

# Mercury Repository-to-Neuron Addressability

Chen, Ho Yiing (norika)<sup>1</sup>

<sup>1</sup>Independent Researcher, Taiwan, ORCID: [0009-0006-6816-9891](https://orcid.org/0009-0006-6816-9891), norika at  
charenix.com

Draft: June 18, 2026

## Abstract

Most interpretability views begin after the software has already become a tensor. Mercury refuses that cut because the cut is where evidence gets lost. In the Qwen 2.5 3B city view, files, modules, layers, dimensions, attention heads, and measured neuron sites can live in one address chain; this paper formalizes that chain as something a reviewer can test. The core construction is an address lift from code-side structure to model-side structure:  $(r, f, o) \rightarrow (m, \ell, u, d, h)$ . A visible dot is not useful unless it can be traced backward to a file or module handle and forward to a measured site. I define address retention, dependency reach, broken-handle penalty, and link completeness as audit equations.

The claim is simple: an LLM atlas becomes more than a neuron picture when the path from codebase to model interior remains clickable.

## Author Note

This paper is the one I should have written earlier. The city is not only a stack of activation sites. It also has streets made of files, imports, configs, tools, tests, assets, and broken handles. If those streets are cut away, the atlas loses one of its sharpest teeth.

## 1 The Cut That Should Not Happen

Many tools begin with activations, and that is reasonable, but it hides a wound. Before the model is an activation tensor, it is also a codebase, a set of modules, a loading path, a config, a tokenizer, projection matrices, shared K/V contracts, and runtime choices.

When those pieces disappear, the researcher gets a glowing neuron with no memory of how it was built or loaded.

Mercury keeps that memory attached to the visible map.

That is the whole point. A dot must remember its street address before I trust it as evidence.

## 2 Two Address Spaces

Let a code-side node be

$$c = (r, f, o, t),$$

where  $r$  is a repo or package,  $f$  is a file path,  $o$  is an object or symbol, and  $t$  is a role such as entry, core, model, config, util, test, asset, broken, or unused.

Let a model-side site be

$$n = (m, \ell, u, d, h),$$

where  $m$  is the model,  $\ell$  is the depth index,  $u$  is a module slot,  $d$  is a hidden dimension or channel, and  $h$  is an optional attention-head address.

The Mercury lift is a partial map:

$$\Lambda : C \rightharpoonup N.$$

Partiality matters. Some code nodes never reach an activation site. Some measured sites do not have a clean code-side explanation. A system that pretends the map is total is lying.

### 3 Address Retention

For a visible item  $v$ , define

$$a(v) = \begin{cases} 1, & \text{if } v \text{ has a valid source or neural address,} \\ 0, & \text{otherwise.} \end{cases}$$

The view-level address retention score is

$$A(V) = \frac{1}{|V|} \sum_{v \in V} a(v).$$

This number is brutal and useful. A beautiful view with low  $A(V)$  is mostly a poster, while a dense view with high  $A(V)$  has become an instrument.

A low score is not a minor defect. It means the viewer is being asked to admire light without being given a handle.

### 4 Dependency Reach

Code nodes are not isolated dots, because they import, call, configure, test, and break each other. Let

$$G_C = (C, E_C)$$

be the source graph, and let  $G_N = (N, E_N)$  be the neural relation graph. The lifted dependency reach of a code node  $c$  is

$$D_k(c) = |\{n' \in N : \exists c' \in \text{Reach}_k(c), \Lambda(c') = n'\}|.$$

If  $D_k(c)$  is zero, the code node is visible but not yet connected to the model body. If it is large, that node is a bridge, and the map should draw it with a different level of attention.

### 5 Broken Objects

Broken handles matter because they are where false confidence enters. A dead file, unused module, stale config, or failed import can still look important in a graph if the visualization only counts edges and never asks whether the handle resolves.

Define a broken penalty:

$$P_{\text{broken}}(V) = \frac{|\{v \in V : t(v) = \text{broken} \wedge a(v) = 0\}|}{|V| + \epsilon}.$$

The corrected usefulness of a view is then

$$U(V) = A(V) \cdot (1 - P_{\text{broken}}(V)).$$

This is not cosmetic. It protects the atlas from turning a messy codebase into fake certainty.

The ugly cases stay visible. That is how the map earns trust.

## 6 Module Contract

For transformer work, the key bridge is often a module path:

$u \in \{\text{input\_layernorm}, \text{q\_proj}, \text{k\_proj}, \text{v\_proj}, \text{o\_proj}, \text{gate\_proj}, \text{up\_proj}, \text{down\_proj}, \text{final\_norm}\}.$

For a site  $n = (m, \ell, u, d, h)$ , define module completeness:

$$M(n) = \mathbf{1}[\ell \text{ exists}] \cdot \mathbf{1}[u \text{ exists}] \cdot \mathbf{1}[\text{shape}(u) \text{ is known}].$$

An attention head has another contract:

$$H(n) = (q_{\ell, h}, k_{\ell, g(h)}, v_{\ell, g(h)}, o_{\ell, h}),$$

where  $g(h)$  is the grouped-query K/V owner when the architecture uses GQA. This is why a head panel is not merely a head number. It is a small contract about ownership.

## 7 Why Existing Views Are Not Enough

BertViz made attention inspectable as an interactive visual object [1]. TransformerLens and CircuitsVis made component-level work practical [2, 3]. The Transformer Circuits program gave a strong language for transformer components and residual streams [4]. Activation Atlas showed that features can be spatially explored [5].

Mercury is not trying to replace those instruments. It asks for the missing bridge: can the user click from the visible city object back to source-side structure and forward to a neural address without losing the chain?

## 8 Falsification

The address lift fails whenever the chain cannot be walked. A code node may have no model endpoint:

$$\Lambda(c) = \emptyset.$$

A measured site may be addressable but code-orphaned:

$$\Lambda^{-1}(n) = \emptyset.$$

A link may also exist while carrying the wrong module role. Define role mismatch:

$$R(c, n) = \mathbf{1}[t(c) \not\sim u(n)].$$

The total address failure is

$$F = \frac{|\{c : \Lambda(c) = \emptyset\}| + |\{n : \Lambda^{-1}(n) = \emptyset\}| + \sum_{(c, n)} R(c, n)}{|C| + |N| + \epsilon}.$$

This is the number that keeps the visualization honest. If  $F$  is high, the map is not ready to make strong claims.

## 9 What This Paper Adds

This paper gives Mercury a spine between software and measured sites. Without that spine, a city can still impress; with it, the city can be audited.

The difference is concrete in the actual Qwen atlas. A handle can be seen as a repo cluster, then as a layer project, then as a module entry, then as a dimension or head contract. The viewer does not need to trust a caption, because the caption has an address chain behind it.

## 10 Limits

This paper does not claim that every repo item has a meaningful model role; many items are only packaging, scaffolding, or residue. It also does not claim that code structure fully explains runtime behavior, because a tokenizer, quantization path, or execution backend can change the story.

The narrower claim is stronger: a public LLM atlas should not sever software handles from measured sites. When the chain exists, keep it visible, and when it breaks, mark the break instead of smoothing it away.

## 11 Conclusion

Mercury Repository-to-Neuron Addressability is about keeping the black box open from both sides. Files and measured sites should not live in separate worlds. If a glowing point cannot be traced, it should be treated as scenery; if it can be traced, it becomes a handle, and handles are where real work begins.

## References

- [1] Jesse Vig. A Multiscale Visualization of Attention in the Transformer Model. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2019. <https://arxiv.org/abs/1906.05714>
- [2] Neel Nanda and Joseph Bloom. TransformerLens. 2022. <https://transformerlensorg.github.io/TransformerLens/content/citation.html>
- [3] Alan Cooney and Neel Nanda. CircuitsVis. 2023. <https://github.com/TransformerLensOrg/CircuitsVis>
- [4] Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A Mathematical Framework for Transformer Circuits. *Transformer Circuits*, 2021. <https://transformer-circuits.pub/2021/framework/index.html>
- [5] Shan Carter, Zan Armstrong, Ludwig Schubert, Ian Johnson, and Chris Olah. Activation Atlas. *Distill*, 2019. doi:10.23915/distill.00015. <https://distill.pub/2019/activation-atlas/>