Transformer-Specific Interpretability Part 3: Mechanistic Interpretability

Hosein Mohebbi, Jaap Jumelet, **Michael Hanna**,
Afra Alishahi, Willem Zuidema

What is mechanistic interpretability?

You're not the only one asking!

Sasha Rush
@srush_nlp

I recently asked pre-PhD researchers what area they were most excited about, and overwhelmingly the answer was "mechanistic interpretability". Not sure how that happened, but I am interested how it came about.

Jacob Andreas @jacobandreas · Jan 23
I still don't totally understand the difference between "mechanistic" and "non-mechanistic" interpretability but it seems to be mainly a distinction of the authors' social network?

Andrew Gordon Wilson @andrewgwils · Jan 24
Did they seem to know much about it and the foundations? I've also noticed a major increase in interest in this area, and alignment, but I suspect unfortunately for many it's just trendy buzzwords.

Mark Riedl @mark_riedl · Jan 23
Mechanistic explainability doesn't require human-participant studies for evaluation. Pesky humans always being noisy and requiring IRB protocols and requiring months and months of time.

Mechanistic XAI as a term exists to differentiate from human-centered

Sarah Wiegreffe @sarahwiegreffe · Jan 24
FWIW, I gave a talk at ACL in July on this topic. The framework in the talk doesn't capture everything, but I think it gives some credence as to why the terminology might be useful.

"Two Views of LM Interpretability" (starting at 7:46):

What is mechanistic interpretability?

Mechanistic interpretability: subfield of interpretability that aims to reverse-engineer neural network behavior, mapping low-level mechanisms to higher-level human-interpretable algorithms.

It frequently uses causal interpretability techniques, and studies the sub-layer level (e.g. attention heads, MLPs, or neurons).

This contrasts with work that:

operates at the input-output level (behavioral interpretability)

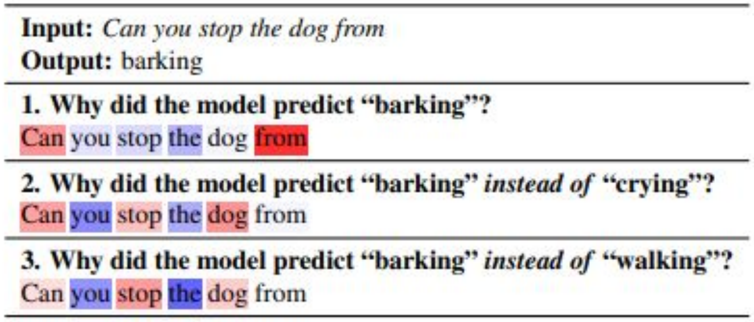
What is mechanistic interpretability?

Mechanistic interpretability: subfield of interpretability that aims to reverse-engineer neural network behavior, mapping low-level mechanisms to higher-level human-interpretable algorithms.

It frequently uses causal interpretability techniques, and studies the sub-layer level (e.g. attention heads, MLPs, or neurons).

This contrasts with work that:

finds important input tokens
(input attribution)



Yin and Neubig, 2022

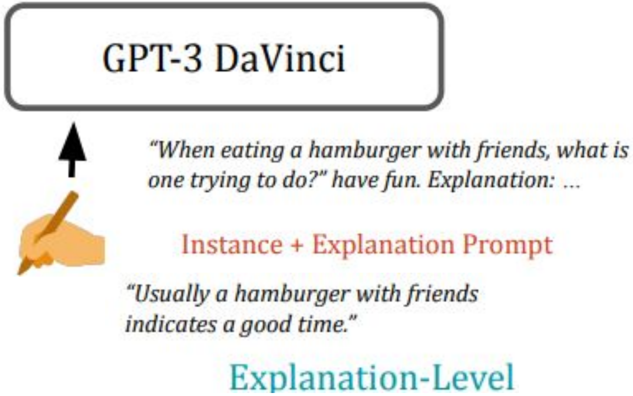
What is mechanistic interpretability?

Mechanistic interpretability: subfield of interpretability that aims to reverse-engineer neural network behavior, mapping low-level mechanisms to higher-level human-interpretable algorithms.

It frequently uses causal interpretability techniques, and studies the sub-layer level (e.g. attention heads, MLPs, or neurons).

This contrasts with work that:

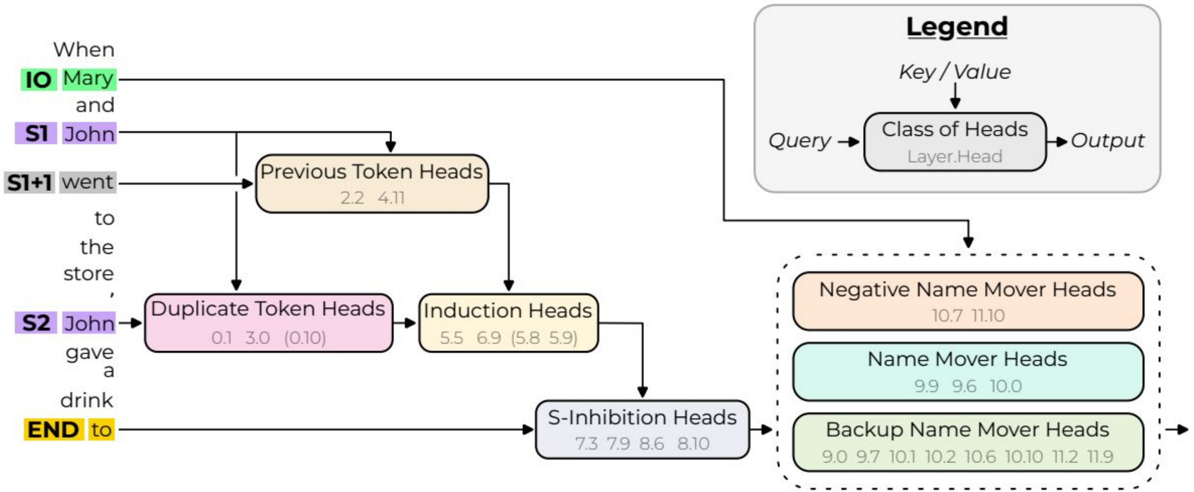
is human/user-centered (HCXAI)



Wiegreffe et al., 2022

What is mechanistic interpretability?

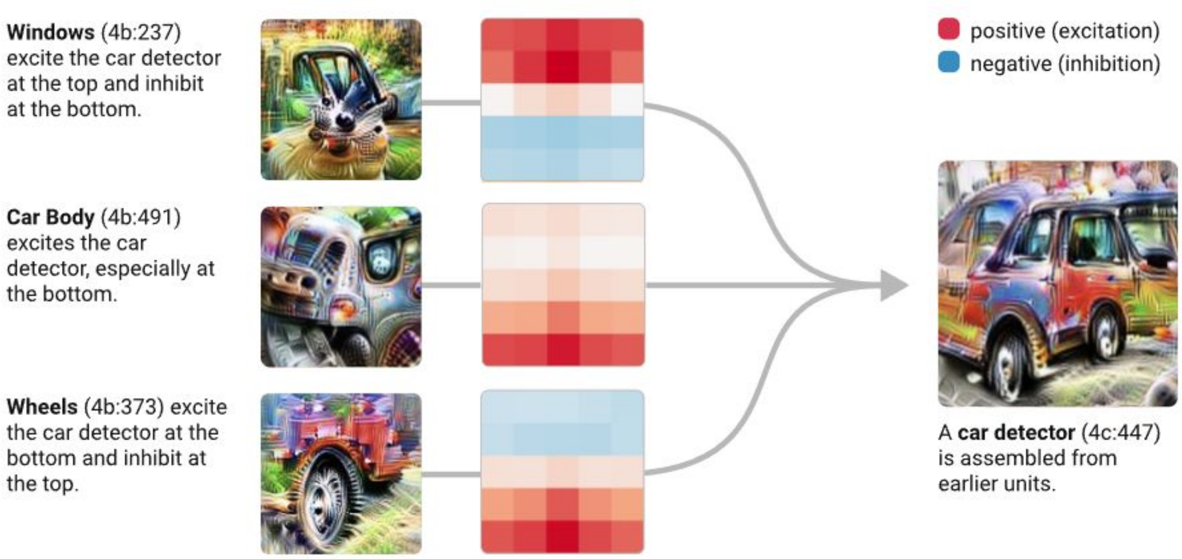
There are still many ways to reverse-engineer model behavior! But today, I'll focus on **circuits**: one particular framework for characterizing model behaviors.



Wang et al., 2023

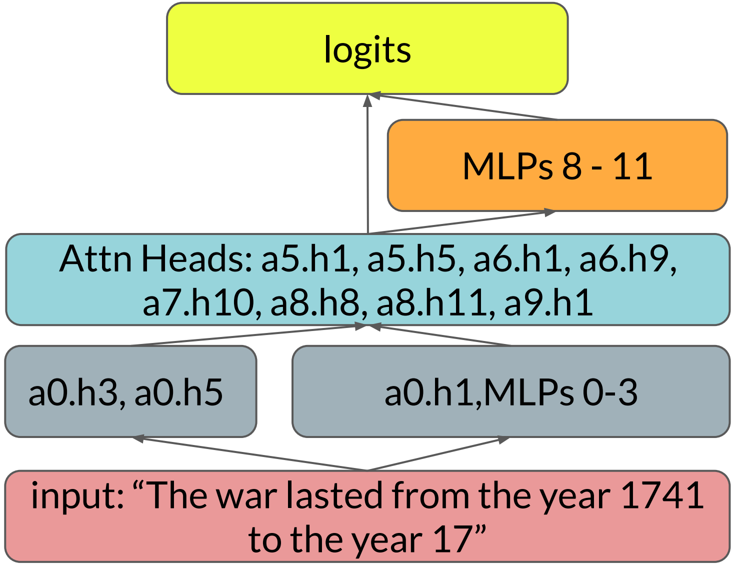
What are circuits?

Olah et al. (2020) defined circuits as “sub-graphs of the network, consisting a set of tightly linked features and the weights between them”



What are *transformer* circuits?

Transformer circuits localize and characterize transformer LM behavior in a (small) set of components of the model.



Upweight years > 41

Identify and upweight 41

Subject enrichment

Hanna et al., 2023

Circuits

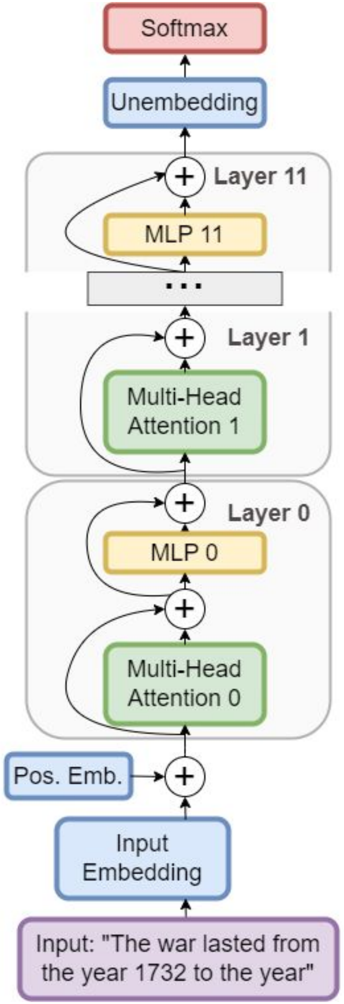
Circuit: minimal computational subgraph of a model that is faithful to model performance on a given task

- **Minimal computational subgraph**: minimal set of model nodes and edges
- **Task**: collection of **inputs** and **expected outputs**, measured by some **loss**.
- **Faithful**: loss remains the same when all non-circuit edges are *ablated*

But what does that mean?

What computational subgraph? The transformer LM architecture

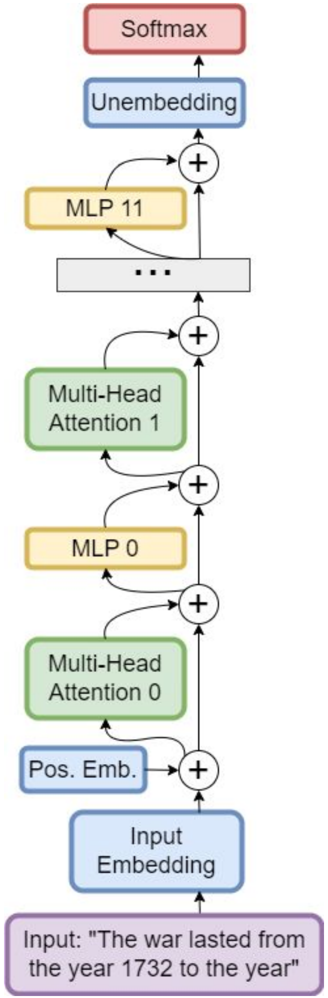
This is a traditional diagram of an autoregressive decoder-only LM.



The Residual Stream View

Centering the residuals reveals:

- Every component reads from and writes to the residual stream!
- Every component's input is the sum of the outputs of the components that came before

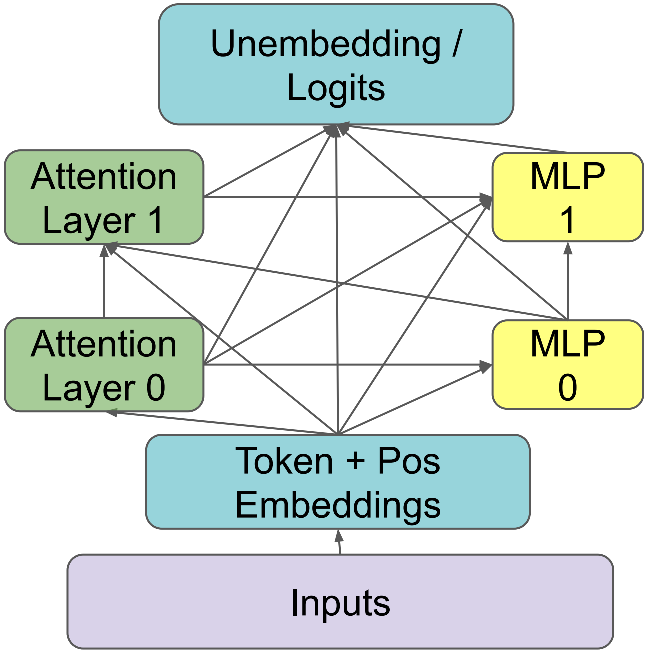


Elhage et al. (2021)

Computational Graph

For our circuit, we want the **minimal subgraph** that is faithful to model behavior. This view lets us specify the specific node-node interactions that count.

Note: we could have chosen other levels of granularity for this graph!



Task: Greater-Than

A **task** consists of:

Inputs: "The war lasted from 1741 to 17"
Expected outputs: a 2-digit number greater than 41
Metric: $\sum_{y>41} p(y) - \sum_{y<=41} p(y)$

Tasks should be solvable by your model, and evaluable in one forward pass.

Average Metric Value: 0.817

For circuit-finding, we also need corrupted inputs.

Corrupted inputs: "The war lasted from 1701 to 17"

Task: Subject-Verb Agreement

A **task** consists of:

Input: “The keys on the cabinet”
Expected output: a verb that agrees with the subject (“keys”)
Metric: $\sum_{y,\text{agree}(y, \text{“keys”})} p(y) - \sum_{y,\text{disagree}(y, \text{“keys”})} p(y)$

Tasks should be solvable by your model, and evaluable in one forward pass.

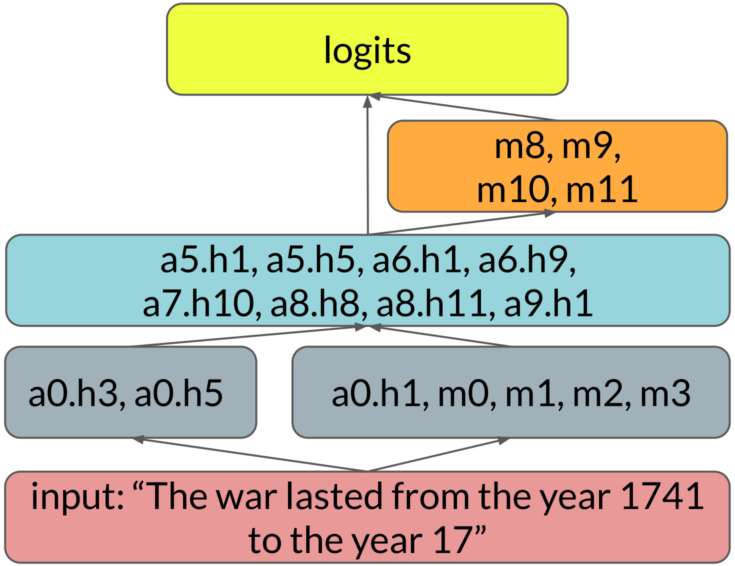
Average Metric Value: 0.351

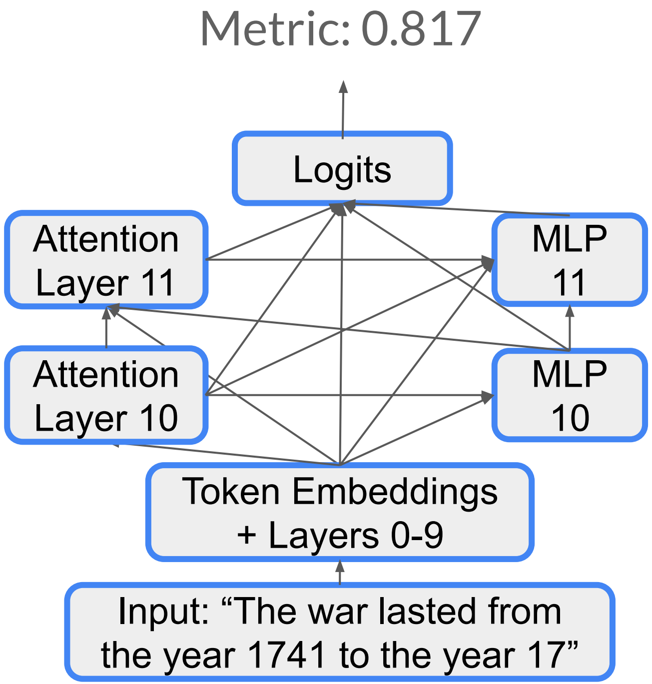
For circuit-finding, we also need corrupted inputs.

Corrupted Input: “The key on the cabinet”

Faithfulness

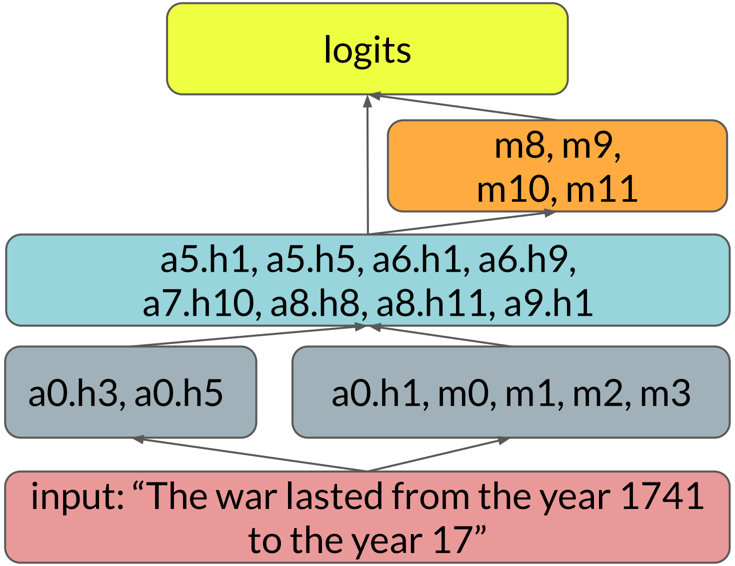
If a circuit is faithful to model behavior, we can ablate all nodes outside the circuit, with little to no behavior change!

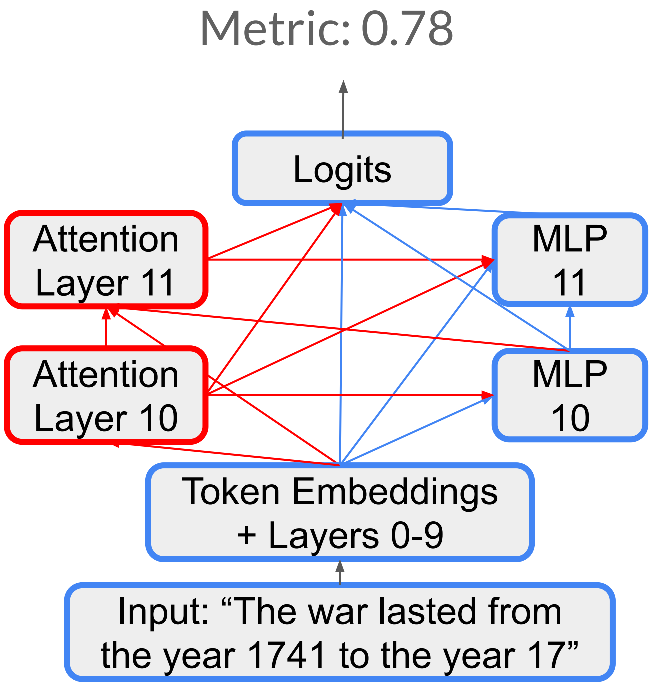




Faithfulness

If a circuit is faithful to model behavior, we can ablate all nodes outside the circuit, with little to no behavior change!





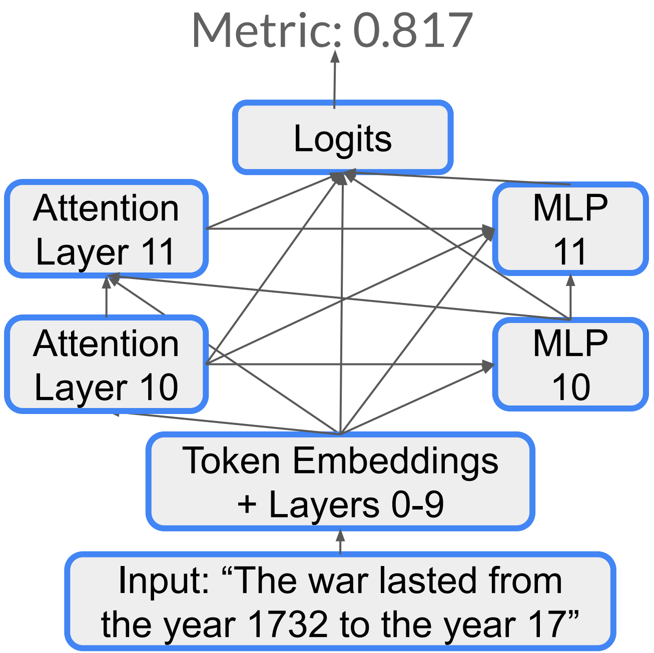
Finding Circuit Structure

Finding important nodes

We want to find nodes / edges that are important for a task.

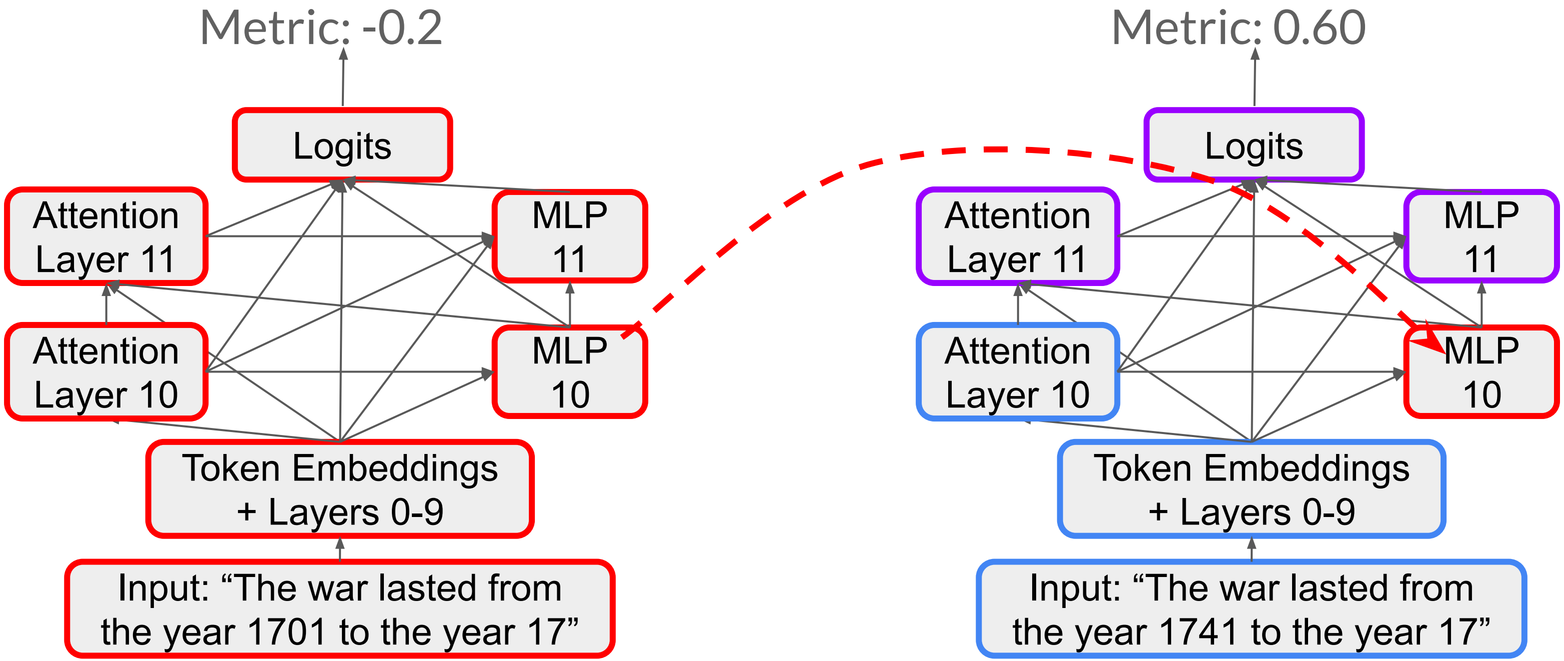
Core Idea: Important nodes / edges can't be ablated without hurting model performance.

But how do we ablate? Don't use zero ablations!



Activation Patching

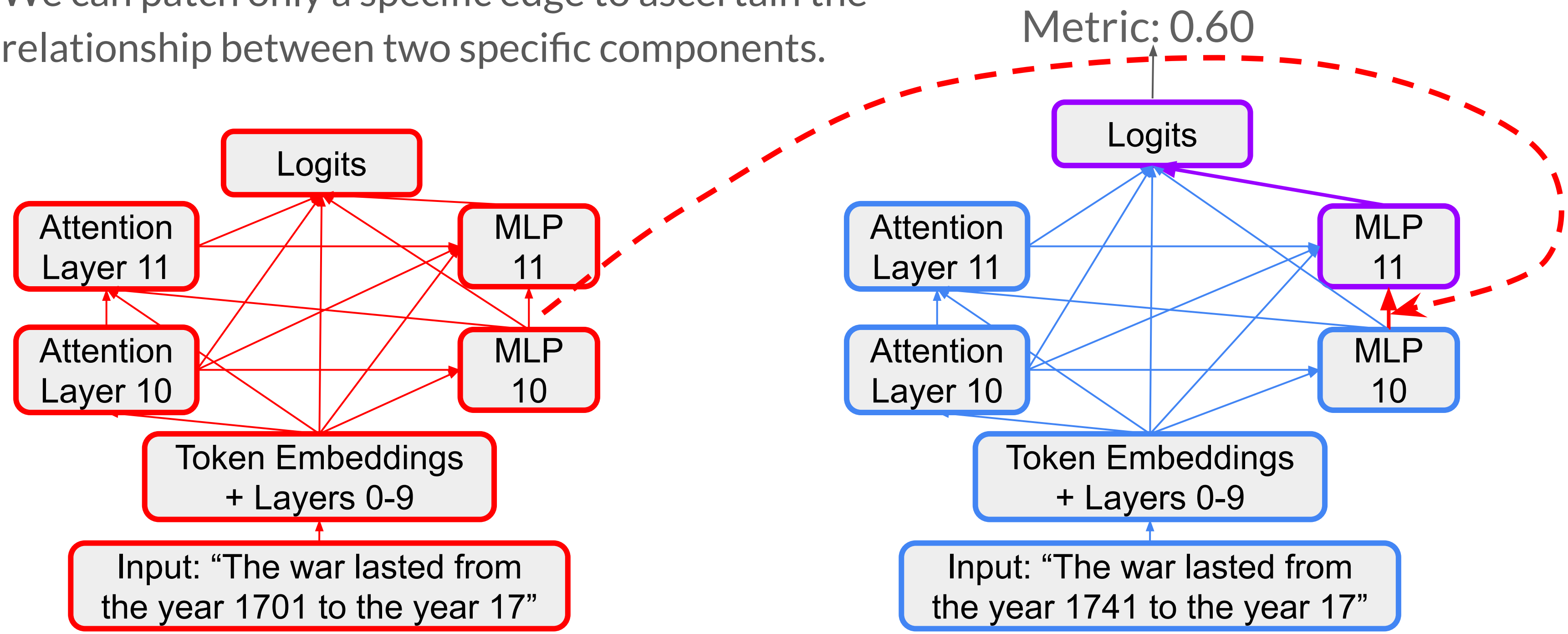
Replace a component's activation on one example, with an activation from another!



Activation patching (Vig et al., 2020; Geiger et al., 2020) predates LM circuits work

Edge Patching

We can patch only a specific edge to ascertain the relationship between two specific components.



Wang et al. (2022), Goldowsky-Dill et al. (2023)

How to perform patching?

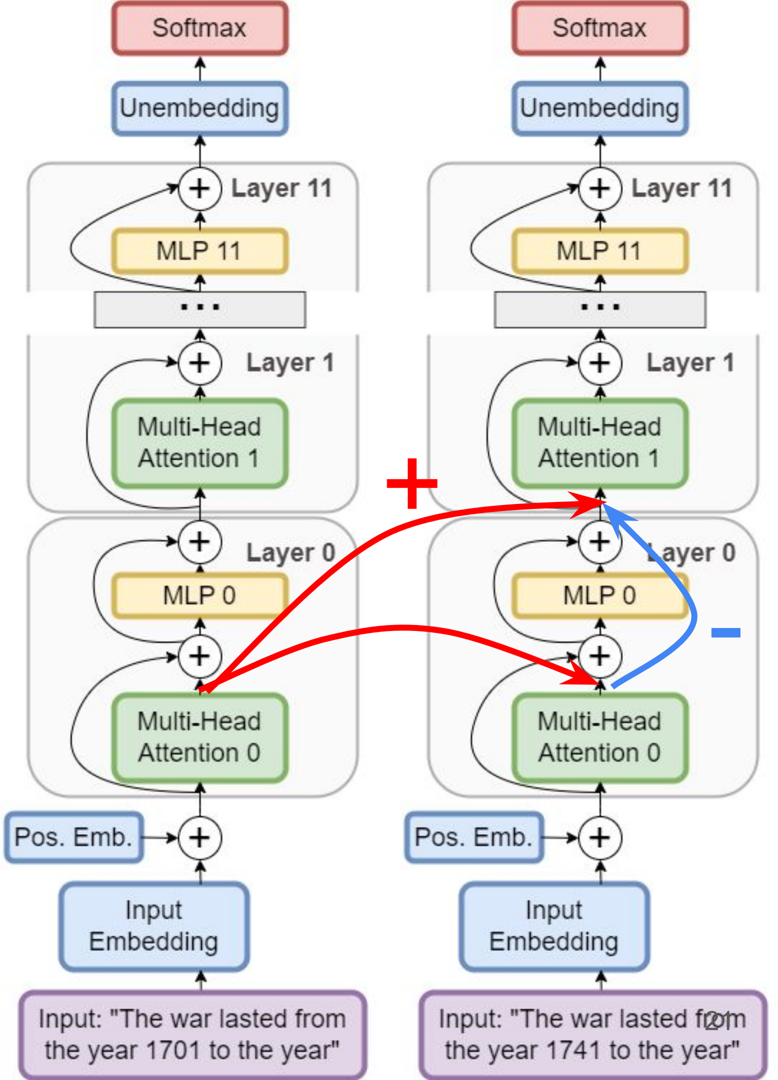
Node-level patching:

Replace the output of the node (e.g. Attn 0) with its output on another input!

Edge-level patching:

Exploit the linearity of the residual stream! Say we're patching the edge Attn0->Attn1.

- Take the input to Attn1
- Subtract the output of Attn0 on normal input
- Add in the output of Attn0 on corrupted input

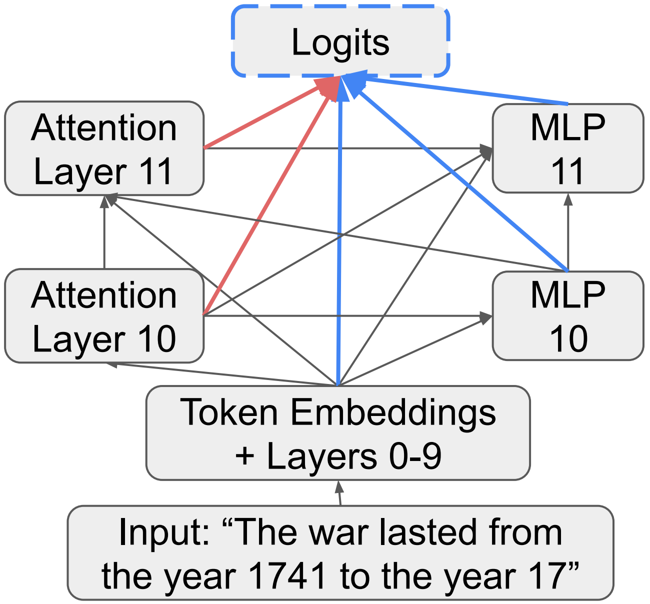


Circuit Finding: Activation Patching

How can we use patching to find an entire circuit?

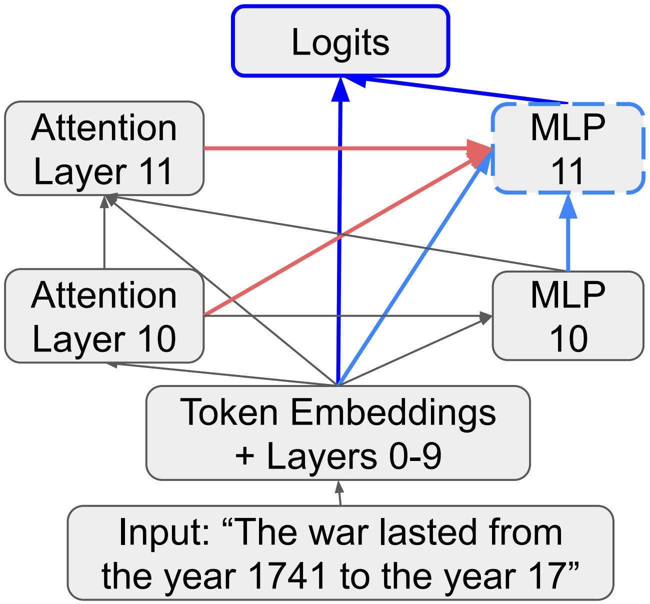
One approach: iteratively patch to find important nodes / edges.

First, find the nodes connected directly to the logits…



Circuit Finding: Activation Patching

Then find the nodes directly connected to those nodes, and then…

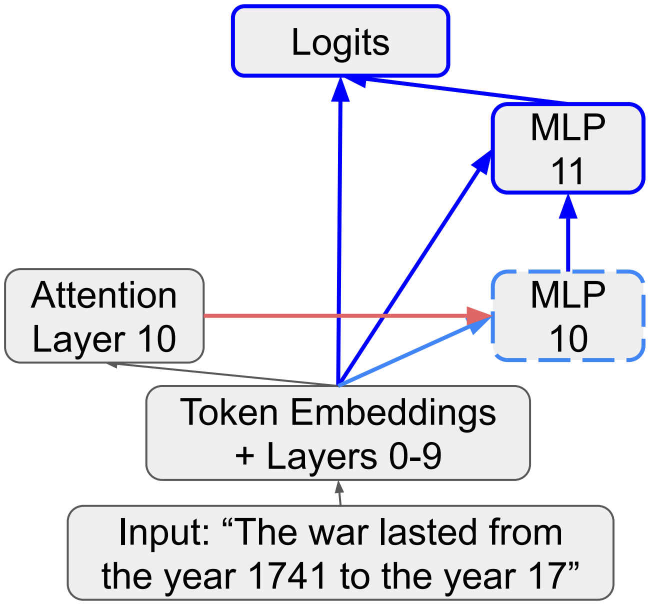


Circuit Finding: Activation Patching

Once we've reached the embeddings, we've found the circuit.

Techniques like automatic circuit discovery (ACDC, Conmy et al. (2023)) use similar approaches.

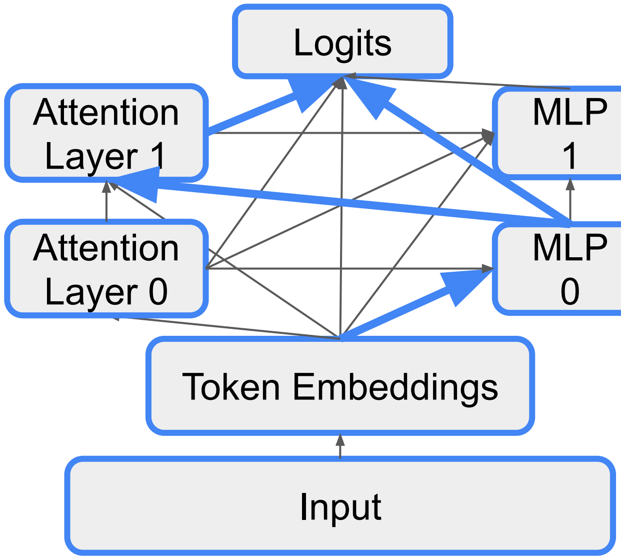
This is very slow! The solution: approximations to activation patching



Proving Circuit Faithfulness

How to prove circuit faithfulness?

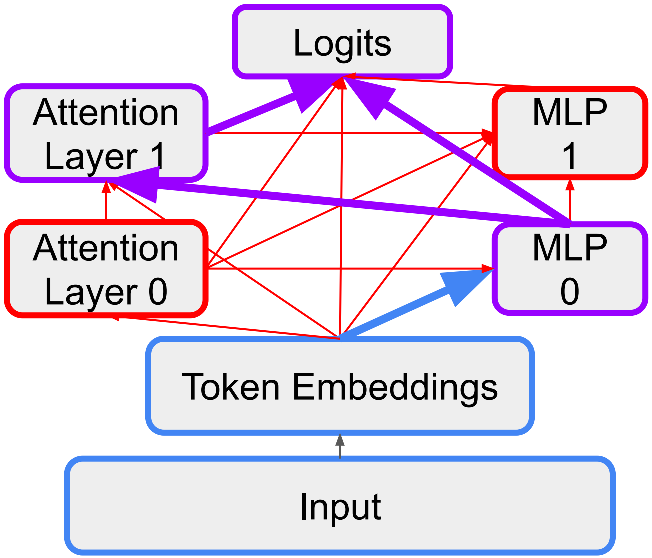
Perform another patching experiment!
Corrupt everything but your circuit.



Proving Circuit Faithfulness

A faithful circuit will have task performance close to that of the whole model! See also:

- **Completeness**: Have we discovered all components, even negative ones?
- **Minimality**: Are all components in the circuit necessary?

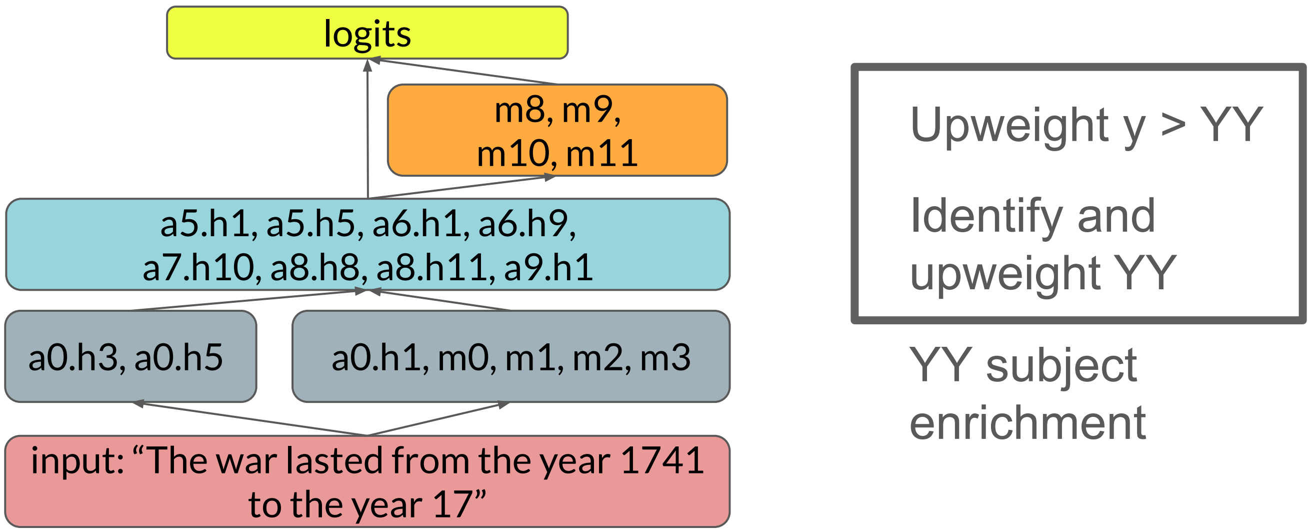


Uncovering Circuit Semantics

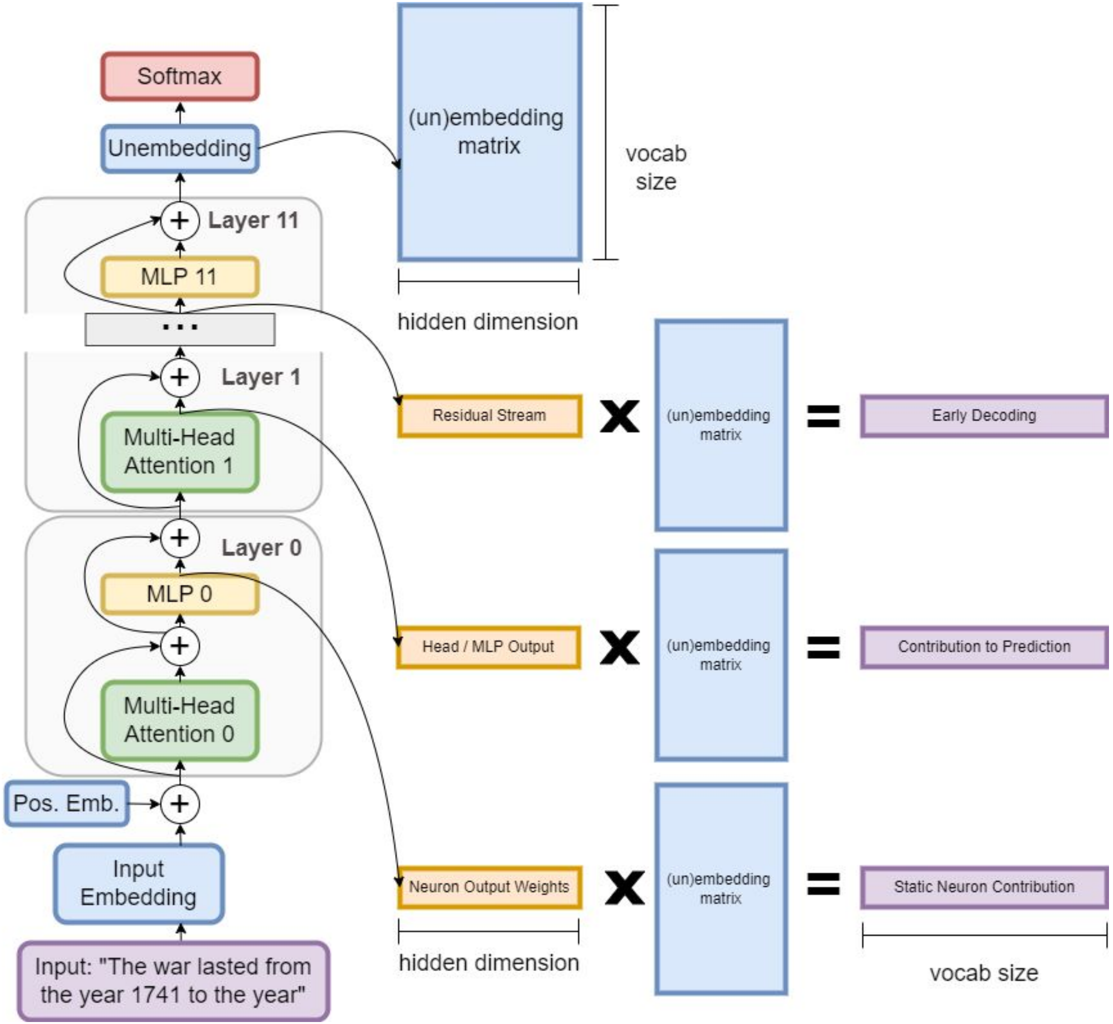
Circuit Semantics

Now we've found the structure of a circuit. How do we get to the semantics?

- This is harder than structure-finding!
- We'll stick with one method: the logit lens



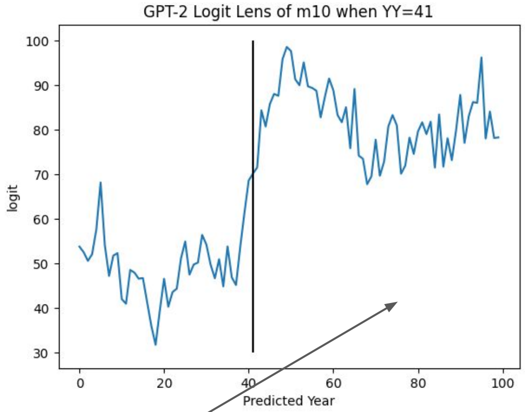
Logit Lens

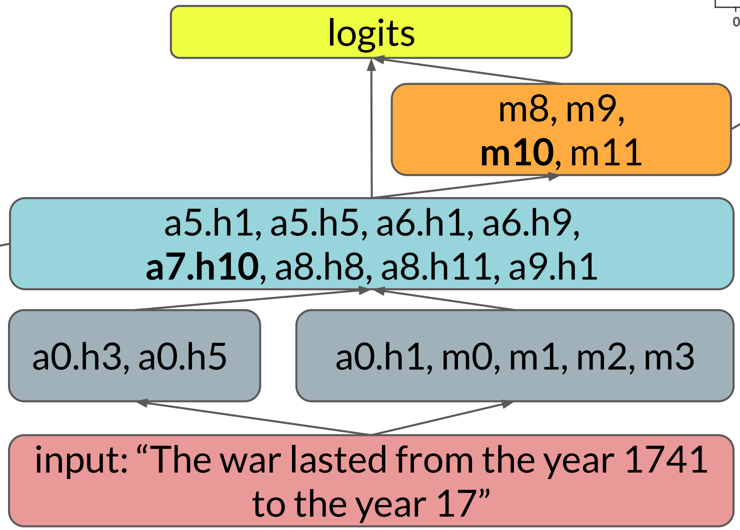


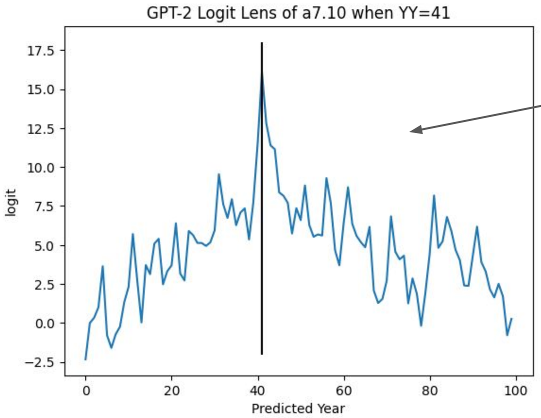
Nostalgebraist (2020), Geva et al. (2020)

The Logit Lens, Applied

How can we use the logit lens to characterize the circuit from before?

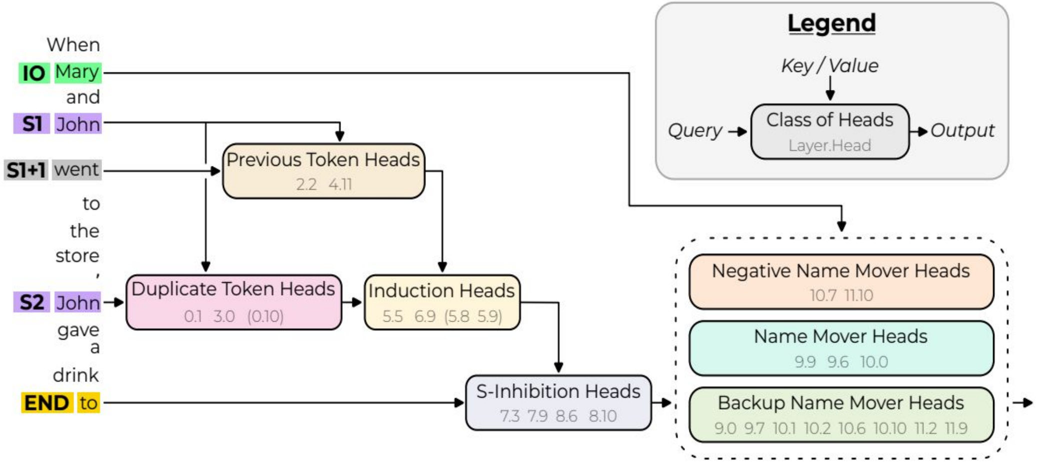






What are circuits good for?

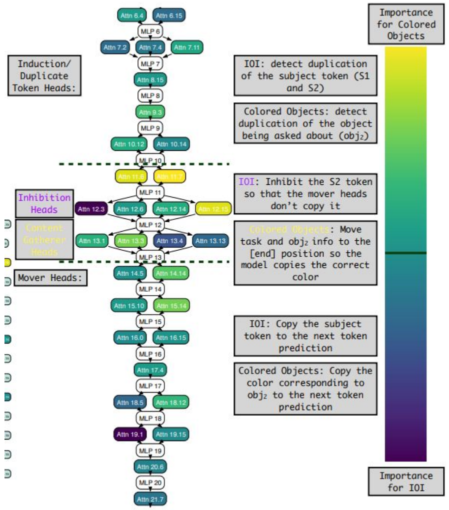
INTERPRETABILITY IN THE WILD: A CIRCUIT FOR INDIRECT OBJECT IDENTIFICATION IN GPT-2 SMALL



Wang et al., 2023

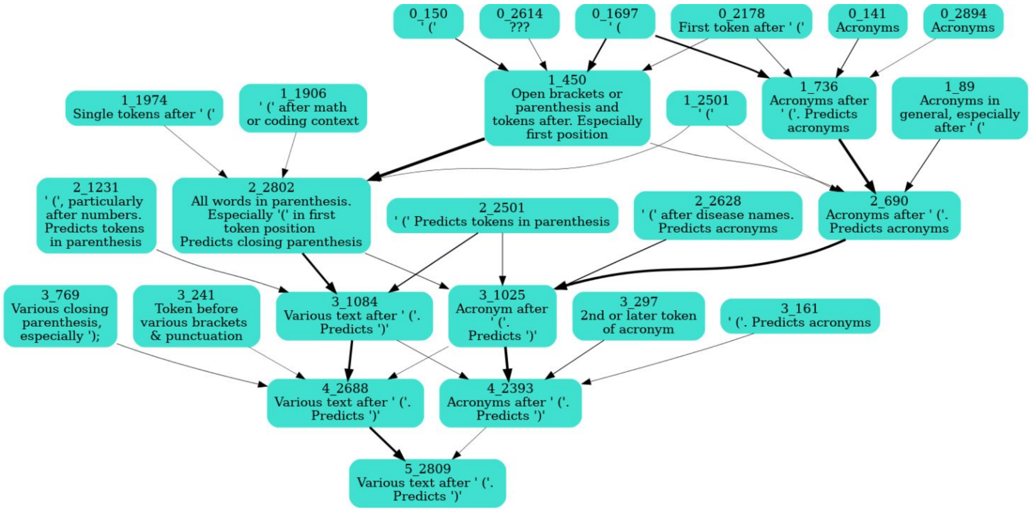
What are circuits good for?

CIRCUIT COMPONENT REUSE ACROSS TASKS IN TRANSFORMER LANGUAGE MODELS



Merullo et al., 2024

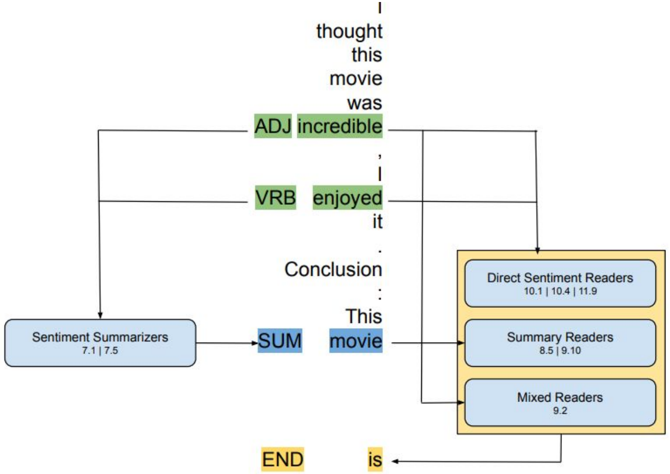
SPARSE AUTOENCODERS FIND HIGHLY INTERPRETABLE FEATURES IN LANGUAGE MODELS



Cunningham et al., 2024

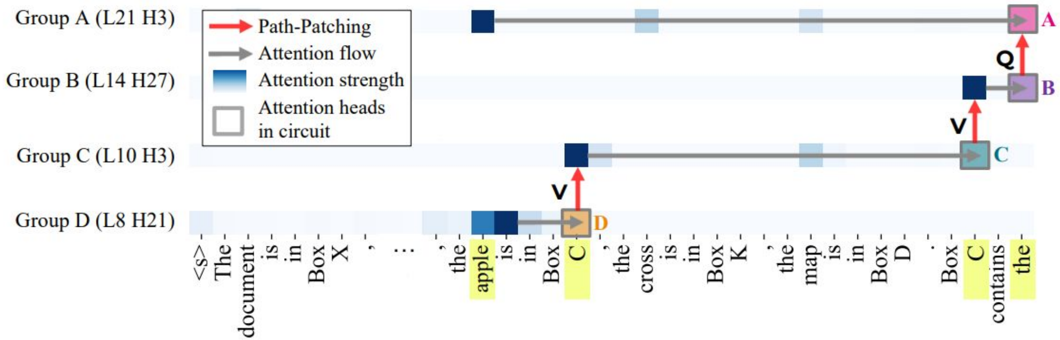
What are circuits good for?

Linear Representations of Sentiment in Large Language Models



Tigges et al., 2023

Fine-Tuning Enhances Existing Mechanisms: A Case Study on Entity Tracking



Prakash et al., 2024

Try the notebooks!

The QR code leads to a folder with two notebooks:

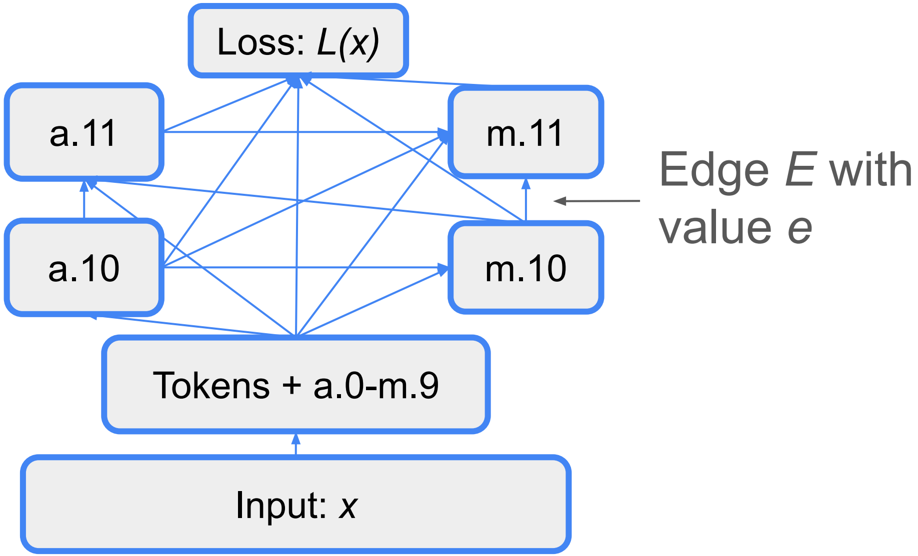
- **high_level_circuit_finding.ipynb**: shows you how to use high-level automated methods to find circuits for the tasks you care about!
- **low_level_circuit_finding.ipynb**: shows you how to do low-level circuit-finding, like path / edge patching, and the logit lens

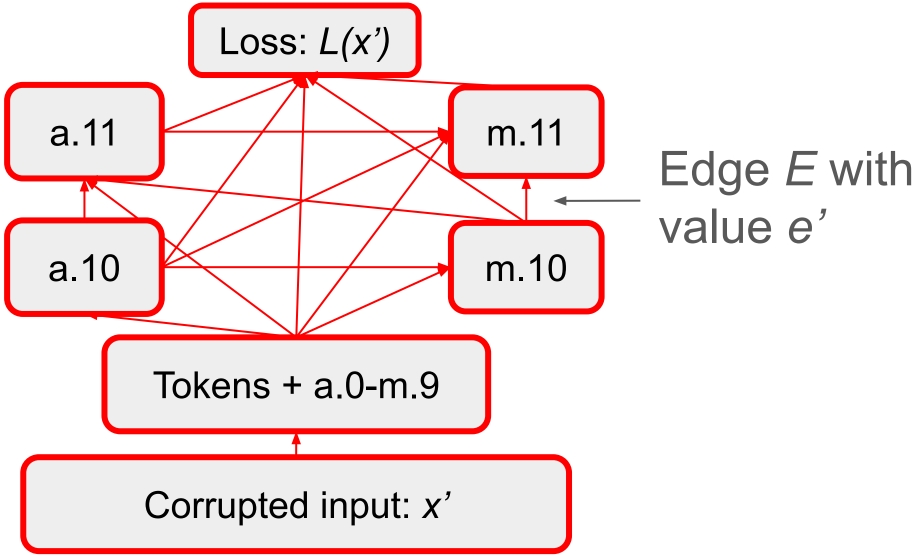
https://shorturl.at/bsEJW

Backup Slides

Circuit Finding: Attribution Patching

Activation patching's scaling is bad, *O(# edges)*. We can approximate it instead!

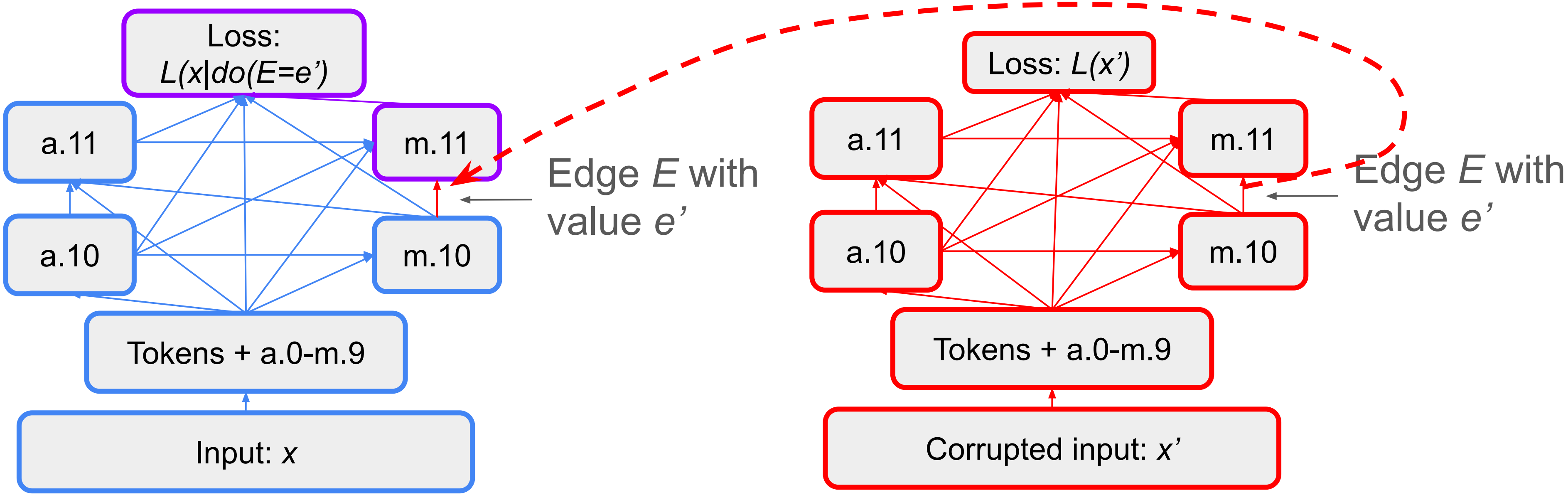




Nanda (2023)

Circuit Finding: Attribution Patching

Activation patching's scaling is bad: *O(# edges)*. We can approximate it instead!



Activation patching computes a score $s(E) = |L(x|do(E = e')) - L(x)|$

Circuit Finding: Attribution Patching

Activation patching computes a score $s(E) = |L(x|do(E = e')) - L(x)|$

We substitute the first term with its first-order Taylor approximation at *e*:

$$L(x|do(E = e')) \approx L(x) + (e' - e)^\top \frac{\partial}{\partial e} L(x|do(E = e))$$

Leaving us with:

$$s(E) \approx (e' - e)^\top \frac{\partial}{\partial e} L(x|do(E = e)) = (e' - e)^\top \frac{\partial}{\partial e} L(x)$$

This is computable for all edges in just two forward and one backward pass!

Syed et al. (2023)