

Mercury: An Addressable Million-Cell Sensor Grid for Browser-Accessible LLM Observability on Consumer Hardware

Chen, Ho Yiing¹

¹Independent researcher, charenix.com, ORCID: 0009-0006-6816-9891
abbychen981008@gmail.com

2026-05-20

Abstract

I present **Mercury**, an open-source observability layer for transformer language models that places up to one million addressable sensors across a model’s hidden state space, captures activation patterns through a standard logits-processor hook, and produces an interactive 3D visualization in a single 2 MB self-contained HTML file. Existing interpretability methods (sparse autoencoders, dictionary learning) typically require auxiliary model training and substantial GPU resources. Mercury runs on consumer hardware (one RTX 3060 12 GB) and completes a full observation pass in 3.5 minutes. I demonstrate Mercury on qwen2.5:7B, identifying $N_{\text{active}} = 14,912$ active sensor cells across ten multilingual prompts and 944 distinct per-cell firing signatures. The a-priori Rosen-bridge topology, designed before any data was collected, recovers 18,113 visible co-firing edges where chance would predict 1,313, a $13.8\times$ enrichment ($p < 10^{-50}$).

1 Introduction

There is a gap between “I do not understand this model” and “I cannot see this model.” Mechanistic interpretability has been gnawing at the first gap for years. This paper takes a swing at the second.

The setup is simple. Pick a transformer language model. Decide on a lattice of *addressable sensors*: discrete units that sit at known coordinates in the model’s hidden state space. Run a prompt through and record which sensors fired, then repeat over a small prompt set. The output is a heat map at the resolution you chose. I usually pick one million units.

The Mercury toolkit makes all of that take 3.5 minutes on a desktop GPU.

The premise is that visibility comes first. Before I can ask whether a model has internal language lanes [2], or whether scaling laws extend to internal specialization [3], I need a picture I can point at. Existing tools either render the architecture without live data [7, 6], or render learned features only after a costly preprocessing step [4, 8], or expose Python hooks and leave the rendering to me [5]. None of them open in a browser, on commodity hardware, in under five minutes.

This work fills that gap. The contributions are:

1. A placement scheme that puts $L \cdot H \cdot Q$ addressable sensors across L transformer layers, H hidden dimensions, and Q quantile buckets, with a 24-byte-per-cell catalog that is byte-for-byte deterministic from the grid spec.
2. An a-priori topology over those sensors, the Rosen-bridge graph, designed without looking at any activations. It encodes the prior that adjacent dimensions and adjacent quantiles are likely to co-fire.

3. A logits-processor observer that runs end-to-end on one RTX 3060 12 GB and completes a ten-prompt observation pass in 3.5 minutes for a 1,003,520-cell grid over qwen2.5:7B.
4. A companion 3D viewer [1] that renders the resulting heat map and bridges in a single 2 MB self-contained HTML file.

I want the bar to be low. Mercury is not the most precise interpretability tool published. It is the most accessible one I could build in a 24-hour sprint with the hardware I owned.

2 Related Work

The closest neighbours of Mercury fall into three groups.

Dictionary learning and sparse autoencoders. Anthropic’s monosemanticity work [4] trains an auxiliary SAE on top of a frozen transformer and produces a feature dictionary of monosemantic directions. The picture is rich and the features are interpretable, but every model and every layer of interest needs its own training run, and the GPU bill is non-trivial. I do no auxiliary training here; the live model is observed through a hook.

Inspection libraries. TransformerLens [5] exposes Python hooks into every layer of a transformer and lets the user query whatever they want. As a library, it leaves visualization, persistence, and shareability explicitly out of scope. I borrow the same hook idea and add a fixed addressable lattice plus a single-file viewer on top.

Architecture and feature visualizers. Bycroft’s web visualizer [7] draws the architecture of a transformer in three dimensions but shows no live data. BertViz [6] visualizes attention. Activation Atlas [8] maps learned features after a trained dimensionality reducer. None of these can read raw activations from a running local model in under five minutes on commodity hardware.

What I inherit from each: the hook-based capture pattern from TransformerLens, the visual-first stance from Activation Atlas, the single-file portability from Bycroft. What I add: a fixed coordinate system, an a-priori topology, and a release-quality artifact for independent researchers.

3 Architecture

The pipeline has four pieces:

lattice build $\longrightarrow$ observer hook $\longrightarrow$ heat counter $\longrightarrow$ viewer.

The first three pieces share a single mmap-backed binary region. The fourth reads from it.

3.1 The UMA region

A single contiguous on-disk file, the *Unified Mercury Address* (UMA) region, holds three byte arrays:

- the **catalog**: 24 bytes per unit, packing the coordinate (L, D, Q) and the calibrated activation threshold;
- the **bridges**: a packed uint32 adjacency list for the Rosen-bridge graph;
- the **heat counter**: a writable uint16 per unit, incremented by the observer on each fire.

The whole UMA region is byte-addressable. A unit ID $k \in [0, N)$ maps to catalog offset $24k$ with no indirection. The observer needs no map structure, only a base pointer and an index.

For qwen2.5:7B ($L = 28$, $H = 3584$, $Q = 10$) the catalog is 24,084,480 bytes describing 1,003,520 units, the edges file is 23,791,040 bytes, and the heat counter is 2,007,040 bytes (uint16). Total UMA footprint: 49.9 MB on disk, mmap’d at runtime.

3.2 Sensor placement

A lattice is specified by three integers and a quantile schedule $\mathbf{q} = (q_1, \dots, q_Q) \in [0, 1]^Q$:

$$\mathcal{L}(L, H, Q, \mathbf{q}) = \{ (l, d, j) : l \in [0, L), d \in [0, H), j \in [0, Q) \}.$$

The total sensor count is

$$N = L \cdot H \cdot Q.$$

Sensor (l, d, j) fires on a given token iff the activation at layer l , dimension d exceeds the per-unit threshold $\tau_{l,d,j} = F_{l,d}^{-1}(q_j)$, where $F_{l,d}$ is the empirical CDF of the activation at (l, d) over a calibration corpus. Sensors with $j = 0$ are the least selective (50th percentile by default); those with $j = Q - 1$ are the most selective (99.5th percentile).

The same dimension is therefore covered by Q sensors at different selectivity levels. This costs a factor of Q in storage but yields a free quantile axis in the visualization.

3.3 The Rosen-bridge graph

The edge graph is a hypothesis encoded in geometry. Two sensors (l_1, d_1, j_1) and (l_2, d_2, j_2) are connected by a Rosen bridge iff

$$|l_1 - l_2| \leq 1 \wedge |d_1 - d_2| \leq 1 \wedge |j_1 - j_2| \leq 1 \wedge (l_1, d_1, j_1) \neq (l_2, d_2, j_2).$$

The prior: nearby sensors in the (L, D, Q) lattice are more likely to co-fire than distant ones. Each interior sensor has 26 neighbours (the corners of a $3 \times 3 \times 3$ cube minus itself); boundary sensors have fewer. Across the full 7B lattice, the edge count is

$$E = 5,947,760,$$

small compared to the $\binom{N}{2}$ possible pairs.

Edges are computed once at build time and never rebuilt. If the prior is correct, observed co-firing pairs should land disproportionately inside the edge set. Section 5 shows the enrichment is $13.8\times$.

3.4 The observer hook

The observer captures activations through a logits-processor hook. The hook receives the model’s last-layer hidden state at every generated token. For each token t and each sensor (l, d, j) with l equal to the hooked layer:

$$\text{fire}_t(l, d, j) = \mathbb{I}[h_{t,d} > \tau_{l,d,j}],$$

where $h_{t,d}$ is the activation at dimension d on token t . When $\text{fire}_t = 1$, the heat counter increments and the sensor ID is appended to the per-prompt fired-units buffer.

This release observes only the last transformer layer (“Tier-A”). Section 7 discusses the Tier-B extension that hooks all L layers.

4 Implementation

4.1 Catalog record

Each catalog record is 24 bytes, laid out as:

Offset	Type	Field
0	uint32	sensor ID k
4	uint16	layer l
6	uint16	quantile bucket j
8	uint32	dimension d
12	float32	threshold $\tau_{l,d,j}$
16	float32	calibration mean (reserved)
20	float32	calibration stddev (reserved)

Byte-aligned. A reader that knows L, H, Q can decode any record without scanning.

4.2 Edge record

Edges store $u \rightarrow v$ as a single uint32 per pair with v packed into the low 32 bits and u recoverable from a separate offset array (CSR layout). For the 7B lattice, total edge storage is $4 \cdot E = 23,791,040$ bytes.

4.3 Heat counter

The heat counter is a uint16 per unit. Per-prompt observations sum to under 2^{16} in practice (a prompt of 100 tokens can light a unit at most 100 times). Across ten prompts the counter still does not saturate. If a future use case pushed past 65,535 firings per unit, the counter type would change to uint32; the catalog layout would not.

4.4 Build cost

The build is a Python script that writes the catalog and edge files in one pass. The full 7B lattice (1,003,520 sensors, 5,947,760 edges) builds in

$$T_{\text{build}} = 2.49 \text{ seconds.}$$

The 3B lattice (368,640 sensors, 2,211,840 edges) builds in 0.94 seconds. Both numbers were measured on the same RTX 3060 host but neither uses the GPU.

5 Demonstration on qwen2.5:7B

I built the 1M-sensor lattice for qwen2.5:7B (commit fd43cd8 on the project repository), then ran a ten-prompt observation pass. The prompts span four topics (physics, algorithms, machine learning, casual) crossed with two languages (Chinese, English), matched one-for-one between the 3B and 7B observations released in [1].

5.1 Headline numbers

Quantity	Value
Lattice size N	1,003,520
Sensors active in any prompt	14,912
Distinct per-unit firing signatures	944
A-priori edges	5,947,760
Visible edges (both endpoints active)	18,113
Observation wall time (10 prompts)	3.5 min
Lattice build wall time	2.49 s

The active fraction is small: $14,912/1,003,520 = 1.49\%$ of the lattice lit up across the entire prompt set. Of those active sensors, 944 distinct firing signatures appeared, where a signature is a binary string of length ten encoding which prompts activated the sensor.

5.2 Edge enrichment

The interesting question for the topology is whether observed co-firing pairs land on the edge set more often than chance. Let A be the set of active sensors with $|A| = a$. The expected number of edges fully inside A under a uniform null is

$$\mathbb{E}[E_{\text{null}}] = E \cdot \binom{a}{2} / \binom{N}{2}.$$

For the 7B observation, $a = 14,912$, $N = 1,003,520$, $E = 5,947,760$:

$$\mathbb{E}[E_{\text{null}}] \approx 5.95 \times 10^6 \cdot \frac{14,912 \cdot 14,911/2}{1,003,520 \cdot 1,003,519/2} \approx 1,313.$$

The observed visible-edge count is 18,113, a $13.8\times$ enrichment. A hypergeometric test on the same numbers gives $p < 10^{-50}$.

The interpretation: the a-priori topology was right enough that observed co-firing concentrates massively on the designed adjacencies. The prior survives the data.

5.3 Wall-time breakdown

Per-prompt wall times across the ten prompts ranged from 17 to 20 seconds on the RTX 3060 12 GB. Summed: 3 minutes 31 seconds. The hook overhead is dominated by the GPU forward pass; the observer adds only a per-token threshold comparison and a counter increment.

6 Visualization

The viewer is described in detail in the companion paper [1]; I include here only the lines that matter for the method. The output of one observation pass is a small set of binary files. A Python script reads them, applies the cylindrical coordinate transform

$$(x, y, z) = \left((Q_k + 1) \cos\left(\frac{2\pi D_k}{H}\right), c_k, (Q_k + 1) \sin\left(\frac{2\pi D_k}{H}\right) \right),$$

where c_k is the number of prompts in which sensor k was active, and writes a 2,031,982-byte self-contained HTML file with inlined Three.js [9]. The file opens in any browser, runs offline, exposes a lane filter and a hover inspector, and renders all 14,912 active points and 18,113 visible edges at interactive frame rates.

7 Limitations

This release observes only the last transformer layer. Earlier layers exist in the catalog but are not yet hooked, because the Python observer routes through `logits_processor`, which only sees the final residual stream. A Tier-B build that hooks every layer is in preparation and requires a patched `llama.cpp`. The catalog and viewer already encode all $L \cdot H \cdot Q$ sensors, so Tier-B will not need a new format.

The current threshold calibration is approximate. The released lattices ship with $\tau_{l,d,j}$ filled by a coarse first pass over a small calibration corpus; the quantiles are close to the targets but not provably tight. A second-pass calibration is on the to-do list.

Demonstration has covered only the Qwen2.5 family at two sizes. Multi-family comparison (Llama 3, Mistral, Yi at comparable scales) is out of scope for this paper.

8 Conclusion

The toolkit makes the inside of a 7B-parameter transformer visible on a desktop GPU in under five minutes, end to end. The lattice is addressable, the topology is fixed before any data is seen, and the

visualization is a 2 MB HTML file. The a-priori prior turns out to be 13.8 times more accurate than random over the 7B observation.

The point is not that this is the best interpretability tool. The point is that it is one an independent researcher can run alone, at home, in an afternoon.

Acknowledgements

The Mercury pipeline and this paper were drafted in a 24-hour sprint across 2026-05-19 and 2026-05-20, in collaboration with an Anthropic Claude Code session acting as a pair-engineer. Every numeric claim in this paper traces to a git commit hash and an on-disk binary in the companion Zenodo dataset record [1]. If a number does not trace, that is a bug, and I want to know.

References

- [1] Chen, Ho Yiing. Mercury-Viewer: A self-contained browser-based interactive 3D visualization of LLM internal sensor maps. Zenodo, 2026. DOI: 10.5281/zenodo.20312290.
- [2] Chen, Ho Yiing. Unsupervised discovery of language and topic lanes in transformer models via multilingual co-firing signatures. In preparation, 2026.
- [3] Chen, Ho Yiing. Internal specialization scales with model size: Quantitative evidence from qwen2.5 3B vs 7B vs 14B. In preparation, 2026.
- [4] 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., Tamkin, A., Nguyen, K., McLean, B., Burke, J., Hume, T., Carter, S., Henighan, T., and Olah, C. Towards Monosemanticity: Decomposing Language Models With Dictionary Learning. Transformer Circuits Thread, Anthropic, October 2023. <https://transformer-circuits.pub/2023/monosemantic-features>.
- [5] Nanda, N. and Bloom, J. TransformerLens. <https://github.com/TransformerLensOrg/TransformerLens>, 2022.
- [6] Vig, J. A Multiscale Visualization of Attention in the Transformer Model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 37–42, Florence, Italy, 2019. DOI: 10.18653/v1/P19-3007.
- [7] Bycroft, B. LLM Visualization. <https://bbycroft.net/llm>, December 2023.
- [8] Carter, S., Armstrong, Z., Schubert, L., Johnson, I., and Olah, C. Activation Atlas. *Distill*, 4(3):e15, 2019. DOI: 10.23915/distill.00015.
- [9] Cabello, R. (mr.doob) et al. Three.js: JavaScript 3D library. <https://threejs.org>, 2010–present.