., and Wang, X.-J. A hierarchy of intrinsic timescales across primate cortex. *Nat. Neurosci.*, 17(12): 1661–1663, December 2014.

Ravfogel, S., Elazar, Y., Gonen, H., Twiton, M., and Goldberg, Y. Null it out: Guarding protected attributes by iterative nullspace projection, 2020. URL https://arxiv.org/abs/2004.07667.

Ren, Z. and Liu, Z. Are your reasoning models reasoning or guessing? a mechanistic analysis of hierarchical reasoning models, 2026. URL https://arxiv.org/abs/2601.10679.

Shen, X., Wang, Y., Shi, X., Wang, Y., Zhao, P., and Gu, J. Efficient reasoning with hidden thinking, 2025. URL https://arxiv.org/abs/2501.19201.

Su, J., Lu, Y., Pan, S., Murtadha, A., Wen, B., and Liu, Y. RoFORMer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

Templeton, A., Conerly, T., Marcus, J., Lindsey, J., Bricken, T., Chen, B., Pearce, A., Citro, C., Ameisen, E., Jones, A., Cunningham, H., Turner, N. L., McDougall, C., MacDiarmid, M., Freeman, C. D., Sumers, T. R., Rees, E., Batson, J., Jermyn, A., Carter, S., Olah, C., and Henighan, T. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

Theodorus, J., Swaytha, V., Gautam, S., Ward, A., Shah, M., Blondin, C., and Zhu, K. Finding sparse autoencoder representations of errors in cot prompting. In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*, 2025. URL https://openreview.net/forum?id=oCprwPRqwW.

Trinh, T. H., Wu, Y., Le, Q. V., He, H., and Luong, T. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, January 2024. ISSN 1476-4687. doi: 10.1038/s41586-023-06747-5. URL http://dx.doi.org/10.1038/s41586-023-06747-5.

Vilas, M. G., Yousefi, S., Nushi, B., Horvitz, E., and Balachandran, V. Tracing the traces: Latent temporal signals for efficient and accurate reasoning, 2025. URL https://arxiv.org/abs/2510.10494.

Wang, B., Min, S., Deng, X., Shen, J., Wu, W., Zettlemoyer, L., and Sun, H. Towards understanding chain-of-thought prompting: An empirical study of what matters. In *Proc. of The 61st Annual Meeting of the Association for Computational Linguistics*, 2023. URL https://arxiv.org/pdf/2212.10001.pdf. ACL 2023.

Wang, G., Li, J., Sun, Y., Chen, X., Liu, C., Wu, Y., Lu, M., Song, S., and Yadkori, Y. A. Hierarchical reasoning model, 2025a. URL https://arxiv.org/abs/2506.21734.

Wang, W., Ma, Z., Ding, M., Zheng, S., Liu, S., Liu, J., Ji, J., Chen, W., Li, X., Shen, L., and Yuan, Y. Medical reasoning in the era of llms: A systematic review of enhancement techniques and applications, 2025b. URL https://arxiv.org/abs/2508.00669.

Wang, Z., Jiang, J., Qiu, T., Liu, H., Tang, X., and Yao, H. Efficient long cot reasoning in small language models, 2025c. URL https://arxiv.org/abs/2505.18440.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

Wu, S., Irsoy, O., Lu, S., Dabravolski, V., Dredze, M., Gehrmann, S., Kambadur, P., Rosenberg, D., and Mann, G. Bloomberggpt: A large language model for finance, 2023. URL https://arxiv.org/abs/2303.17564.

Zelikman, E., Wu, Y., Mu, J., and Goodman, N. D. Star: Bootstrapping reasoning with reasoning, 2022. URL https://arxiv.org/abs/2203.14465.

Zhang, A. L., Kraska, T., and Khattab, O. Recursive language models, 2026. URL https://arxiv.org/abs/2512.24601.

Zhang, Y., Tang, B., Ju, T., Duan, S., and Liu, G. Do latent tokens think? a causal and adversarial analysis of chain-of-continuous-thought, 2025. URL https://arxiv.org/abs/2512.21711.

Zhou, A. Rt-2: Vision-language-action models for generalizable robotic control: A comprehensive review. *Advances in Engineering Technology Research*, 15(1):1423, November 2025. ISSN 2790-1688. doi: 10.56028/aetr.15.1.1423.2025. URL http://dx.doi.org/10.56028/aetr.15.1.1423.2025.

Zhou, Y., Wang, Y., Yin, X., Zhou, S., and Zhang, A. R. The geometry of reasoning: Flowing logics in representation space, 2026. URL https://arxiv.org/abs/2510.09782.

A. Additional Experimental Details

A.1. Full HRM Architecture and Training Details

This appendix expands the brief HRM description of Section 3.3.

Components. HRM is composed of four learned components:

1. *Input Embedding Network* $f_{\mathrm{I}} : \mathcal{V}^S \to \mathbb{R}^{S\times d}$ that maps input tokens $\mathbf{x}$ to d-dimensional embeddings $\tilde{\mathbf{x}}{=}f_{\mathrm{I}}(\mathbf{x})$. In the released model, f_{I} is a learned token embedding; positional information is injected inside each

Transformer block via Rotary Position Embeddings (RoPE) (Su et al., 2021). The embedding is computed once per example and reused across recurrent steps.

2. *Low-Level Reasoning Module* f_L: a 4-layer post-norm Transformer encoder with bidirectional self-attention, SwiGLU MLPs, and rotary position encodings.

3. *High-Level Reasoning Module* f_H: same architecture as f_L with separate parameters.

4. *Output Head* $f_{\mathrm{O}} : \mathbb{R}^{S\times d} \to \mathbb{R}^{S\times|\mathcal{V}|}$, a linear projection from the final high-level state to per-position logits.

Both f_L and f_H operate on tensors of shape $S \times d$ with d=512 and 8 attention heads, totalling ≈27M parameters.

Latent States. HRM maintains z_L, $z_H \in \mathbb{R}^{S\times d}$, where z_L is the fast low-level state and z_H is the slow high-level state. "Fast" and "slow" refer purely to update rate. Both states are initialized from a single fixed normal vector held constant across training and inference.

Segments. A segment consists of N high-level cycles, each containing T low-level updates followed by a single high-level update:

$$z_L \leftarrow f_L\big(z_L,\ z_H + \tilde{\mathbf{x}}\big), \tag{1}$$

$$z_H \leftarrow f_H\big(z_H,\ z_L\big), \tag{2}$$

where u is added to the running state as an *input injection*. Equation (1) is applied T times, then equation (2) once; with N=T=2 each segment performs 4 updates of z_{L} and 2 updates of z_{H}. After all in-segment updates, the output head is applied to z_{H}.

Adaptive Halting and the Carry. After each segment an ACT-style halting head (Graves, 2017) predicts whether to halt or continue, up to $M_{\max}$=16 segments. The pair (z_H, z_L) propagated between segments is the *carry*; it is reset when the halting head fires or when the input $\mathbf{x}$ changes. We index segments by $t \in \{0, \ldots, 15\}$ and refer to t as the *recursive step*.

Training Procedure. HRM is trained with three ingredients. **Deep supervision**: a loss $\mathcal{L}$ is applied after every segment, $\sum_{t=0}^{M_{\max}-1} \mathcal{L}(\hat{\mathbf{y}}^{(t)}, \mathbf{y})$, so each segment learns to produce a usable prediction without backpropagation through earlier segments. **One-step gradient approximation**: only the final z_{L} and z_{H} updates within a segment are differentiated; gradients flow through the output head and these final updates. **Adaptive halting head**: trained with a Q-learning objective so the predicted "halt" value approximates the expected solution quality of the current segment. We do not retrain HRM in this work; we use the publicly released Sudoku-Extreme checkpoint and run all 16 segments unless stated otherwise.

A.2. Sudoku-Extreme Dataset Construction

The Sudoku-Extreme dataset is constructed from publicly available Sudoku corpora and filtered to keep only puzzles whose human-solver rating is "extreme" difficulty, i.e., puzzles requiring multi-step logical deductions and long sequences of constraint propagation. We follow the original HRM training procedure (Wang et al., 2025a) and build a 1,001,000-puzzle training corpus by drawing a uniform random sample of 1,000 base puzzles and adding 1,000 structure-preserving augmentations per puzzle. Evaluations use a deterministic subset of the first 500 puzzles retrieved from the test data loader.

A.3. Non-Recurrent Baseline Details

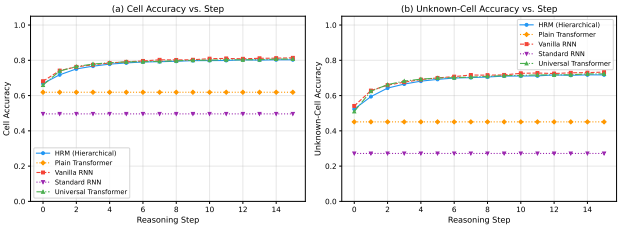

Figure 6. Cell accuracy across ACT steps for all five architectures on the held-out 500-puzzle test set. The recurrent models (HRM, Recurrent Transformer, Universal Transformer) improve monotonically; the non-recurrent baselines (Plain Transformer, GRU) are flat single-pass.

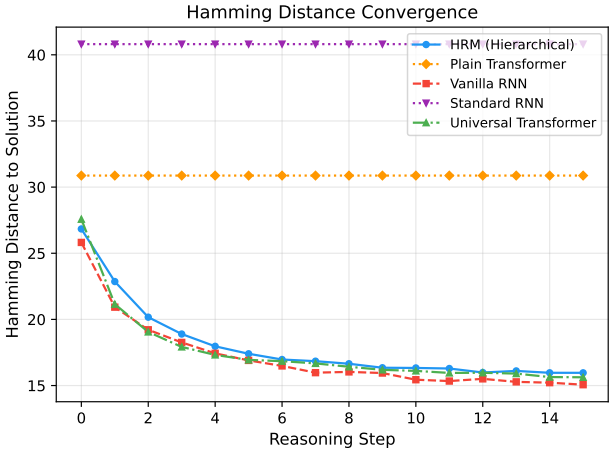

Figure 7. Hamming distance to the ground-truth solution across ACT steps for the same five architectures.

Plain Transformer results across training checkpoints (step 0 output = final output):

GRU results (partial evaluation):

Checkpoint	Cell%	Puzzle%	Hamming
Plain Trans. step 10416	60.9	0.2	31.7
Plain Trans. step 15624	**61.9**	0.2	30.9
Plain Trans. step 20832	61.0	0.4	31.6
Plain Trans. step 26040	61.0	0.4	31.6
Plain Trans. step 31248	60.1	0.4	32.3
Plain Trans. step 36456	60.8	0.0	31.7
Plain Trans. step 41664	60.7	0.0	31.8

Checkpoint	Cell%	Puzzle%	Hamming
GRU step 10416	48.0	0.0	42.1
GRU step 15624	49.2	0.0	41.1
GRU step 20832	49.4	0.0	41.0
GRU step 26040	49.5	0.0	40.9
GRU step 31248	49.5	0.0	40.9
GRU step 36456	49.6	0.0	40.9
GRU step 41664	**49.6**	0.0	40.8

A.4. z_{H} Ablation Per-Step Results

Full per-step zero-ablation results (5,000 puzzles; baseline: 82.6% cell accuracy), corresponding to the experiment summarised in Section 4.2:

Step ablated	Δacc	Accuracy
All steps	-20.5pp	62.1%
Step 0	-7.8pp	74.8%
Step 15	-27.6pp	55.0%

A.5. z_{H} Directional Convergence Metrics

A.6. Cross-Puzzle Patching: Recipient Puzzle

A.7. Activation Patching Details

Cross-puzzle patching results (source puzzle s, target puzzle t):

A.8. Non-Linear (MLP) Probe Comparison

To test whether constraint information is encoded non-linearly, we trained 2-layer MLP probes (512→256→1, ReLU) alongside the linear probes of Section 5.2 on identical data splits. The mean MLP gain across all six targets at step 15 is $<$ 1pp, indicating that the constraint information in z_{H} is essentially linearly accessible.

A.9. Probe PCA Analysis

Variance explained by top 3 principal components of the 4 constraint probe weight vectors:

A.10. SAE Top Features (Full Table)

Table 8 gives the full list of top features retrieved by SAE.

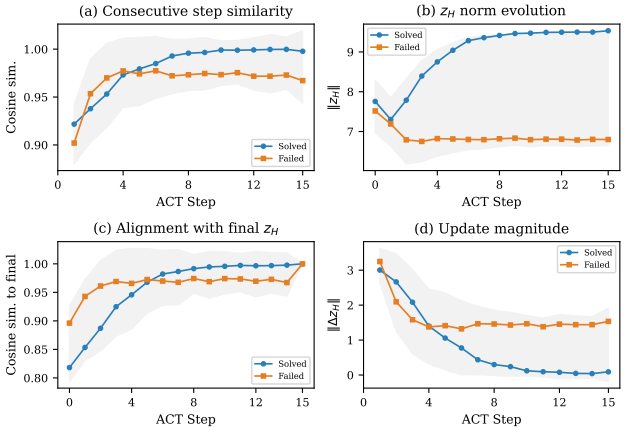

Figure 8. Population-level directional convergence metrics for z_{H} across 16 ACT steps, separated by solved (blue) and failed (orange) puzzles. **Left**: consecutive cosine similarity $\cos(z_{\mathrm{H}}^{(t)}, z_{\mathrm{H}}^{(t-1)})$. **Center**: update magnitude $\|\Delta z_{\mathrm{H}}^{(t)}\|$. **Right**: state norm $\|z_{\mathrm{H}}^{(t)}\|$. Solved puzzles converge (cosine $>$0.99, $\|\Delta z_{\mathrm{H}}\| \to 0$, norms growing to $\sim$9.5) while failed puzzles oscillate ($\|\Delta z_{\mathrm{H}}\| \approx 1.5$ at step 15, norms plateauing at $\sim$6.8).

4	.	.	.	.	1	.	.	9
.	8	.	.	7	.	3	.	.
.	.	3	.	.	.	.	4	.
.	.	.	6	9	.	.	.	1
6	.	.	.	.	.	.	2	.
.	.	1	.	.	2	4	.	.
.	5	.	.	.	.	.	8	.
2	.	.	.	.	9	.	.	4
.	.	7	5	.	.	.	.	.

4	6	2	3	8	1	5	7	9
5	8	9	4	7	6	3	1	2
7	1	3	9	2	5	6	4	8
8	2	5	6	9	4	7	3	1
6	9	4	1	3	7	8	2	5
3	7	1	8	5	2	4	9	6
1	5	6	2	4	3	9	8	7
2	3	8	7	6	9	1	5	4
9	4	7	5	1	8	2	6	3

Figure 9. Recipient puzzle used in Figure 4: given clues (left) and ground-truth solution (right).

A.11. SAE Dead Feature Analysis

Of 2,048 SAE dictionary features: 704 dead (0% fire rate), 1,095 with $< 0.1\%$ fire rate, 65 with 1–10%, 124 with 10–50%, and 38 with $> 50\%$ fire rate.

A.12. Causal Importance Hierarchy Figure

Refer Figure 10

Patched component	Steps	Δacc
z_{H} (early)	1–3	-53.1pp
z_{H} (late)	5–7	-61.7pp
Both levels	all	-65.4pp
Row-masked z_{H}	1–4	-77.8pp

Step	PC1 (%)	PC2 (%)	PC3 (%)
0	79.0	12.4	8.6
4	75.1	13.5	11.4
8	73.8	13.8	12.4
12	74.9	14.9	10.2
15	76.9	13.0	10.1

Table 8. The most-specialized SAE features (d=2,048, λ=0.01). r_{best} is the Pearson correlation with the feature's best-matching symbolic target; r_{next} is the largest absolute correlation against any other target. No feature satisfies $|r_{\text{best}}| > 2|r_{\text{next}}|$.

Feat.	Best target	Peak step	r_{best}	r_{next}
263	box violation	0	$+0.41$	0.40
597	is-given	13	$+0.39$	0.33
1809	is-given	–	$+0.39$	0.21
1542	per-cell correct	–	-0.37	0.34
654	is-given	–	$+0.35$	0.31
211	is-given	–	-0.35	0.33
1317	per-cell correct	–	$+0.33$	0.26
2015	per-cell correct	14	$+0.33$	0.32
2031	box violation	–	$+0.32$	0.32
1715	per-cell correct	–	-0.32	0.28
1576	is-given	–	-0.32	0.23
639	is-given	–	-0.31	0.26
1228	per-cell correct	–	-0.31	0.31
302	is-given	–	-0.31	0.28

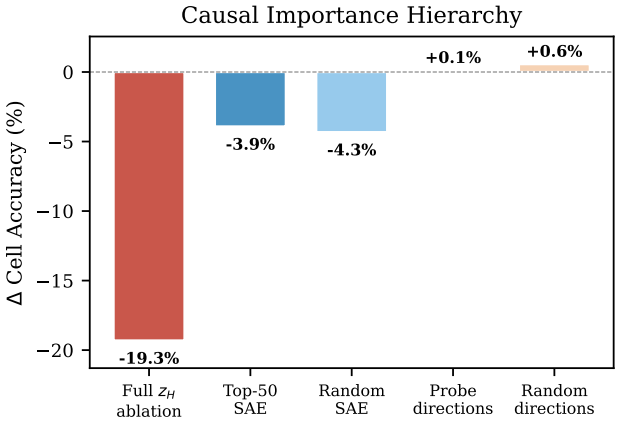

Figure 10. Causal importance hierarchy. Full z_{H} ablation (-19.3%) dominates SAE feature ablation (-3.9%), which in turn dominates probe direction ablation ($\approx 0\%$). Top-50 SAE features are no more causal than random SAE features (p=0.20). Computation is deeply distributed across the entire 512-d z_{H} space.