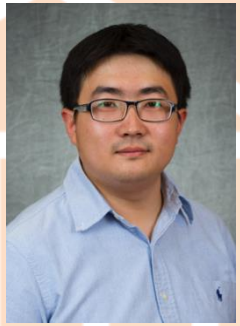
Discovery, Validation and Editing of Large Language Model Mechanisms







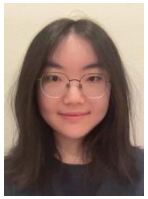
Yinhan He, Wendy Zheng, Tianyi Zhao, Chen Chen, Jundong Li

University of Virginia

Speaker Introduction

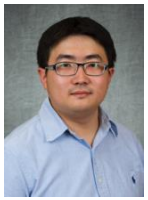
Yinhan He — PhD candidate, ECE @ UVA; LLM interpretability and efficiency

Wendy Zheng — PhD student, CS @ UVA; LLM mechanistic interpretability



Tianyi Zhao — PhD student, CS @ UVA; trustworthy AI, hallucination, miscalibration, model editing

Chen Chen — Assistant Professor, CS @ UVA; complex networks & trustworthy ML



Jundong Li — Associate Professor, ECE & CS @ UVA (corresponding); graph ML, trustworthy ML, LLMs

Tutorial Agenda

Part 1: Motivation & Significance (10 min)

Part 2: Notions & Background (15 min)

Part 3: Mechanism Discovery (40 min)

Part 4: Mechanism Validation (40 min)

Part 5: Mechanism Editing (40 min)

Part 6: Summary and Future Directions (15 min)

Part 1: Motivation & Significance

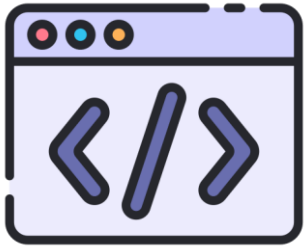
Explaining the need for mechanistic interpretability (MI) in the big picture

Presenter: Jundong Li

Part 1: LLMs Empowers Every Industry

Healthcare

Clinical Q&A,
drug discovery



Coding

Code understanding,
generation

Education

Personalized tutoring,
content creation

Science

Literature search,
experiment design

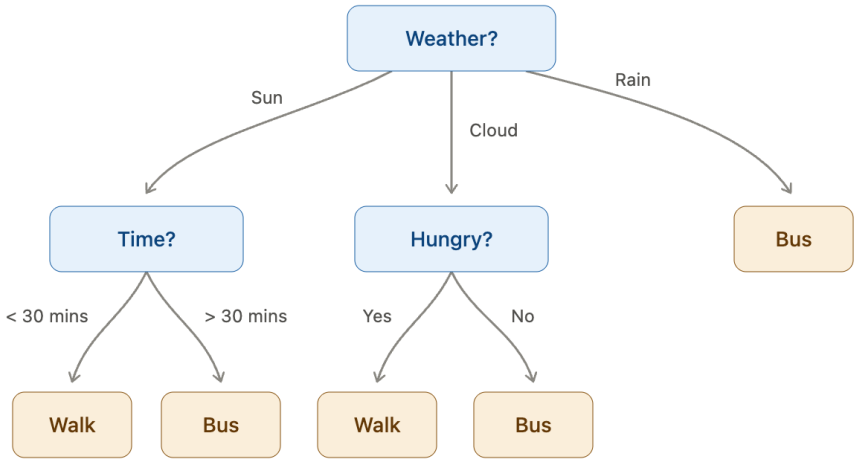
Law

Contract analysis,
legal Q&A

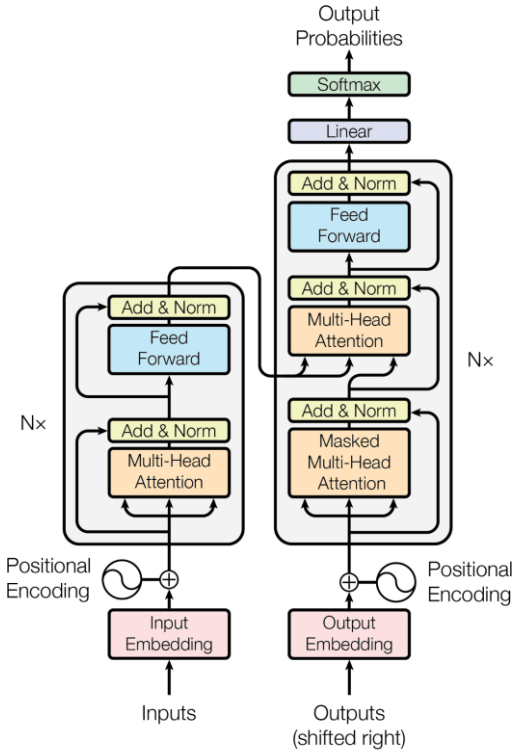
Part 1: Interpretability Challenges in LLMs

Today is sunny. How should I go to the grocery store that is ~25 minutes walk away?

Decision Tree



Large Language Models

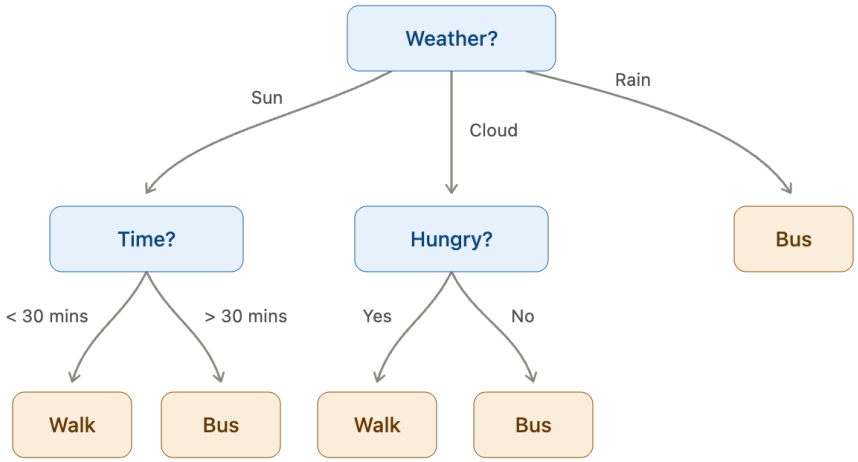


Wang, K., et al. "Interpretability in the Wild: A Circuit for Indirect Object Identification in GPT-2 Small." ICLR, 2023

Part 1: Interpretability Challenges in LLMs

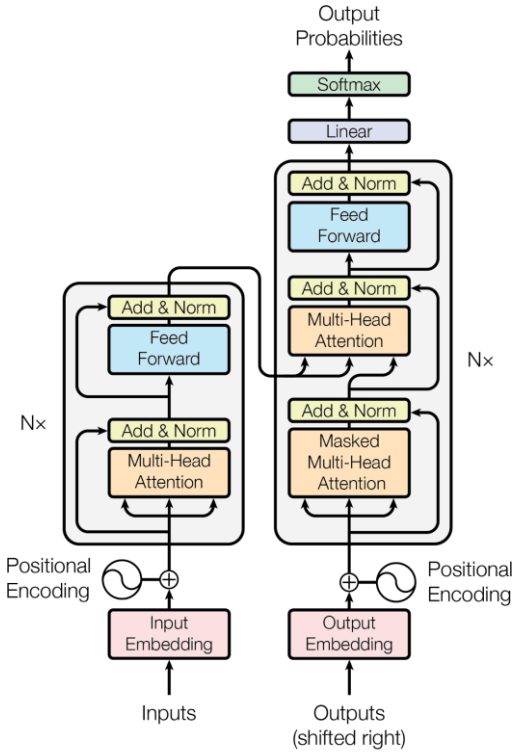
Today is sunny. How should I go to the grocery store that is ~25 minutes walk away?

Decision Tree



We know clearly how a decision tree reach the question's answer, but we do not know how LLMs do.

Large Language Models



Wang, K., et al. "Interpretability in the Wild: A Circuit for Indirect Object Identification in GPT-2 Small." ICLR, 2023

Part 1: What is Mechanistic Interpretability?

MI *reverse-engineers* a model's computation into a **human-readable algorithm** and **the parts of the network that execute it**.

Mechanism = the high-level algorithm the model implements to perform a task or behavior.

Circuit = the subset of components (attention heads, MLP layers, residual stream) that implements that mechanism.

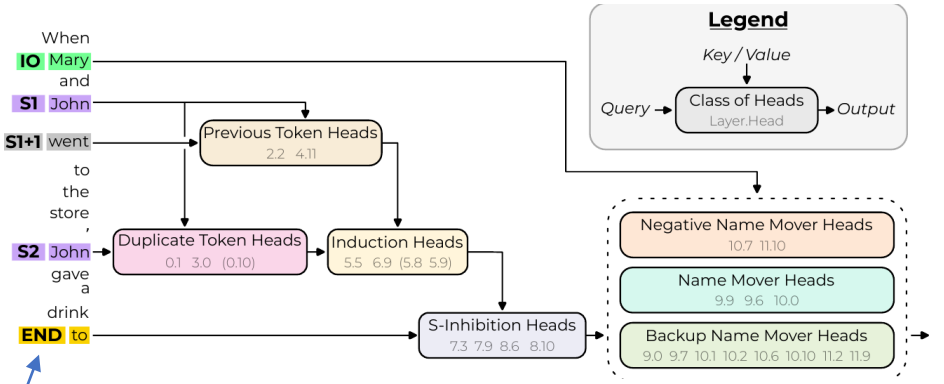
MECHANISM — the algorithm

IOI task: "answer the name that is NOT repeated"

1. **Detect the repeated name (S = "John")**
   ↳ *implemented by Duplicate-Token & Induction heads*
2. **Suppress attention to that name**
   ↳ *implemented by S-Inhibition heads*
3. **Copy the remaining name → "Mary"**
   ↳ *implemented by Name-Mover heads*

implemented by

CIRCUIT — the components that run it and their functional interpretation



'When Mary and John went to the store, John gave a drink to?

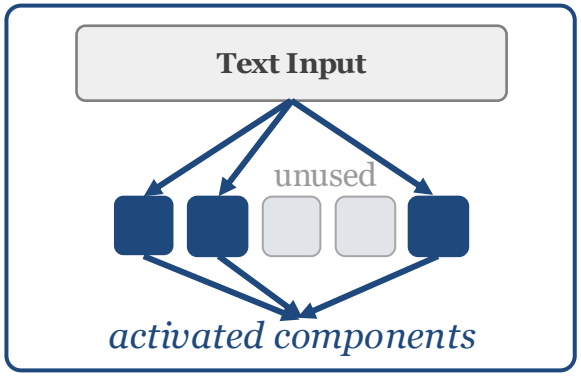
Wang, K., et al. "Interpretability in the Wild: A Circuit for Indirect Object Identification in GPT-2 Small." ICLR, 2023

Part 1: Why is MI Necessary?

1. Behavior can hide the truth

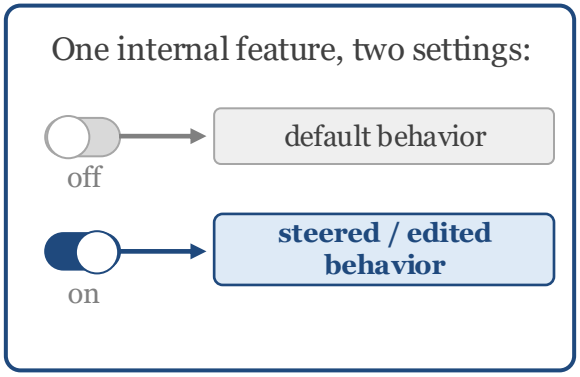
Input–output tests can pass while deceptive goals, backdoors, or shortcut reasoning persist inside. Only inspecting the computation surfaces them.

2. Models can have unused components



LLMs trained via gradient descent often have redundant or inactive components. MI can identify these components, which can then be pruned to improve efficiency.

3. Understanding enables control



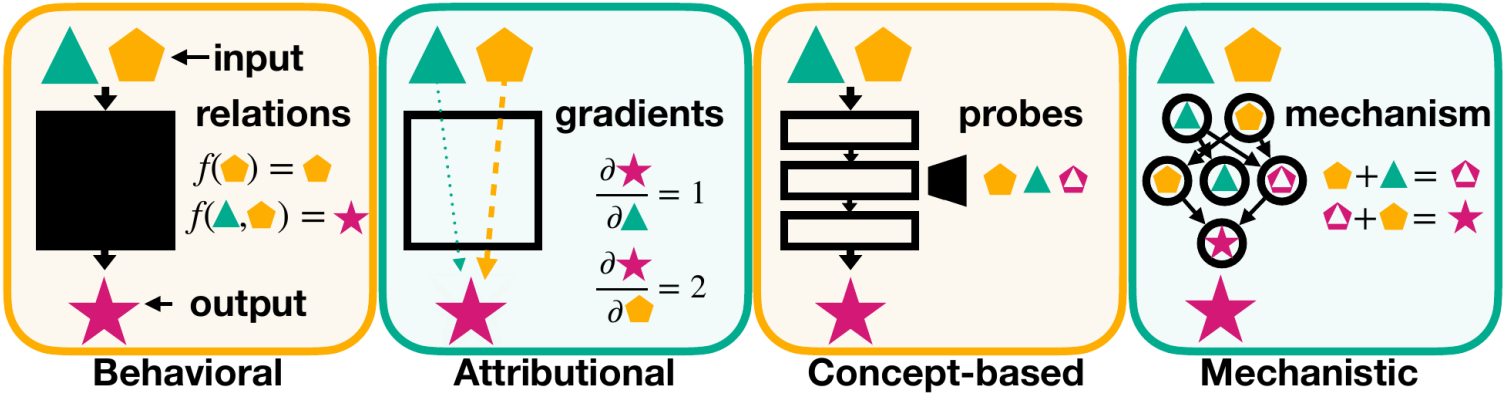
A validated mechanism lets us edit facts, steer behavior, or unlearn knowledge. Further, we can oversee models which are too capable to judge by behavior alone.

Hubinger, E., et al. "Sleeper Agents: Training Deceptive LLMs That Persist Through Safety Training." arXiv:2401.05566
Yu, L., et al. "Mechanistic Understanding and Mitigation of Language Model Non-Factual Hallucinations." Findings of EMNLP 2024
Meng, K., et al. "Locating and Editing Factual Associations in GPT." NeurIPS 2022
Templeton, A., et al. "Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet." Transformer Circuits Thread 2024

Part 1: Four Paradigms of Interpretability

❑ Interpretability paradigms

- The first three paradigms are 'traditional' explainable AI (XAI);
- MI exposes the model's internal causal mechanisms.



Behavioral	Attributional	Concept-based	Mechanistic
Analyze input-output patterns	Quantify individual input feature influences	Identifies representations governing behaviors	Uncover precise causal mechanisms

Part 1: Why Traditional Explainability Fall Short

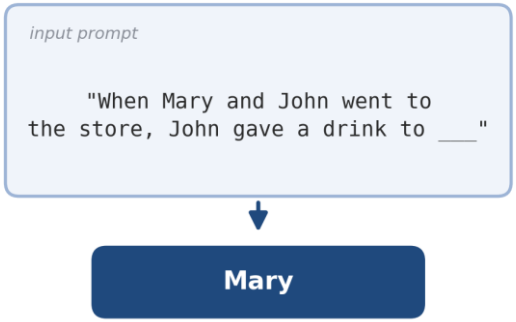
Traditional paradigm (example methods)	What it tells you	Why it is not enough
Behavioral *e.g., input perturbation, occlusion, ablation, behavioral tests*	input–output relations only	Never opens the box, i.e., sees only behavior, not mechanism
Attributional *e.g., saliency, Integrated Gradients, LIME , SHAP , attention maps*	importance of input features	Reveals where in the input the model looks, not how it computes
Concept-based *e.g., probing classifiers, TCAV*	which concepts are encoded	Only shows a concept is decodable from a representation, not that how model uses it

All three paradigms are at the input or representation level, observationally, none recover the causal algorithm. **Mechanistic interpretability adds causal intervention.**

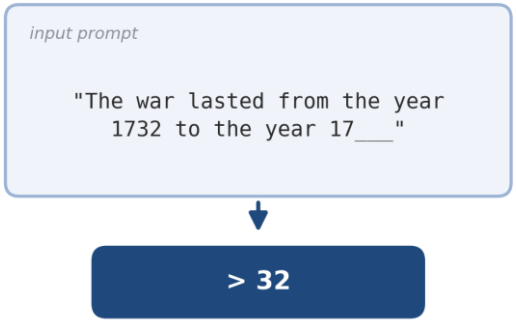
Jain, S., & Wallace, B. C. "Attention Is Not Explanation." NAACL 2019
Belinkov, Y. "Probing Classifiers: Promises, Shortcomings, and Advances." Computational Linguistics 2022
Ribeiro, M., et al. "Why should i trust you?" Explaining the predictions of any classifier." SIGKDD. 2016.
Lundberg, S. & Lee, S. "A unified approach to interpreting model predictions." NeurIPS, 2017.
Kim, B, et al. "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)." ICML, 2018.

Part 1: Language Tasks Studied in MI Research

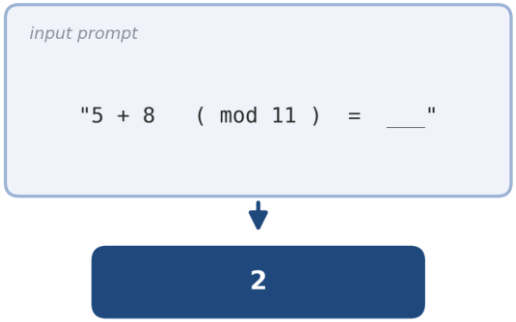
“Indirect Object Identification” (IOI)



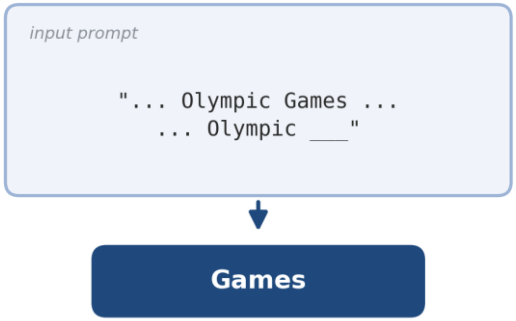
“Greater-than”



“Modular arithmetic"

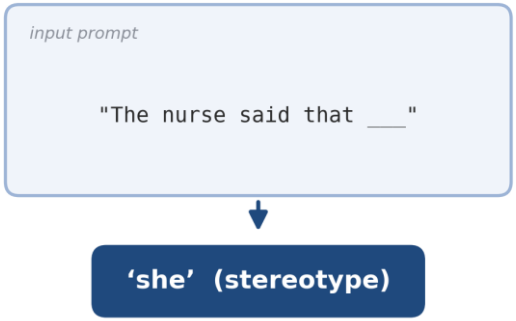


“Induction”



“Factual recall"

“Gender-bias"

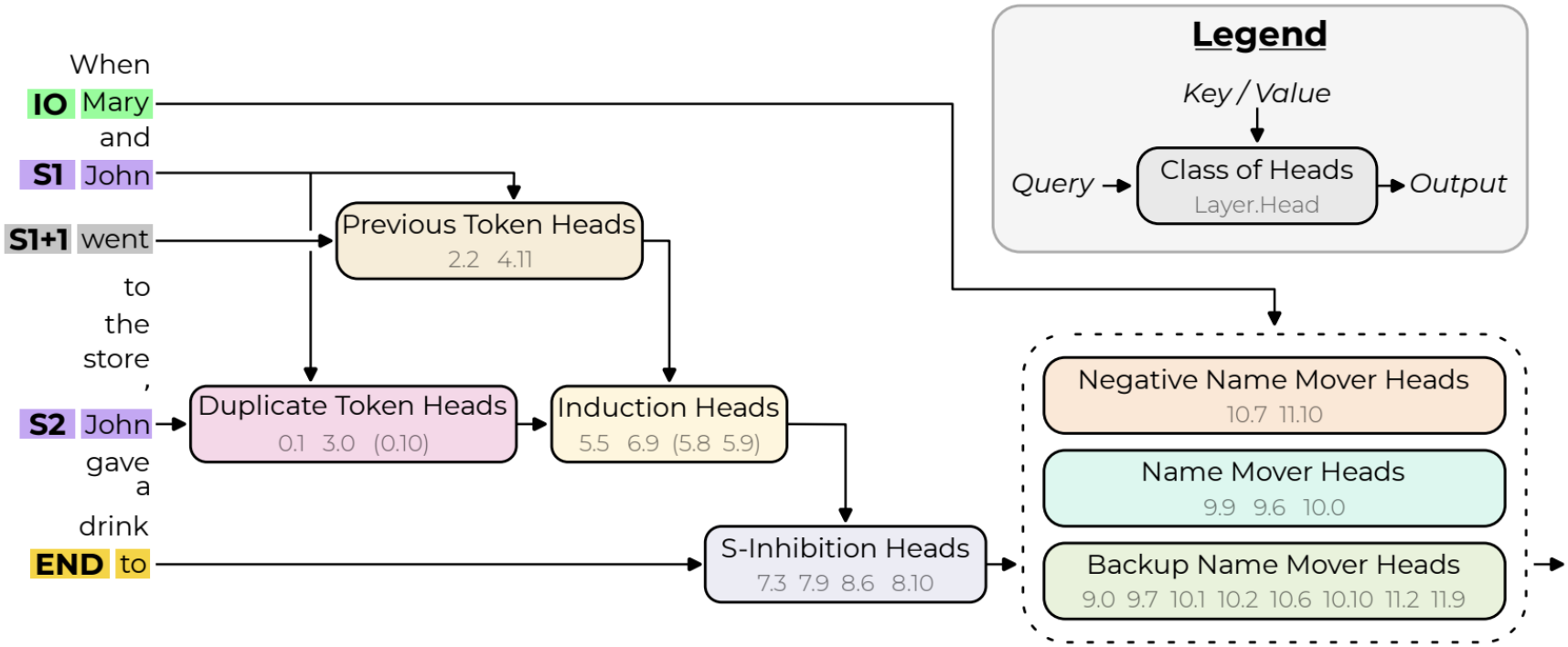


Wang, K., et al. "Interpretability in the Wild: A Circuit for Indirect Object Identification in GPT-2 Small." ICLR 2023
Hanna, M., et al. "How Does GPT-2 Compute Greater-Than? Interpreting Mathematical Abilities in a Pre-trained Language Model." NeurIPS 2023
Nanda, N., et al. "Progress Measures for Grokking via Mechanistic Interpretability." ICLR 2023
Olsson, C., et al. "In-Context Learning and Induction Heads." Transformer Circuits Thread 2022
Meng, K., et al. "Locating and Editing Factual Associations in GPT." NeurIPS 2022
Vig, J., et al. "Investigating Gender Bias in Language Models Using Causal Mediation Analysis." NeurIPS 2020

Part 1: Perspective I — Discovery

❑ Find the circuit responsible for a behavior

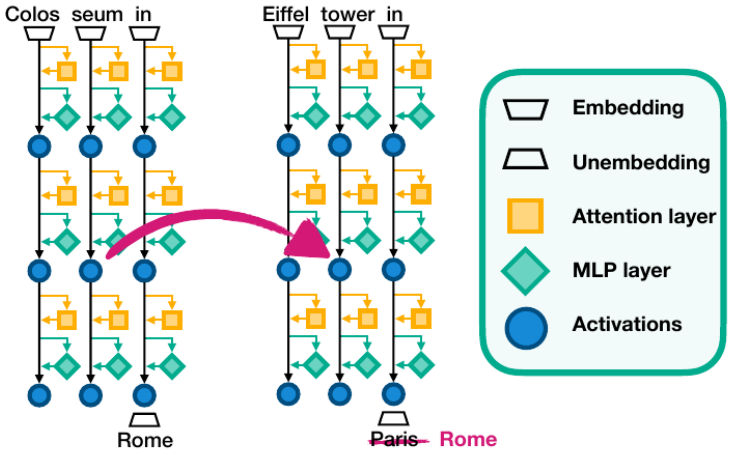
- E.g., for indirect object identification (IOI) task, prompt "John and Mary went to the store. Mary had a good day. John gave a bottle of milk to ___."



Wang, K., et al. "Interpretability in the Wild: A Circuit for Indirect Object Identification in GPT-2 Small." ICLR 2023

Part 1: Perspective II — Validation

- **A discovered mechanism must be validated causally before we trust it**



(a)

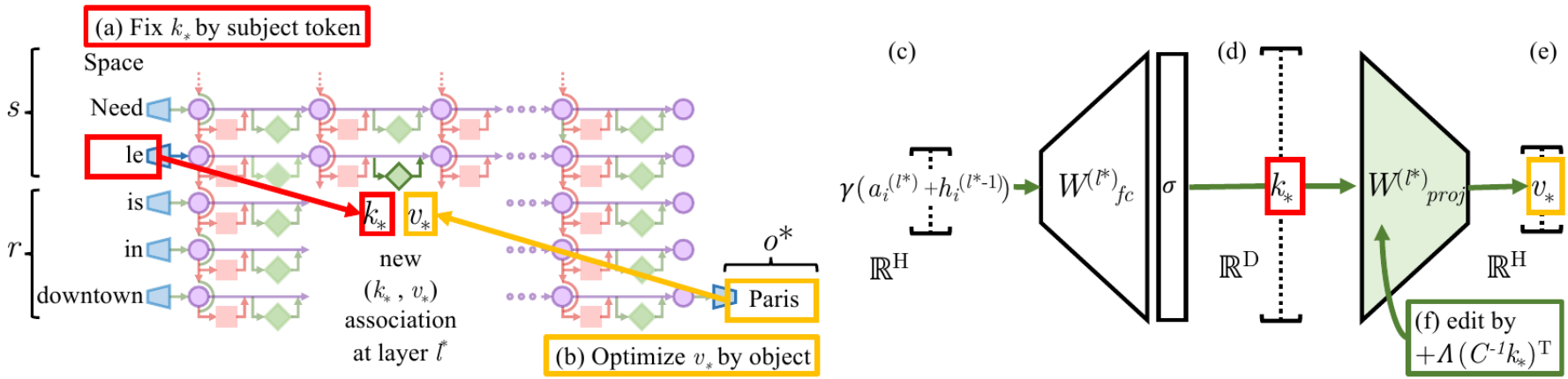
e.g., ***Validation by intervention****: patch a component from a clean run into a corrupted run. If the prediction flips to the correct answer, that component is causally responsible.*

Faithfulness	**Minimality**	**Completeness**
the circuit alone reproduces the behavior	every component is necessary	nothing important is left outside

Bereska, L., & Gavves, E. "Mechanistic Interpretability for AI Safety—A Review." arXiv:2404.14082.

Part 1: Perspective III — Editing

- **A validated mechanism becomes an actionable target, where we can edit LLMs' behavior**



e.g., Locate, then edit: a single rank-one weight update rewrites a stored fact (ROME).

Knowledge editing	**Reasoning editing**	**Unlearning**	**Steering & mitigation**
ROME / MEMIT, knowledge circuits	Circuit reshaping	remove private / unsafe knowledge	representation-level control

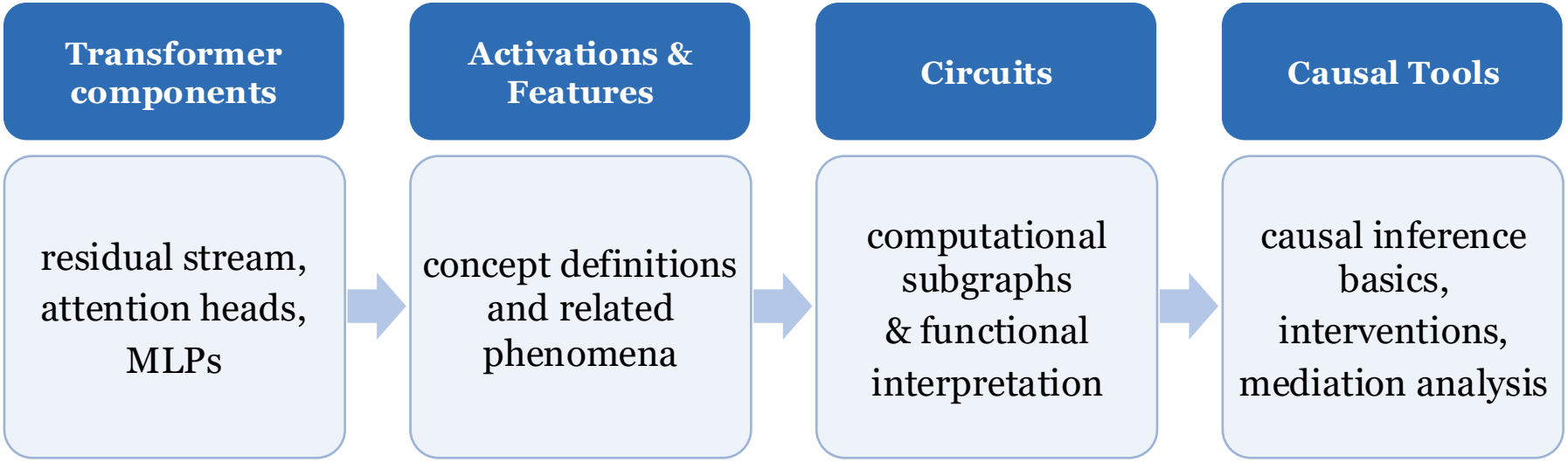
Meng, K., et al. "Locating and Editing Factual Associations in GPT." NeurIPS 2022

Part 2: Notations & Background

Introducing the background knowledge of this tutorial

Presenter: Jundong Li

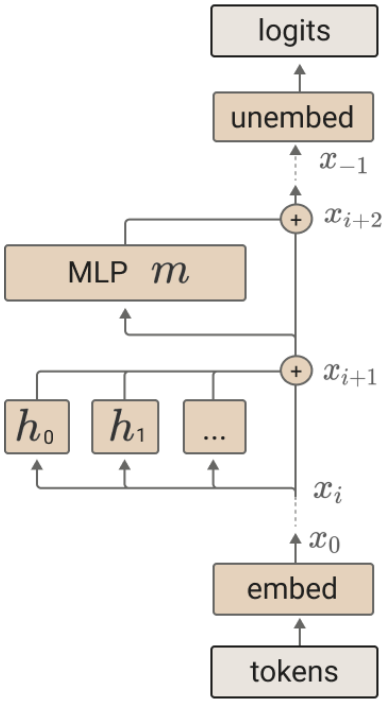
Part 2: From Inputs to Model Internals



Part 2: Transformer as a Computational Graph

❑ Transformers computational graph

- **Nodes**: attention heads, MLP layers;
- **Edges**: computation flow.



The final logits are produced by applying the unembedding.

$$T(t) = W_U x_{-1}$$

An MLP layer, m, is run and added to the residual stream.

$$x_{i+2} = x_{i+1} + m(x_{i+1})$$

Each attention head, h, is run and added to the residual stream.

$$x_{i+1} = x_i + \sum_{h \in H_i} h(x_i)$$

One residual block

Token embedding.

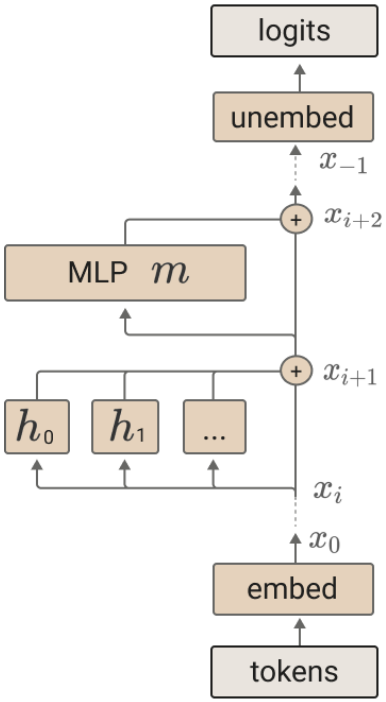
$$x_0 = W_E t$$

Elhage, N., et al. "A Mathematical Framework for Transformer Circuits." Transformer Circuits Thread 2021

Part 2: Transformer as a Computational Graph

❑ Transformers computational graph

- ➢ **Residual stream**: a shared linear channel running the full LLM depth
- ➢ Every component writes its output back to residual stream by addition



The final logits are produced by applying the unembedding.

$$T(t) = W_U x_{-1}$$

An MLP layer, m, is run and added to the residual stream.

$$x_{i+2} = x_{i+1} + m(x_{i+1})$$

Each attention head, h, is run and added to the residual stream.

$$x_{i+1} = x_i + \sum_{h \in H_i} h(x_i)$$

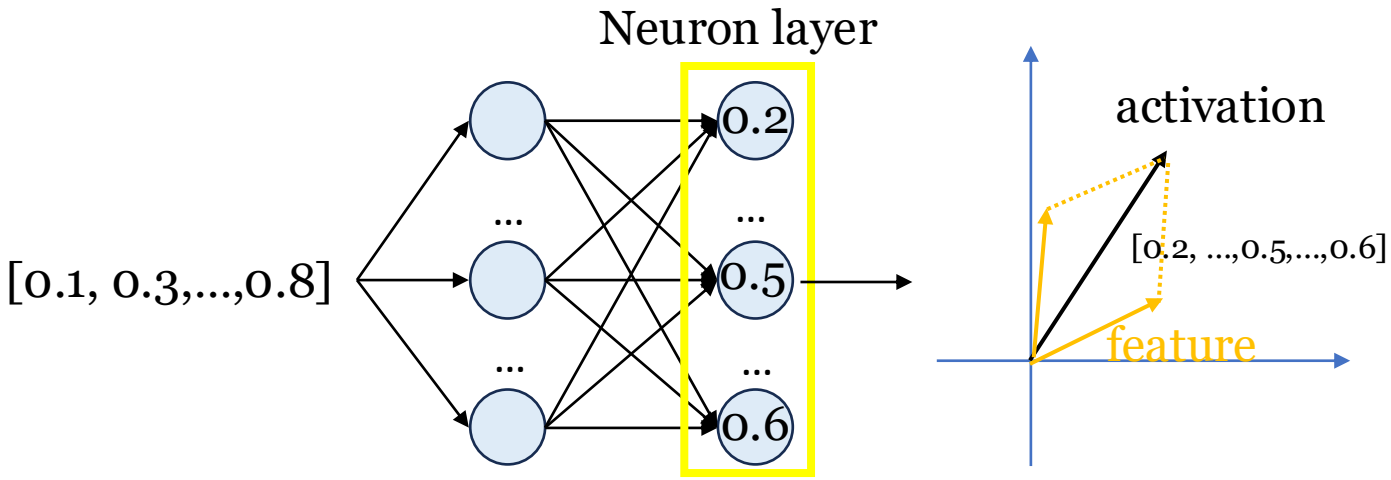
One residual block

Token embedding.

$$x_0 = W_E t$$

Elhage, N., et al. "A Mathematical Framework for Transformer Circuits." Transformer Circuits Thread 2021

Part 2: Activations & Features



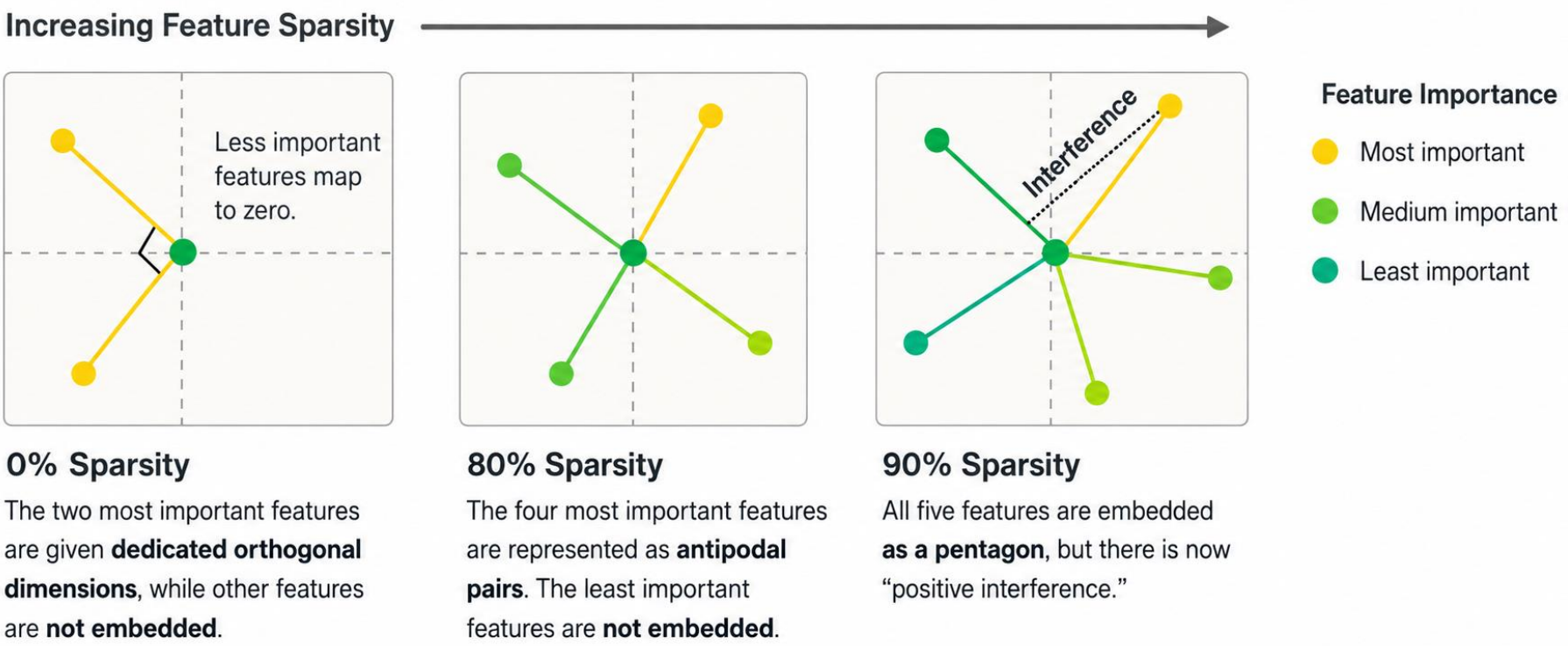
- **Activation:** the output vector of a neuron layer of an LLM
- **Feature:** a direction in activation space representing a semantic concept
  - Based on the Linear Representation Hypothesis (high-level concepts are represented linearly in the activation space of a model)
- A single neuron may activate for many unrelated features

Park, K., et al. "The linear representation hypothesis and the geometry of large language models." ICML 2024.
Olah, C., et al. "Zoom In: An Introduction to Circuits." Distill, 2020.

Part 2: Activations & Features

❑ Superposition

- **Definition:** A model packs more features than it has neurons *by encoding them* as overlapping non-orthogonal directions.

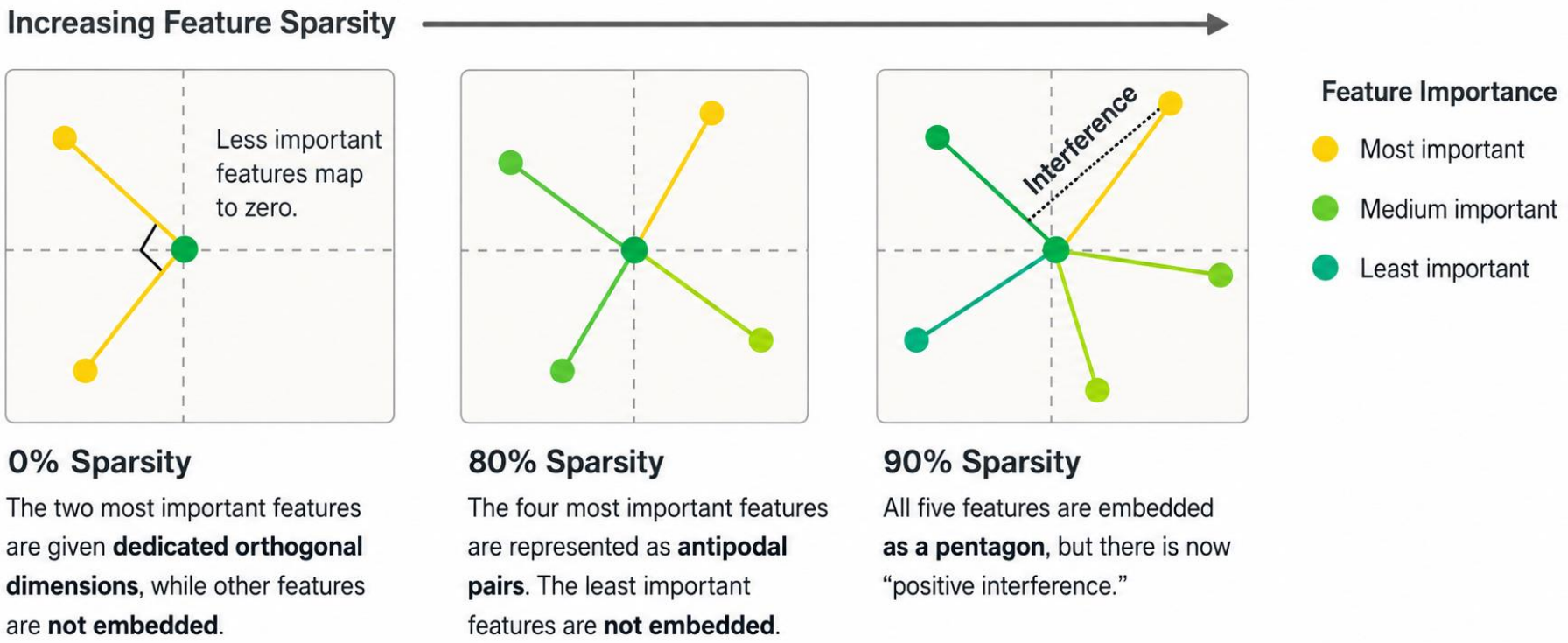


Elhage, N., et al. "Toy Models of Superposition." Transformer Circuits Thread 2022

Part 2: Activations & Features

❑ Superposition

- **Why it minimally impacts LLM performance:** only a few features are active on any given input (they're sparse), so overlapping directions rarely collide.



Elhage, N., et al. "Toy Models of Superposition." Transformer Circuits Thread 2022

Part 2: Activations & Features

❑ Superposition leads to Polysemanticity

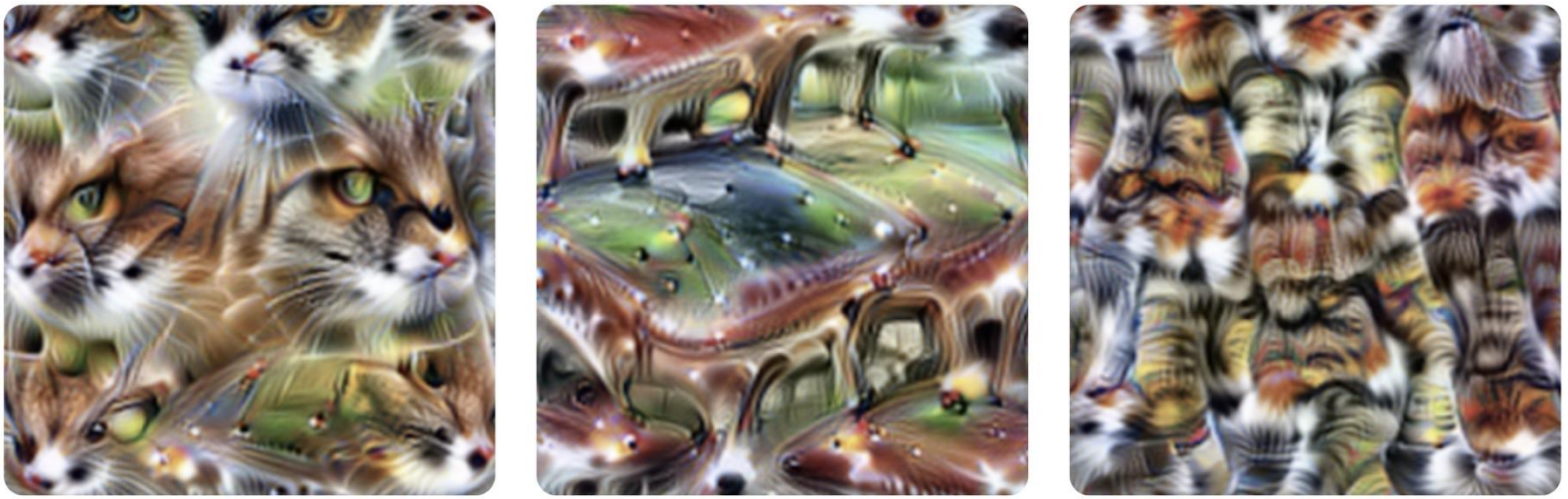
- **Polysemanticity:** a single neuron activates several unrelated features at inference time

Olah, C., et al. "Zoom In: An Introduction to Circuits." Distill 2020

Part 2: Activations & Features

❑ Polysemanticity

- **Example:** In a convolutional neural network (CNN) with vision input, one neuron activates three unrelated concepts: cat faces, fronts of cars, cat legs.

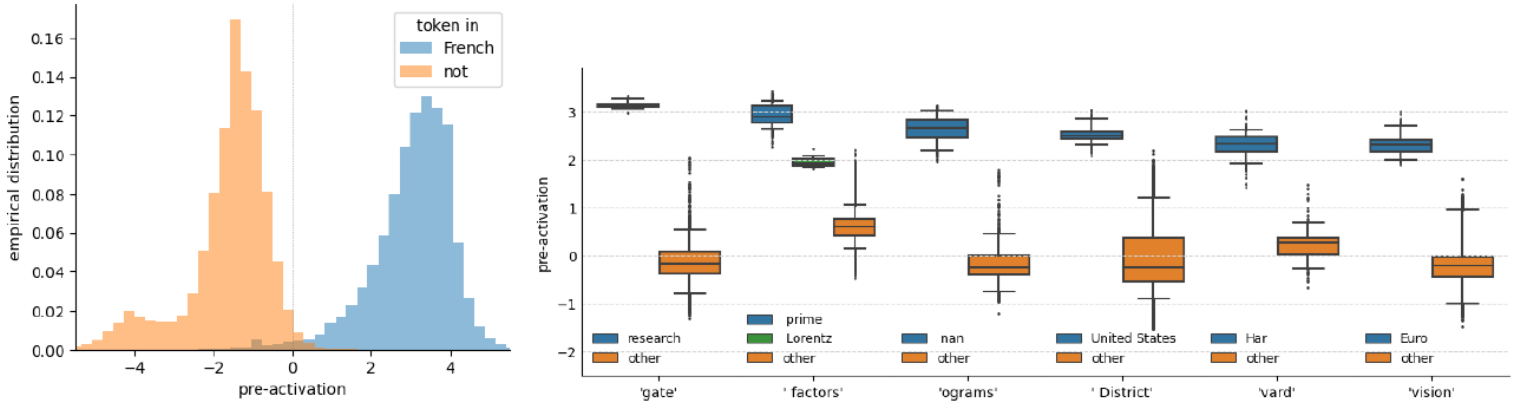


Olah, C., et al. "Zoom In: An Introduction to Circuits." Distill 2020

Part 2: Activations & Features

❑ Polysemanticity

- **Example:** in a language model, a mono-semantic neuron (left: activates only on French) vs. a polysemantic neuron (right: activates on six unrelated n-grams).



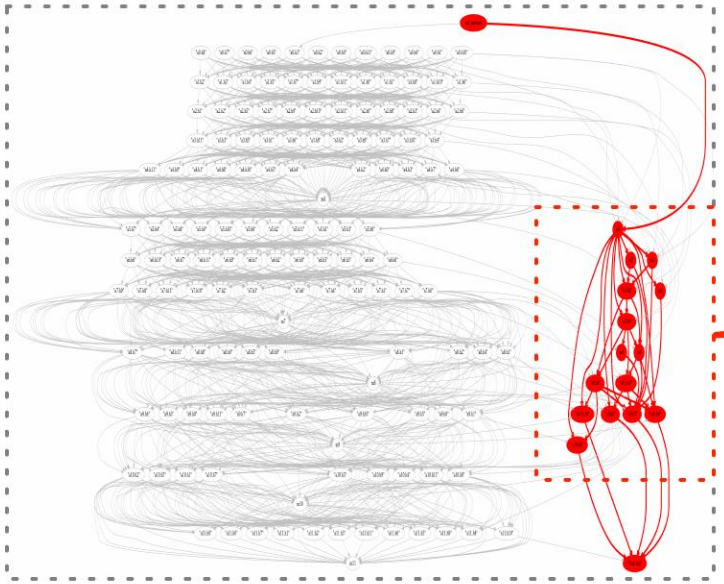
(a) A monosemantic French neuron

(b) A polysemantic neuron activating on six unrelated n-grams

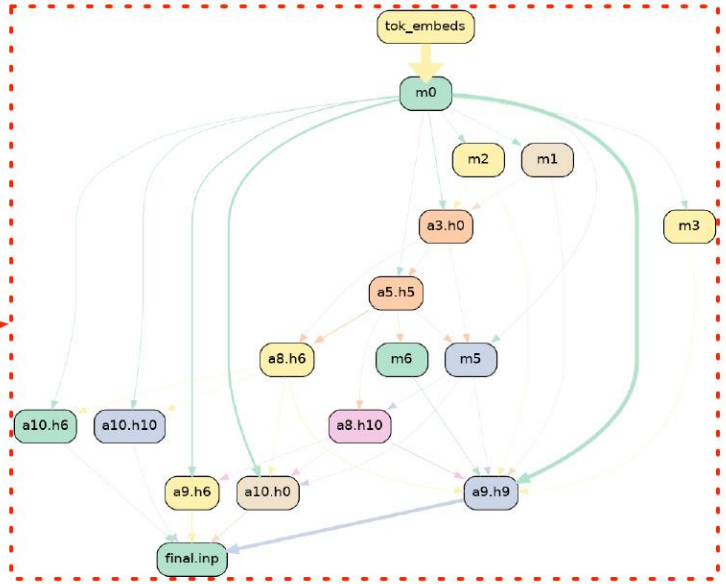
Olah, C., et al. "Zoom In: An Introduction to Circuits." Distill 2020

Part 2: Circuits

- **Definition:** Computational subgraph of the LLM that implements a certain task or a behavior



Full computational graph of the model (left)

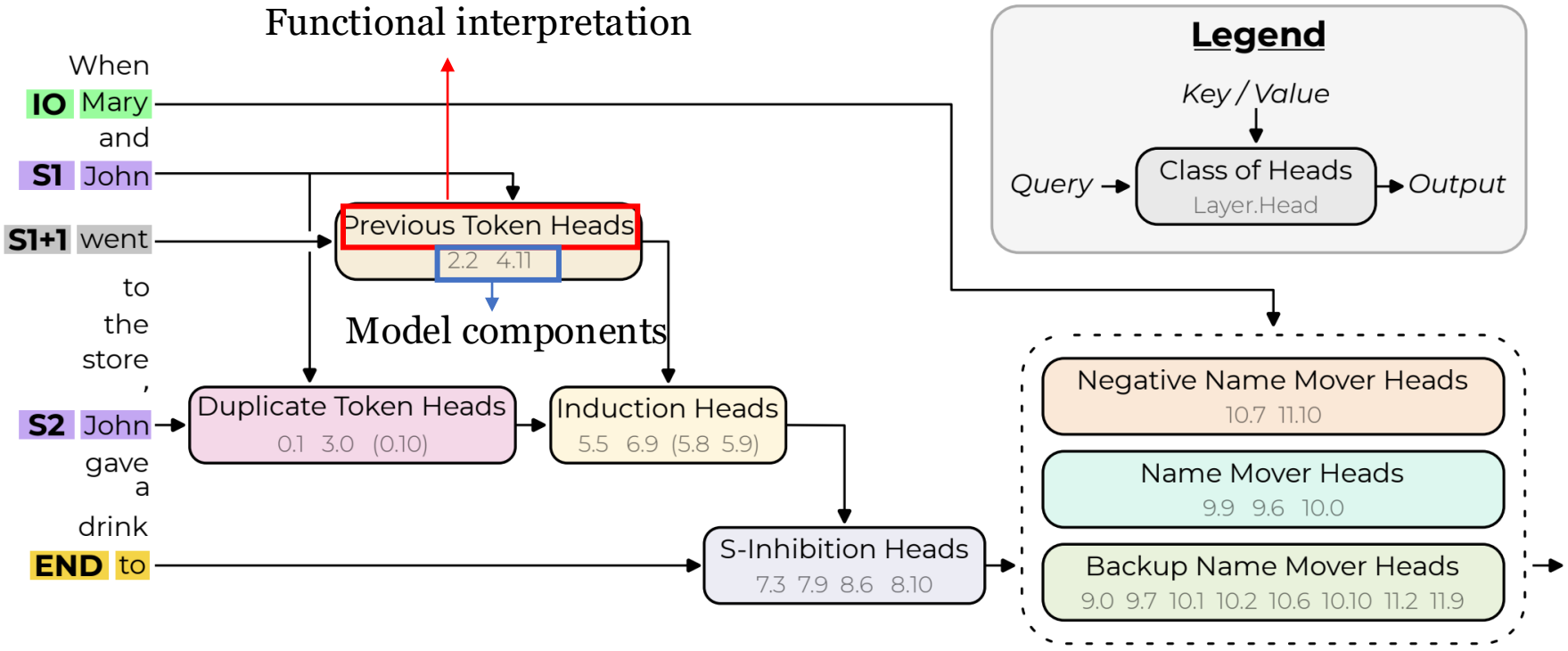


Circuit recovered for a task (right)

Conmy, A, et al. "Towards automated circuit discovery for mechanistic interpretability." NeurIPS, 2023

Part 2: Circuits

- **Definition:** Computational subgraph of the LLM that implements a certain task or a behavior

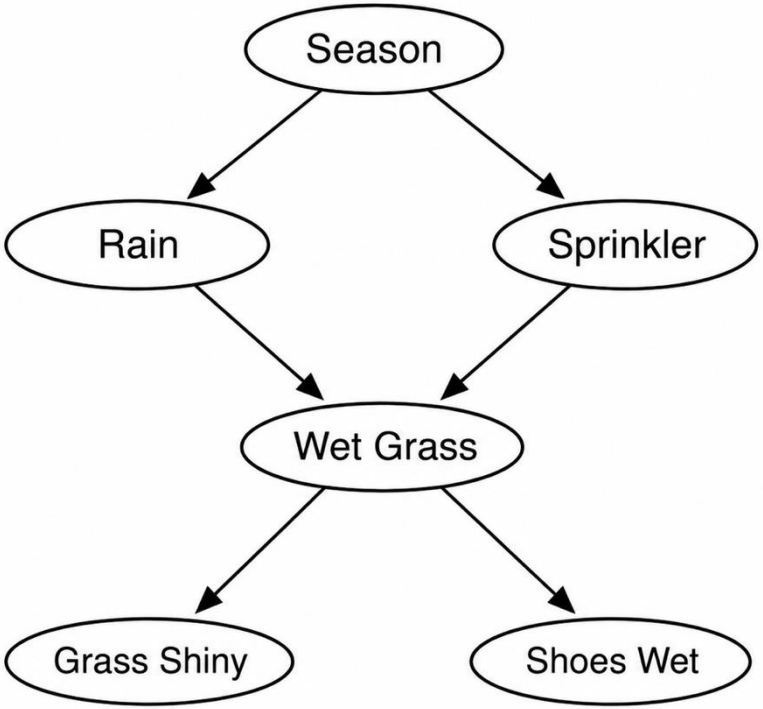


Wang, K, et al. "Interpretability in the wild: a circuit for indirect object identification in gpt-2 small." ICLR 2023

Part 2: Causal Inference Basics

❑ Structural Causal Model (SCM)

- **Definition:** A structural causal model is a directed acyclic graph of variables, where an edge means "directly causes"
- E.g.,

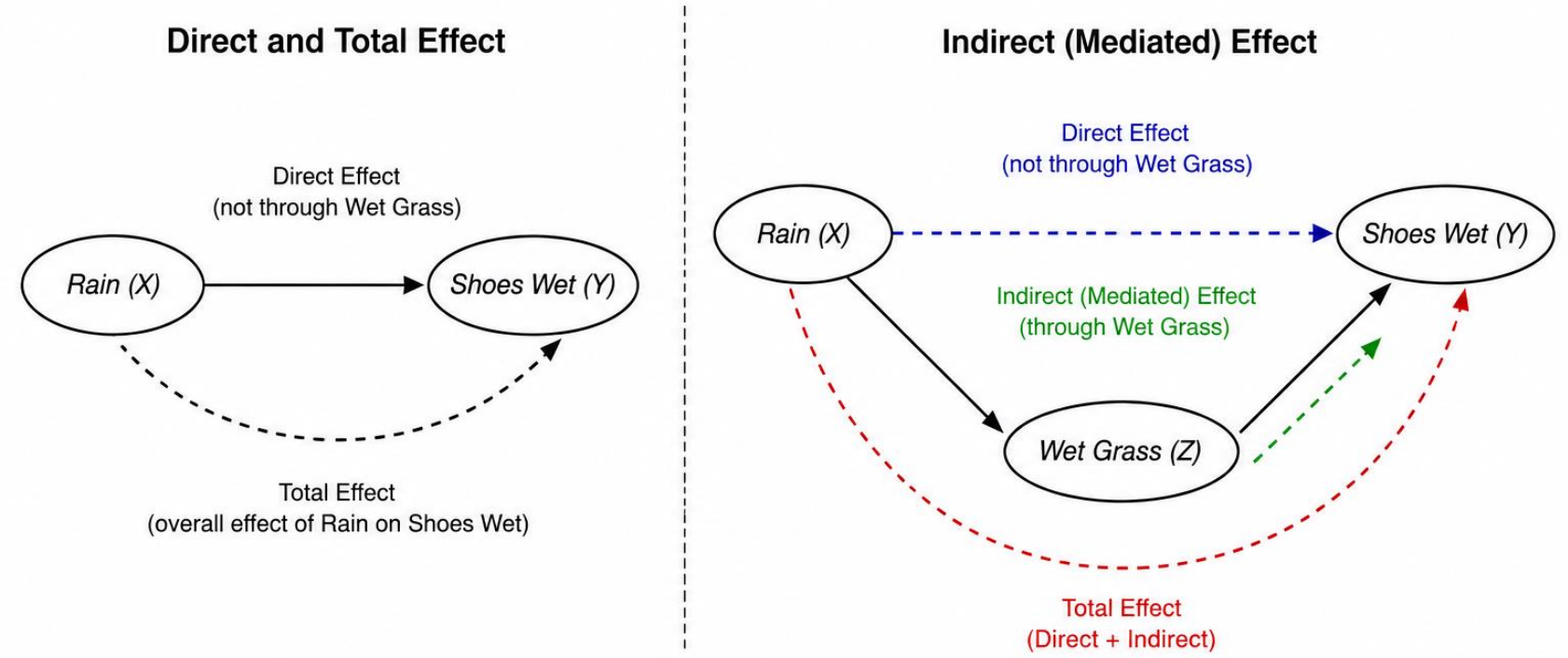


Pearl, J. Causality. Cambridge university press, 2009.

Part 2: Causal Inference Basics

❑ Causal mediation analysis

- **Definition:** decompose a total effect into a direct effect and an indirect effect routed through a mediator.

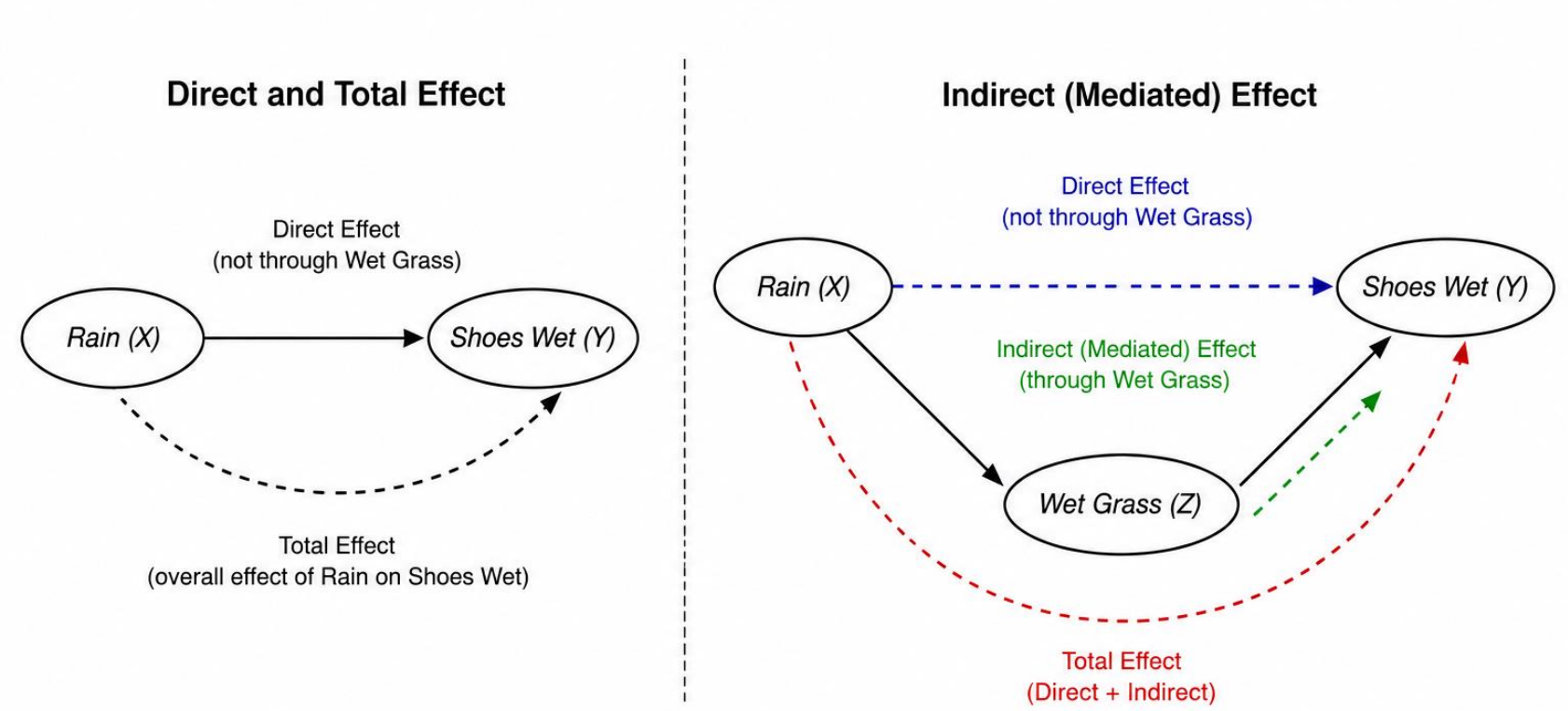


Pearl, J. Causality. Cambridge university press, 2009.

Part 2: Causal Inference Basics

❑ Causal mediation analysis

- **Mediator**: an internal component (e.g., in LLM, neuron, attention head, layer, subspace) on the path

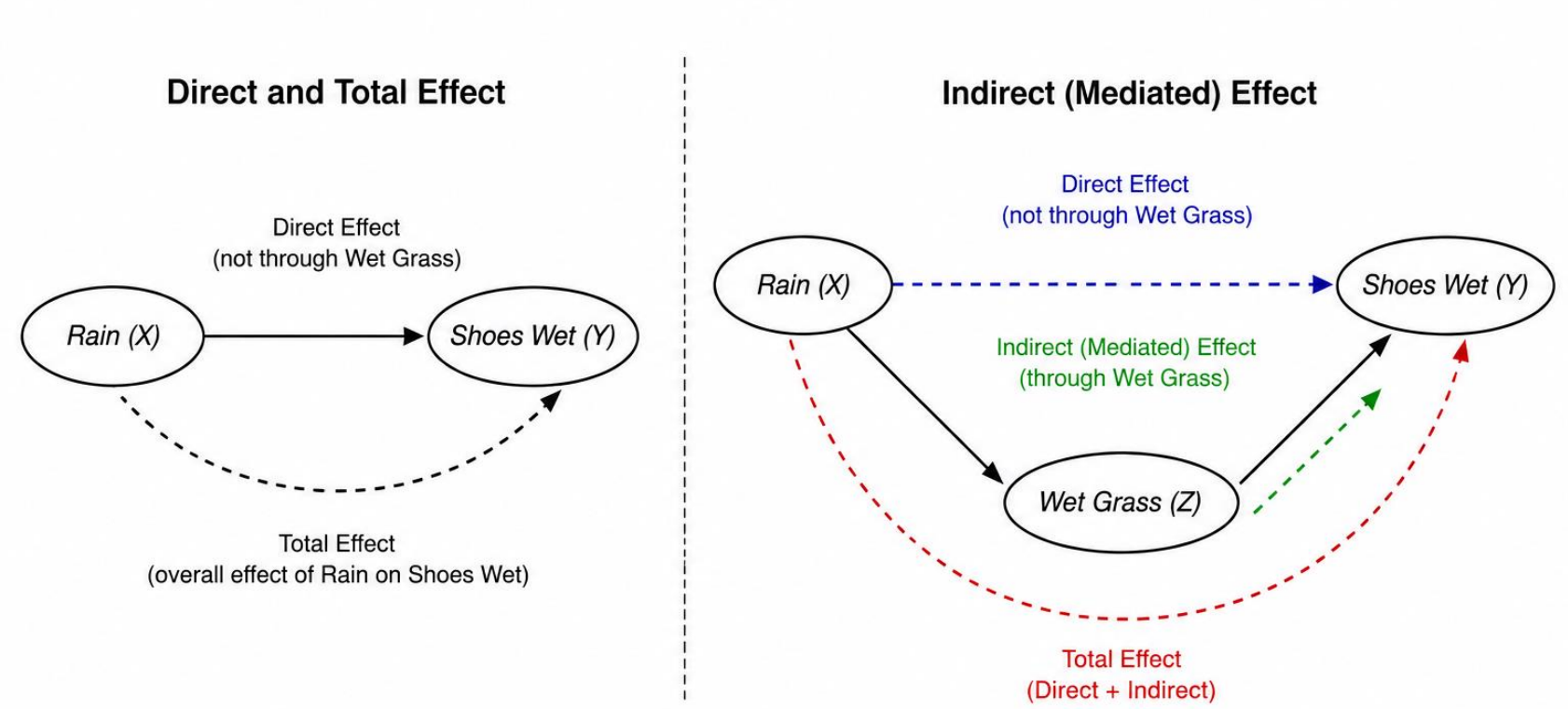


Pearl, J. Causality. Cambridge university press, 2009.

Part 2: Causal Inference Basics

❑ Causal mediation analysis

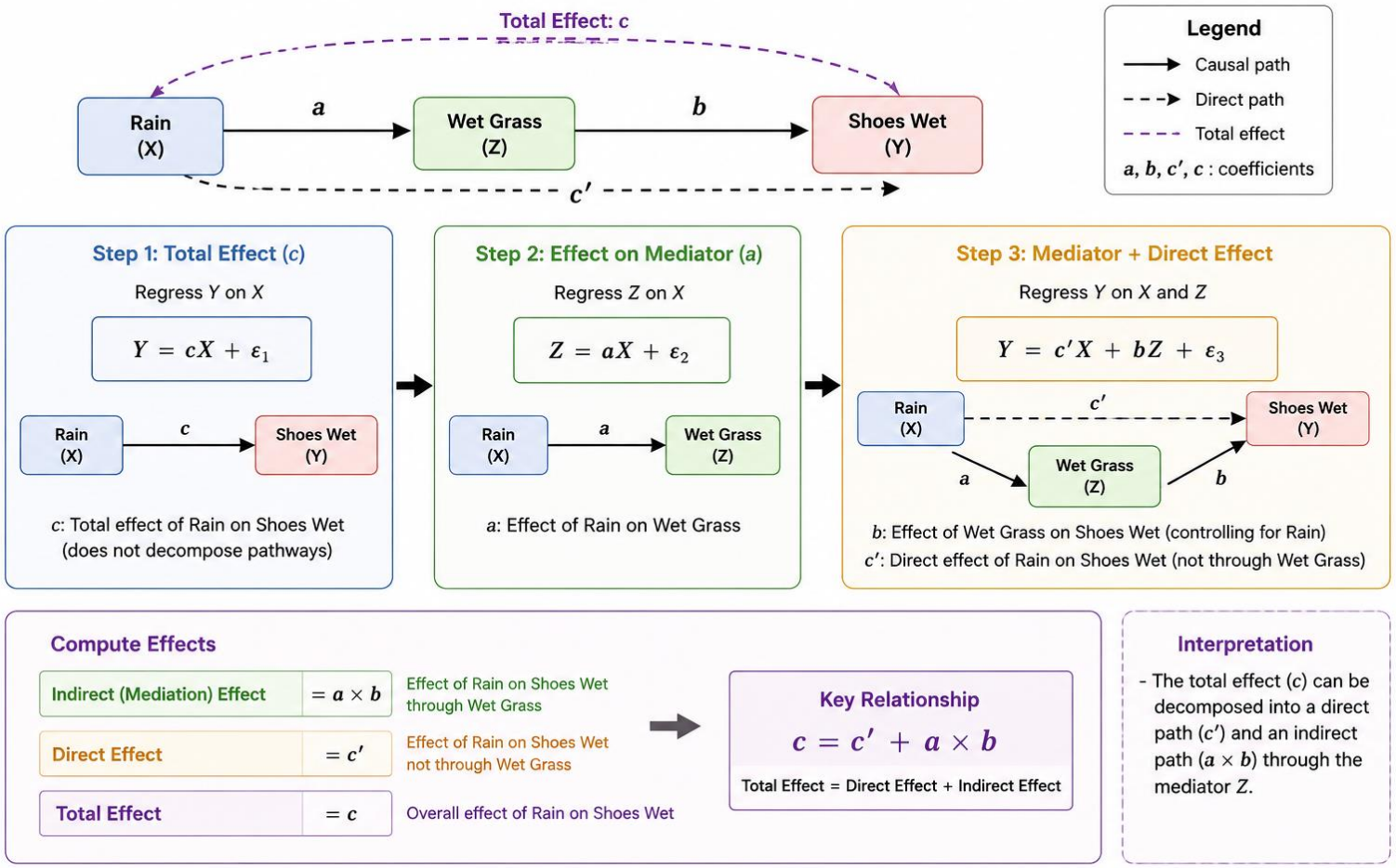
- ➢ **Direct effect**: input → output, holding the mediator fixed
- ➢ **Indirect effect**: input → mediator → output (the routed path)



Vig, J., et al. "Investigating Gender Bias in Language Models Using Causal Mediation Analysis." NeurIPS 2020

❑ Causal mediation analysis

➢ Causal direct/indirect effect of X to Y under linear mediation model

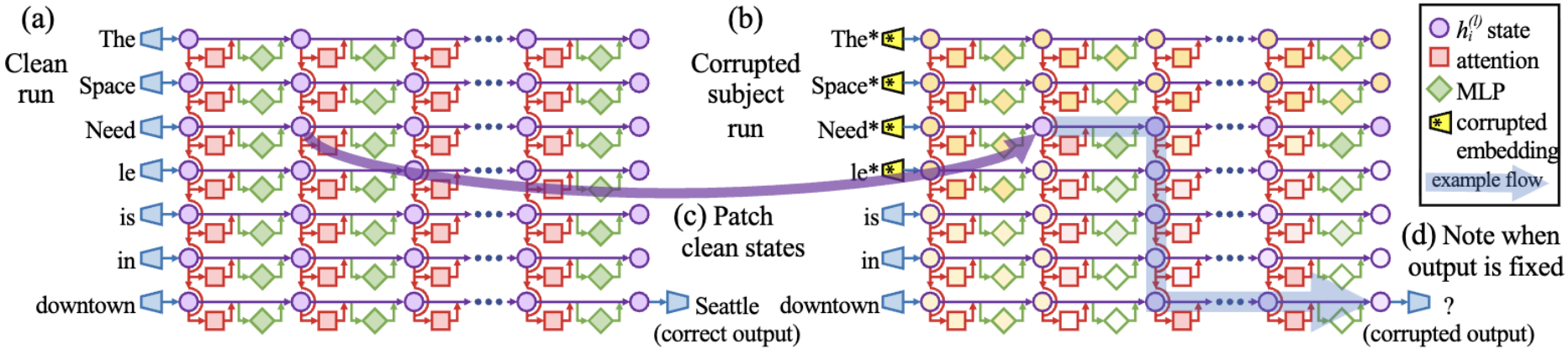


Pearl, J. Causality. Cambridge university press, 2009.

Part 2: Causality in LLMs

❑ Application in MI

- ➢ We treat a computational graph as a causal model and design methods to measure causal indirect effect between input and output caused by intermediate component of interest.



Meng, K., et al. "Locating and Editing Factual Associations in GPT." NeurIPS 2022

Part 3: Mechanism Discovery

Uncovering the mechanisms within black-box LLMs

Presenter: Wendy Zheng

Part 3: Mechanism Discovery

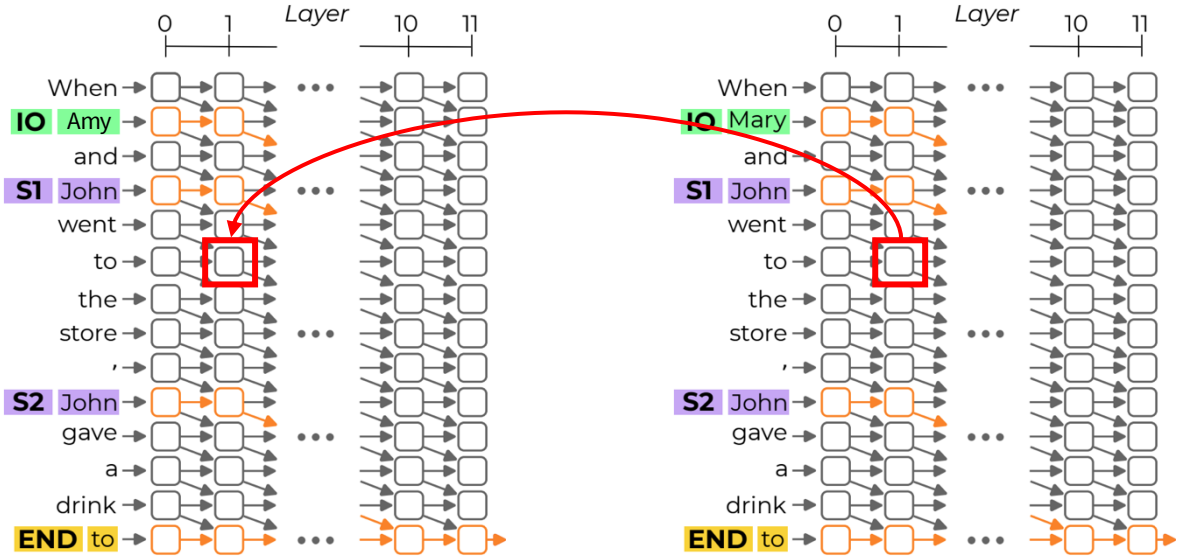
❑ Four families of discovery methods

- **Causal Mediation**
  - Measure each component's importance through interventions
    - Activation Patching, Path Patching, ACDC
- **Attribution**
  - Estimate each component's importance via approximations
    - EAP, EAP-IG, ModCirc
- **Sparse Decomposition**
  - Disentangle activations into interpretable features
    - SAEs, Transcoders, Crosscoders, Sparse Feature Circuits
- **Optimization-based**
  - Use optimization objectives to find mechanisms
    - Edge Pruning, IBCircuit, DAS

Part 3.1: Causal Mediation

❑ Counterfactual interventions

- **Clean input**
  - "When Mary and John went to the store, John gave a drink to [Mary]"
- **Corrupted input**
  - "When Amy and John went to the store, John hands a drink to [Amy]"

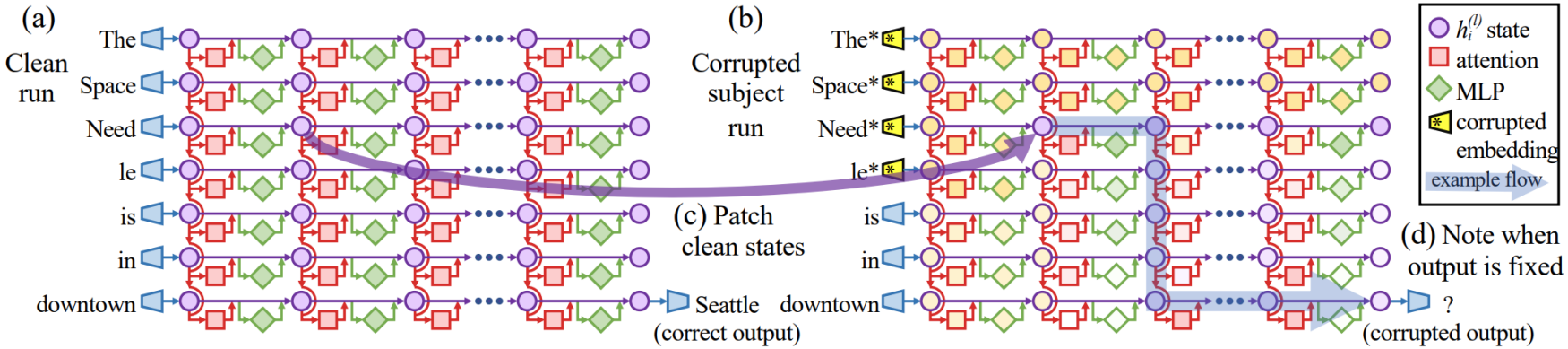


Wang, K., et al. "Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 Small." ICLR 2023

Part 3.1: Causal Mediation

❑ Activation Patching / Causal Tracing

- ➢ **Counterfactual interventions to find important components**
  - Clean input – "The Space Needle is in downtown [Seattle]"
  - Corrupted input – adds Gaussian noise to "The Space Needle"
- ➢ **Iteratively restore component activations for discovery**
  - Measures the marginal contribution of each component
  - Simple and brute force



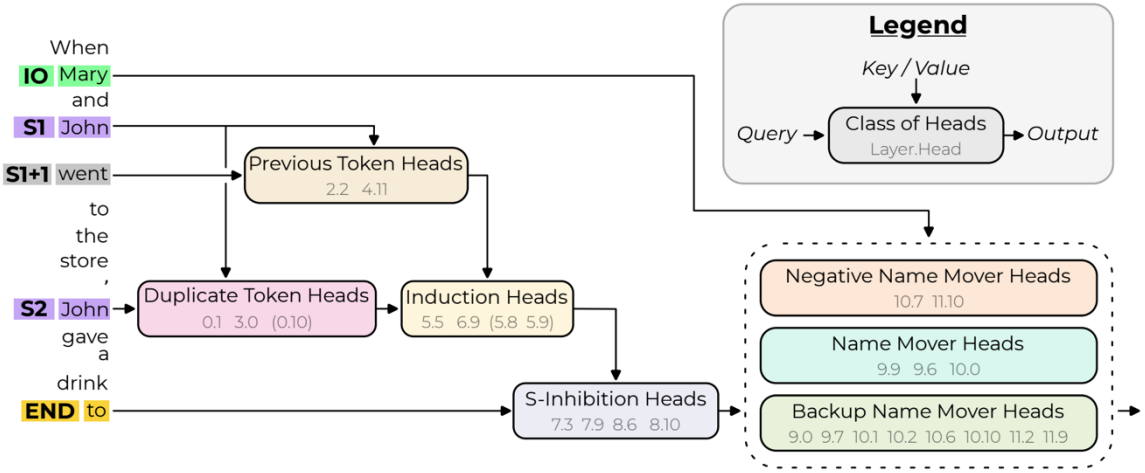
Overview of Activation Patching

Meng, K., et al. "Locating and Editing Factual Associations in GPT." NeurIPS 2022

Part 3.1: Causal Mediation

❑ Path Patching

- ➢ **Applies interventions on specific edges between components**
  - Provides finer-grained insight into how information propagates
  - Reveals which pathways are causally responsible for a given prediction
- ➢ **E.g.: Indirect Object Identification – 26 attention heads**
  - Identify and output the indirect object in a sentence
    - "When John and Mary went to the store, John handed an apple to [Mary]"

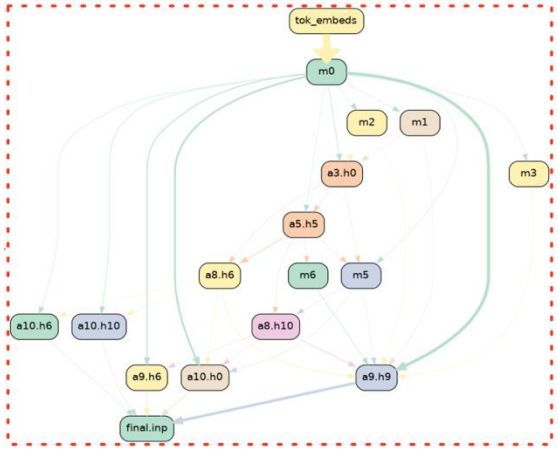


Wang, K., et al. "Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 Small." ICLR 2023

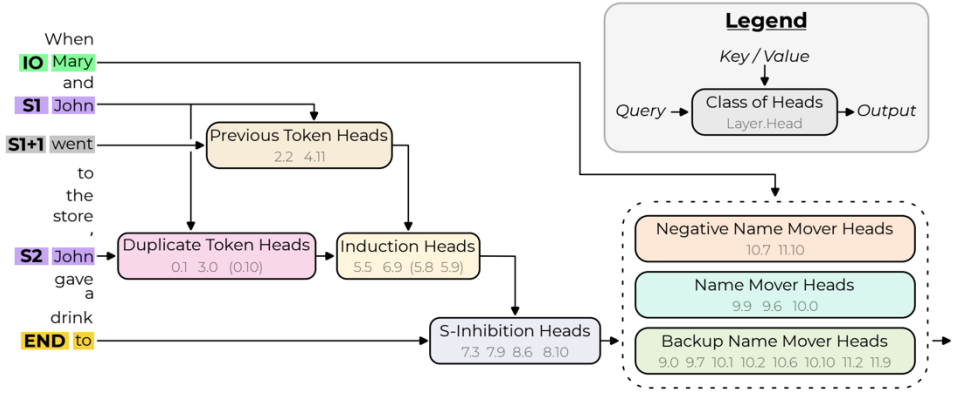
Part 3.1: Causal Mediation

❑ Automatic Circuit DisCovery (ACDC)

- ➢ **Automates the path patching process for circuit discovery**
  - Recursively apply patching starting from the output
  - At each node, remove the edges that minimally impact model behavior
  - Final output – sparse subgraph that performs well on the specific task
- ➢ **Limitations**
  - Fail to fully recover all functionalities in discovered circuits
  - Sensitive to hyperparameters and selected behavioral metric



ACDC Circuit



Original IOI Circuit

Conmy, A., et al. "Towards automated circuit discovery for mechanistic interpretability." NeurIPS 2023

Part 3.2: Attribution

❑ Edge Attribution Patching (EAP)

➢ **Use first-order Taylor approximation to estimate the effect of corrupting a single edge**

- Attribution score = $\left|(e_{corr} - e_{clean})^{T} \frac{\partial}{\partial e_{clean}} L(x_{clean} | \, do(E = e_{clean})\right|$
  - $(e_{corr} - e_{clean})$ is the change in an edge's value under corruption
    - e_{clean} is the component's activation when processing x_{clean}
    - e_{corr} is the activation when processing x_{corr}
  - $\frac{\partial}{\partial e_{clean}} L(x_{clean} | \, do(E = e_{clean})$ measures model behavior sensitivity
    - L represents the behavioral metric that measures how well the model performs a behavior
    - $do(E = e_{clean})$ refers to the do operator from causality, setting the edge E to the clean activation e_{clean}

Syed, A., et al. "Attribution patching outperforms automated circuit discovery." BlackboxNLP 2024

Part 3.2: Attribution

❑ Edge Attribution Patching (EAP)

➢ Empirical performance and limitations

- Only requires two forward passes and one backward pass for a model
  - Avoid edge-by-edge interventions in activation patching
- Outperforms ACDC for identifying important circuit components
  - Applying ACDC on a circuit discovered by EAP yields better results
- Performance remains sensitive to the choice of behavioral metric

Syed, A., et al. "Attribution patching outperforms automated circuit discovery." BlackboxNLP 2024

Part 3.2: Attribution

❑ Edge Attribution Patching with Integrated Gradients (EAP-IG)

➢ **Motivation: Circuits found by EAP are not always faithful**

- Faithful circuits preserve task performance even when all edges outside of the circuit are corrupted
- EAP is limited when the gradient of the behavioral metric is zero
  - attribution score = activation difference x metric gradient
  - Important components (i.e., large activation difference) can still get a zero attribution score due to zero local gradient

Hanna, M., et al. "Have Faith in Faithfulness: Going Beyond Circuit Overlap When Finding Model Mechanisms." COLM 2024

Part 3.2: Attribution

❑ Edge Attribution Patching with Integrated Gradients (EAP-IG)

➢ **Solution: combine EAP with integrated gradients**

$$(e_{corr} - e_{clean})^{T} \frac{1}{m} \sum_{k=1}^{m} \frac{\partial}{\partial e} L(x_{clean} | \, do(E = e_{corr} + \frac{k}{m}(e_{clean} - e_{corr}))$$

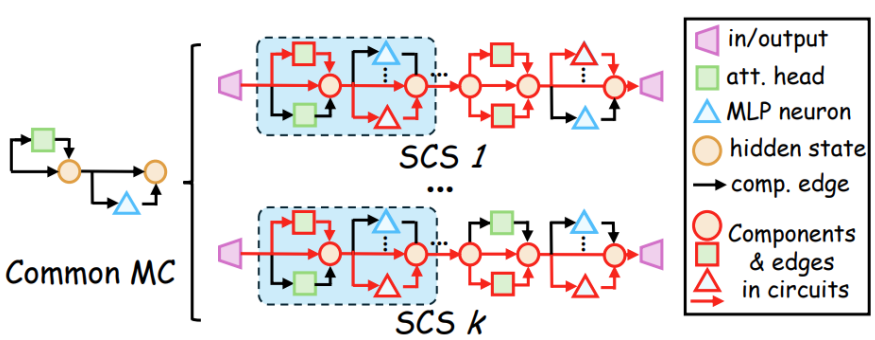
- Accumulate gradients along the straight-line path from e_{corr} to e_{clean}
  - Identify m equidistant steps along the path from corrupted to clean.
  - At each step, set all edge values to $e_{corr} + \frac{k}{m}(e_{clean} - e_{corr})$ and compute the gradient at that step
  - Average the gradients across all steps and multiply by the activation difference to compute the final attribution score for each edge

Hanna, M., et al. "Have Faith in Faithfulness: Going Beyond Circuit Overlap When Finding Model Mechanisms." COLM 2024

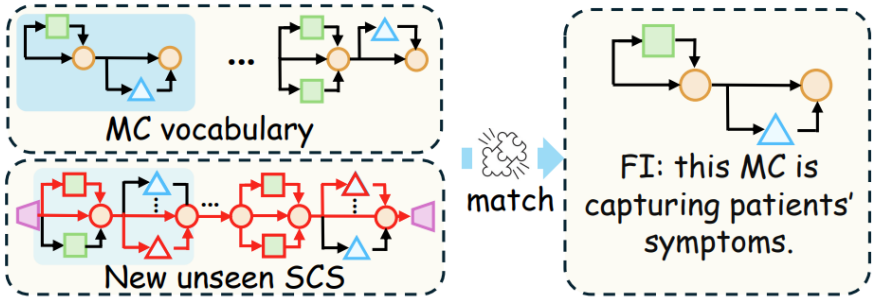
Part 3.2: Attribution

❑ ModCirc

- **Task-agnostic modular circuits (MCs) enable global-level interpretability of LLMs**
  - Prior work focus on task-specific circuits (SCSs) → lack shared patterns
  - ModCirc identifies a global vocabulary across similar tasks
  - Reveals cross-task mechanisms with reduced costs



(a) MCs provide cross-task insight.

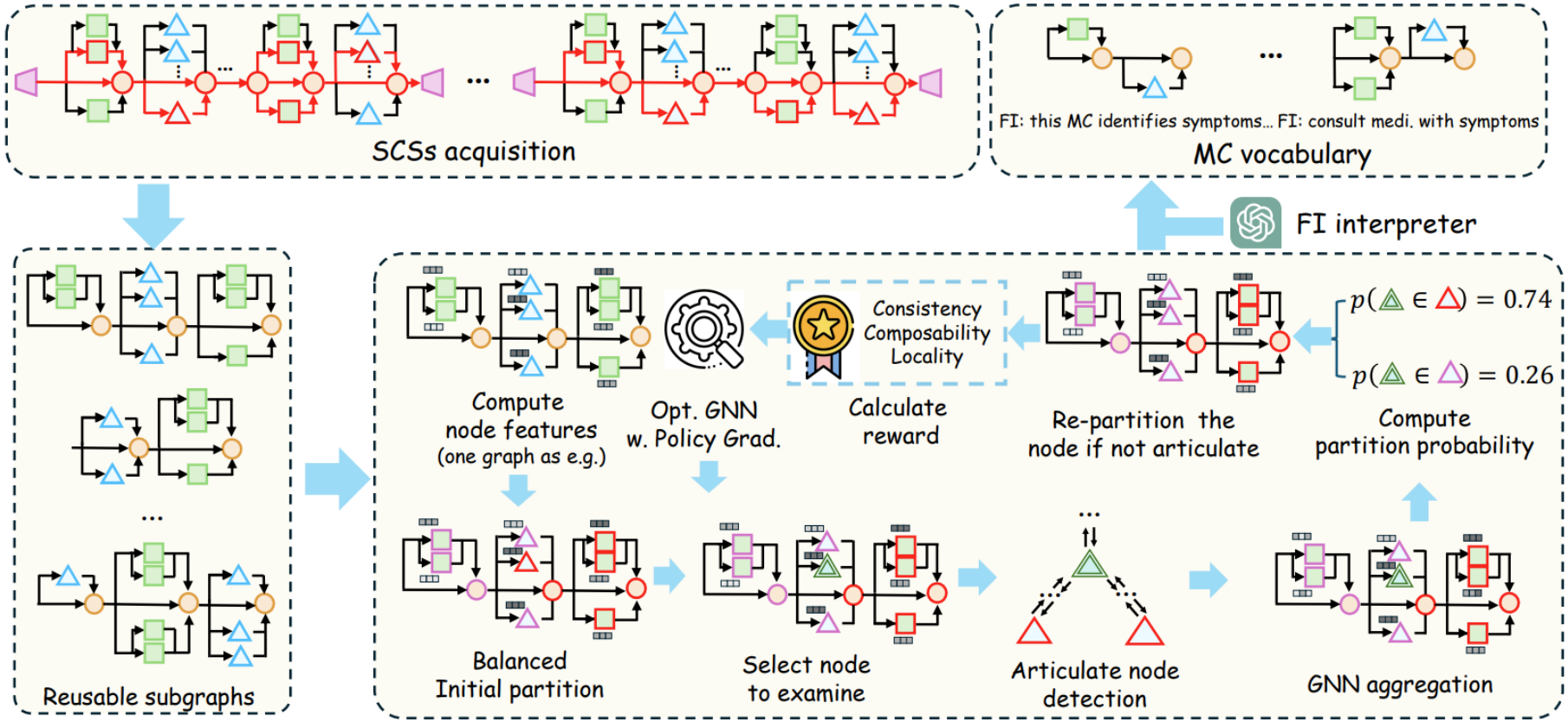


(b) Func. interpretations (FI) are extracted from MC vocab.

He, Y., et al. "Towards global-level mechanistic interpretability: A perspective of modular circuits of large language models." ICML 2025

Part 3.2: Attribution

❑ ModCirc



He, Y., et al. "Towards global-level mechanistic interpretability: A perspective of modular circuits of large language models." ICML 2025

Part 3.3: Sparse Decomposition

❑ Sparse Autoencoders (SAEs)

- **Use dictionary learning to find monosemantic features**
  - Train a sparse dictionary to reconstruct model activations using few active features
    - Encoder maps model activations to latent dictionary features
    - Decoder reconstructs activations linearly using features
- **Enforce interpretability via sparsity regularization**
  - Encourages monosemantic and disentangled features

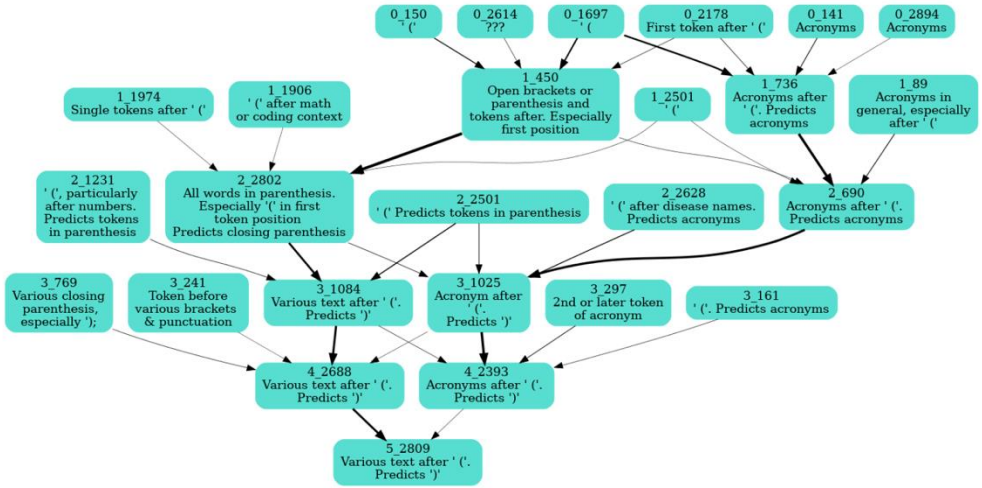
Huben, R., et al. "Sparse autoencoders find highly interpretable features in language models." ICLR 2024

Part 3.3: Sparse Decomposition

❑ Sparse Autoencoders (SAEs)

➢ Circuit discovery using learned features

- Treat features as nodes in the computational graph
- Recursively ablate individual features to measure its causal effect on target feature
- Repeat to uncover the circuit of relevant features



Circuit for predicting the closing parenthesis

Huben, R., et al. "Sparse autoencoders find highly interpretable features in language models." ICLR 2024

Part 3.3: Sparse Decomposition

❑ Sparse Autoencoders (SAEs)

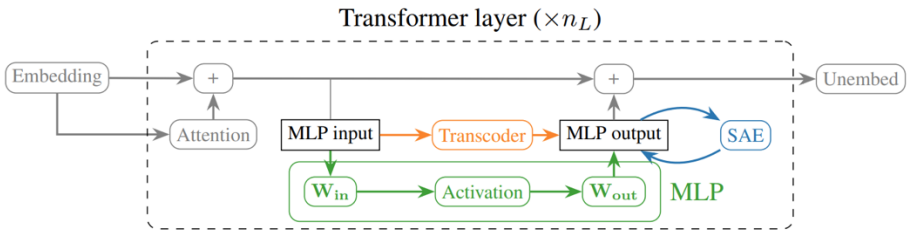
- **SAE features are more interpretable than neurons**
  - SAE features are more often judged to be monosemantic by humans
  - Automated tests show that LLMs better explain and predict features
- **Scalable interpretability for large language models**
  - Gemma Scope is a comprehensive suite of thousands of SAEs on Gemma 2 2B, 9B, and 27B models
  - GPT-4 can be interpretated using a 16M parameter SAE

Huben, R., et al. "Sparse autoencoders find highly interpretable features in language models." ICLR 2024
Bricken, T., et al. "Towards Monosemanticity: Decomposing Language Models With Dictionary Learning." Transformer Circuits Thread, 2023
Lieberum, T., et al. "Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2." BlackboxNLP 2024
Gao, L., et al. "Scaling and evaluating sparse autoencoders." ICLR 2025

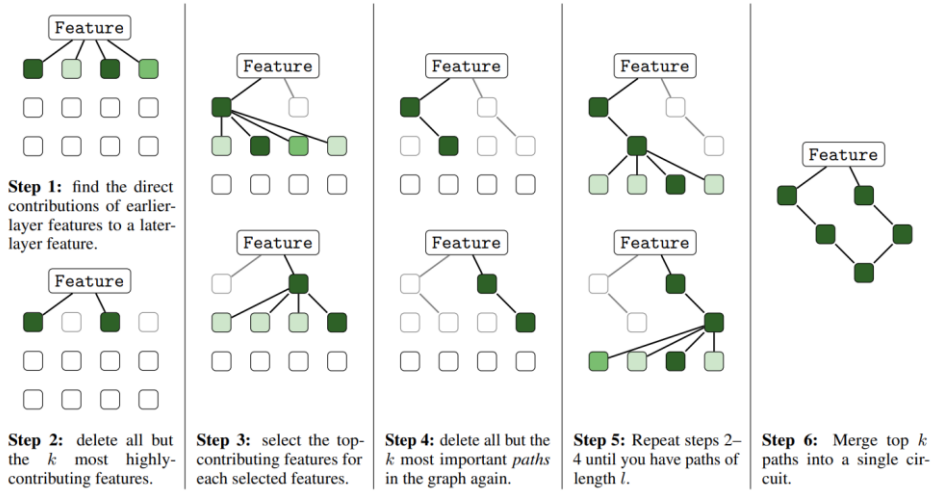
Part 3.3: Sparse Decomposition

❑ Transcoders

- ➢ **Train SAE to approximate output of dense MLP sublayers**
  - Enable edge-wise attribution-based circuit analysis
- ➢ **Comparable performance to SAEs**
  - Achieve better reconstruction fidelity with same number of features



A comparison between SAEs, MLP transcoders, and MLP sublayers



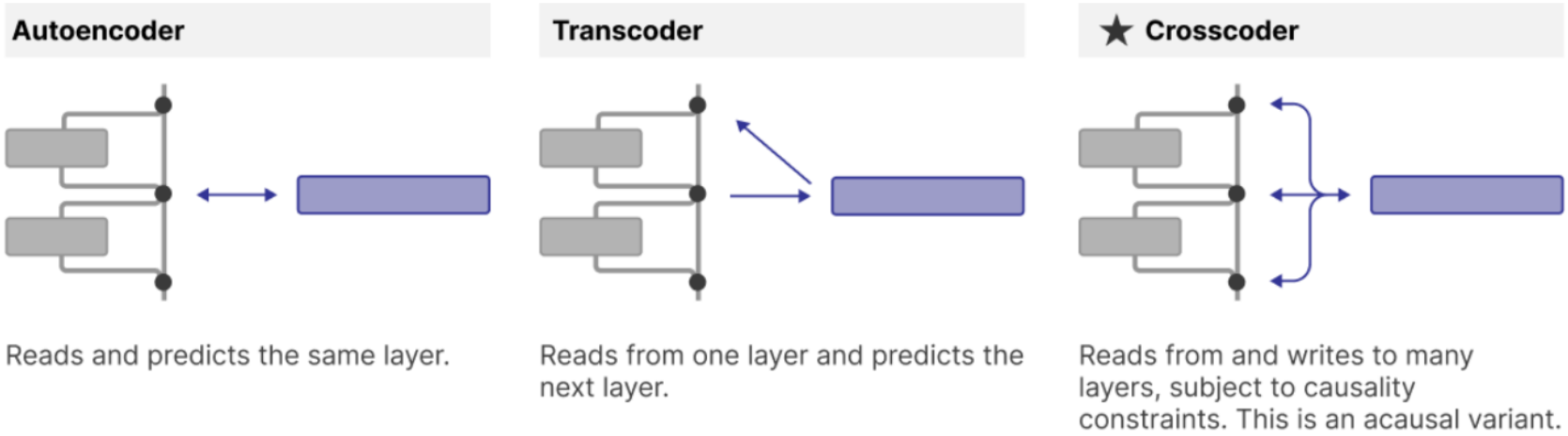
Circuit analysis with transcoder features

Dunefsky, J., et al. "Transcoders find interpretable llm feature circuits." NeurIPS 2024

Part 3.3: Sparse Decomposition

❑ Crosscoders

- ➢ **Shared dictionary learning across layers and even models**
  - Residual connections make layers “almost parallel”
  - Learning across layers can capture superposed features
- ➢ **Improved performance compared to SAEs**
  - Crosscoders trained on all layers achieve lower reconstruction error
  - Tradeoff – higher training compute cost



Lindsey, J., et al. "Sparse Crosscoders for Cross-Layer Features and Model Diffing." Transformer Circuits Thread 2024

Part 3.3: Sparse Decomposition

❑ Crosscoders

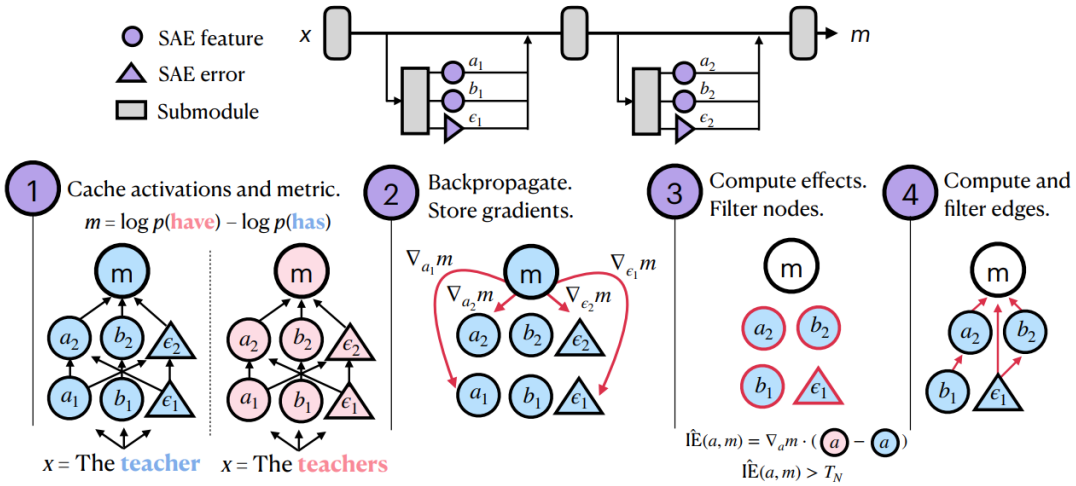
- **Model Diffing – tells us how a model changed during training**
  - Learned shared features have similar decoder vectors → represent the same concepts across models
- **Comparison to SAEs**
  - Shared features often correspond to consistent concepts across models → enables prompt-specific model diffing
  - Crosscoders support structural comparison across models (global model diffing)

Lindsey, J., et al. "Sparse Crosscoders for Cross-Layer Features and Model Diffing." Transformer Circuits Thread 2024

Part 3.3: Sparse Decomposition

❑ Sparse Feature Circuits

- **Feature-level circuit discovery with SAEs**
  - Rewrites model computations using SAE dictionary features and reconstruction errors → nodes become SAE features / errors
- **Attribution-based identification of important features**
  - Use attribution patching to estimate the importance of each node
  - Filter out nodes below predefined threshold
  - Repeat for edges



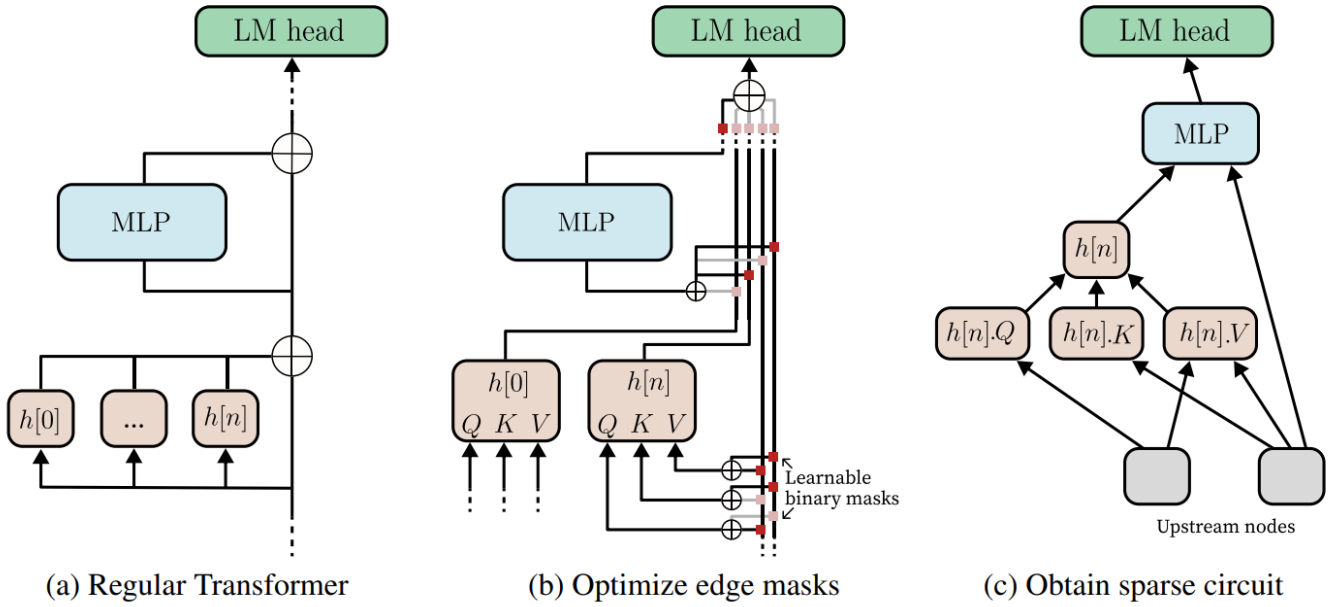
Marks, S., et al. "Sparse feature circuits: Discovering and editing interpretable causal graphs in language models." ICLR 2025

Part 3.4: Optimization-based

❑ Edge Pruning

➢ **Uses gradient-based pruning to find circuits**

- Adapts model compression techniques to circuit discovery
- Learns edge-level masks to prune edges rather than entire components
- Replaces pruned edges with counterfactual activations (edge ablation)



(a) Regular Transformer (b) Optimize edge masks (c) Obtain sparse circuit

Bhaskar, A., et al. "Finding transformer circuits with edge pruning." NeurIPS 2024

Part 3.4: Optimization-based

❑ Edge Pruning

➢ **Achieves faithful interpretations more efficiently**

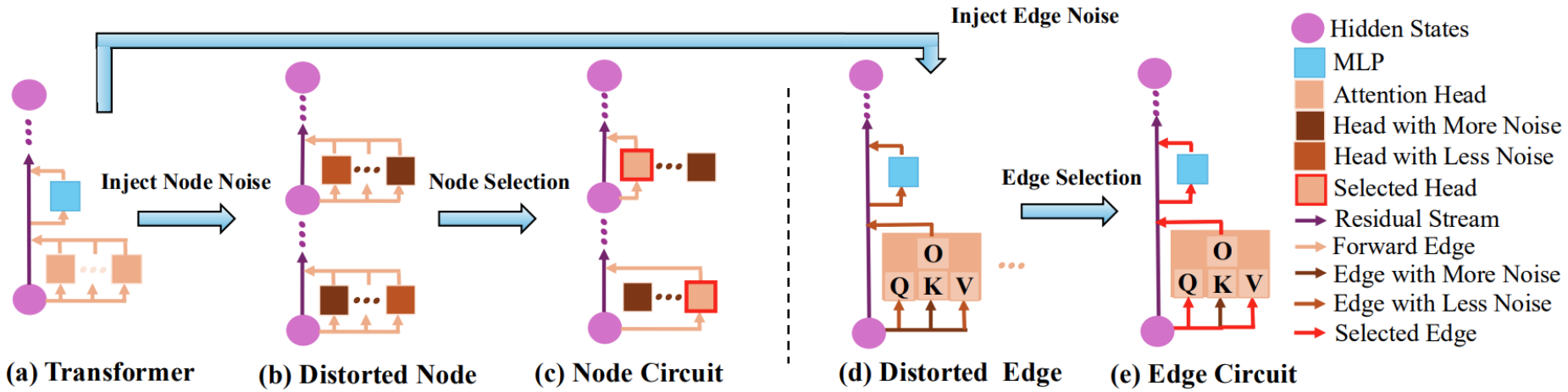
- Produces sparse circuits that closely replicate original model behavior
  - More faithful than EAP and often outperforms ACDC
- Can scale up to 100k examples while maintaining performance
  - ACDC performs well but is computationally expensive
  - EAP is much faster but performs poorly

Bhaskar, A., et al. "Finding transformer circuits with edge pruning." NeurIPS 2024

Part 3.4: Optimization-based

❑ IBCircuit

- **Information Bottleneck (IB) principle – minimize input data while maximizing the useful information for prediction**
  - Learns minimal yet faithful circuits by compressing model computation
  - Focuses on retaining only information relevant for the target task



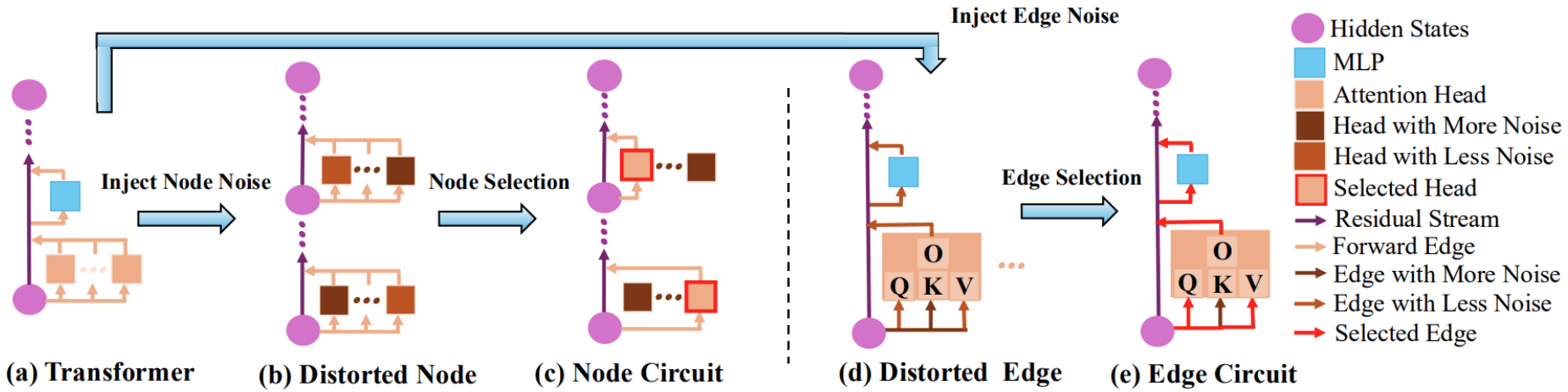
An overview of the IBCircuit framework

Bian, T., et al. "IBCircuit: Towards Holistic Circuit Discovery with Information Bottleneck." PMLR 2025

Part 3.4: Optimization-based

❑ IBCircuit

- ➢ **Learn masks (IB weights) over nodes and edges**
- ➢ **Optimization objective: compression + faithfulness tradeoff**
  - Preserves model behavior (e.g., KL divergence constraint)
  - Minimizes irrelevant information in the retained circuit



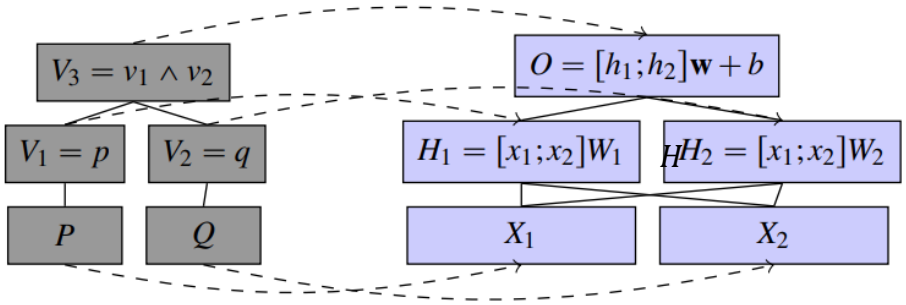
An overview of the IBCircuit framework

Bian, T., et al. "IBCircuit: Towards Holistic Circuit Discovery with Information Bottleneck." PMLR 2025

Part 3.4: Optimization-based

❑ Distributed Alignment Search (DAS)

- ➢ **Challenge – concepts are distributed**
- ➢ **Goal – align neural representations with causal variables**
  - Build a symbolic causal model of the specific task
  - Learn a mapping between intermediate activations and interpretable causal variables



$$W_1 = \begin{bmatrix} \cos(20^\circ) & -\sin(20^\circ) \\ \sin(20^\circ) & \cos(20^\circ) \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} 1 & 1 \end{bmatrix}$$
$$W_2 = \begin{bmatrix} \sin(20^\circ) & \cos(20^\circ) \end{bmatrix} \quad b = -1.8$$

Symbolic Causal Model

Low-level Neural Network

Geiger, A., et al. "Finding alignments between interpretable causal variables and distributed neural representations." PMLR 2024

Part 3.4: Optimization-based

❑ Distributed Alignment Search (DAS)

- **Key mechanism – rotation-based disentanglement**
  - Uses rotation matrices to identify subspaces encoding causal variables
  - Handles polysemantic representations via subspace alignment
  - Disentangle causal variables into more distinct dimensions
- **Causal validation via interventions**
  - Performs interventions in aligned representation space
  - Tests whether neural network behavior matches the causal model

Geiger, A., et al. "Finding alignments between interpretable causal variables and distributed neural representations." PMLR 2024

Part 3.4: Optimization-based

❑ Boundless DAS

➢ **Removes the need to predefine causal subspace dimensionality**

- DAS requires a specified dimension for the rotation matrices (unknown)
  - Need to perform parameter search for DAS
- Boundless DAS learns a continuous mask to select relevant dimensions
  - Apply rotation only after identifying the important subspace
  - Eliminates need to brute-force search

Wu, Z., et al. "Interpretability at scale: Identifying causal mechanisms in alpaca." NeurIPS 2023

Part 4: Mechanism Validation

Validating correctness of the discovered mechanisms

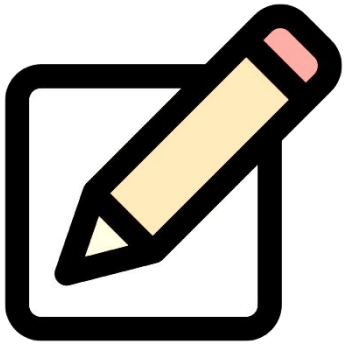
Presenter: Yinhan He

Part 4: Why Validation Matters

Discovery only determines importance

Behavior success does not mean ground truth

Empirical claims have escape values



Correctness is necessary for editing

Part 4.1: Formalizing MI

❑ Four Axioms of MI

➢ Preparation

- d_t: neural network function
- d_h: "human recipe", i.e., the high-level mechanism (algorithm) the model perform a task/behavior
- $d_h[i]$ represents the human recipe operating its ith step, $d_t[i]$ represents the neural network operating its computations corresponding to the ith step of the "human recipe"
- $\alpha_i(\cdot)$ function mapping network numbers to human symbols at step i,
- $\gamma_i(\cdot) = \alpha_i^{-1}(\cdot)$

Palumbo, N., et al. "Validating Mechanistic Interpretations: An Axiomatic Approach." ICML 2025

Part 4.1: Formalizing MI

❑ Four Axioms of MI

Axiom 1 — Prefix Equivalence

The human recipe's first i steps must produce the same result as the real network's first i steps, when translated to the same symbol space.

$$\Pr_{x \sim \mathcal{D}} \left[\alpha_i \circ d_t[:i{+}1](x) \;=\; d_h[:i{+}1] \circ \alpha_0(x) \right] \;\geq\; 1 - \varepsilon$$

α_i
Translate network numbers → human symbols at step i

$d_t[:i{+}1]$
Real network running its first i steps

$d_h[:i{+}1]$
Human recipe running its first i steps

$\alpha_0(x)$
Translate the raw input x into symbols first

$x \sim \mathcal{D}$
Random realistic input drawn from the data distribution

$\geq 1 - \varepsilon$
Must be true on almost every input — only ε fraction allowed to fail

Palumbo, N., et al. "Validating Mechanistic Interpretations: An Axiomatic Approach." ICML 2025

Part 4.1: Formalizing MI

❑ Four Axioms of MI

Axiom 2 — Component Equivalence

Each single step of the human recipe must match the real network at that exact step — not just the accumulated result, but the individual piece.

$$\Pr_{x\sim\mathcal{D}}\left[\alpha_i \circ d_t[:i{+}1](x) \;=\; d_h[i] \circ \alpha_{i-1} \circ d_t[:i](x)\right] \;\geq\; 1-\varepsilon$$

$\alpha_i \circ d_t[:i{+}1]$

Real network's output at step i, decoded into symbols

$d_h[i]$

Just ONE step (step i) of the human recipe

$\alpha_{i-1} \circ d_t[:i]$

Real network's output from step i−1, decoded — fed as input to the human step

$x \sim \mathcal{D}$

Random realistic input from the training distribution

$\geq 1-\varepsilon$

This single step must match on almost every input

Palumbo, N., et al. "Validating Mechanistic Interpretations: An Axiomatic Approach." ICML 2025

Part 4.1: Formalizing MI

❑ Four Axioms of MI

Axiom 3 — Prefix Replaceability

Swap the real network's first i steps with the human recipe steps (converting back to numbers with γ). The final answer must still be correct.

$$\Pr_{x \sim \mathcal{D}}\left[t(x) \;=\; d_t[i{+}1:] \circ \gamma_i \circ d_h[:i{+}1] \circ \alpha_0(x)\right] \;\geq\; 1-\varepsilon$$

$t(x)$
The real network's correct final answer

$d_t[i{+}1:]$
Real network finishing the job from step i+1 onward

γ_i
Reverse-translate: human symbols → network numbers at step i

$d_h[:i{+}1] \circ \alpha_0$
Human recipe's first i steps, starting from the translated input

$\geq 1-\varepsilon$
Swapping in the human prefix must give the same final answer on almost every input

Palumbo, N., et al. "Validating Mechanistic Interpretations: An Axiomatic Approach." ICML 2025

Part 4.1: Formalizing MI

❑ Four Axioms of MI

Axiom 4 — Component Replaceability

Swap in just ONE human step (step i) into the real network's pipeline. The final answer must still be correct.

$$\Pr_{x \sim \mathcal{D}}\left[t(x) \;=\; d_t[i{+}1:] \circ \gamma_i \circ d_h[i] \circ \alpha_{i-1} \circ d_t[:i](x)\right] \;\geq\; 1-\varepsilon$$

$t(x)$	$d_t[i{+}1:]$	γ_i	$d_h[i]$	$\alpha_{i-1} \circ d_t[:i]$
Real network's correct final answer	Real network's remaining steps after i	Symbols → numbers reverse-translator	ONE human step only (step i)	Real network up to step i−1, decoded for the human step to consume

$\geq 1-\varepsilon$

Even swapping just one step must preserve the final answer on almost every input

Palumbo, N., et al. "Validating Mechanistic Interpretations: An Axiomatic Approach." ICML 2025

Part 4.1: Formalizing MI

❑ Provable Guarantees

➢ Input Domain Robustness

- A circuit is input-robust if replacing all components outside the circuit with fixed values still preserves the network's output for **every input** within a neighborhood of interest.

Definition 1 (Input-robust circuit). *Given a neural network f_G and a union of continuous domains $\mathcal{Z} \subseteq \mathbb{R}^n$ (e.g., a union of ℓ_p-balls of radius ϵ_p around a set of discrete points $\{\boldsymbol{x}_j\}_{j=1}^k$), we say that a subgraph $C \subseteq G$ is an* input-robust circuit *with respect to $\langle f_G, \mathcal{Z} \rangle$, a fixed patching vector α applied to the complement set $\overline{C} := G \setminus C$, and a tolerance level $\delta \in \mathbb{R}^+$, if and only if:*

$$\forall \boldsymbol{z} \in \mathcal{Z} := \bigcup_{j=1}^{k} \mathcal{B}_{\epsilon_p}^p(\boldsymbol{x}_j). \quad \|f_C(\boldsymbol{z} \mid \overline{C} = \alpha) - f_G(\boldsymbol{z})\|_p \leq \delta, \tag{1}$$

$$s.t.\ \mathcal{B}_{\epsilon_p}^p(\boldsymbol{x}_j) := \{\boldsymbol{z}_j \in \mathbb{R}^n \mid \|\boldsymbol{z}_j - \boldsymbol{x}_j\|_p \leq \epsilon_p\}$$

A ball centered at p with radius ϵ_p

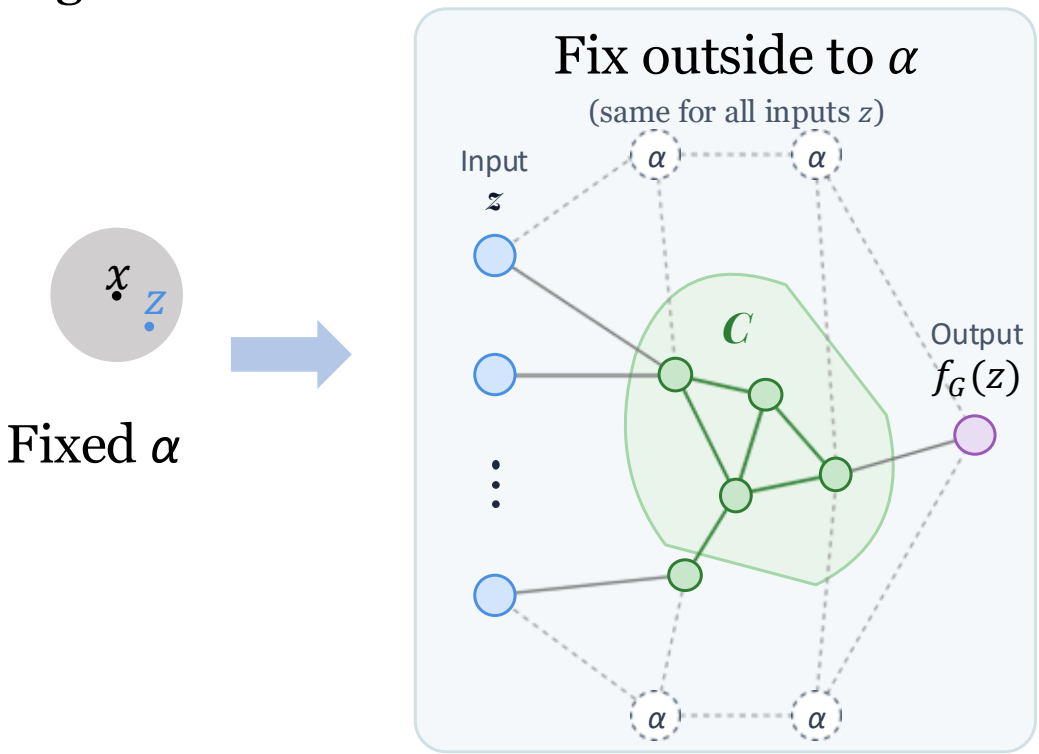
Hadad, I., et al. "Formal Mechanistic Interpretability: Automated Circuit Discovery with Provable Guarantees." ICLR 2026

Part 4.1: Formalizing MI

❑ Provable Guarantees

➢ Input Domain Robustness

- A circuit is input-robust if replacing all components outside the circuit with fixed values still preserves the network's output for **every input** within a neighborhood of interest.



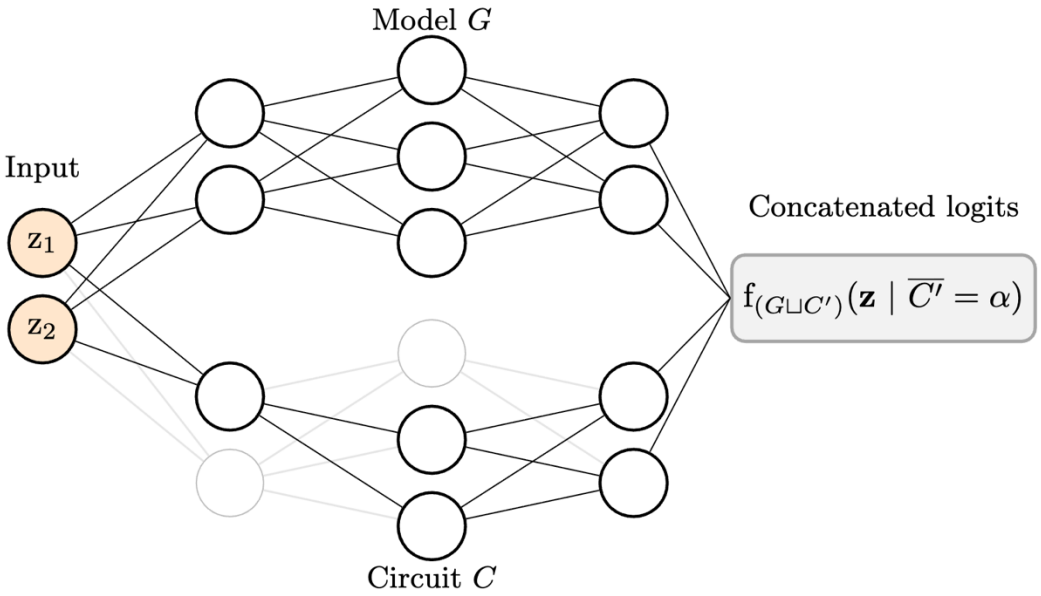
Hadad, I., et al. "Formal Mechanistic Interpretability: Automated Circuit Discovery with Provable Guarantees." ICLR 2026

Part 4.1: Formalizing MI

❑ Provable Guarantees

➢ Input Domain Robustness

- Certification: A siamese encoding transforms circuit verification into standard neural network verification by evaluating the model and circuit jointly.



Hadad, I., et al. "Formal Mechanistic Interpretability: Automated Circuit Discovery with Provable Guarantees." ICLR 2026

Part 4.1: Formalizing MI

❑ Provable Guarantees

➢ Robust Patching

- A circuit is patching-robust if its behavior remains unchanged regardless of how the rest of the network is patched.

Definition 2 (Patching-robust circuit). *Given a neural network f_G, a continuous input domain $\mathcal{Z} \subseteq \mathbb{R}^n$, and a reference set of inputs $\{\boldsymbol{x}_j\}_{j=1}^k \subseteq \mathcal{X}$, we say that $C \subseteq G$ is a* patching-robust circuit *with respect to $\langle f_G, \mathcal{Z} \rangle$ and a tolerance level $\delta \in \mathbb{R}^+$, iff for every $\boldsymbol{x}_j$ in $\{\boldsymbol{x}_j\}_{j=1}^k$:*

$$\forall \alpha \in \mathcal{H}_{\overline{C}}(\mathcal{Z}). \quad \|f_C(\boldsymbol{x}_j \mid \overline{C} = \alpha) - f_G(\boldsymbol{x}_j)\|_p \leq \delta \tag{2}$$

Activation space of $\bar{C}$ given all inputs within Z

Hadad, I., et al. "Formal Mechanistic Interpretability: Automated Circuit Discovery with Provable Guarantees." ICLR 2026

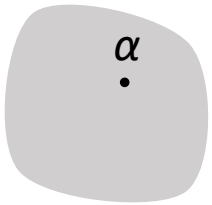
Part 4.1: Formalizing MI

❑ Provable Guarantees

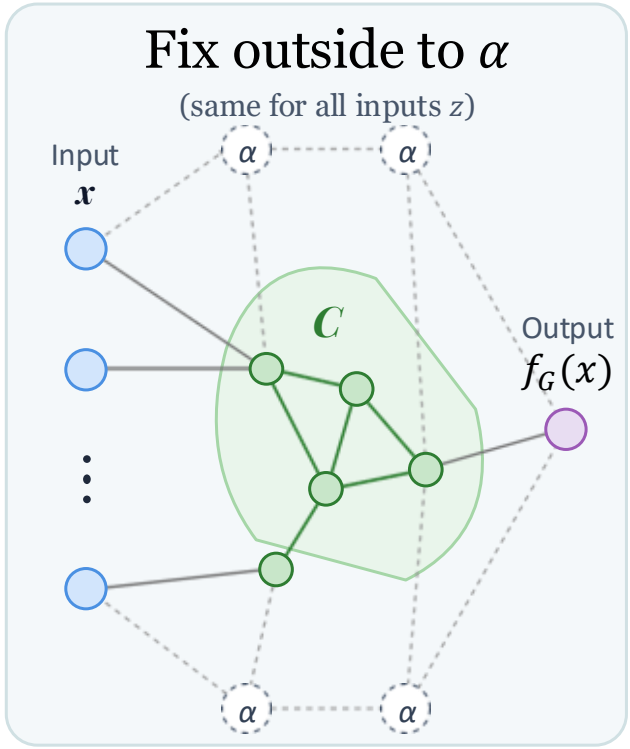
➢ Robust Patching

- A circuit is patching-robust if its behavior remains unchanged regardless of how the rest of the network is patched.

For any $\alpha \in H_{\bar{C}}(Z)$



Fixed x



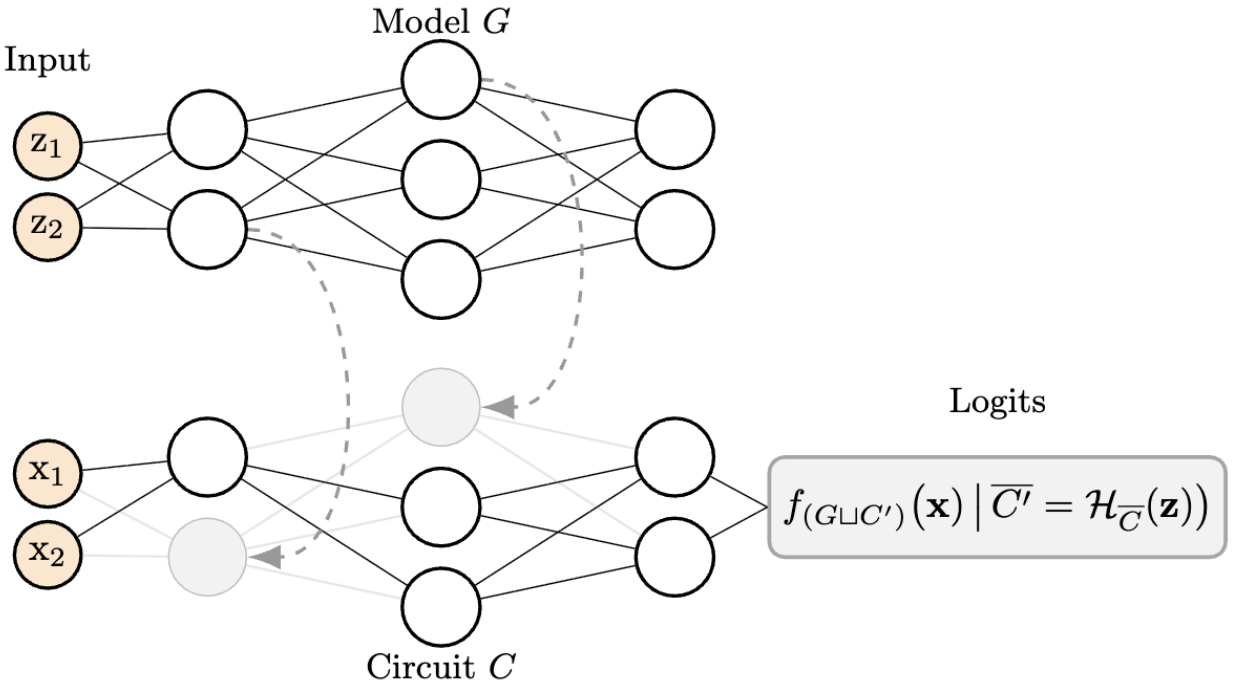
Hadad, I., et al. "Formal Mechanistic Interpretability: Automated Circuit Discovery with Provable Guarantees." ICLR 2026

Part 4.1: Formalizing MI

❑ Provable Guarantees

➢ Robust Patching

- Certification: Duplicate the circuit, connect it to the full model through patched activations, and verify that both produce similar outputs across all feasible patches.



Hadad, I., et al. "Formal Mechanistic Interpretability: Automated Circuit Discovery with Provable Guarantees." ICLR 2026

Part 4.1: Formalizing MI

❑ Provable Guarantees

➢ Minimality guarantee

- General Idea:
  - Discovered circuits are required to be near-minimal
  - No redundant components included beyond what is necessary for correct behavior

Hadad, I., et al. "Formal Mechanistic Interpretability: Automated Circuit Discovery with Provable Guarantees." ICLR 2026

Part 4.1: Formalizing MI

❑ Provable Guarantees

➢ Minimality guarantee

- Preparation:
  - A general faithfulness predicate $\Phi(C, G)$: returns True if the circuit C in computation the graph G is faithful otherwise False.

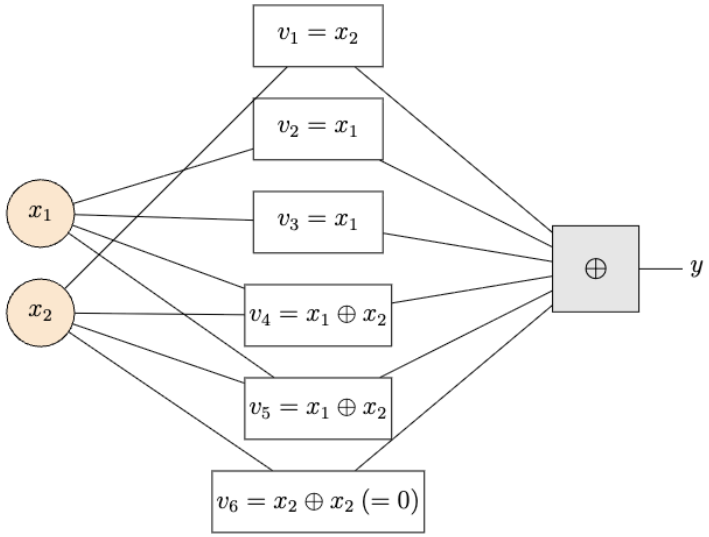
Hadad, I., et al. "Formal Mechanistic Interpretability: Automated Circuit Discovery with Provable Guarantees." ICLR 2026

Part 4.1: Formalizing MI

❑ Provable Guarantees

➢ Minimality guarantee

- Preparation:
  - A general faithfulness predicate $\Phi(C, G)$: returns True if the circuit C in computation the graph G is faithful otherwise False.
  - E.g., $x_i \in \{0,1\}$



$$f_G(x_1, x_2) := x_2 \oplus x_1 \oplus x_1 \oplus (x_1 \oplus x_2) \oplus (x_1 \oplus x_2) \oplus (x_2 \oplus x_2)$$
$$= x_2 \oplus \underbrace{(x_1 \oplus x_1)}_{=0} \oplus \underbrace{[(x_1 \oplus x_2) \oplus (x_1 \oplus x_2)]}_{=0} \oplus \underbrace{(x_2 \oplus x_2)}_{=0} = x_2.$$

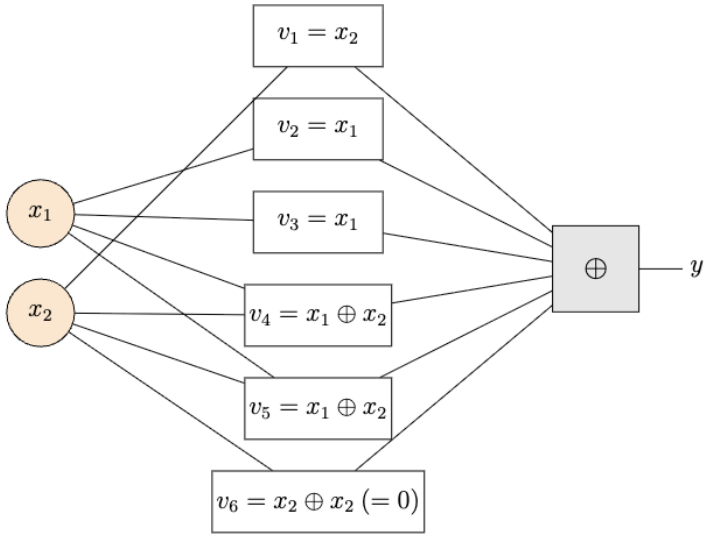
Hadad, I., et al. "Formal Mechanistic Interpretability: Automated Circuit Discovery with Provable Guarantees." ICLR 2026

Part 4.1: Formalizing MI

❑ Provable Guarantees

➢ Minimality guarantee

- Preparation:
  - A general faithfulness predicate $\Phi(C, G)$: returns True if the circuit C in computation the graph G is faithful otherwise False.
  - E.g., $x_i \in \{0,1\}$



$$\Phi(C, G) = \mathbb{I}[f_C(x_1, x_2) = f_G(x_1, x_2), \forall x_1, x_2)]$$

Hadad, I., et al. "Formal Mechanistic Interpretability: Automated Circuit Discovery with Provable Guarantees." ICLR 2026

Part 4.1: Formalizing MI

❑ Provable Guarantees

➢ Minimality guarantee

- Minimal Circuits
  - Quasi-Minimal Circuit: $\Phi(C, G) \wedge [\exists\, i \in C, \neg\Phi\big((C \setminus \{i\}), G\big)]$
  - Locally-Minimal Circuit: $\Phi(C, G) \wedge [\forall\, i \in C,\ \neg\Phi\big((C \setminus \{i\}), G\big)]$
  - Subset-Minimal Circuit: $\Phi(C, G) \wedge [\forall\, S \subseteq C,\ \neg\Phi\big((C \setminus S), G\big)]$
  - Cardinally-Minimal Circuit: $\nexists\, C', s.t. \Phi(C', G) \wedge |C'| < |C|$

Hadad, I., et al. "Formal Mechanistic Interpretability: Automated Circuit Discovery with Provable Guarantees." ICLR 2026

Part 4.1: Formalizing MI

❑ Certified Circuits

- **Goal – make circuits provably stable under data changes**
  - Certification – a circuit remains unchanged for all datasets *D'* within an edit-distance radius *r* of *D*
  - Addresses the issue that circuit discovery can be brittle
- **Key idea – worst-case robustness guarantee**
  - Ensures the selected circuit components are invariant under bounded data perturbations

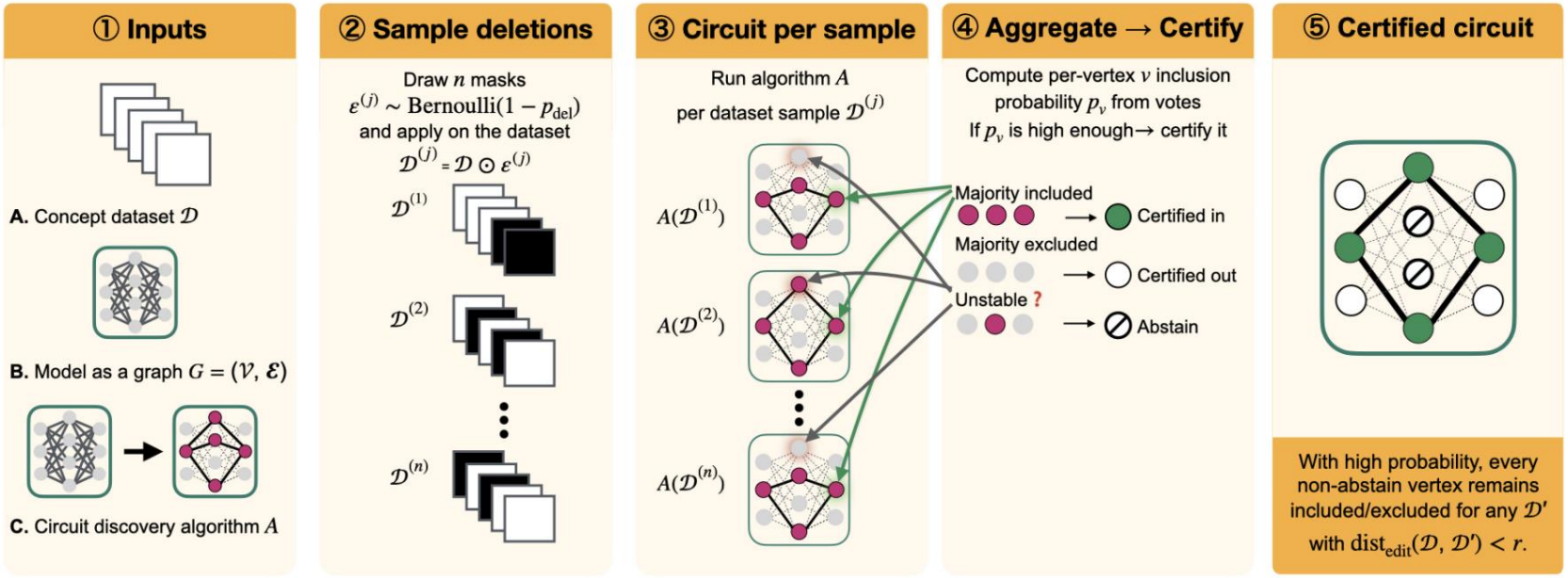
Anani, A., et al. "Certified Circuits: Stability Guarantees for Mechanistic Circuits." arXiv:2602.22968

Part 4.1: Formalizing MI

❑ Certified Circuits

➢ Algorithm-agnostic framework

- Works as a wrapper around any existing circuit discovery method
- Does not require redesigning how circuits are found



Anani, A., et al. "Certified Circuits: Stability Guarantees for Mechanistic Circuits." arXiv:2602.22968

Part 4.2: Validation Metrics

- **Faithfulness (the smaller the better):**

$$|L(C) - L(G)|$$

- **Can the circuit alone reproduce model behavior?**
  - Compare original model behavior $L(G)$ to the behavior when only the circuit is enabled $L(C)$
    - C is the candidate circuit to validate
    - G is the entire computational graph
    - $L(\cdot)$ is the behavioral metric representing the importance of a model component / a group of components (the larger the more important)

Wang, K., et al. "Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 Small." ICLR 2023

Part 4.2: Validation Metrics

- **Faithfulness (the smaller the better):**

$$|L(C) - L(G)|$$

- **Lacks robustness**
  - Different performance when ablating nodes vs edges
  - Sensitive to the ablation method (i.e., mean, zero, resample, corrupt)
  - Wide variance in performance across and within datasets
- **Alternative definition: can the method recover "ground truth?"**
  - Ground truth defined as previously discovered circuits (i.e., IOI)
  - Limitation – requires a defined ground truth → cannot be used for unexplored tasks

Wang, K., et al. "Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 Small." ICLR 2023
Miller, J., et al. "Transformer Circuit Evaluation Metrics Are Not Robust." COLM 2024
Syed, A., et al. "Attribution patching outperforms automated circuit discovery." BlackboxNLP 2024

Part 4.2: Validation Metrics

- **Completeness (the smaller the better):**

$$\max_{K \subseteq C} |L(C \backslash K) - L(G \backslash K)|$$

- **Are there any important components not in the circuit?**
  - Circuit should perform like the original model under same ablations
    - K is any valid subset of nodes in circuit C
    - $|L(C \backslash K) - L(G \backslash K)|$ is the incompleteness score
      - Measures the difference in circuit and model performance when K is removed
    - Want to minimize the incompleteness score to achieve completeness

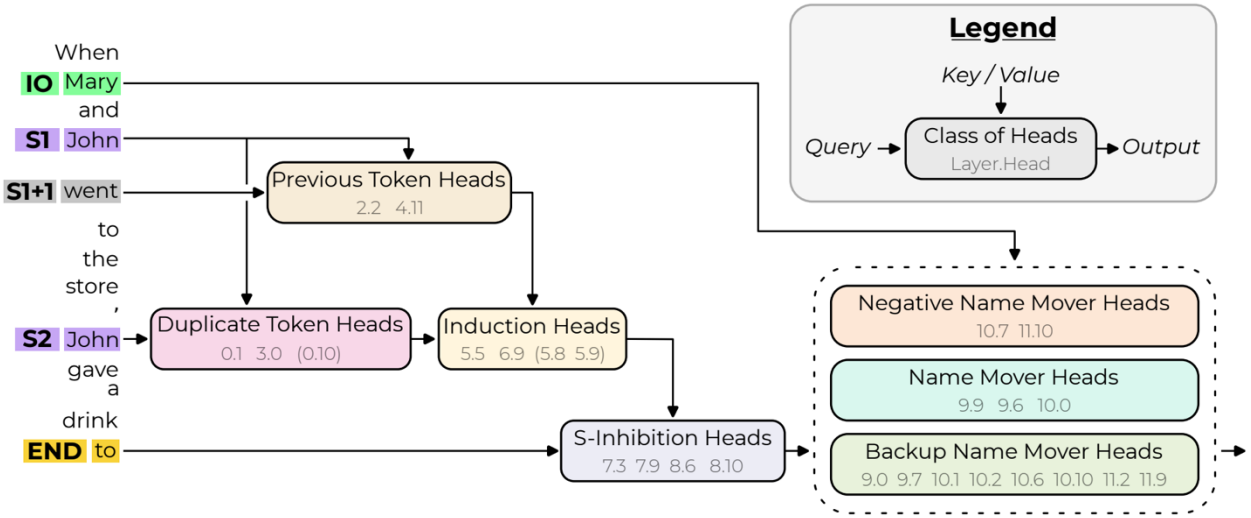
Wang, K., et al. "Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 Small." ICLR 2023

Part 4.2: Validation Metrics

- **Completeness (the smaller the better):**

$$\max_{K \subseteq C} |L(C \backslash K) - L(G \backslash K)|$$

- **E.g.: K is the set of name-mover heads in IOI**
  - $L(C \backslash \mathrm{K})$ should be small because the circuit is missing a key node group
  - If C is complete, $L(G \backslash \mathrm{K})$ should be small because C captures all of the necessary nodes; If C is incomplete, $L(G \backslash \mathrm{K})$ should be large because $G \backslash K$ contains redundant name-mover heads



Wang, K., et al. "Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 Small." ICLR 2023

Part 4.2: Validation Metrics

- **Minimality (the larger the better):**

$$\min_{v \in C} \max_{K \subseteq C \backslash \{v\}} |L(C \backslash (K \cup \{v\})) - L(C \backslash K)|$$

- **Are there any irrelevant components in the circuit?**
  - Validate that all nodes in the circuit are necessary
    - v is any node in circuit C
    - K is any valid subset of nodes in circuit C without v
    - $|L(C \backslash (K \cup \{v\})) - L(C \backslash K)|$ is the minimality score
      - Measures the performance recovered from adding node v

Why $|L(C \backslash (K \cup \{v\})) - L(C \backslash K)|$ rather than $|L(C \backslash \{v\}) - L(C)|$?

An important component v may have backup components in C that perform the same functionality, keeping $|L(C\{v\}) - L(C)|$ small and causing us to mistakenly consider v unnecessary. We therefore first remove these backups using K, allowing us to more faithfully measure the minimality score of v.

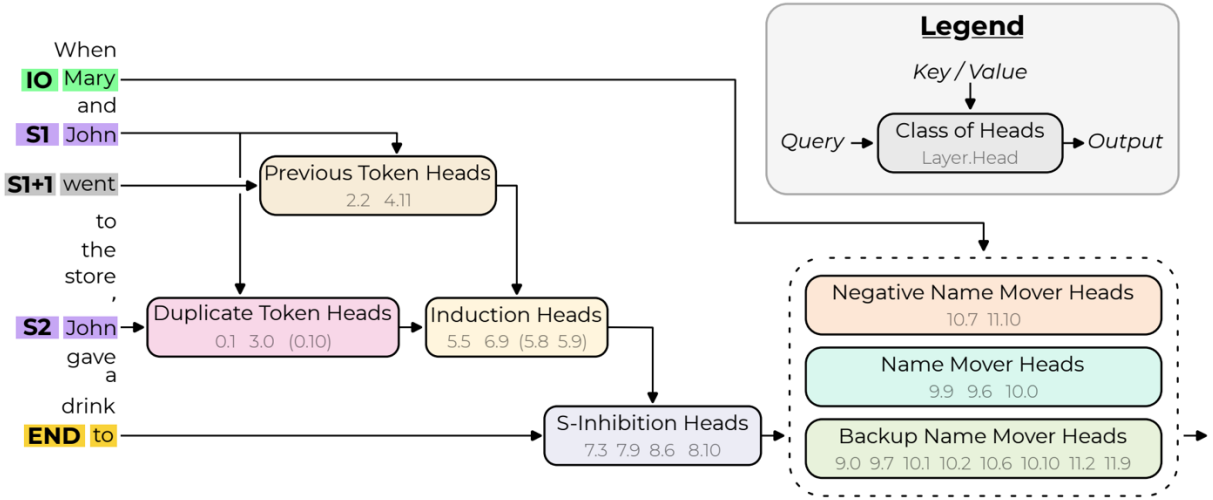
Wang, K., et al. "Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 Small." ICLR 2023

Part 4.2: Validation Metrics

- **Minimality (the larger the better):**

$$\min_{v \in C} \max_{K \subseteq C \setminus \{v\}} |L(C \setminus (K \cup \{v\})) - L(C \setminus K)|$$

- **E.g.:** Assume the circuit has three name mover heads {A,B,C}. K is the set of name-mover heads {A, B} in IOI, v is name mover head C
  - $L(C \setminus (K \cup \{v\}))$ should be small because the IOI circuit is missing a key node group; If C is minimal, $L(C \setminus K)$ should be large because v is relevant and important; If C is not minimal, $L(C \setminus K)$ should remain a small value because v is irrelevant.



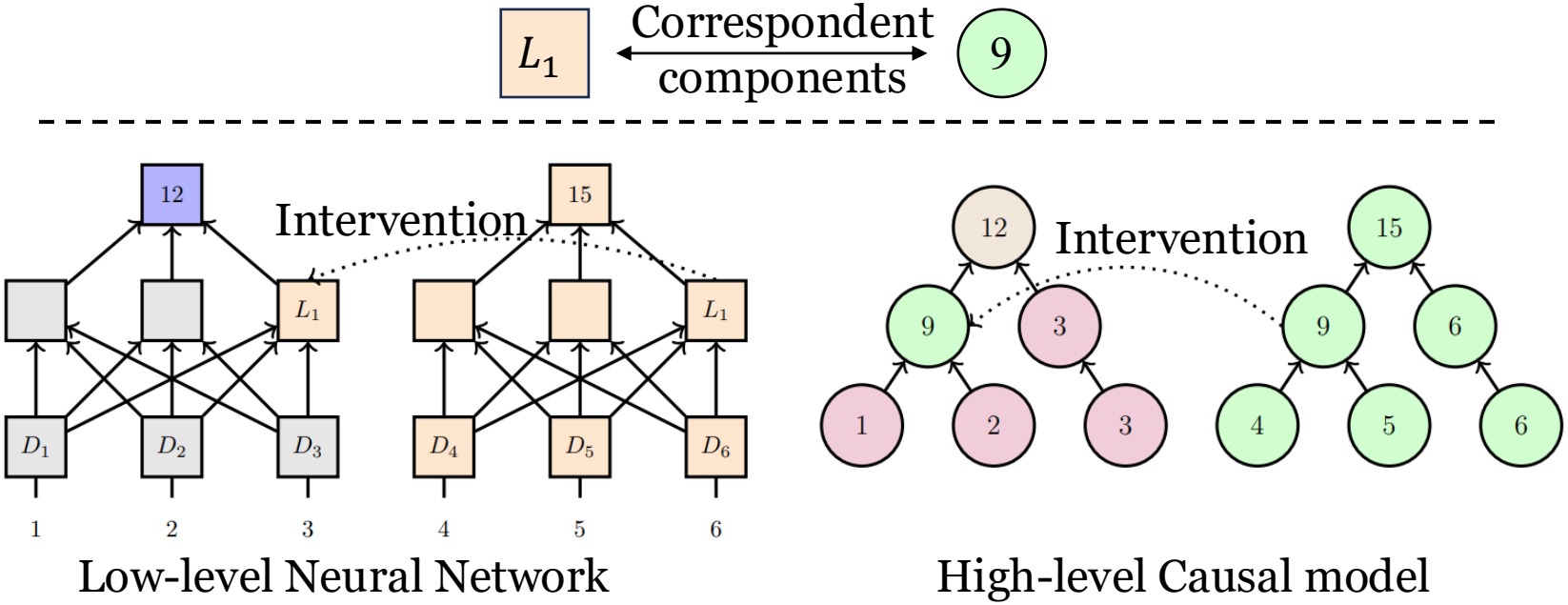
Wang, K., et al. "Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 Small." ICLR 2023

Part 4.3: Validation Tools

❑ Interchange Intervention

➢ **Tests whether the model implements an abstract causal model**

- Define high-level causal explanation of a task using variables
- Hypothesize where these variables are located in the model
- Perform component-wise corresponding interventions on high and low-level model, check if output of high and low-level models are equal

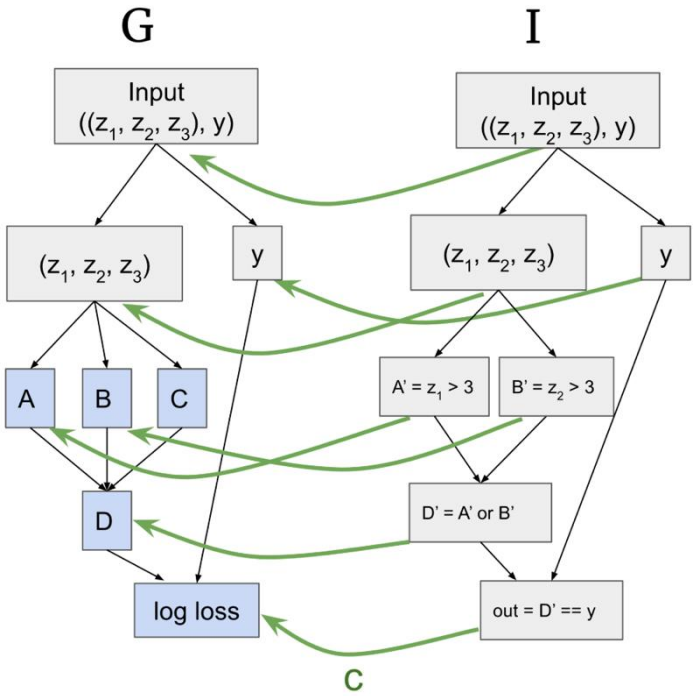


Geiger, A., et al. "Causal abstractions of neural networks." NeurIPS 2021

Part 4.3: Validation Tools

❑ Causal Scrubbing

- **Used to evaluate hypotheses *h = (G, I, c)***
  - *G* is the computational graph
  - *I* is the interpretation that *h* claims to explain the model behavior
  - *c* is the mapping of nodes between *I* and *G*



Chan, L., et al. "Causal scrubbing, a method for rigorously testing interpretability hypotheses." AI Alignment Forum 2022

Part 4.3: Validation Tools

❑ Causal Scrubbing

- **Key Idea:** Scrub (i.e. remove) all causal relationships that *h* deems unnecessary
  - If model behavior is preserved → hypothesis captures the causal structure of the computation
  - If behavior changes → hypothesis is incomplete or incorrect

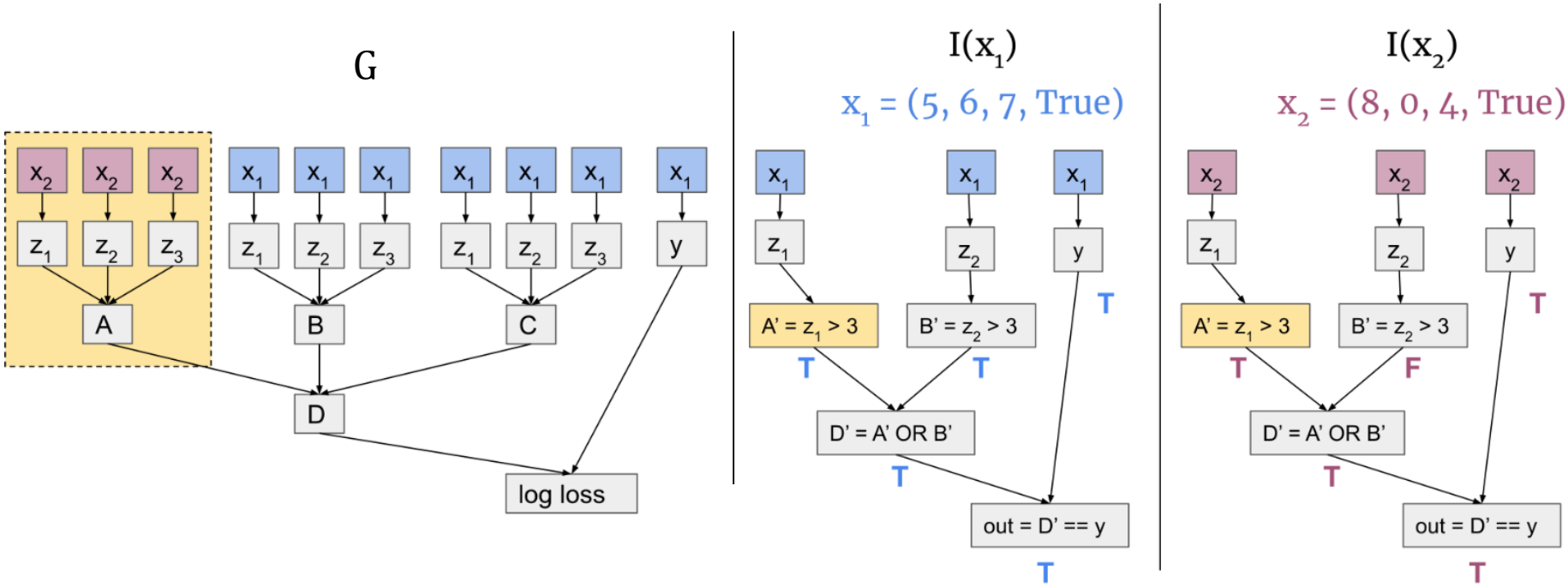
Chan, L., et al. "Causal scrubbing, a method for rigorously testing interpretability hypotheses." AI Alignment Forum 2022

Part 4.3: Validation Tools

❑ Causal Scrubbing

➢ Scrubbing for Important Components

- Important components means the components of G in $c(I)$
- Find the input x_2 whose activation at component of interest match that of the reference input x_1



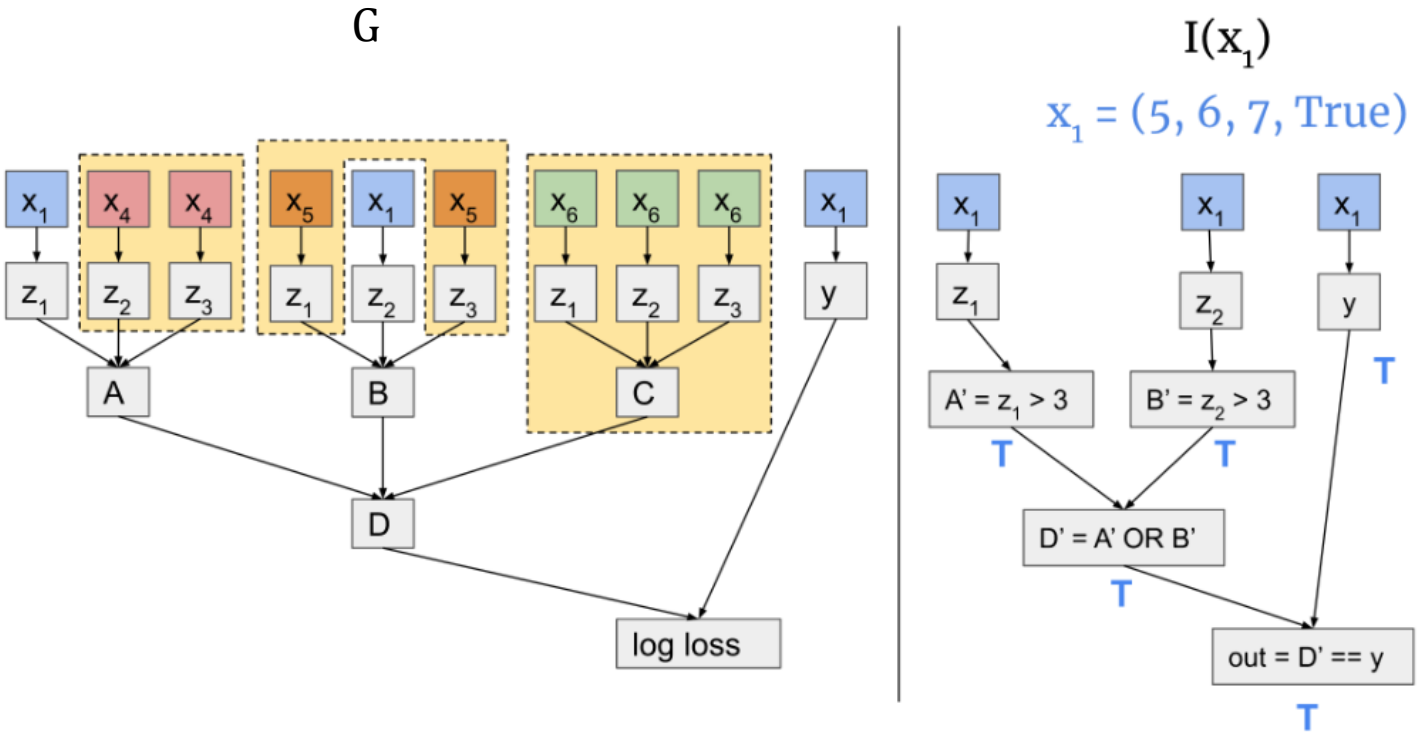
Chan, L., et al. "Causal scrubbing, a method for rigorously testing interpretability hypotheses." AI Alignment Forum 2022

Part 4.3: Validation Tools

❑ Causal Scrubbing

➢ Scrubbing for Unimportant Components

- Random possible input can serve as x_2 as those components are not in $I(\cdot)$



Chan, L., et al. "Causal scrubbing, a method for rigorously testing interpretability hypotheses." AI Alignment Forum 2022

Part 4.3: Validation Tools

❑ Causal Scrubbing – the process

➢ **Input – randomly sample 2 instances (x_1, x_2) from the dataset**

- x_1 – initial starting point for scrubbing (reference input)
- x_2 – randomly resampled activations

```
def run_scrub(node_I, x_1):
    Initialize input_G to hold the output activations for each node in G
    Use c to find node node_G in G corresponding to node_I
    If node_G is input:
        Return x_1
    For parent_node_G of node_G:
        Use c to find node parent_node_I of parent_node_G
        If parent_node_I in I:
            Find x_new that satisfies parent_node_I(x_1) == parent_node_I(x_new)
            Set input_G[parent_node_G] = run_scrub(parent_node_I, x_new)
        Else:
            Set input_G[parent_node_G] = parent_node_G(x_2)
    Gather the new outputs of all parent_node_G from input_G
    Recompute and return the output of node_G using the new outputs
```

Chan, L., et al. "Causal scrubbing, a method for rigorously testing interpretability hypotheses." AI Alignment Forum 2022

Part 4.3: Validation Tools

❑ Causal Scrubbing – the process

➢ **Input – randomly sample 2 instances (x_1, x_2) from the dataset**

- x_1 – initial starting point for scrubbing (reference input)
- x_2 – randomly resampled activations

```
def run_scrub(node_I, x_1):
    Initialize input_G to hold the output activations for each node in G
    Use c to find node node_G in G corresponding to node_I
    If node_G is input:
        Return x_1
    For parent_node_G of node_G:
        Use c to find node parent_node_I of parent_node_G
        If parent_node_I in I:
            Find x_new that satisfies parent_node_I(x_1) == parent_node_I(x_new)
            Set input_G[parent_node_G] = run_scrub(parent_node_I, x_new)
        Else:
            Set input_G[parent_node_G] = parent_node_G(x_2)
    Gather the new outputs of all parent_node_G from input_G
    Recompute and return the output of node_G using the new outputs
```

Scrubbing for important components

Chan, L., et al. "Causal scrubbing, a method for rigorously testing interpretability hypotheses." AI Alignment Forum 2022

Part 4.3: Validation Tools

❑ Causal Scrubbing – the process

➢ **Input – randomly sample 2 instances (x_1, x_2) from the dataset**

- x_1 – initial starting point for scrubbing (reference input)
- x_2 – randomly resampled activations

```
def run_scrub(node_I, x_1):
    Initialize input_G to hold the output activations for each node in G
    Use c to find node node_G in G corresponding to node_I
    If node_G is input:
        Return x_1
    For parent_node_G of node_G:
        Use c to find node parent_node_I of parent_node_G
        If parent_node_I in I:
            Find x_new that satisfies parent_node_I(x_1) == parent_node_I(x_new)
            Set input_G[parent_node_G] = run_scrub(parent_node_I, x_new)
        Else:
            Set input_G[parent_node_G] = parent_node_G(x_2)
    Gather the new outputs of all parent_node_G from input_G
    Recompute and return the output of node_G using the new outputs
```

Scrubbing for unimportant components

Chan, L., et al. "Causal scrubbing, a method for rigorously testing interpretability hypotheses." AI Alignment Forum 2022

Part 4.3: Validation Tools

❑ Interpretive equivalence

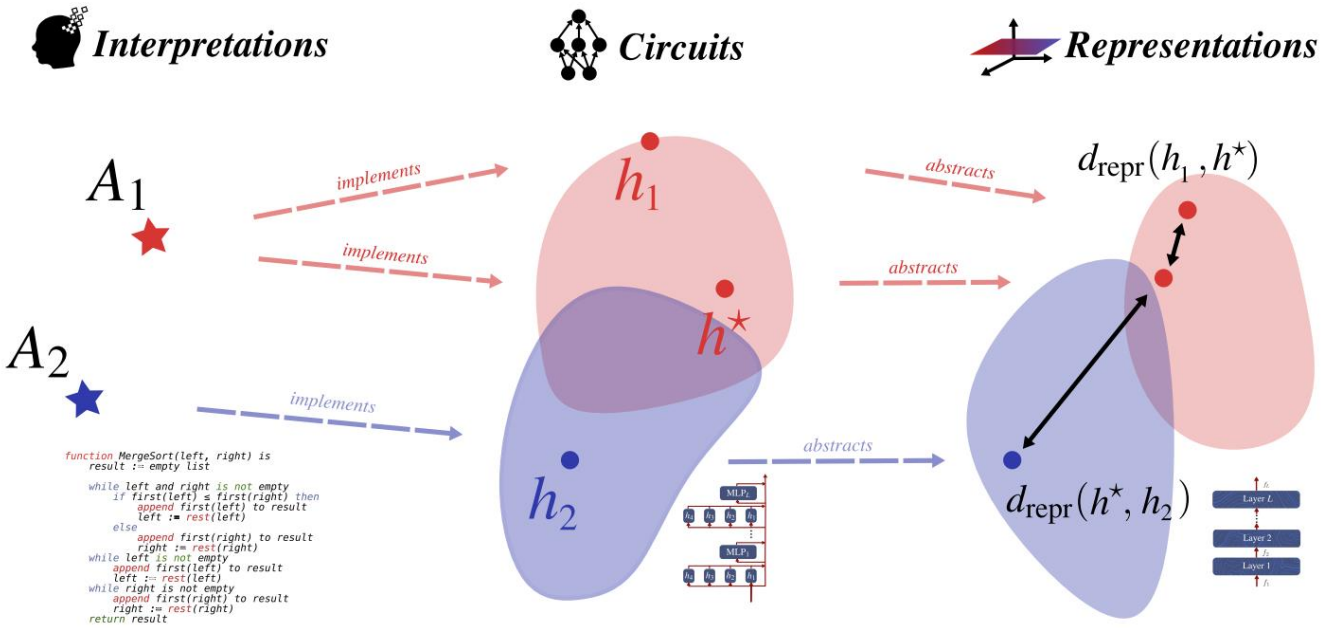
- ➢ **Models are equivalent if they share the same interpretation**
  - Ignores low-level implementation details (i.e. parameters, circuits, etc.)
  - Equivalent models respond similarly under the same interventions

Sun, A., & Toneva, M. "Tracking Equivalent Mechanistic Interpretations Across Neural Networks." ICLR 2026

Part 4.3: Validation Tools

❑ Interpretive equivalence

- ➢ **Low-level implementations – h_1, h_2**
- ➢ **High-level interpretations – A_1, A_2**



$$d_{\text{repr}}(h^\star, h_2) \not\approx d_{\text{repr}}(h_1, h^\star) \Rightarrow A_1 \neq A_2$$

$$\sum d_{repr}(h^*, h_2) \leq \sum d_{repr}(h^*, h_1) \Rightarrow A_1 = A_2$$

Sun, A., & Toneva, M. "Tracking Equivalent Mechanistic Interpretations Across Neural Networks." ICLR 2026

Part 4.3: Validation Tools

❑ Interpretive equivalence

- **Model reduction via interpretive equivalence**
  - Large and small models can be treated as equivalent if they share the same causal interpretation under congruity
  - Enables compressing large models into simpler surrogates that preserve only causally relevant computations
- **Task reduction via shared causal structure**
  - Different tasks can be mapped to a common causal abstraction if their models are interpretively equivalent
  - Allows transfer of insights across tasks, reducing complex problems to simpler, already-understood causal forms

Sun, A., & Toneva, M. "Tracking Equivalent Mechanistic Interpretations Across Neural Networks." ICLR 2026

Part 5: Mechanism Editing

From discovered mechanisms to controlled behavior change

Presenter: Tianyi Zhao

Part 5: Mechanism Editing

❑ Three Granularities of Mechanism Editing

- **Parameter-level**
  - Edit model weights at located components
- **Representation-level**
  - Edit activations at inference; weights untouched
- **Circuit / feature-level**
  - Edit the mechanistic graph itself

❑ More Applications

- **Mechanistic Unlearning**
  - Remove targeted knowledge or capability while preserving the rest
- **Bias and Hallucination Mitigation**
  - Reduce systematic failures via direct intervention on mechanism

Part 5.1: Parameter-Level Editing

❑ Core Premise

Edit the model's weights at located components

❑ Category

- **ROME-lineage**
  - Closed-form rank-one updates at causally-located components
- **Hypernetwork Editors**
  - Learn a meta-network that maps (edit, gradient) → weight update
- **Mechanism-aware Multi-hop Editing**
  - Edit the circuit (not a single layer) so edits propagate to logical consequences
- **Lifelong / Streaming Editing**
  - Route edits through auxiliary storage to preserve base weights across many sequential edits

Part 5.1: Parameter-Level Editing

❑ ROME

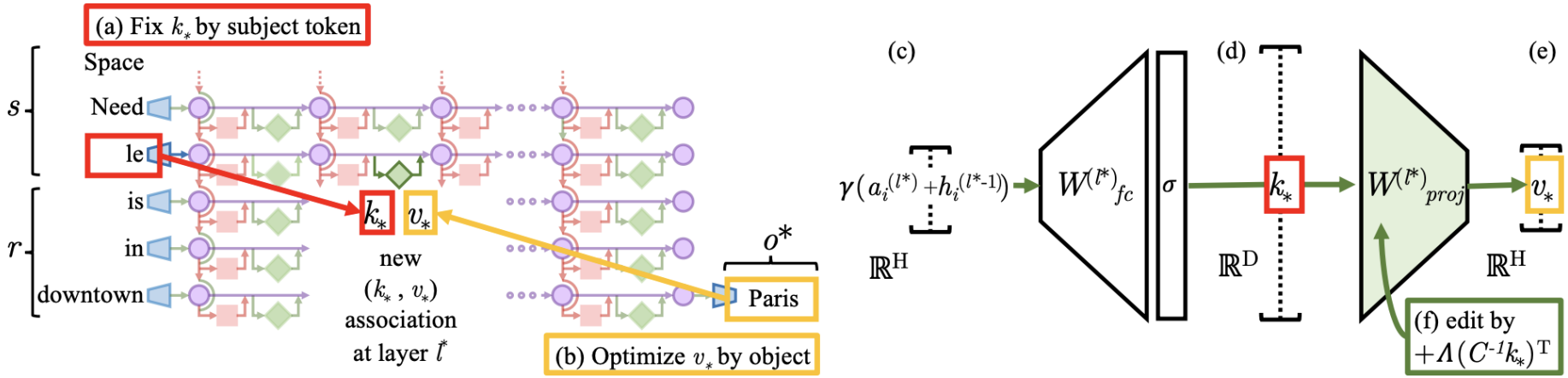
➢ **Premise: factual recall is mediated by mid-layer MLPs**

Causal tracing identifies the layer ℓ* for the fact.

Use the MLP as key–value memory; compute the new value vector.

Apply a closed-form rank-one update: $W \leftarrow W + \Delta$

1 locate ℓ* → **2 optimize v*** → **3 update W**



Meng, K. et al. "Locating and Editing Factual Associations in GPT." NeurIPS 2022

Part 5.1: Parameter-Level Editing

❑ Scaling and Refining ROME

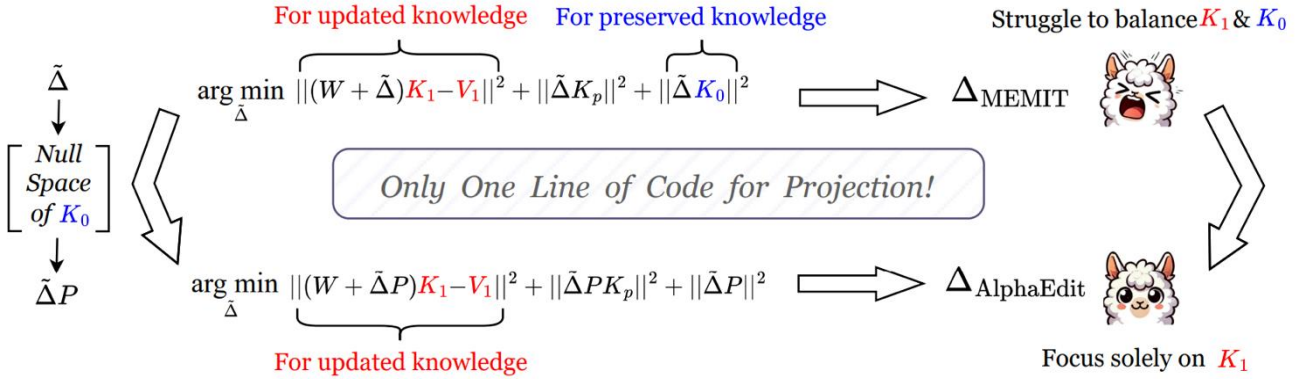
➢ MEMIT

- Mass-editing memory: thousands of edits in one pass via least-squares over multiple MLPs

$$W_1 \triangleq \underset{\hat{W}}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \left\| \hat{W} k_i - m_i \right\|^2 + \sum_{i=n+1}^{n+u} \left\| \hat{W} k_i - m_i \right\|^2 \right).$$

➢ AlphaEdit

- Projects ROME-style updates onto the null space of preserved knowledge to reduce catastrophic forgetting



Meng, K. et al. "Mass-Editing Memory in a Transformer." ICLR 2023
Fang, J., et al. "AlphaEdit: Null-Space Constrained Knowledge Editing for Language Models." ICLR 2025

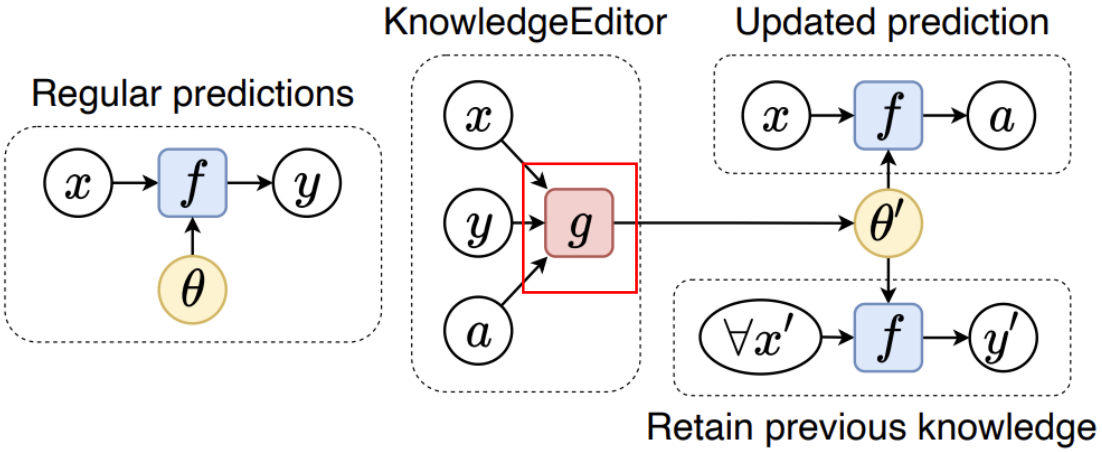
Part 5.1: Parameter-Level Editing

❑ Hypernetwork Editors

Treat editing as a learned function: train an auxiliary hypernetwork that maps (edit samples, gradient features…) to weight update

Unlike ROME, it makes no mechanistic assumption about where knowledge is stored; the editor learns to edit from data

➢ KnowledgeEditor



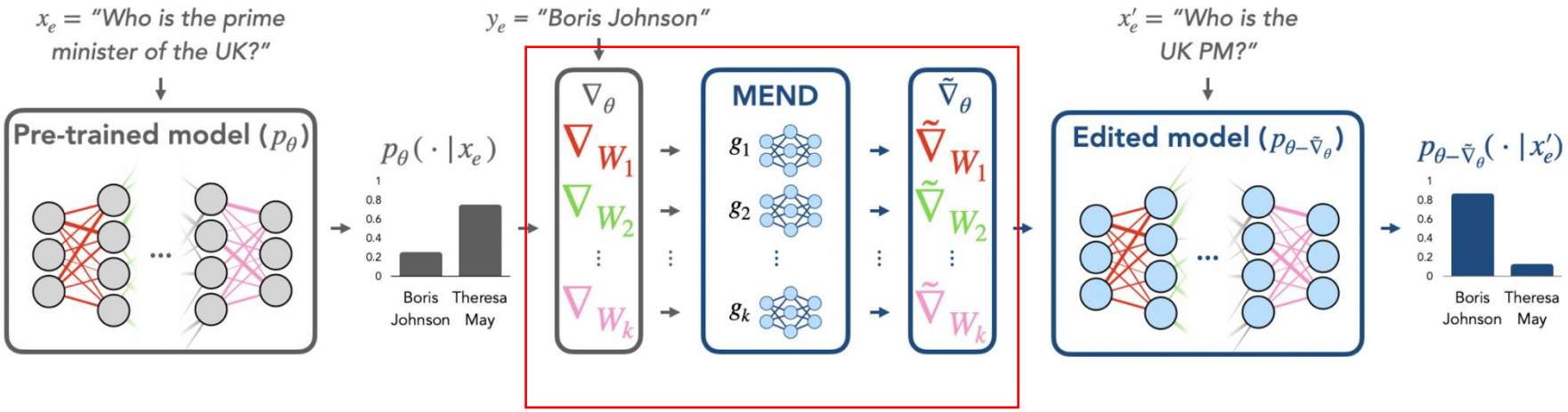
Train a hypernetwork g to predict edit deltas from gradient features

De Cao, N., et al. "Editing Factual Knowledge in Language Models." EMNLP 2021

Part 5.1: Parameter-Level Editing

❑ Hypernetwork Editors

➢ MEND



Train a collection of MLPS $\{g_i\}_{i=1}^k$ to trasnform raw gradients into low-rank edit updates

Mitchell, E., et al. "Fast Model Editing at Scale." ICLR 2022

Part 5.1: Parameter-Level Editing

❑ Mechanism-aware Multi-hop Editing

➢ Problem

- Parameter edits localized to one component do not propagate along reasoning chains
- When a single fact is edited, the model struggles to accurately update related facts in a multi-hop query

➢ Key Idea

A relation lives in a **circuit** (multiple attention heads + MLPs), not a single layer

Edit the whole circuit, or control how the edit propagates along it

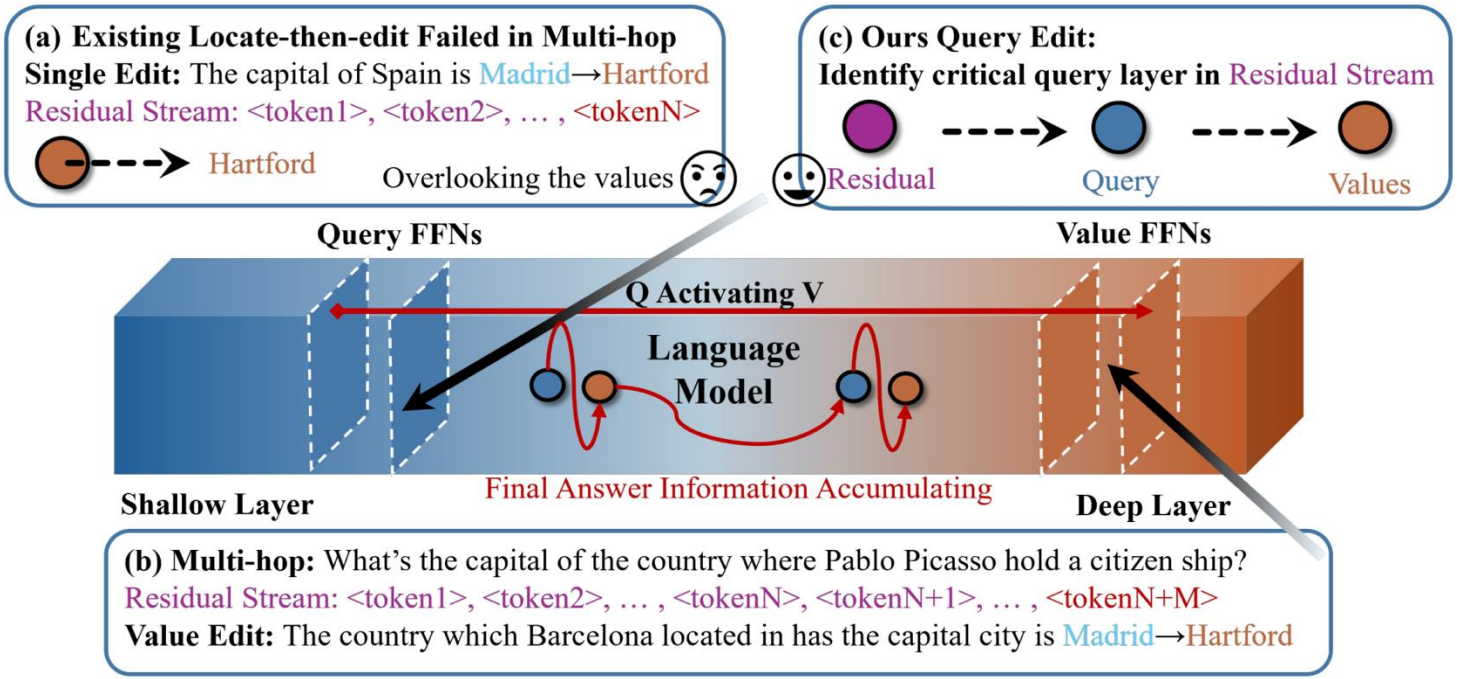
Cohen, R., et al. "Evaluating the Ripple Effects of Knowledge Editing in Language Models." TACL 2024
Zhong, Z., et al. "MQuAKE: Assessing Knowledge Editing in Language Models via Multi-Hop Questions." EMNLP 2023

Part 5.1: Parameter-Level Editing

❑ Mechanism-aware Multi-hop Editing

➢ ACE

- Controls the attribution path along which an edit propagates



Yang, J., et al. "ACE: Attribution-Controlled Knowledge Editing for Multi-hop Factual Recall," ICLR 2026

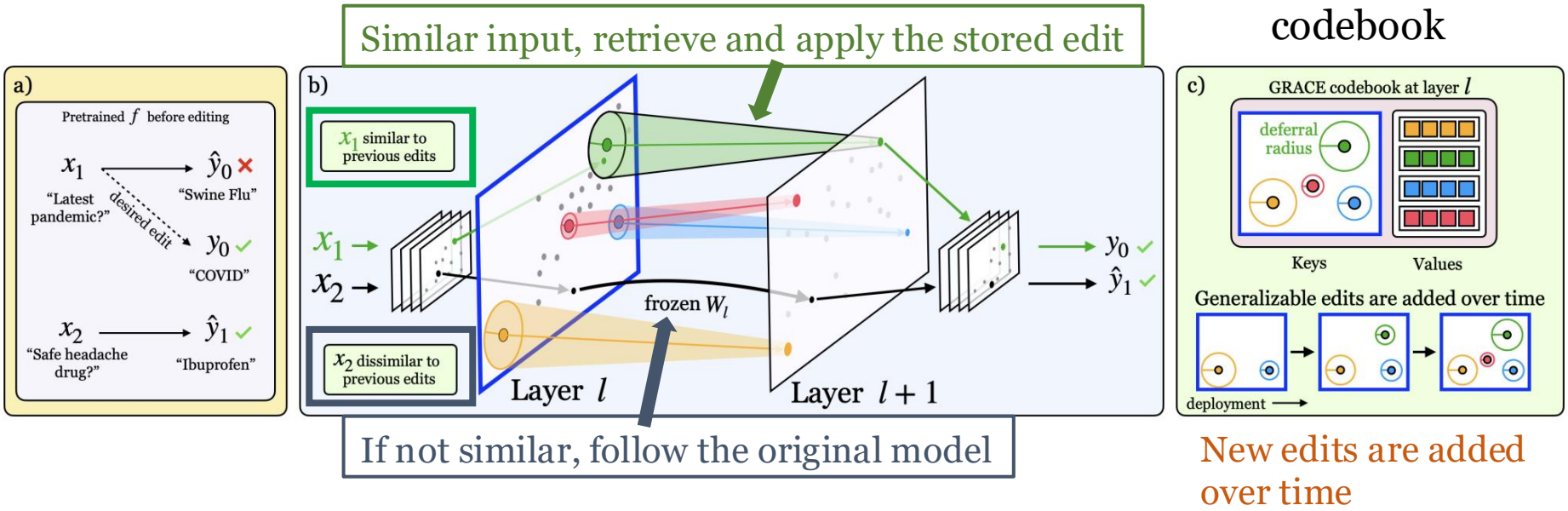
Part 5.1: Parameter-Level Editing

❑ Lifelong/Streaming Editing

- Stop overwriting base weights.
- Route edits through auxiliary storage (adapters, added neurons, or context) so the original model stays intact

➢ **GRACE**

Key idea: freeze the model, store edits in a codebook, and reuse them for similar inputs.



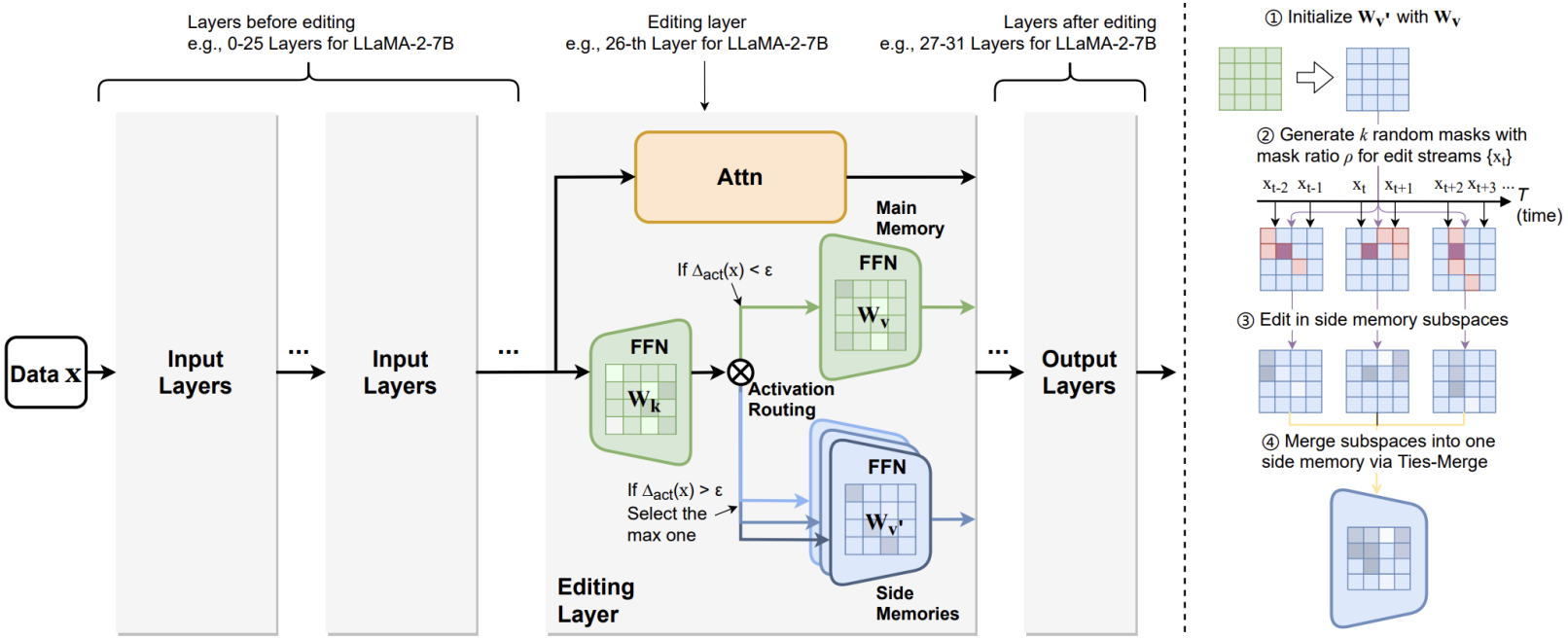
Hartvigsen, T., et al. "Aging with GRACE: Lifelong Model Editing with Discrete Key-Value Adaptors." NeurIPS 2023

Part 5.1: Parameter-Level Editing

❑ Lifelong/Streaming Editing

➢ WISE

- **Dual parametric memory scheme:** main memory (green) for pretrained knowledge + side memory (blue) for edited knowledge
- Train a router to decide which to use



(a) Workflow Overview with Knowledge Routing

(b) Knowledge Sharding and Merging

Wang, P., et al. "WISE: Rethinking the Knowledge Memory for Lifelong Model Editing of Large Language Models." NeurIPS 2024

Part 5.1: Parameter-Level Editing

❑ Benchmarks

➢ Single-hop Edits

- ZsRE

  Wikipedia QA pairs; single-fact edits evaluated for efficacy and paraphrase robustness

- CounterFact

  Counterfactual edits

➢ Multi-hop Edits

- MQuAKE

  Multi-hop QA edits

- RippleEdits

  Tests whether an edit consistently updates the web of facts logically implied by it

➢ Comprehensive Benchmark

- KnowEdit

  Unified benchmark bundling 6 datasets across insertion, modification, and erasure tasks

Levy, O., et al. "Zero-Shot Relation Extraction via Reading Comprehension," CoNLL 2017
Meng, K., et al. "Locating and Editing Factual Associations in GPT." NeurIPS 2022
Zhong, Z., et al. "MQuAKE: Assessing Knowledge Editing in Language Models via Multi-Hop Questions." EMNLP 2023
Cohen, R., et al. "Evaluating the Ripple Effects of Knowledge Editing in Language Models." TACL 2024
Zhang, N., et al. "A Comprehensive Study of Knowledge Editing for Large Language Models." arXiv:2401.01286

Part 5.1: Parameter-Level Editing

❑ Benchmarks

➢ Lifelong / Large-scale Edits

- WikiBigEdit

  Large-scale lifelong benchmark of 500K sequential real-world Wikidata edits

➢ Unstructured Knowledge

- UnKEBench

  The first unstructured editing benchmark: 1K long-form counterfactual passages going beyond (subject, relation, object) triples to free-form narratives

➢ Framework

- EasyEdit

  A unified open-source library implementing 10+ editors over all standard editing benchmarks

Thede, L., et al. "WikiBigEdit: Understanding the Limits of Lifelong Knowledge Editing in LLMs." ICML 2025
Deng, J., et al. "Everything is Editable: Extend Knowledge Editing to Unstructured Data in Large Language Models." ICLR 2025
Wang, P., et al. "EasyEdit: An Easy-to-use Knowledge Editing Framework for Large Language Models." ACL 2024

Part 5.2: Representation-Level Editing

❑ Core Premise

Linear Representation Hypothesis: many behaviors (truth, refusal, sentiment, sycophancy) are mediated by a **single direction** in residual-stream space → adding/projecting along it steers behavior

❑ The Steering Recipe

Training-free, reversible

$$h^{(l)} \leftarrow h^{(l)} + \alpha \cdot v$$

At chosen layer l, add scalar α times direction v to the residual stream.
$\alpha > 0$ amplifies the behavior, $\alpha < 0$ suppresses it, $\alpha = 0$ = original model.

Central Question: How do we find v?

Park, K., et al. "The Linear Representation Hypothesis and the Geometry of Large Language Models." ICML 2024

Part 5.2: Representation-Level Editing

❑ Three Routes to a Steering Direction

- **Contrastive Activations**
  - Direction = mean activation difference on prompt pairs
- **Supervised Probe**
  - Direction = classifier weights on labeled examples
- **Unsupervised**
  - Direction satisfying a consistency property

All three regimes produce a linear direction; they differ only in how it's found

Part 5.2: Representation-Level Editing

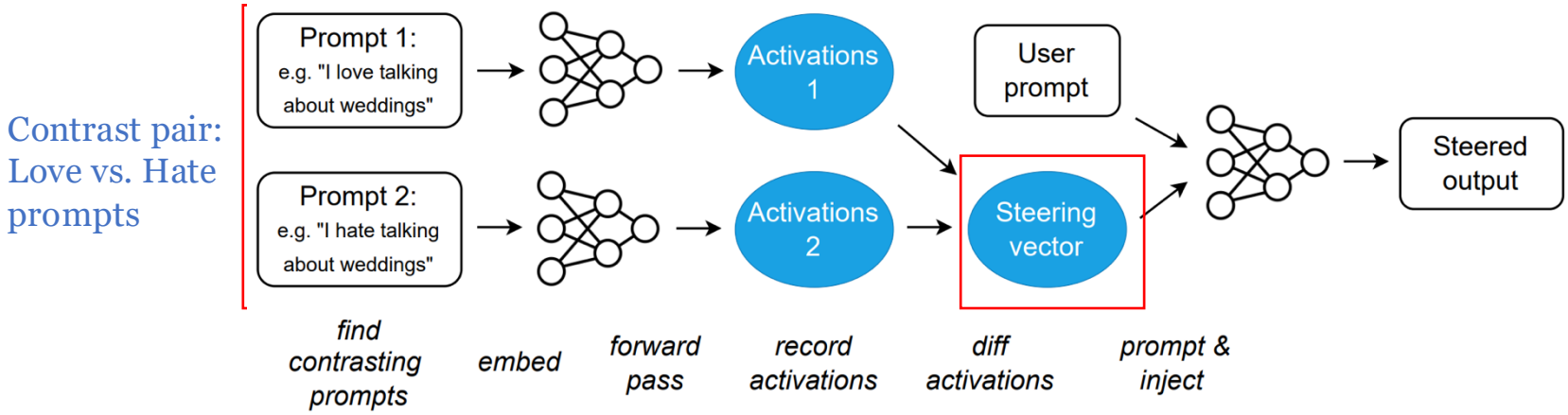
❑ Contrastive Activation Directions

$$v^{(l)} = \mathbb{E}_{x \in \mathcal{D}^+}\left[h^{(l)}(x)\right] - \mathbb{E}_{x \in \mathcal{D}^-}\left[h^{(l)}(x)\right]$$

Direction = **mean activation** on "positive" prompts minus **mean activation** on "negative" prompts.

➢ ActAdd

Discovered direction: sentiment



Turner, A. M., et al. "Steering Language Models With Activation Engineering." arXiv:2308.10248

Part 5.2: Representation-Level Editing

❑ Inference-Time Intervention

supervised variant

➢ Pipeline

- For each attention head, train a linear probe distinguishing truthful vs. false outputs on labeled QA data
- Probe weight vector = head's **truth direction** v_{truth}
- At inference, shift each head's output along v_{truth} :

$$h^{(l,h)} \leftarrow h^{(l,h)} + \alpha \cdot v_{truth}^{(l,h)}$$

➢ Result

- TruthfulQA accuracy ↑ substantially on LLaMA (e.g., +15 pts), with no fine-tuning and negligible impact on MMLU

Requires labeled data; recent studies find contrastive methods achieve comparable or better results without labels

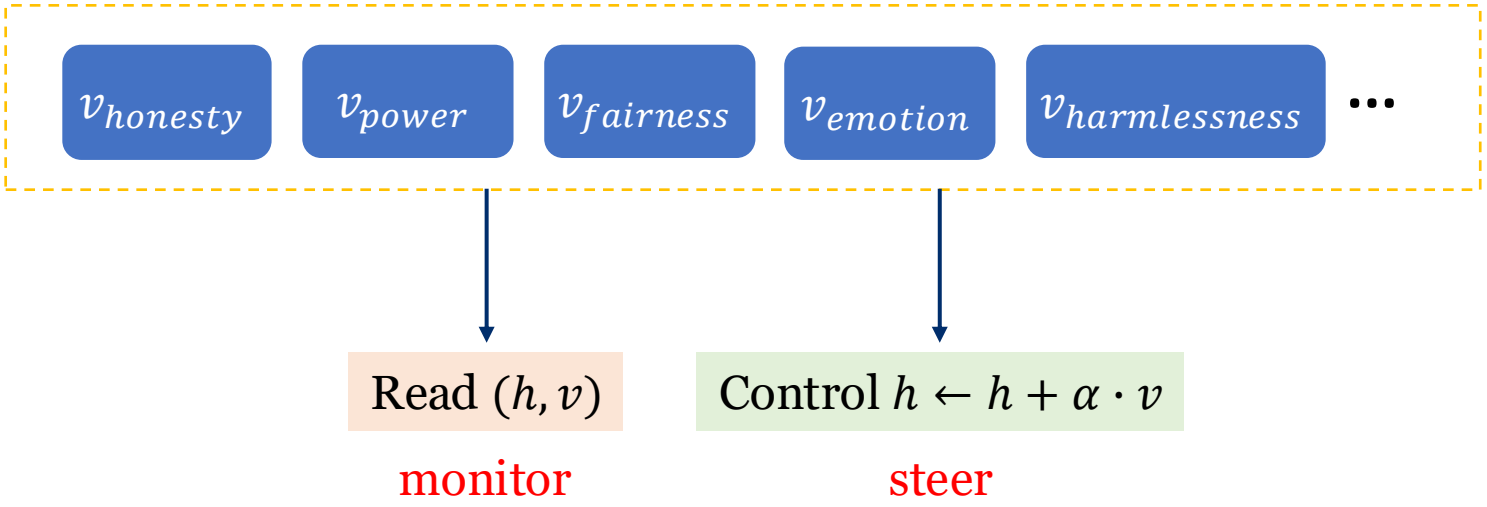
Li, K., et al. "Inference-Time Intervention: Eliciting Truthful Answers from a Language Model." NeurIPS 2023
Im, S., et al. "A Unified Understanding and Evaluation of Steering Methods." arXiv:2502.02716

Part 5.2: Representation-Level Editing

❑ Systematized Framework: RepE

➢ Formalization

- Treat **representations** (not neurons or circuits) as the primary unit of analysis → generalize the steering recipe across many high-level concepts



Zou, A., et al. "Representation Engineering: A Top-Down Approach to AI Transparency." arXiv:2310.01405

Part 5.2: Representation-Level Editing

❑ Practical Results

➢ Refusal is Mediated by a Single Direction

A contrastive-activations method applied to a safety-critical behavior

Two interventions, two outcomes

$h \leftarrow h - \alpha \cdot v_{refusal}$ Jailbreak (model complies with harmful prompts)

$h \leftarrow h + \alpha \cdot v_{refusal}$ Hardened Refusal

Why this matters?

- A safety-critical behavior reduces to a 1D linear intervention; the same direction recurs across model families
- **Defensive:** monitoring one direction may flag jailbreaks
- **Offensive:** open-weight safety training is one matrix multiplication from removal

Arditi, A., et al. "Refusal in Language Models Is Mediated by a Single Direction." NeurIPS 2024

Part 5.2: Representation-Level Editing

❑ Recent Advances

➢ Persona Vectors at frontier scale

- **Automated pipeline:** natural-language trait description → contrastive prompts → steering vector
- Identified vectors for many traits: evil, sycophancy, hallucination, optimism, humor, …

➢ Task-level generalization

- **Function Vectors:** a single vector triggers an ICL task without demonstrations; compositional across subtasks

➢ Trainable interventions

- **ReFT:** fine-tune **low-rank interventions on activations** instead of weights; far more parameter-efficient than LoRA

Chen, R., et al. "Persona Vectors: Monitoring and Controlling Character Traits in Language Models." arXiv:2507.21509
Todd, E., et al. "Function Vectors in Large Language Models." ICLR 2024
Wu, Z., et al. "ReFT: Representation Finetuning for Language Models." NeurIPS 2024

Part 5.3: Circuit/Feature-Level Editing

❑ Core Premise

- **Edit the *mechanism***

Target the mechanistic graph itself; more expensive but more localized and robust

- **Two editing granularities**
  - Atomic: SAE feature
  - Compositional: subgraphs

Part 5.3: Circuit/Feature-Level Editing

❑ SAE Feature Clamping

After SAE training, activations decompose as:

$$h \approx \sum_i z_i \cdot f_i + \text{error}$$

where each f_i is an interpretable feature direction.

➢ **The intervention:** fix the activation of a chosen feature

$$z_i \leftarrow c, \qquad \text{then reconstruct } \tilde{h} \approx \sum_j z_j \cdot f_j$$

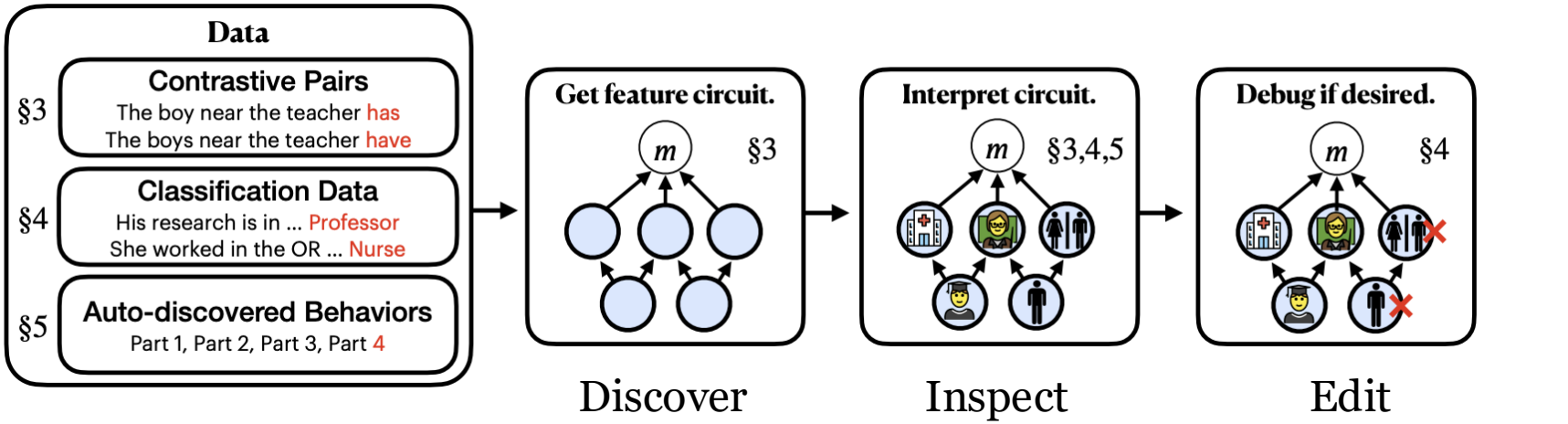
$c >$ typical activation **amplifies** feature i
$c = 0$ **ablates** feature i

Part 5.3: Circuit/Feature-Level Editing

❑ Sparse Feature Circuits

➢ SHIFT

Demonstration of the Discovery → Editing loop closing

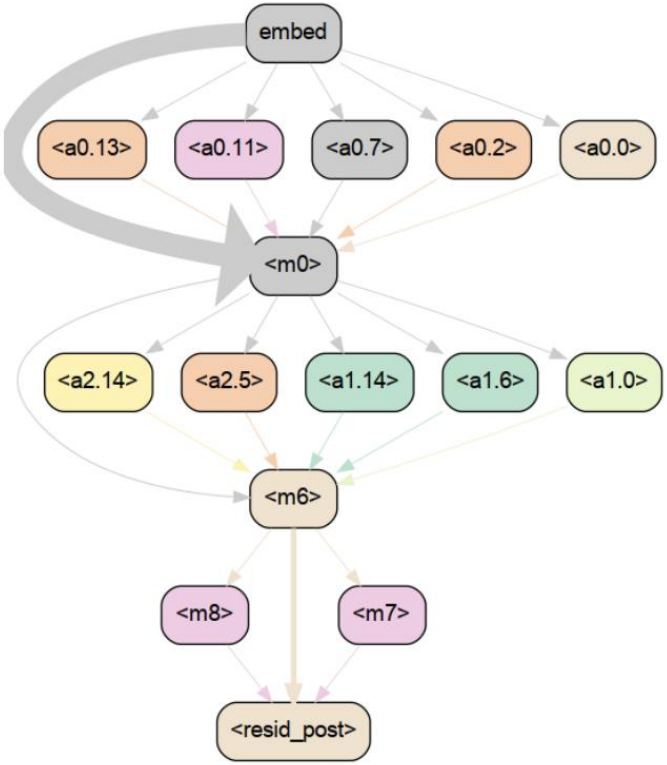


Marks, S., et al. "Sparse Feature Circuits: Discovering and Editing Interpretable Causal Graphs in Language Models." ICLR 2025

Part 5.3: Circuit/Feature-Level Editing

❑ Knowledge Circuits

- Circuit-level editing of factual knowledge; extends ROME-style edits from single MLPs to **full circuits**



Factual knowledge is not stored in one MLP; it's computed by a multi-component circuit: information heads, relation heads, and MLPs cooperating across layers.

Yao, Y., et al. "Knowledge Circuits in Pretrained Transformers." NeurIPS 2024

Part 5.3: Circuit/Feature-Level Editing

❑ Reasoning Circuit Reshaping

➢ REdit

- Generalizes mechanism-grounded editing from **factual knowledge** to **reasoning patterns**

Problem Formulation

- Different reasoning patterns (propositional logic, multi-hop, arithmetic) live in **overlapping circuits**. Editing one pattern interferes with others.

Pipeline

1. **Discover** the circuits of target and adjacent reasoning patterns
2. **Reshape —** actively reduce circuit overlap before editing
3. **Edit** with standard LoRA, achieving both Generality and Locality

Lei, Z., et al. "Reforming the Mechanism: Editing Reasoning Patterns in LLMs with Circuit Reshaping." ICLR 2026

Part 5.4: Mechanistic Unlearning

❑ Mechanistic Unlearning

Locate the components storing the target capability → **modify** them, leaving the rest untouched

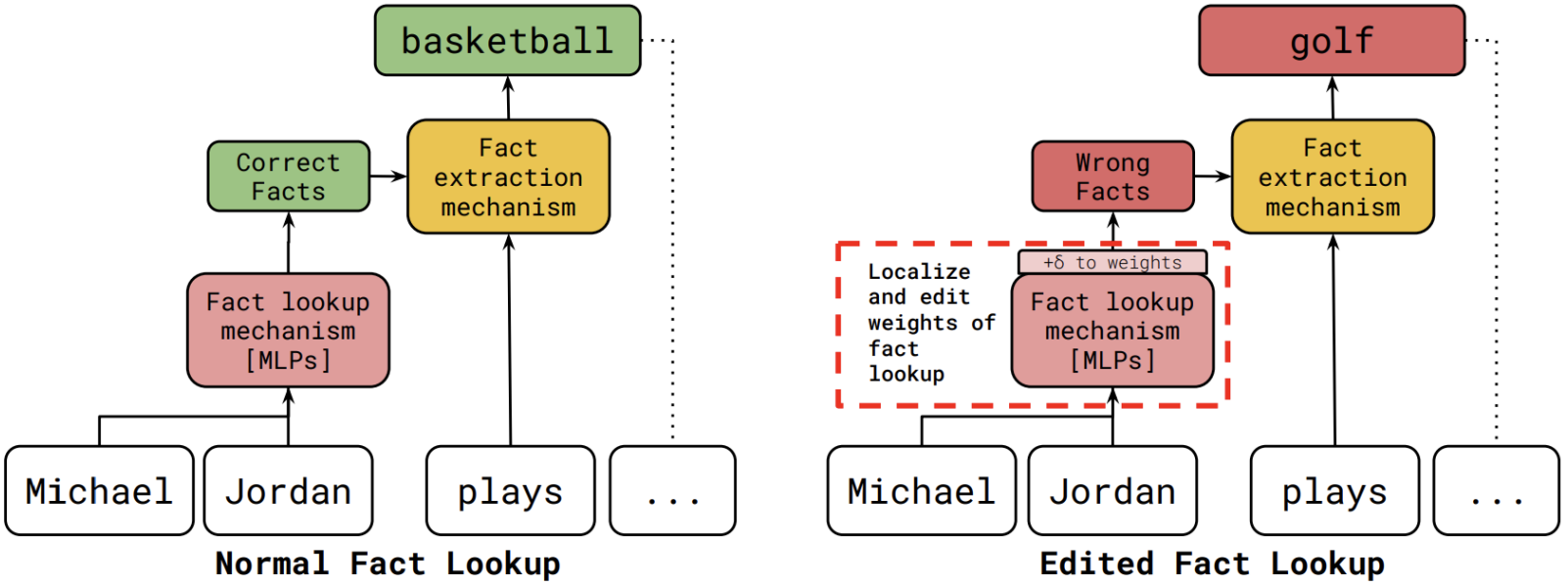
- Mechanistic localization
  - Localize fact's mechanism, then update those weights
- RMU; LAT
  - Misdirect activations of forget-set content
- Selective neuron pruning
  - Ablate discovered features / circuits encoding target capability

Guo, P., et al. "Mechanistic Unlearning: Robust Knowledge Unlearning and Editing via Mechanistic Localization." arXiv:2410.12949
Li, N., et al. "The WMDP Benchmark: Measuring and Reducing Malicious Use With Unlearning." ICML 2024
Sheshadri, A., et al. "Latent Adversarial Training Improves Robustness to Persistent Harmful Behaviors in LLMs." TMLR 2025
Pochinkov, N., & Schoots, N. "Dissecting Language Models: Machine Unlearning via Selective Pruning." arXiv:2403.01267

Part 5.4: Mechanistic Unlearning

❑ Mechanistic Unlearning via Localization

Apply ROME-style localization to identify components storing a target knowledge cluster and then selectively suppress them.



Guo, P., et al. "Mechanistic Unlearning: Robust Knowledge Unlearning and Editing via Mechanistic Localization." arXiv:2410.12949

Part 5.4: Mechanistic Unlearning

❑ Representation Misdirection for Unlearning

➢ Overview

- At a chosen layer l, push the activation of forget-set content toward a random vector while keeping retain-set activations unchanged

$$\mathcal{L}_{\text{forget}} = \mathbb{E}_{x_f \sim D_{\text{forget}}} \left[\frac{1}{L_f} \sum_{\text{token } t \in x_f} \| M_{\text{updated}}(t) - c \cdot \mathbf{u} \|_2^2 \right]$$

$$\mathcal{L}_{\text{retain}} = \mathbb{E}_{x_r \sim D_{\text{retain}}} \left[\frac{1}{L_r} \sum_{\text{token } t \in x_r} \| M_{\text{updated}}(t) - M_{\text{frozen}}(t) \|_2^2 \right]$$

Forget-set content gets mapped to noise; retain-set content is anchored to its original representation

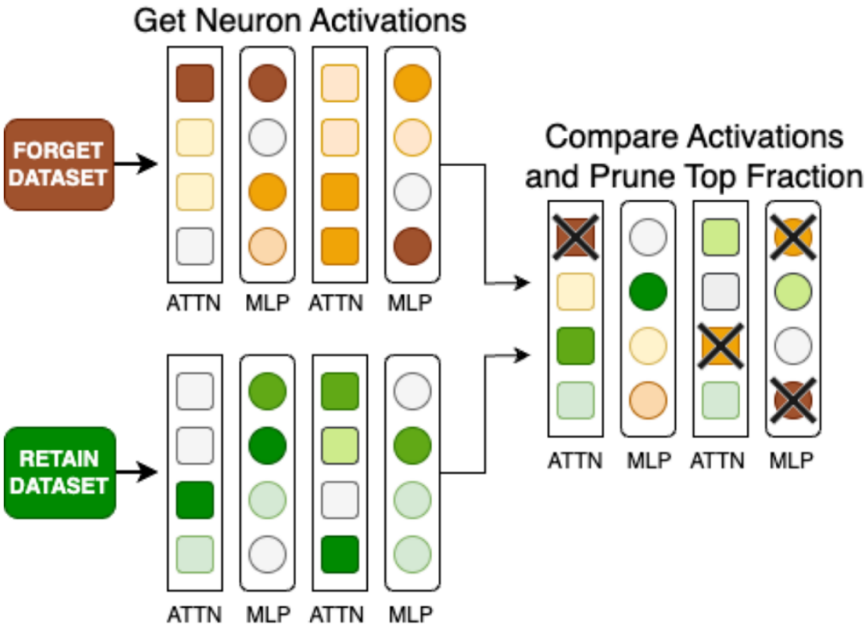
Li, N., et al. "The WMDP Benchmark: Measuring and Reducing Malicious Use With Unlearning." ICML 2024

Part 5.4: Mechanistic Unlearning

❑ Selective Neuron Pruning

➢ Overview

- Identify neurons selectively activated for the target capability → structurally ablate them



Instead of overwriting weights, **remove the neurons** that carry knowledge

Pochinkov, N., & Schoots, N. "Dissecting Language Models: Machine Unlearning via Selective Pruning." arXiv:2403.01267

Part 5.5: Bias and Hallucination Mitigation

❑ Locate the bias-carrying components

➢ Causal Mediation Analysis for Gender Bias

Gender bias effects are

- **Sparse** (concentrated in a small subset of neurons and attention heads)
- **Synergistic** (amplified/repressed across components)
- **Decomposable** into direct + indirect effects through specific mediators.

Vig, J., et al. "Investigating Gender Bias in Language Models Using Causal Mediation Analysis." NeurIPS 2020

Part 5.5: Bias and Hallucination Mitigation

❑ Mitigate bias

- **Targeted Ablation**

  Ablate attention heads identified by casual mediation; stereotype propagation ↓ with general capability preserved.

- **Linear Concept Erasure**

  Closed-form orthogonal projection that provably erases a target concept (e.g. gender) from activations.

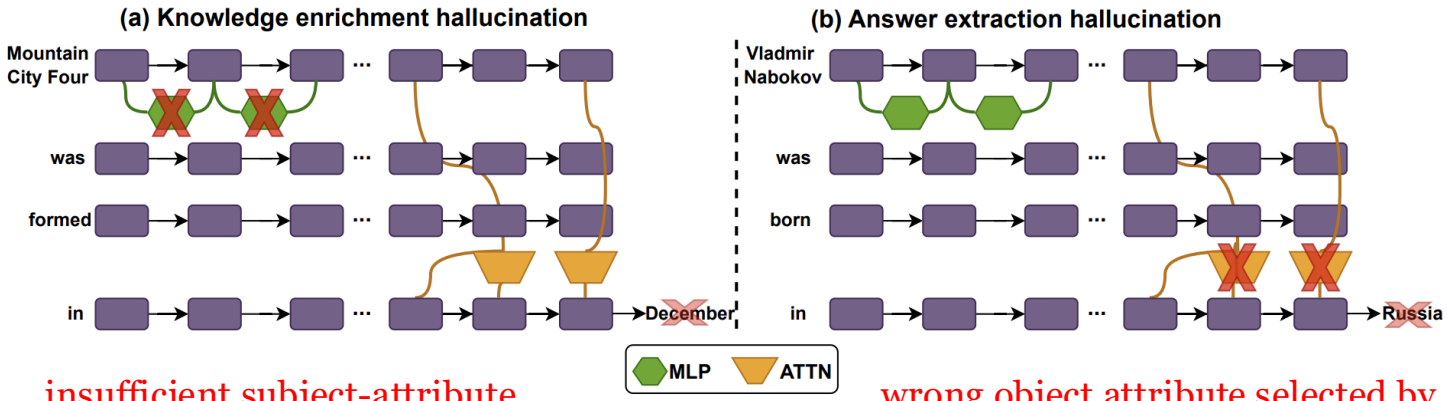
  Complements Targeted Ablation: instead of modifying the component, **removes the concept's linear signature from the residual stream**

Chintam, A., et al. "Identifying and Adapting Transformer-Components Responsible for Gender Bias in an English Language Model." BlackboxNLP 2023
Belrose, N., et al. "LEACE: Perfect linear concept erasure in closed form." NeurIPS 2023

Part 5.5: Bias and Hallucination Mitigation

❑ Mechanistic Understanding of Non-Factual Hallucination

➢ Two mechanistic causes of hallucination



insufficient subject-attribute knowledge in **lower-layer MLPs**

wrong object attribute selected by **upper-layer attention heads**

➢ Mitigation

- Patch the failing component with corrected activations

Yu, L., et al. "Mechanistic Understanding of Non-Factual Hallucination." EMNLP 2024

Part 5.5: Bias and Hallucination Mitigation

❑ Frontier

- **Mechanistic generalization of bias**
  - Political ideology is encoded linearly across models.
- **Persona vector**
  - Automated extraction of trait directions including bias-relevant traits at production scale.
- **Reasoning hallucinations**
  - Long CoT amplifies misalignment; new failure mode requiring mechanism-aware mitigation.
- **Multimodal hallucinations**
  - VLM hallucinations follow similar mechanistic patterns with cross-modal causes.

Kim, J., et al. "Linear Representations of Political Perspective Emerge in Large Language Models." ICLR 2025
Chen, R., et al. "Persona Vectors: Monitoring and Controlling Character Traits in Language Models." arXiv:2507.21509
Yan, H., et al. "When Thinking Backfires: Mechanistic Insights Into Reasoning-Induced Misalignment." ICLR 2026
Basu, S., et al. "Understanding Information Storage and Transfer in Multi-modal Large Language Models." NeurIPS 2024

Part 6: Summary, Challenges, Future Directions

Where MI stands in 2026

Presenter: Chen Chen

Part 6.1: Summary

❑ Closing the Discovery-Validation-Editing Loop

- **Discovery (Part 3)**
  - Causal Mediation
  - Attribution
  - Sparse Features
  - Optimization

What internal mechanism in the model causes this behavior?

- **Validation (Part 4)**
  - Faithfulness / Completeness / Minimality
  - Causal Scrubbing
  - Causal Abstraction
  - Certified Validation

Is the discovered mechanism validated?

- **Editing (Part 5)**
  - Parameter-level Editing
  - Representation-level Editing
  - Circuit- / Feature-level Editing
  - Mechanistic Unlearning
  - Bias and Hallucination Mitigation

Can we leverage the mechanism to control the behavior?

Part 6.2: Challenges

❑ Challenge 1: Scalability

➢ Scaling MI to Frontier Models

Large frontier models remain largely uninterpreted

- Activation patching becomes prohibitive
- SAE training scales poorly
- Validation is expensive

➢ Recent Progress

- Attribution graphs
- Crosscoders
- Remote interventions (NNsight/NDIF)

Open questions:
Can attribution graphs become seed-stable?
Can SAEs scale linearly with model width?
Can validation be approximated tractably at frontier scale?

Part 6.2: Challenges

❑ Challenge 2: Evaluation Benchmarks

➢ The Problem

Discovery, validation, and editing each need benchmarks supplying ground truth; but the current benchmark suite is fragmented, partial, and rarely tests the underlying mechanistic claim.

➢ Structural Gaps

- No known mechanism-level ground truth at LLM scale
- No adversarial benchmark
- Frontier-scale gap

Part 6.2: Challenges

❑ Challenge 2: Evaluation Benchmarks

➢ Open Questions

- Can synthetic ground-truth circuits (Transformer Programs, FIND) be lifted to closer-to-realistic LLM tasks?
- Should benchmarks include adversarial circuits constructed to expose interpretability illusions?
- What is a sufficient benchmark suite for "mechanism passes validation"?

Part 6.2: Challenges

❑ Challenge 3: Theoretical Foundations

➢ The Problem

MI is predominantly empirical. The theoretical questions of why methods work lag the empirical findings substantially.

➢ Specific Gaps

- When does the Linear Representation Hypothesis hold?
- When do SAEs identify the model's true features vs. distributional artifacts?
- Is mechanistic universality provable, or only observed?
- How does mechanism relate to generalization, scaling laws, grokking?

Part 6.2: Challenges

❑ Challenge 3: Theoretical Foundations

➢ When linearity fails empirically

- Not all features are 1D

  Some require multi-dimensional, non-linear encoding

- SAE dark matter

  Residual non-linear structure persists after SAE decomposition

- Polysemy at scale

  SAE features may activate ~90% of the time not on their named concept

Engels, J., et al. "Not All Language Model Features Are One-Dimensionally Linear." ICLR 2025
Engels, J., et al. "Decomposing The Dark Matter of Sparse Autoencoders." TMLR 2025
Templeton, A., et al. "Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet." arXiv:2605.29358

Part 6.2: Challenges

❑ Challenge 4: Engineering Rigor & Audit Standards

- **Current MI lacks engineering rigor for high-stakes use**
  - Tools fail to catch trojans inserted by adversaries
  - Mechanistic descriptions rarely transfer to audit-quality use

- **Reproducibility concerns**
  - Hyperparameter sensitivity of faithfulness metrics
  - Patching protocol pitfalls

- **What's needed**
  - Audit-quality MI

Casper, S., et al. "Toward Transparent AI: A Survey on Interpreting the Inner Structures of Deep Neural Networks." SaTML 2023
Casper, S., et al. "Black-Box Access is Insufficient for Rigorous AI Audits." FAccT 2024

Part 6.3: Future Directions

❑ From Post-Hoc to Training-Time MI

➢ The Shift

- Current MI is post-hoc: train a model, then analyze.
- A growing research thread asks: can we integrate mechanism into the training process itself?

➢ Three Active Research Threads

Train models to be mechanistically tractable (reshape the model)

Exploit training-stage information (observe the training process)

Use mechanism insights to shape training objectives (reshape the loss)

Geiger, A., et al. "Inducing Causal Structure for Interpretable Neural Networks." ICML 2022
Friedman, D., et al. "Learning Transformer Programs." NeurIPS 2023
Lindsey, J., et al. "Sparse Crosscoders for Cross-Layer Features and Model Diffing." Transformer Circuits Thread 2024
Chen, R., et al. "Persona Vectors: Monitoring and Controlling Character Traits in Language Models." arXiv:2507.21509
Lee, A., et al. "A Mechanistic Understanding of Alignment Algorithms: A Case Study on DPO and Toxicity." ICML 2024

Part 6.3: Future Directions

❑ From Post-Hoc to Training-Time MI

➢ Open Questions

- What is the loss-function-level price of mechanistic tractability?
- Can SAE-aware training produce models whose features are guaranteed monosemantic?
- Can MI insights inform RLHF / DPO objectives directly?

Part 6.3: Future Directions

❑ Mechanistic Safety & AI Governance

➢ Three deployment-critical problems

- **Certification at scale** — the technical problem

  Certified circuits and formal MI provide formal guarantees, but are computationally expensive

  > Can certification be made tractable for worst-case safety-relevant circuits in frontier models?

- **Audit standards** — the engineering problem

  > What is a minimal audit-quality MI pipeline, and what should regulators look for?

- **Adversarial robustness of one-dimensional mechanisms** – The theoretical problem

  > Is 1D linearity a training artifact or a fundamental optimization attractor? What does it imply for adversarial defenses?

Anani, A., et al. "Certified Circuits: Stability Guarantees for Mechanistic Circuits." arXiv:2602.22968
Palumbo, N., et al. "Validating Mechanistic Interpretations: An Axiomatic Approach." ICML 2025
Hadad, I., et al. "Formal Mechanistic Interpretability: Automated Circuit Discovery with Provable Guarantees." ICLR 2026

Part 6.3: Future Directions

❑ Multimodal & Agentic Mechanisms

➢ Status

Multimodal Mechanisms

- Mature methods (causal tracing, feature decomposition) have been adapted to multimodal models, but signature results (universal motifs, linear concept directions at scale) remain unconfirmed in VLMs.

Agentic Mechanisms

- Essentially unstudied
- No published mechanistic decomposition of tool-use, planning, or multi-agent dynamics

Part 6.3: Future Directions

❑ Multimodal & Agentic Mechanisms

➢ Open Questions

- Do circuit motifs transfer cross-modality?
- Can mechanism-grounded editing work on VLM-specific failures?
- Do agentic behaviors decompose into discoverable circuits, or are they too distributed to localize?
- Can mechanism-level safety scale to agents?

Part 6.3: Future Directions

❑ Mechanisms as Scientific Objects

➢ Open Questions

- Is universality provable, or only observed?

  Different LLMs converge on similar circuit motifs (induction, retrieval, refusal direction) across architectures and training; remains an empirical observation.

- Can MI explain why scaling laws have the shape they do?
- What can neuroscience learn from MI, and vice versa?
- Is there a "physics of computation" lurking in these models?

  Phase transitions (grokking, ICL emergence, transient circuits), universality, and scaling laws all suggest law-like structure.

Chughtai, B., et al. "A Toy Model of Universality: Reverse-Engineering How Networks Learn Group Operations." ICML 2023
Nanda, N., et al. "Progress Measures for Grokking via Mechanistic Interpretability." ICLR 2023
Olsson, C., et al. "In-context Learning and Induction Heads." arXiv:2209.11895
Singh, A., et al. "The Transient Nature of Emergent In-Context Learning in Transformers." NeurIPS 2023

Thank you!

Q & A