AIMLAI @ ECML PKDD 2026
September 7th 2026

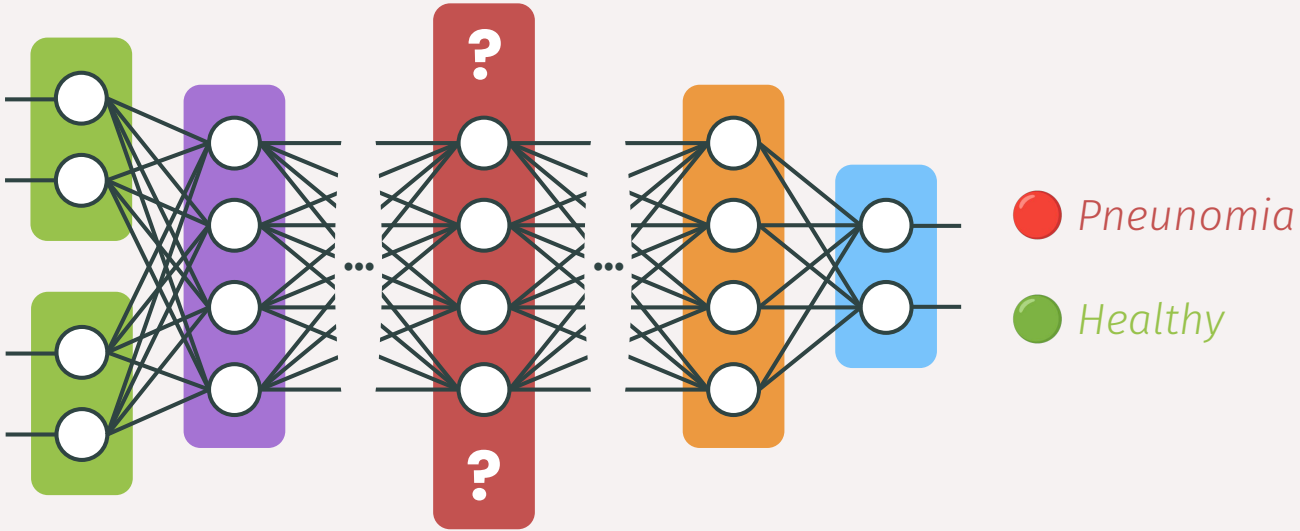
From Classical to Mechanistic Interpretability

Ward Gauderis · José Oramas · Thomas Dooms

VUB ARTIFICIAL INTELLIGENCE RESEARCH GROUP
Universiteit Antwerpen

fwo Flanders AI Research Program

To control, predict, and learn from neural networks, we have to understand their *internal* working.



Monitor & steer

Align & debug

Improve design

Extract knowledge

(Sharkey, *Open Problems in Mechanistic Interpretability*. 2025)

This chapter...

Mechanistic foundations

Ward Gauderis

Methods & Models

José Oramas & Thomas Dooms

Limitations & Open Problems

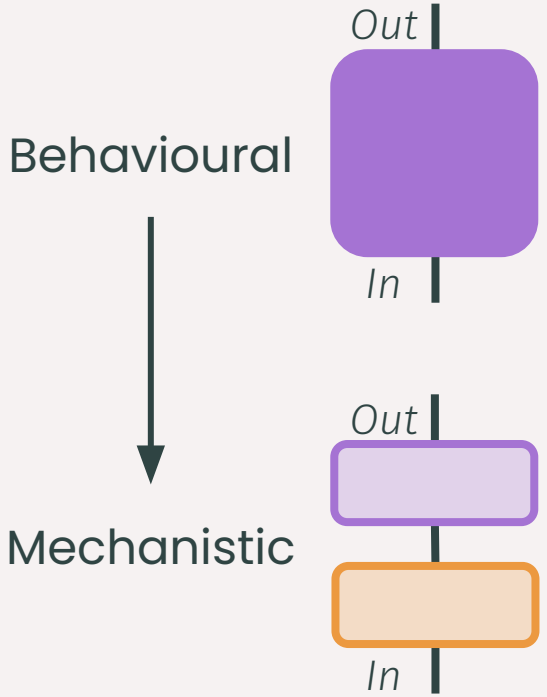
Thomas Dooms

Mechanistic interpretability *reverse-engineers* learned representations into human concepts and algorithms.

Question
How can a model solve a **general** class of problems, not just one instance?

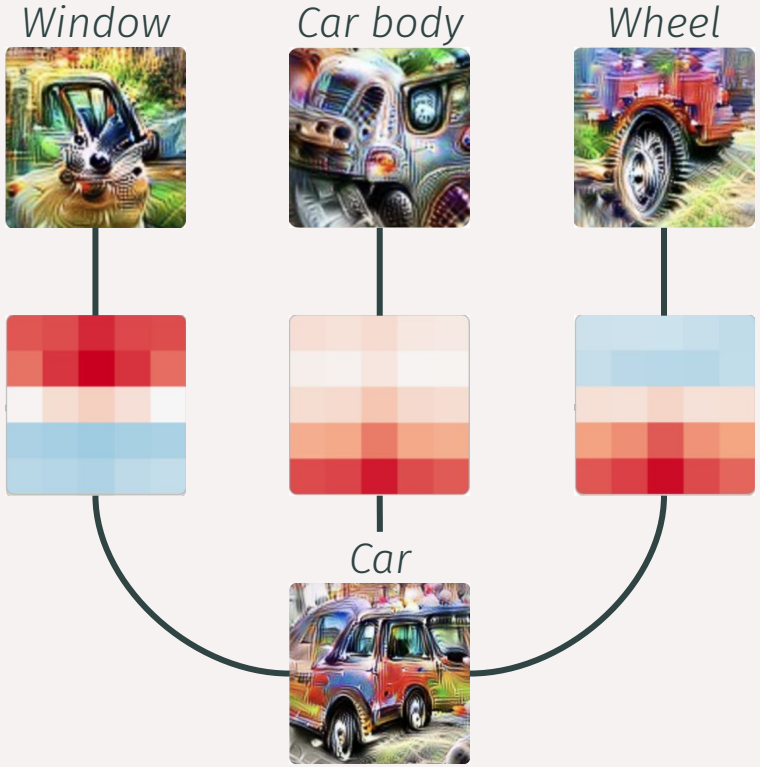
Premise
Generalisation in neural networks arises from shared computational **mechanisms**.

Scope
Any research that describes the internals of a model, including its **activations or weights**.



(Bereska, *Mechanistic Interpretability for AI Safety A review*. 2024; Saphra, *Mechanistic?*. 2024)

Axis 1. Networks are *features*, wired into *circuits*. Both can recur across models and tasks.



1. Object of study

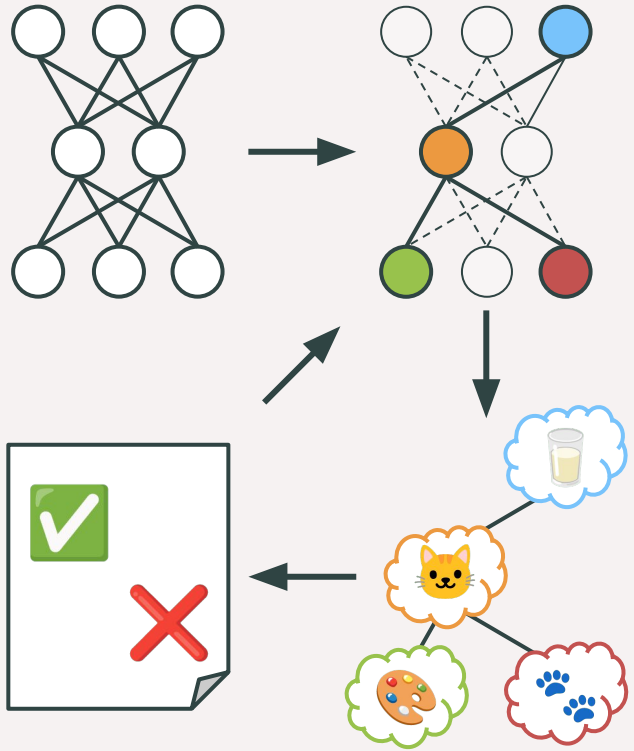
Features
What input properties are encoded and how?

Circuits
How do features compute and interact?

Universality
Are these objects learned across models?

(Olah, *Zoom In: An Introduction to Circuits*. 2020)

Axis 2. Decompose, hypothesise, and test. Reverse-engineering follows the *scientific method*.



1. Object of study

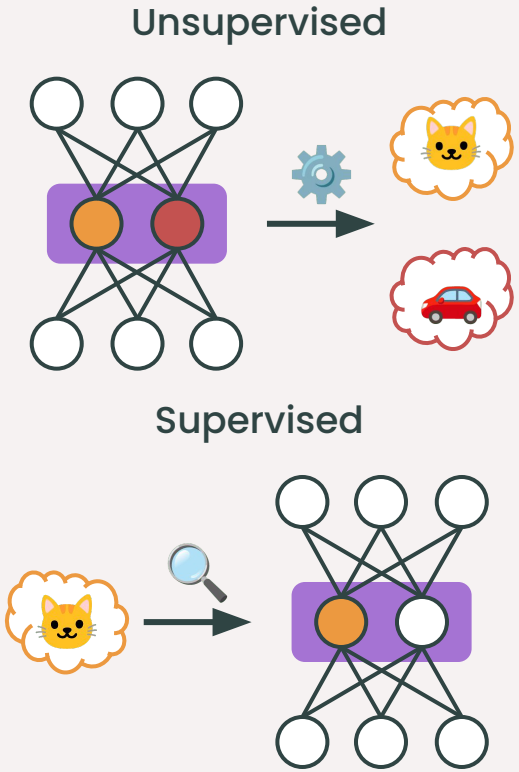
- Features
- Circuits
- Universality

2. Process stage

- **Decomposition** — What are the relevant components?
- **Description** — What does each component do?
- **Validation** — Is a components description accurate?

(Bereska, *Mechanistic Interpretability for AI Safety A review*. 2024; Sharkey, *Open Problems in Mechanistic interpretability*. 2025)

Axis 3. Either we ask what a component means, or we identify where a concept lives.



1. Object of study

Features	Circuits	Universality

2. Process stage

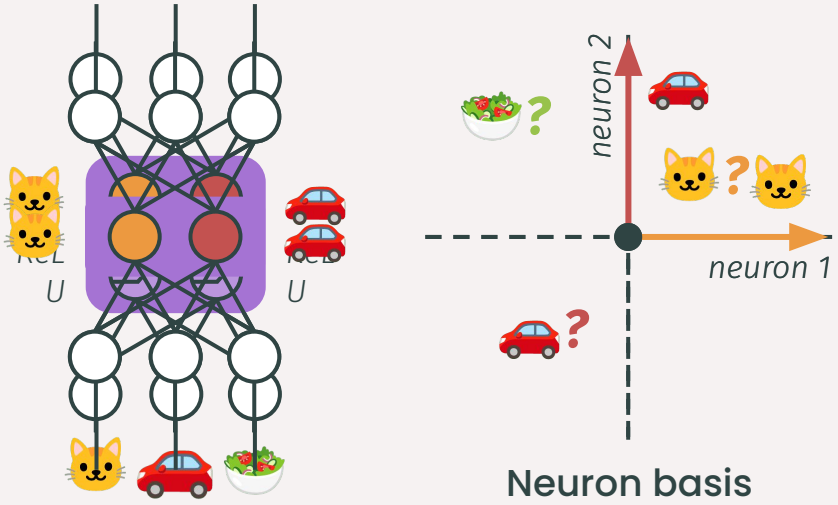
Decomposition	Description	Validation

3. Direction of search

Bottom-up Given a component, what is its role?	**Top-down** Given a role, where is it represented?

(Sharkey, *Open Problems in Mechanistic interpretability*. 2025)

We want features to align with human concepts. Neurons form a *privileged* basis, but are *polysemantic.*

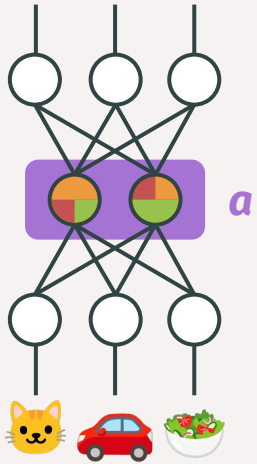


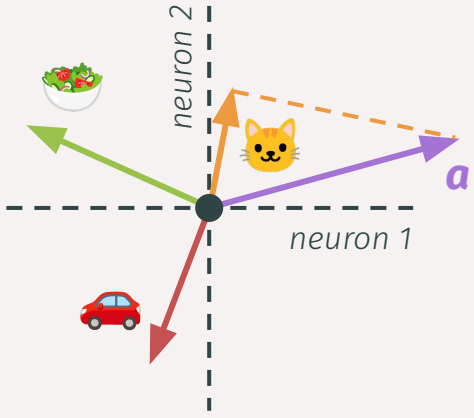
If neurons are features...
A **monosemantic** neuron activates for a single, coherent concept.

...But empirically they are not.
A **polysemantic** neuron activates for multiple unrelated concepts.

(Mu, *Compositional Explanations of Neurons*. 2021; Elhage, *Toy Models of Superposition*. 2022; Scherlis, *Polysemanticity and Capacity in Neural Networks*. 2025)

Under the *Linear Representation Hypothesis (LRH)*, features are linear directions in activation space.





Feature presence is linear projection.

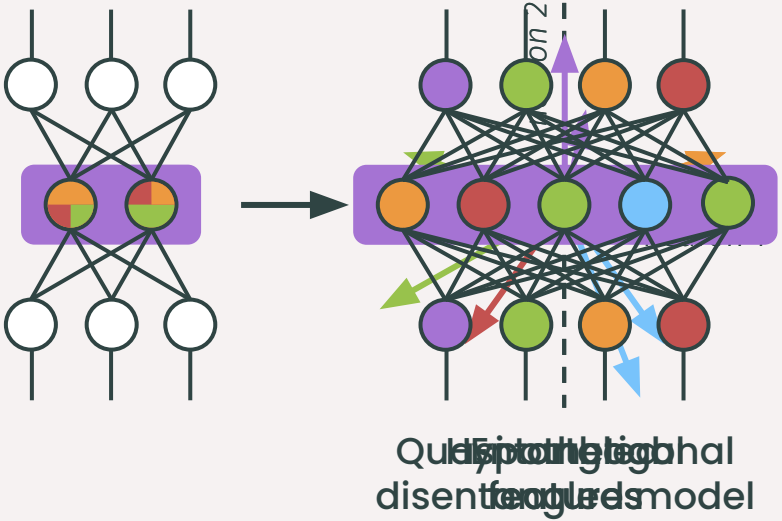
$a_{🐱} \approx \langle d_{🐱} \mid \boldsymbol{a} \rangle$

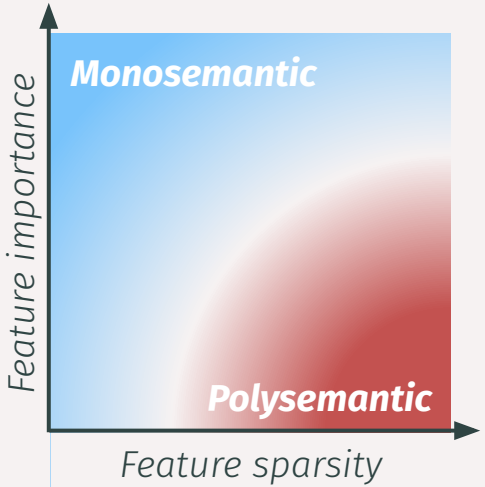
Activations are linear combinations.

$\boldsymbol{a} \approx a_{🐱} \cdot 🐱 + a_{🚗} \cdot 🚗 + a_{🥗} \cdot 🥗$

(Park, *The Linear Representation Hypothesis and the Geometry of LLMs*. 2023)

Under the LRH, *superposition* causes polysemanticity. Activations pack more concepts than neurons.





(Elhage, *Toy Models of Superposition*. 2022; Scherlis, *Polysemanticity and Capacity in Neural Networks*. 2025)

This chapter...

Mechanistic foundations

Ward Gauderis

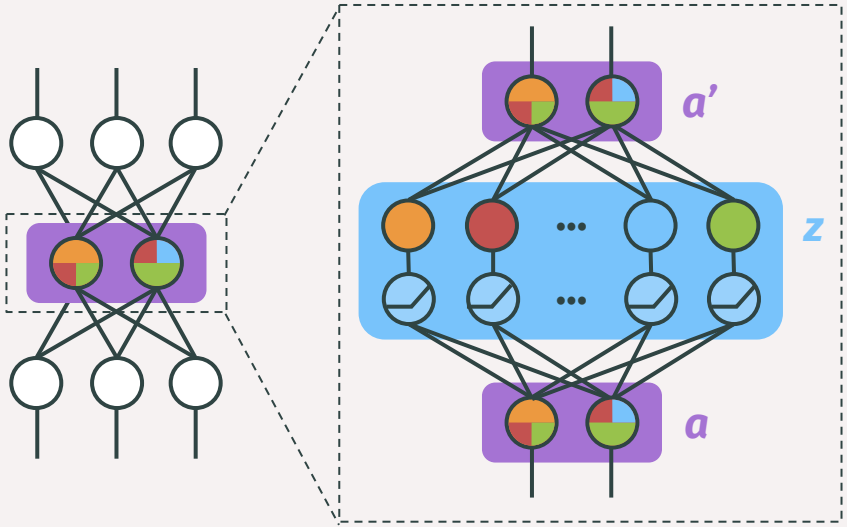
Methods & Models

Ward Gauderis & José Oramas & Thomas Dooms

Limits & Open Problems

Thomas Dooms

Sparse autoencoders learn an overcomplete dictionary of feature directions without labels.



Dictionary learning finds sparse features.

$$\mathbf{a} \approx D\mathbf{z}, \qquad \|\mathbf{z}\|_0 \leq k$$

An encoder replaces the search for $\mathbf{z}$.

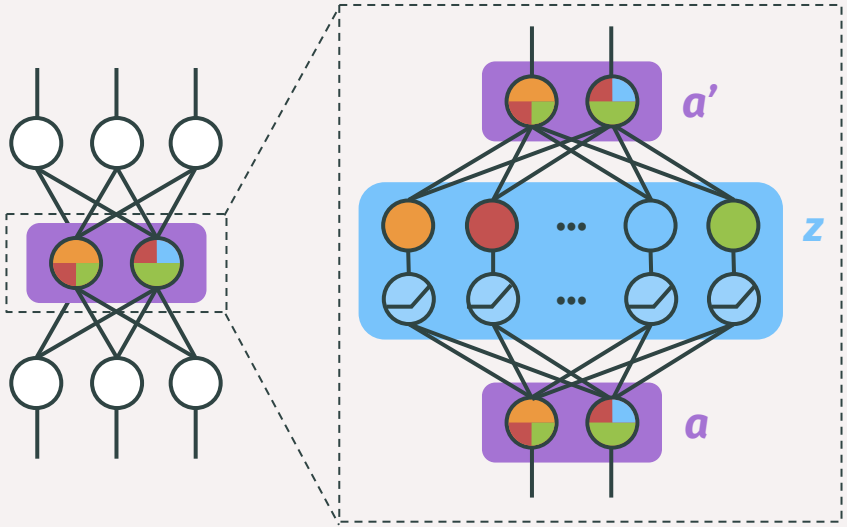
$$\mathbf{z} = \mathrm{ReLU}(E\mathbf{a} + e)$$
$$\mathbf{a'} = D\mathbf{z} + d$$

The loss trades reconstruction and sparsity.

$$L(\mathbf{a}, \mathbf{a'}) = \|\mathbf{a} - \mathbf{a'}\|_2^2 + \lambda \|\mathbf{z}\|_1$$

(Cunningham, *Sparse Autoencoders Find Highly Interpretable Features in Language Models. 2023;* Bricken, *Towards Monosemanticity: Decomposing LMs With Dictionary Learning.* 2023)

Sparsity is not a *proxy* for interpretability. Every feature needs a description and a test.



Features still have to be labelled.

- Decodability does not imply causality.

The decomposition is not unique.

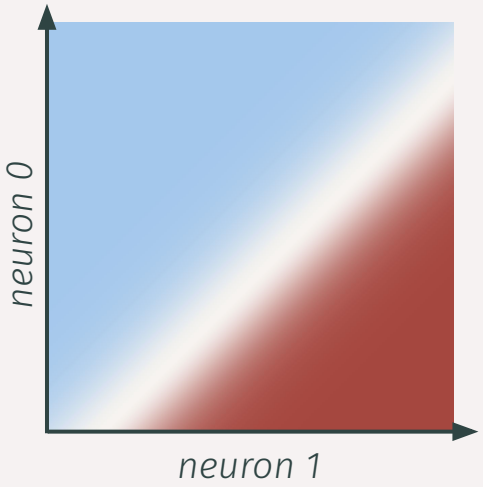
- L1 penalises activation size, not count.
- Dictionary size affects feature granularity.
- Different seeds give different features.

For known concepts, supervision wins.

- Simpler baselines suffice downstream.

(Gao, *Scaling and evaluating sparse autoencoders.* 2024; Bussmann, *BatchTopK Sparse Autoencoders. 2024;* Sharkey, *Open Problems in Mechanistic interpretability.* 2025)

A *probe* uses labels to find the maximal separation between two classes.



Probing asks if a concept is linearly readable.

$y \approx \sigma(wa + b), y \in \{-1, 1\}$

The weights carve a separating hyperplane.

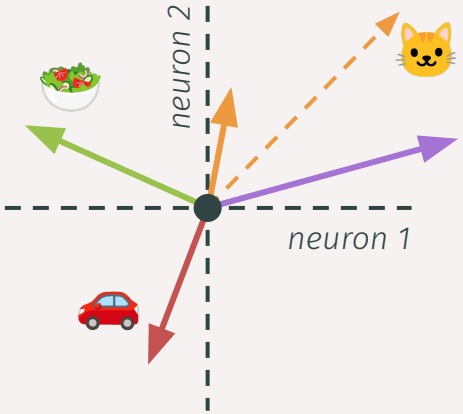
$wa + b = 0$

The loss rewards confidence in the true class.

$L(y, a) = \text{softplus}(-y (wa + b))$

(Alain, Understanding intermediate layers using linear classifier probes. 2016)

A *probe* queries whether representations can *distinguish* a concept, not whether it is used.



Probes can lie about concept presence.

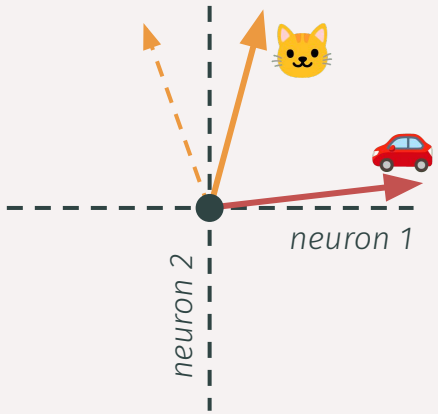
- Decodability still does not imply causality.
- Will pick up on correlations.

Probes miss substructure.

- They answer no questions beyond presence.

(Alain, Understanding intermediate layers using linear classifier probes. 2016)

A *probe* finds the maximal separation, it does not find the concept direction itself.



Probes can lie about concept presence.

- Decodability still does not imply causality.
- Will pick up on correlations.

Probes miss substructure.

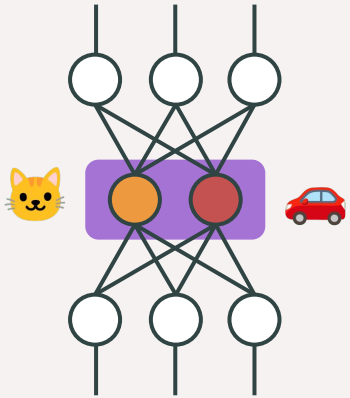
- They answer no questions beyond presence.

Probes find separation; not features.

- Directions don't align with features
- … and should hence not be compared.

(Alain, Understanding intermediate layers using linear classifier probes. 2016)

A *concept bottleneck model* builds the a set of probes into the model.



A bottleneck routes through concepts.

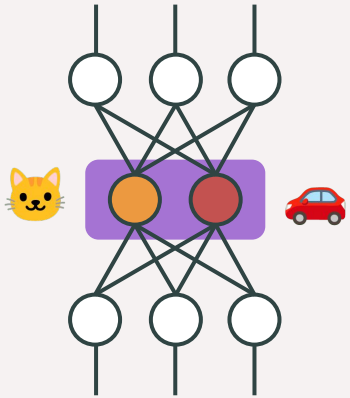
$\hat{c} = g(a), \hat{y} = w\hat{c} + b$

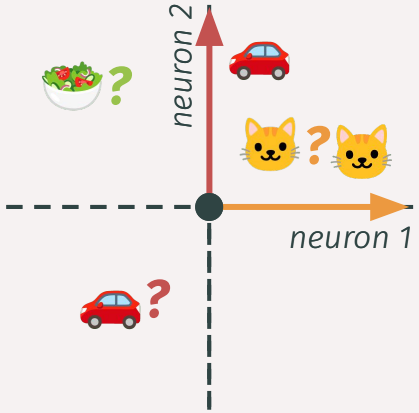
The loss supervises concepts and label jointly.

$L = \ell(\hat{y}, y) + \lambda \Sigma\square\ \ell(\hat{c}\square, c\square)$

(Koh, Concept Bottleneck Models. 2020; Oikarinen, Label-Free Concept Bottleneck Models. 2023; Knab, What's in the Bottle?)

Building in probes inherit probe problems. Concepts leak correlates and are incomplete.





Neuron basis

The concept set must be complete.

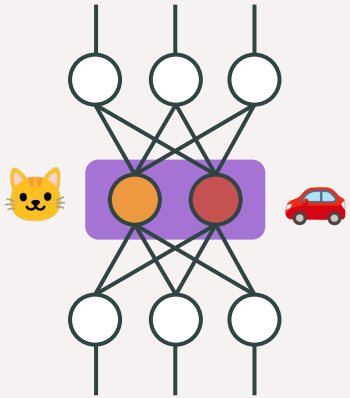
- Missing concepts cap accuracy.
- Residuals can interact weirdly.

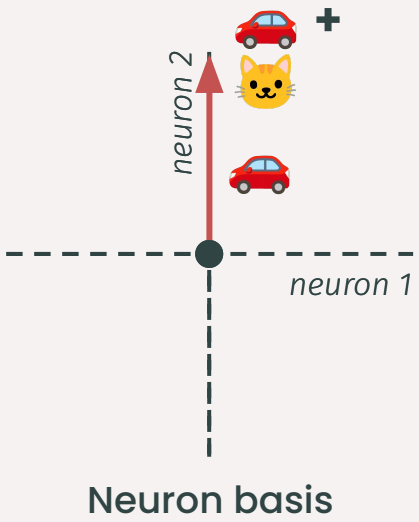
Concept labels are expensive.

- Dense annotation per input
- ‘label-free’ outsources to an oracle.

(Koh, Concept Bottleneck Models. 2020; Oikarinen, Label-Free Concept Bottleneck Models. 2023; Knab, What's in the Bottle?)

Soft bottlenecks still allows *non-linear signal* to go through.





The concept set must be complete.

- Missing concepts cap accuracy.
- Residuals can interact weirdly.

Concept labels are expensive.

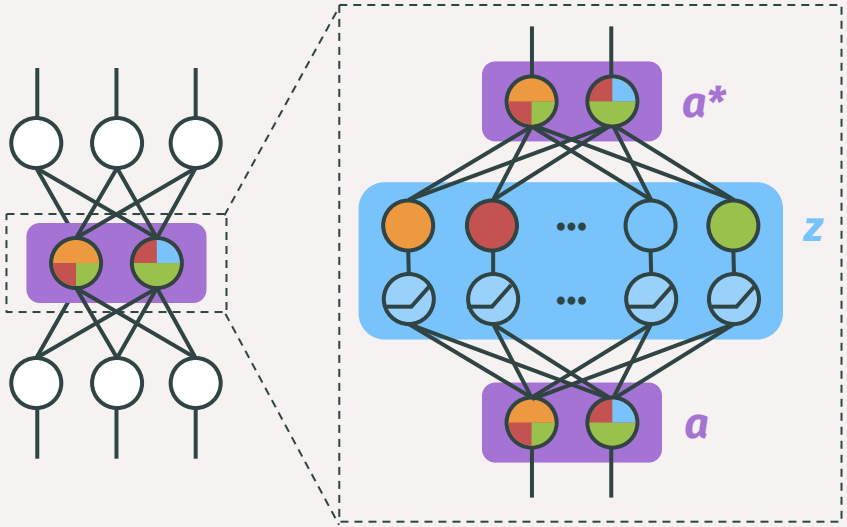
- Dense annotation per input
- ‘label-free’ outsources to an oracle.

Soft concepts leak.

- Carry unlabelled signal past bottleneck.
- Non-linear signal is not penalized

(Koh, Concept Bottleneck Models. 2020; Oikarinen, Label-Free Concept Bottleneck Models. 2023; Knab, What's in the Bottle?)

Transcoders sparsify the computation, they decompose the model into *circuits*.



The dictionary is the same as an autoencoder.

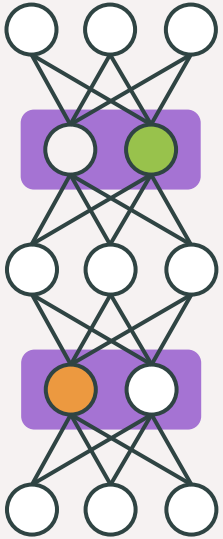
$z = \mathrm{ReLU}(Ea + e)$
$a^* = \mathrm{MLP}(a) \approx Dz + d$

But it reconstructions the layer's output.

$L = \|a^* - Dz - d\|^2 + \lambda\|z\|_1$

(Dunefsky, Transcoders Find Interpretable LLM Feature Circuits. 2024; Ameisen, Circuit Tracing: Revealing Computational Graphs in Language Models. 2025)

Understanding *circuits* allows tracing how a feature is formed.

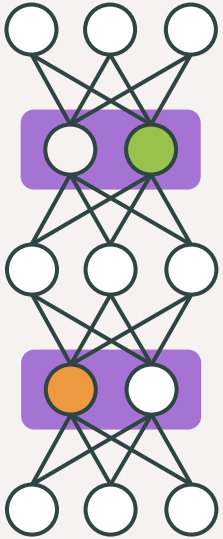


Circuits describe feature interactions.

- Follow how features are computed.
- Test the proposed connections.

(Dunefsky, Transcoders Find Interpretable LLM Feature Circuits. 2024; Ameisen, Circuit Tracing: Revealing Computational Graphs in Language Models. 2025)

Understanding *circuits* allows tracing how a feature is formed.



Circuits describe feature interactions.

- Follow how features are computed.
- Test the proposed connections.

Approximation errors can compound.

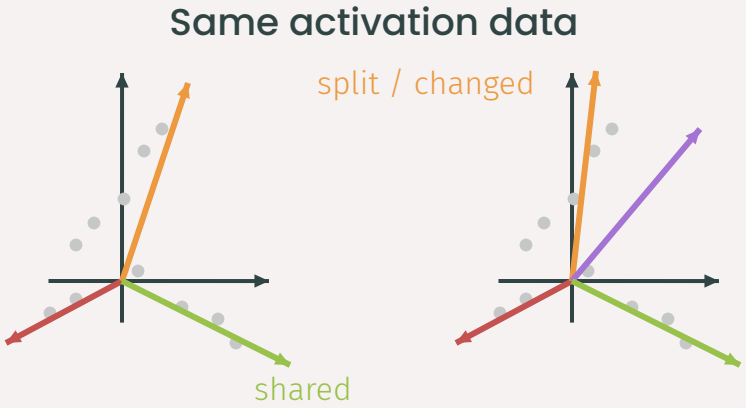
- Layer-wise fits can miss model behaviour.
- Error nodes retain unexplained signal.

SAE caveats still apply.

- Check width, sparsity and stability.
- Sparse activations ≠ sparse circuits

(Dunefsky, Transcoders Find Interpretable LLM Feature Circuits. 2024; Ameisen, Circuit Tracing: Revealing Computational Graphs in Language Models. 2025)

A decomposition need not be unique. *Identifiability* requires assumptions.



Schematic dictionaries

Identifiability asks for uniqueness.

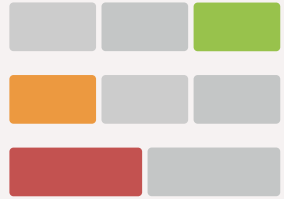
- Allow rescaling and permutation.

Stability tests the fitted solution.

- Vary seeds, data, width, and sparsity.

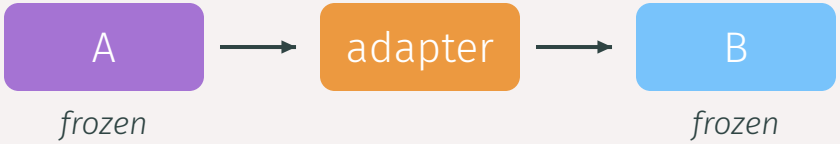
(Hillar & Sommer, *Dictionary Identifiability*, 2015; Paulo & Belrose, *Different Features*, 2025)

Universality depends on the comparison. Features, geometry or function?



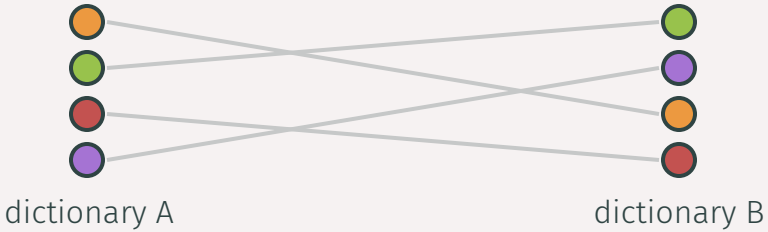
Do models organise inputs similarly?

- Compare the geometry of their activations.



Can their representations work together?

- A learned adapter joins two frozen models.

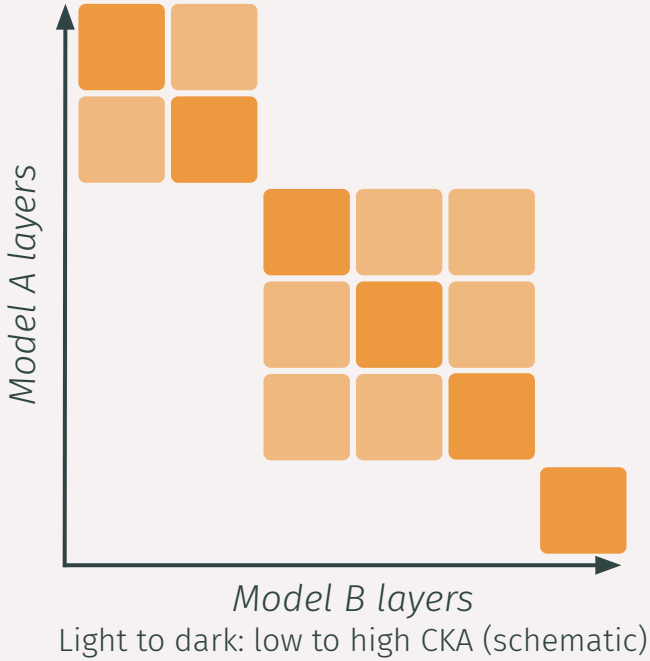


Do they share candidate components?

- Compare features or fit a shared dictionary.

(Kornblith et al., *Representation Similarity*, 2019; Bansal et al., *Model Stitching*, 2021)

Centered kernel alignment compares layers. High similarity does not prove shared circuits.



Light to dark: low to high CKA (schematic)

Use matched input examples.

- Collect and centre activations; feature counts may differ.

Read the layer-by-layer comparison.

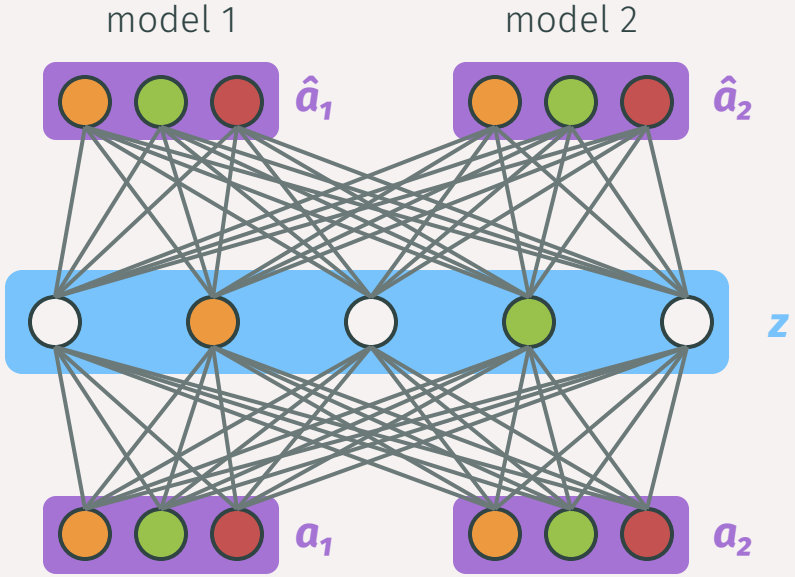
- Darker cells indicate greater similarity; blocks reveal layer groupings.

Similar geometry leaves causal questions.

- Rotations and uniform scaling are ignored; shared roles need separate tests.

(Kornblith, Similarity of Neural Network Representations Revisited. 2019)

Crosscoders learn one shared code. Each site keeps its own decoder.



Paired activations for the same input

One code combines multiple sites.

$$z = \mathrm{ReLU}(E_1 a_1 + E_2 a_2 + e)$$

Each site has its own decoder.

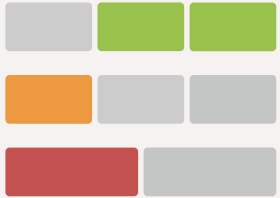
$$\hat{a}_1 = D_1 z + d_1$$
$$\hat{a}_2 = D_2 z + d_2$$

The loss trades reconstruction and sparsity.

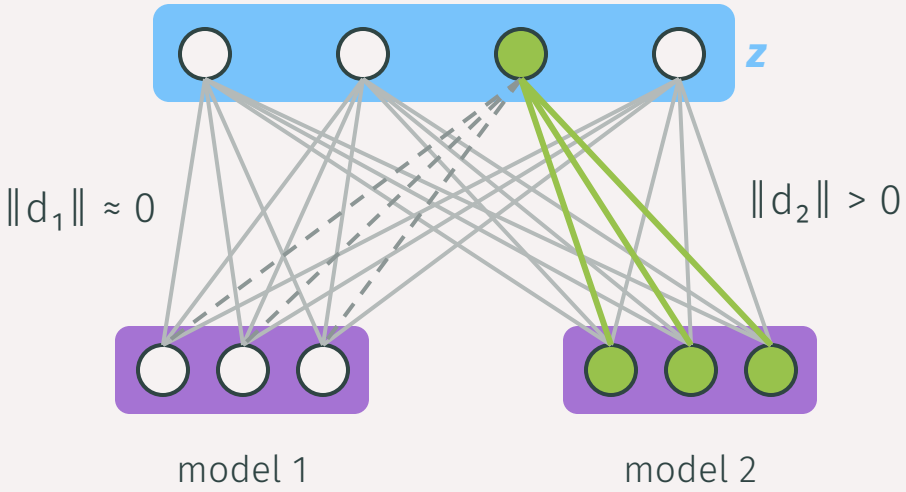
$$L = \|a_1 - \hat{a}_1\|^2 + \|a_2 - \hat{a}_2\|^2 + \lambda R$$

(Lindsey, *Sparse Crosscoders for Cross-Layer Features and Model Diffing*. 2024; Minder, *Overcoming Sparsity Artifacts in Crosscoders*. 2025)

Sparsity can create apparent differences. Model-only latents need targeted checks.



An apparent model-only latent



L1 can create false exclusivity.

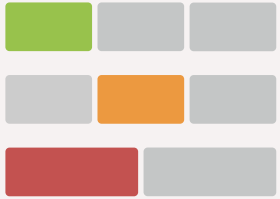
- Shrinkage can erase useful decoder weights.

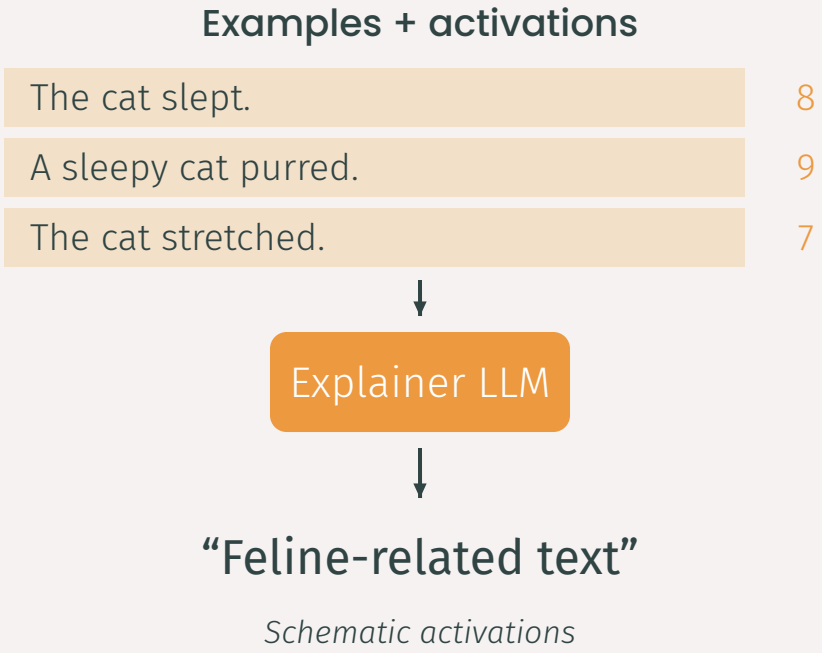
A shared code leaves causal questions.

- Joint encoding does not establish a circuit.

(Lindsey, *Sparse Crosscoders for Cross-Layer Features and Model Diffing*. 2024; Minder, *Overcoming Sparsity Artifacts in Crosscoders*. 2025)

Examples suggest what a feature detects. An LLM turns the pattern into a *hypothesis*.





Input evidence suggests a description.

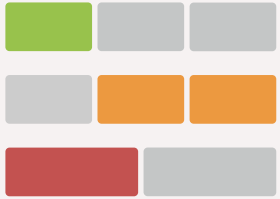
- Inspect activating contexts and token-level attribution.

Output evidence suggests a role.

- Vocabulary projections are clues; interventions test effects.

(Bills et al., 2023; Paulo et al., *Automatically Interpreting Millions of Features*, 2024; Gur-Arieh et al., 2025)

A label must survive new examples. *Counterexamples* reveal its scope.



Hypothesis: feline words

The lion roared.

Any activation? Yes

The lion roared. Correct

The lion roared. Incorrect

Detection checks the context.

- Does any token activate the feature?

Fuzzing checks the marked tokens.

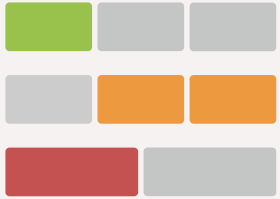
- Are these the positions where it activates?

Compare against a null model.

- Detection and fuzzing can score randomized models well.

(Paulo et al., *Automatically Interpreting Features*, 2024; Heap et al., *Automated Interpretability Metrics*, 2026)

Surprisal tests how much a label helps. It should help most on activating contexts.



Gain = NLL(generic) - NLL(label)

Context	Generic	Label	Gain
cat	12	8	4
lion	13	10	3
dog	11	10.8	0.2

The animal is a ...

Score the description.

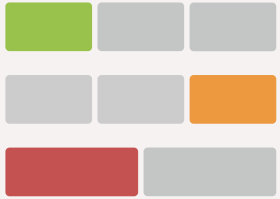
- Measure gain over a generic description.

Rank held-out contexts.

- Use the reduction to separate active from inactive examples.

(Paulo et al., *Automatically Interpreting Millions of Features*, 2024)

Can you spot the *intruder*? Coherence can be tested without a label.



Five held-out contexts

1. The ferry crossed the bay.
2. He took the tram home.
3. She caught the bus today.
4. The apple tasted sweet.
5. The train arrived.

Schematic intruder test

Four positives, one negative.

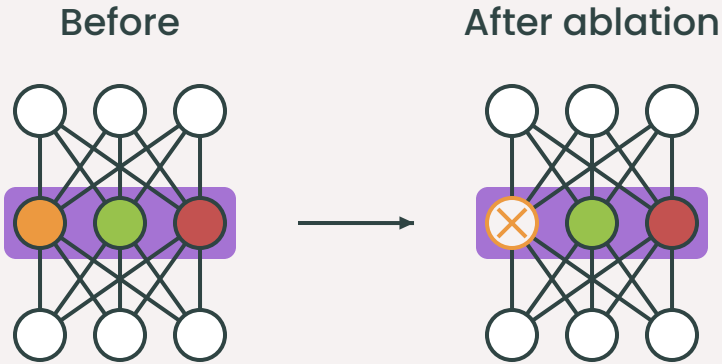
- Choose the example that does not belong.

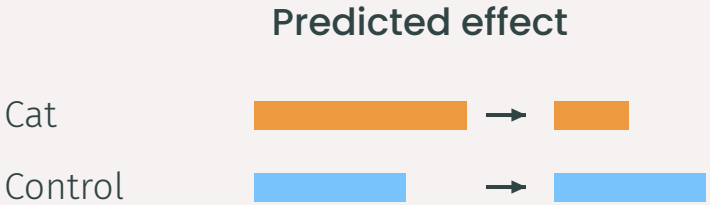
Score accuracy across examples.

- Chance accuracy is 20%.

(Paulo & Belrose, *Evaluating SAE Interpretability Without Explanations*, 2025)

Causal claims predict intervention effects. The setup defines the scope of the test.





Specify the causal hypothesis.

- Set variables, basis, and intervention rules.

Distributed Alignment Search (DAS).

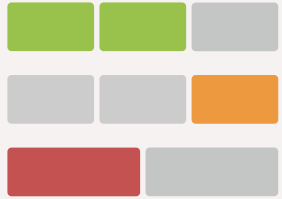
- Align to a fixed causal model.

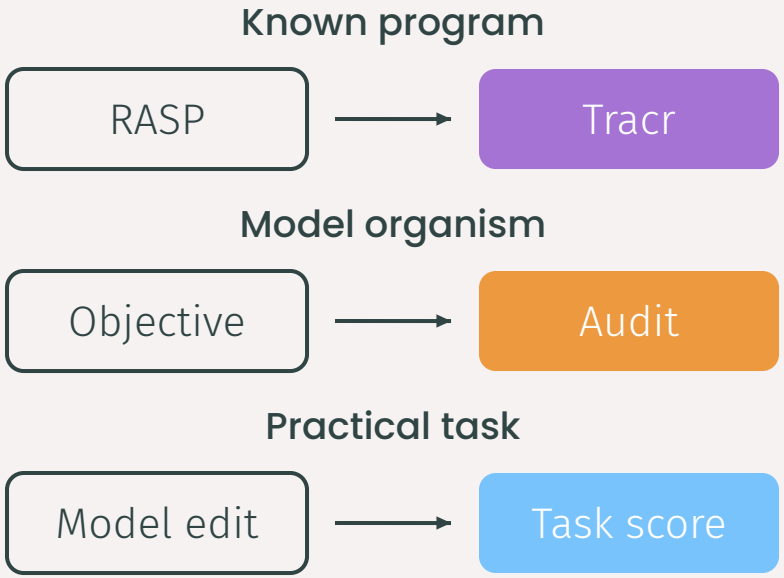
Test predictions against controls.

- Report test scope and uncertainty.

(Geiger et al., *DAS*, 2023; Makelov et al., *Subspace Activation Patching*, 2023)

Validation needs more than one setting. Known mechanisms and real tasks differ.





Known programs test recovery.

- Tracr provides a computation to recover.

Model organisms test failure detection.

- Can an audit find the hidden objective?

Real tasks test usefulness.

- Compare strong baselines for this question.

(Lindner et al., *Tracr*, 2023; Marks et al., *Hidden Objectives*, 2025; Wu et al., *AxBench*, 2025)

This chapter...

Mechanistic foundations

Ward Gauderis

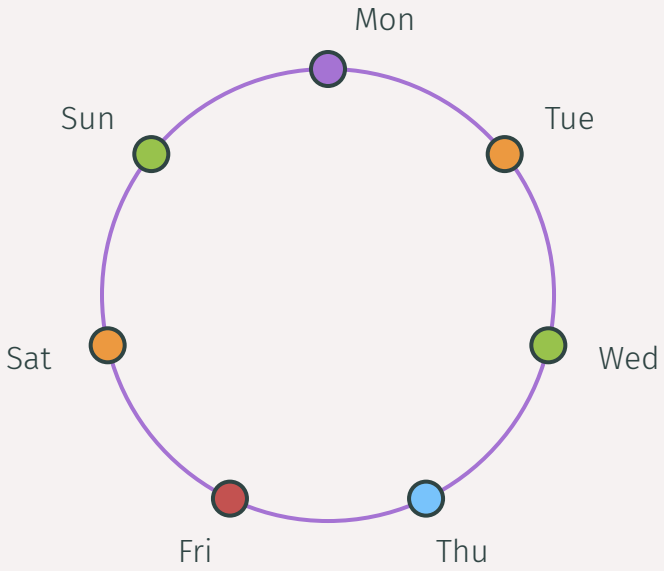
Methods & Models

José Oramas & Thomas Dooms

Limits & Open Problems

Thomas Dooms

Several directions can encode one concept. Interventions test their computational role.



One structured representation

Schematic weekday circle

A feature can have internal geometry.

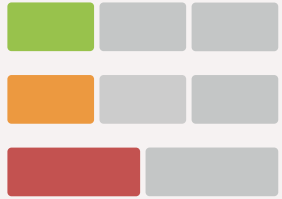
- Weekdays can trace a circle in a plane.

PCA alone does not identify a mechanism.

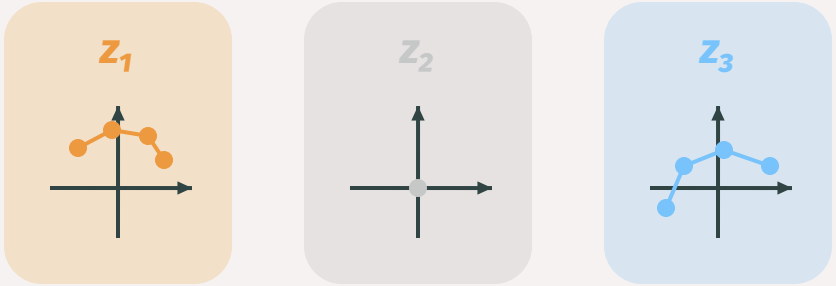
- Intervene on the subspace; test its role.

(Engels et al., *Not All Language Model Features Are One-Dimensionally Linear*, 2024 / ICLR 2025)

Block-sparse featurizers select vectors. Sparsity is imposed across whole blocks.



Select whole vector blocks



$$\hat{x} = D_1 z_1 + D_2 z_2 + D_3 z_3$$

Schematic coordinates across inputs

Select a few vector blocks.

- Top-k keeps the blocks with largest norms.

Keep structure within each block.

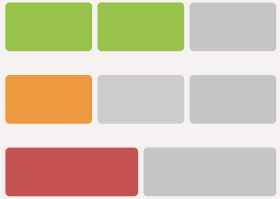
- Signed coordinates can vary together.

Blocks are a modelling assumption.

- A block need not match one concept.

(Fel et al., *Structuring Sparsity*, 2026; Jerpelea & Ananthram, *A Deeper Analysis of BSFs*, 2026)

Weights also admit a decomposition. *APD* searches for reusable components.



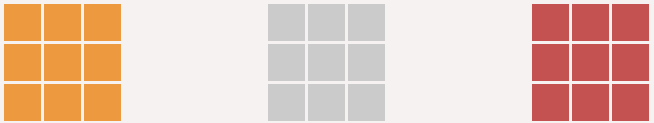
Attention routes OV outputs

QK

OV

APD: additive weight components

$$\theta \approx \theta_1 + \theta_2 + \theta_3$$



active inactive active

QK and OV expose weight circuits.

- QK scores where to attend.
- OV maps what is written.

APD adds reusable components.

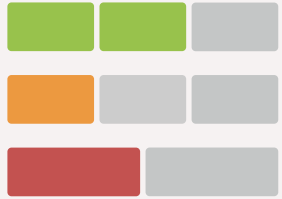
- Rebuild weights; use few pieces per input.

Interactions still matter.

- Parameters add; outputs need not.

(Elhage et al., *A Mathematical Framework*, 2021; Braun et al., *APD*, 2025; Bushnaq et al., *SPD*, 2025)

APD depends on costly fitting and tuning. Its importance scores are approximations.



More components cost more to fit

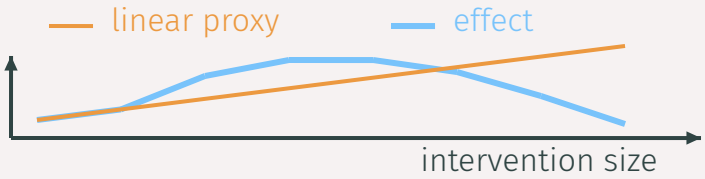
Fitting costs memory and computation.

- More parameter components add overhead.

Sparsity changes the decomposition.

- Tune top-k and simplicity penalties.

A local estimate can miss curvature

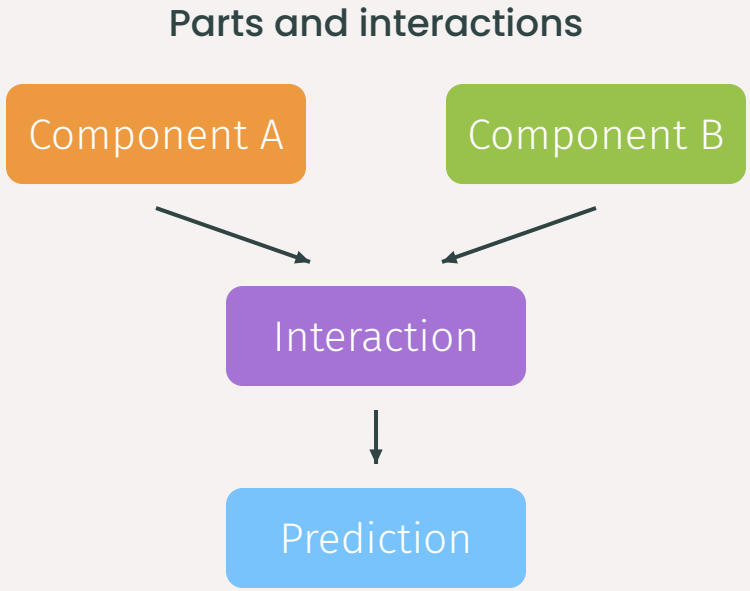


Attribution is a causal proxy.

- Gradients can misestimate ablation effects.

(Braun et al., *Attribution-based Parameter Decomposition*, 2025; Bushnaq et al., *SPD*, 2025)

We can find and test the parts.
When do they explain the *computation*?



Account for parts and interactions.
- How do they combine into the computation?

Specify fidelity and what is missing.
- Which effects are preserved or unexplained?

Compare explanations for a purpose.
- Which helps answer the actual question?

(Gauderis et al., *From Mechanistic to Compositional Interpretability*, 2026; Sharkey et al., *Open Problems*, 2025)

Coming up...

Early Interpretability Methods

José Oramas

From Classical to Mechanistic Interpretability

Ward Gauderis, José Oramas & Thomas Dooms

From Mechanistic to Compositional Interpretability

Ward Gauderis & Thomas Dooms