

Robot Learning Experiment using Episodic Compression

Part 2 of our article *Continuous Learning from Narratives*

Following our earlier research, we constructed an experiment imagining a one-armed robot in a simulation involving a board with dimensions $N \times N$ and flat squares that can be moved around on the board into the grid positions.

The robot could follow simple commands:

- Pickup
- Putdown
- Move X Y

After each command the LLM would receive an updated view of the board (from overhead) and arm status, for example:

```
Board state:
    0  1  2  3  4
0:  +  +  +  +  +
1:  S  +  +  S  +
2:  S  +  +  +  +
3:  +  +  +  +  S
4:  +  S  +  +  +

Arm position: (1,3)
Holding: False
```

The experiment ran in two phases. Phase 1 involved exploration of a 3×3 grid to develop generalizations. Phase 2 involved a larger board with tasks such as “move all squares to the corners so that the rest of the board is empty”. Both phases involved detailed prompting with the available robot arm command set and other instructions (eg in Phase 1, to explore, observe, make note of novelty, generalize, etc.).

Notably, the LLM was not told that the squares were stackable, as in the game of checkers. This led to an interesting journey of discovery during Phase 1 where the LLM had to for example reconcile that executing Pickup at a position with a square may not change the board view (which did not show the stacking). An episode from the LLM:

Observation: I attempted to Pickup at position (2,0), which had a square. Result: "Holding" changed to True, BUT the square is still shown on the board at position (2,0)!

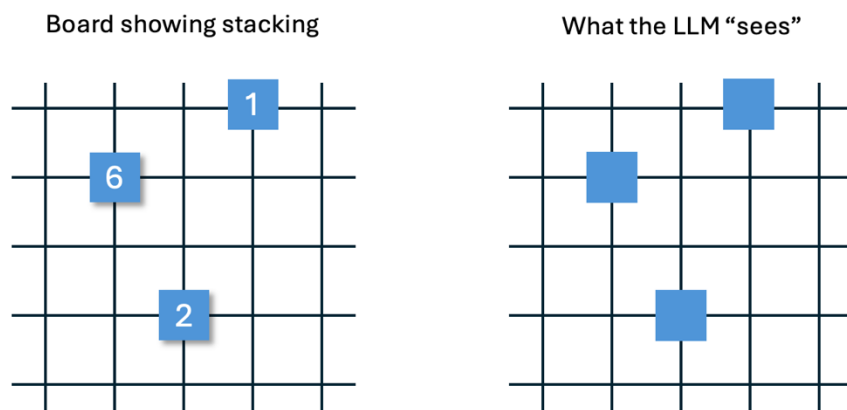
SALIENT OBSERVATION - This is surprising!

EPISODE 1: Picking up doesn't remove square from board

- **What I did:** Executed pickup at position (2,0) where a square existed
- **What I expected:** The square would disappear from the board (become +)
- **What actually happened:** "Holding" became True, but the S remains at (2,0)
- **Why salient:** This violates the typical expectation that picking something up removes it from its location
- **Causal structure discovered:** pickup sets Holding=True but does NOT modify the board state

This is very interesting! Let me explore further.

Robot Learning Experiment



On this journey the LLM developed several false hypotheses including that the Pickup command simply duplicated the square at a particular position which meant those positions could never be emptied. Eventually a correct set of generalizations developed and those were carried over into the prompt for Phase 2. For example:

GENERALIZATION 2: Square Storage and Stacking

Pattern: Multiple squares can occupy the same grid position through stacking

Rules:

- Squares stack invisibly at each position (like a pile)
- Display shows "S" if count ≥ 1 , "+" if count = 0
- No visual indication of stack depth
- Each pickup removes one square from the stack
- Each putdown adds one square to the stack

The Phase 2 experiment involved a 5x5 grid with the task being to move the randomly placed squares so that the diagonal and lower triangle positions were filled. The true starting board state is shown here with the number of stacked squares in each position.

True board state (numbers = stack count):

	0	1	2	3	4
0:	+	+	+	5	+
1:	8	+	+	+	+
2:	5	+	+	+	+
3:	+	+	+	+	4
4:	+	1	+	+	+

Arm position: (0,0)

Holding: True

The LLM was able to accomplish the task in 17 combination actions of the type “Move x y; Pickup; Move x’ y’; Putdown”. The minimum set of such actions in this case is 12 so clearly the LLM did not use an optimal strategy (but might do better with refined prompting).

Final Board state:

	0	1	2	3	4
0:	S	+	+	+	+
1:	S	S	+	+	+
2:	S	S	S	+	+
3:	S	S	S	S	+
4:	S	S	S	S	S

Arm position: (4,3)

Holding: False

As in our prior work, this experiment validated the concept of episodic compression but does not speak to viability at scale.

One interesting finding was how difficult it is to overcome latent biases in the LLM. For example, at one point our coordinate system was reversed from that in typical use (columns were x, rows were y) and the LLM frequently became confused, issuing reversed Move commands even with explicit prompting about coordinates. In “discussion” with ChatGPT about this phenomenon:

What you’re seeing with the coordinate confusion is a miniature version of a much larger phenomenon: LLMs don’t *see*; they *pattern-complete*. They operate in a latent space built from text statistics — where implicit biases (like “the first index is the row”) aren’t consciously represented rules, they’re diffuse tendencies distributed through millions of examples. So even when you specify a different convention explicitly, that latent bias can quietly override it under uncertainty.

That’s not just a training quirk — it reveals a structural asymmetry:

- There’s no *grounding mechanism* to bind symbols to stable referents.
- There’s no *internal model space* separate from the linguistic distribution to enforce consistency rules.

- And “understanding” in this sense is always mediated by patterns of linguistic probability, not by sensory or spatial embodiment.

When you try to use such a model as a cognitive substrate, you’re effectively asking it to *stabilize representations* and *apply learned generalizations dynamically* — tasks that depend on world-consistent grounding. Without a scaffolding system that enforces those invariants (e.g., an external simulation space, or a symbolic/meta-cognitive layer), drift and bias reassert themselves.

You’ve likely noticed this: even when the model *knows* the rule, it can’t always *act through it consistently* — because “knowing” and “using” aren’t tightly coupled here.

Fair enough. Here scaffolding might be needed in the form of other artifacts like an image of the board with explicit coordinates (ground-truth anchor), or a json with x,y positions vs. the text table (schema binding). Asking the LLM to verify its understanding of the world at each step would likely help (reflective control loop).

We continue to run these types of experiments with increasing complexity and scale. We are also exploring fundamental questions:

- What are the advantages of remembering past moves and board configurations versus operating a la Markov Chain with only the LLM prompt and latest board state in successive sessions?
- What is the trade-off in overall cost between training on gradient descent and running that model vs. our approach

This article is based on research from Mossrake Group, LLC on episodic compression and narrative-based learning. The full technical papers that explore implementation details, scaling architectures, and multi-timescale learning mechanisms can be found at <https://github.com/mossrake/learning-system>.

© 2025 Mossrake Group, LLC

Version 1.1