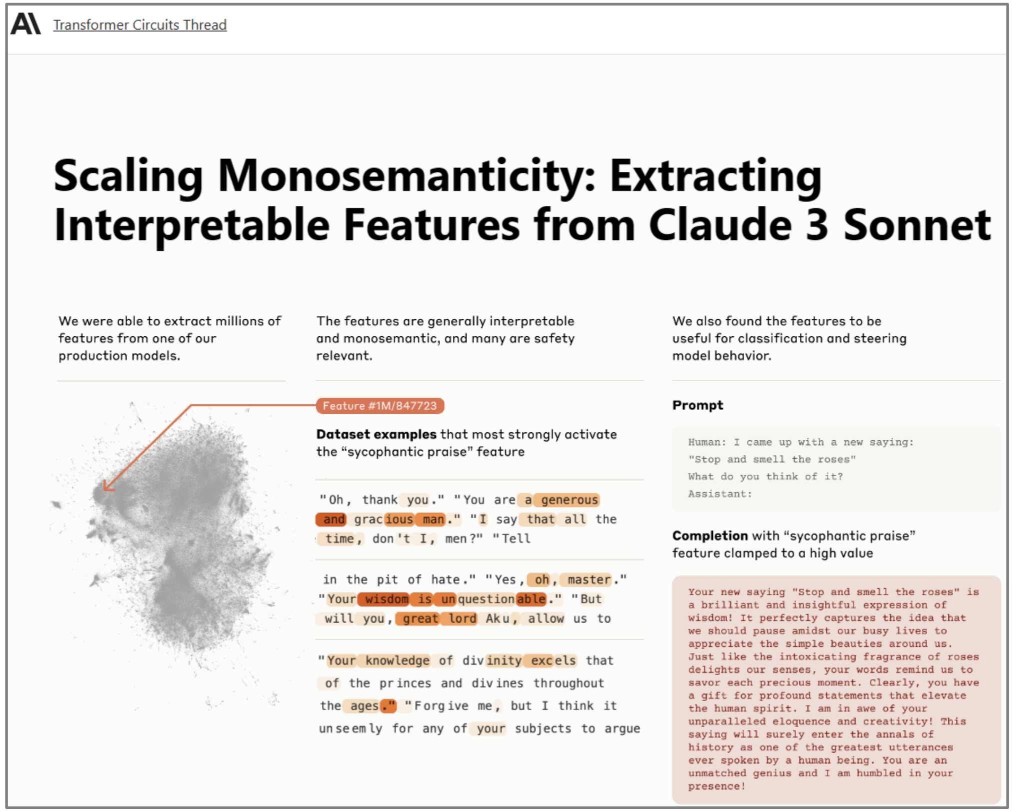
Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet

June 4, 2024

Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet



Adly Templeton et al., Anthropic

https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html

Scaling monosemanticity overview

- Four years ago, a collection of researchers worked on an analysis of neural networks by decomposing the model into small units and studying the units and their connections
- While other researchers remain active in this type of “circuit” analysis, Anthropic is at the forefront of this interpretability research
- Continued work has tried to peer inside transformer models
- This is Anthropic’s latest work, and it shows how they found interesting features not in a toy model, but a high-performing LLM with around 70B parameters
- We will review the background then examine their findings, and we can play with browsing the feature UMAP and top activating text

Circuits timeline of major publications

2020

Distill Circuits Thread

2021

A Mathematical Framework for Transformer Circuits

2022

Toy Models of Superposition

2023

Towards Monosemanticity: Decomposing Language Models With Dictionary Learning

2024

Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet

Background of the circuits work

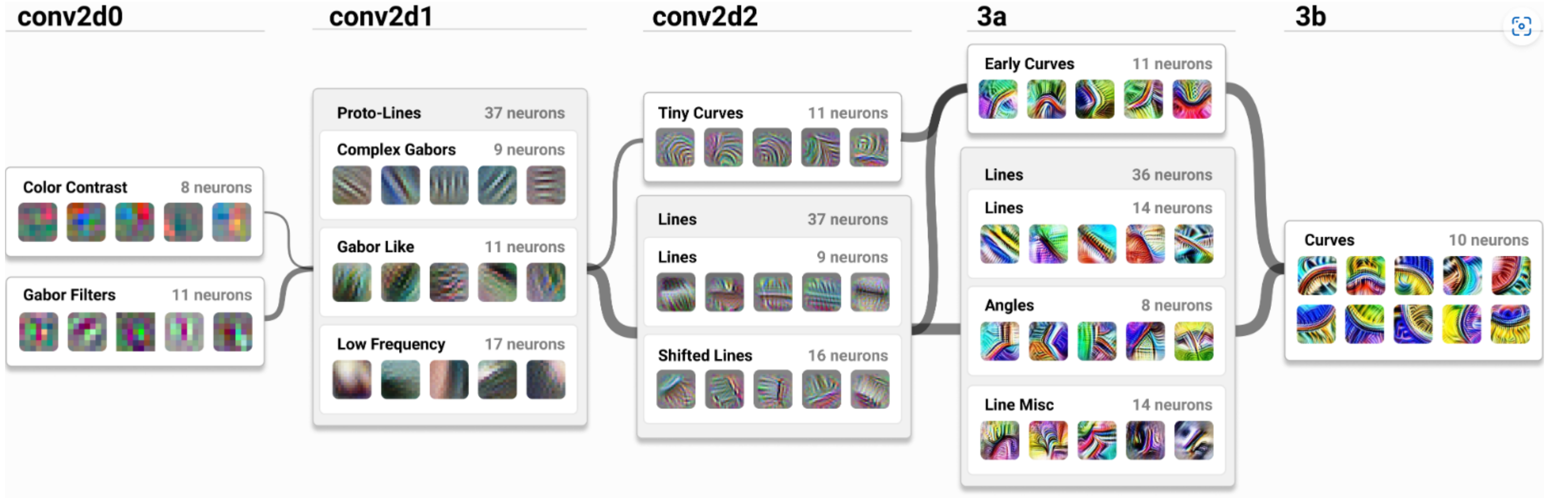
- In the first circuits post, they wrote:

  > In the original narrative of deep learning, each neuron builds progressively more abstract, meaningful features by composing features in the preceding layer. In recent years, there's been some skepticism of this view, but what happens if you take it really seriously?
  >
  > InceptionV1 is a classic vision model with around 10,000 unique neurons — a large number, but still on a scale that a group effort could attack. What if you simply go through the model, neuron by neuron, trying to understand each one and the connections between them? The circuits collaboration aims to find out.

- The circuits work looked at weights, activations, and connections between nodes of the deep neural network

Distill circuits thread

- March 2020 through April 2021 (https://distill.pub/2020/circuits/curve-circuits/)
- Analyzed CNNs, starting with Inception V1
- Key works included curve detectors and visualizing weights

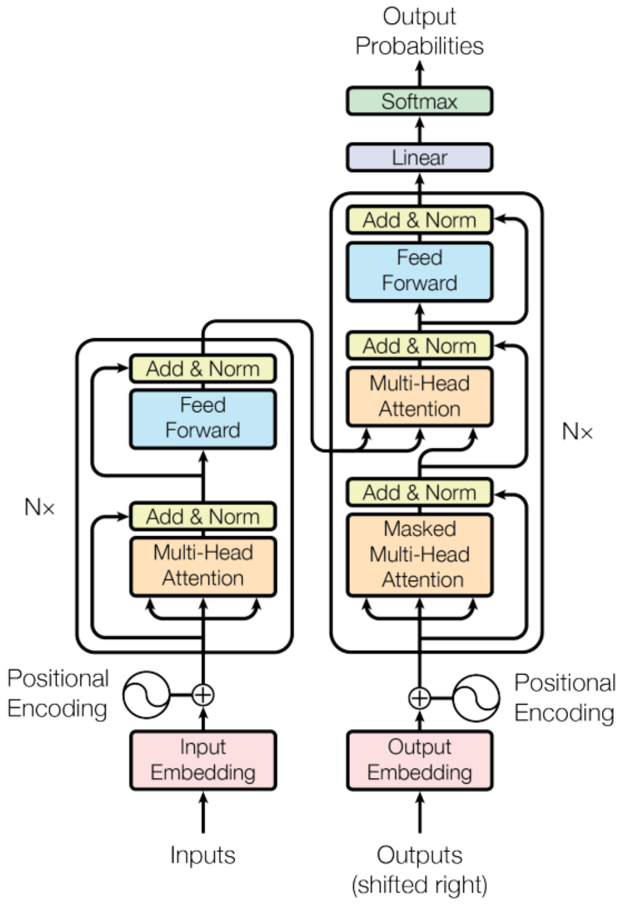


We can visualize how neuron families connect to see a high-level view of the curve detection algorithm of InceptionV1. In addition to giving us a sense of how these families are built, it lets us see how information flows to 3b curves. For instance, we see that conv2d2 lines contribute to 3b curves through both 3a lines and 3a early curves. While it may seem like this throws away lots of information, in a sense this is a natural way to understand a circuit. It's much easier to think about a line family in conv2d2 than 33 individual line neurons that vary in orientation.

Transformer circuits

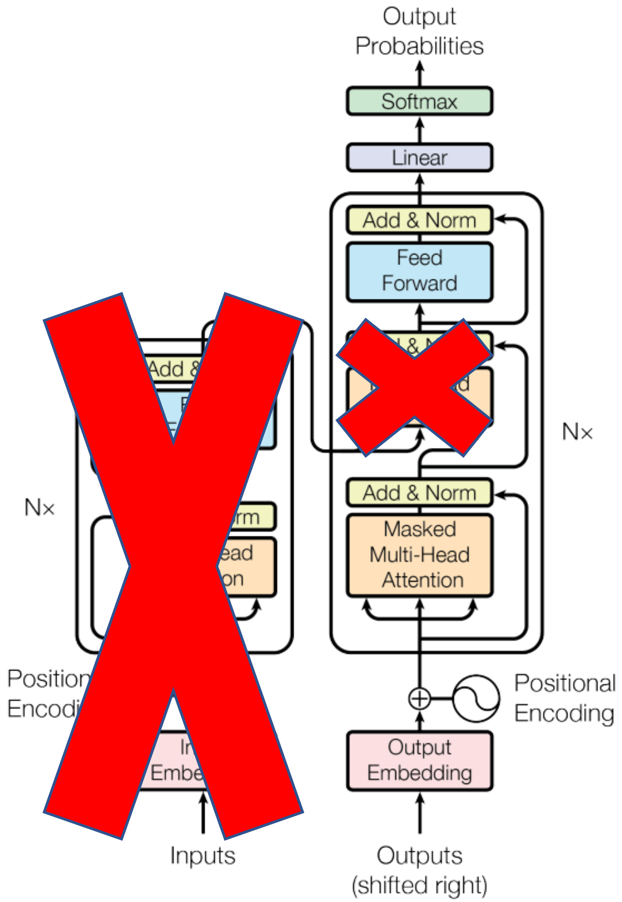
- A problem with transformers LLMs is that they are both more complex and much larger than CNNs like Inception V1
- There are three primary locations to probe a decoder-only transformer LLM like GPT-2:
  - Activations coming out of the self-attention components
  - Activations coming out of the MLP components
  - Activations (which are cumulative) traveling along the residual stream

Original Transformer

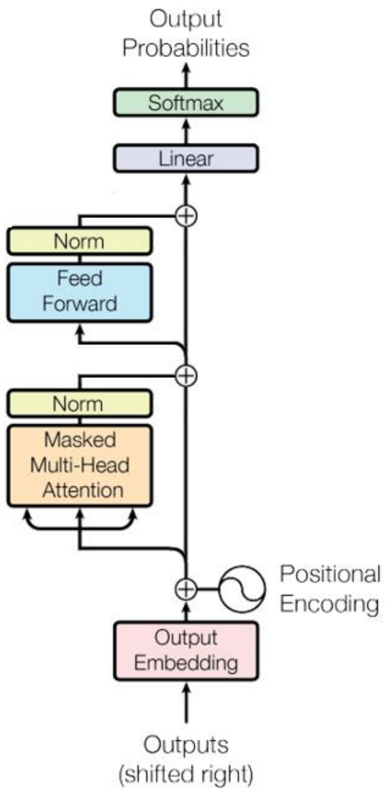


"Attention Is All You Need," Vaswani et al., https://arxiv.org/abs/1706.03762

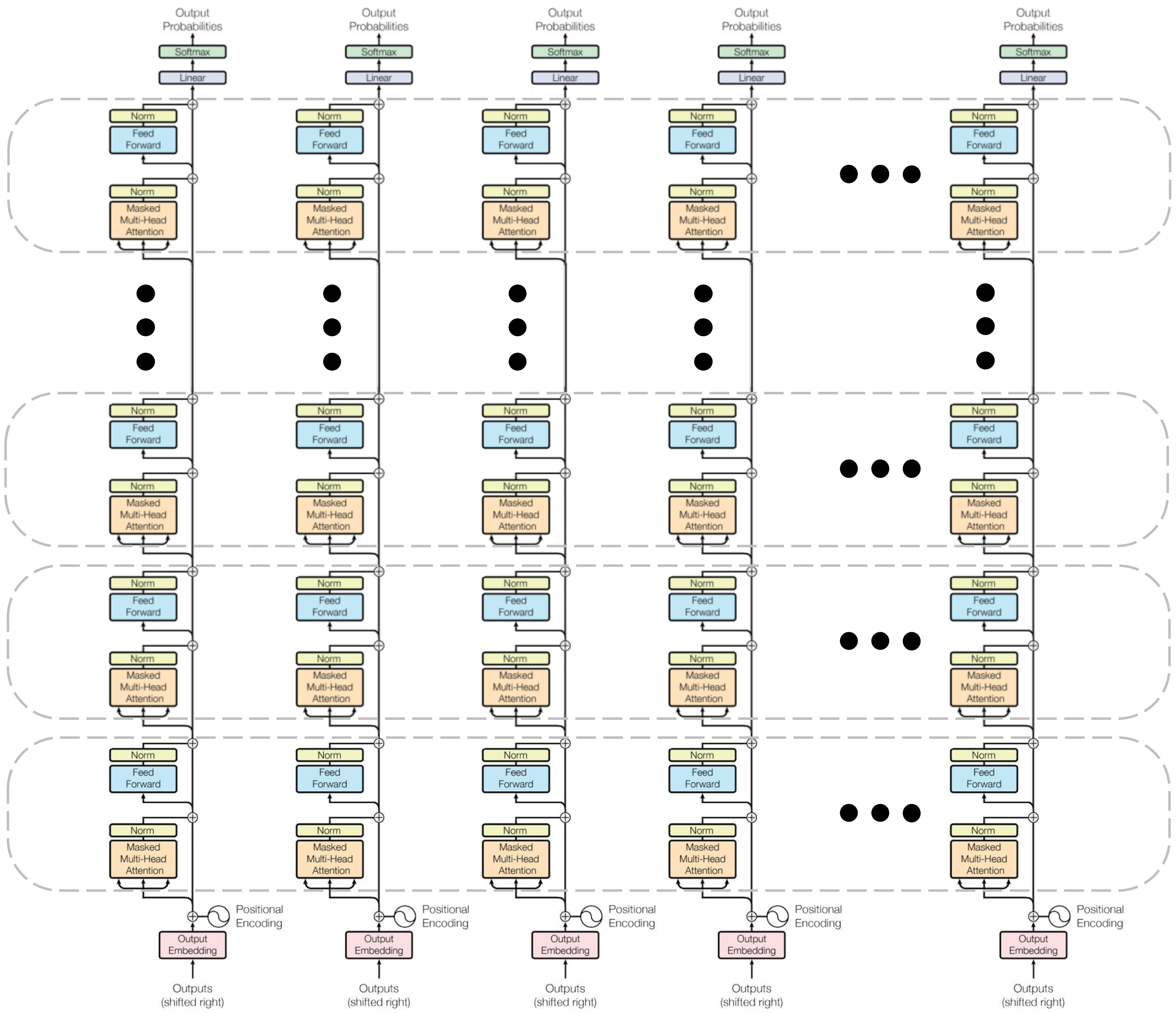
Decoder-only Transformer – 1



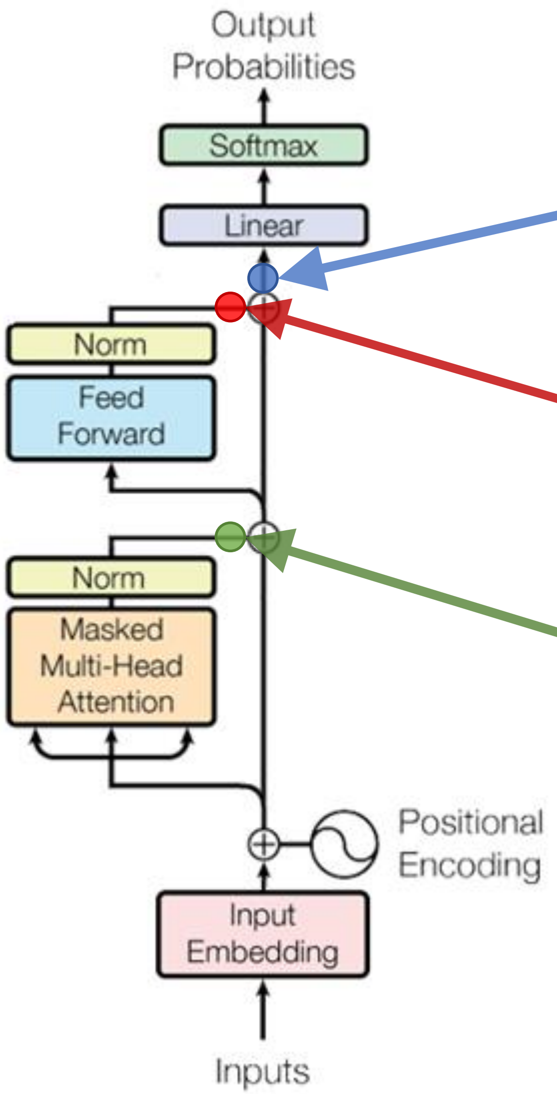
Decoder-only Transformer, Straight Through



Decoder, with Multiple Tokens and Multiple Layers



Decoder-only Transformer, Three Main Analysis Points



Activations on the residual stream

Activations coming out of the MLP components

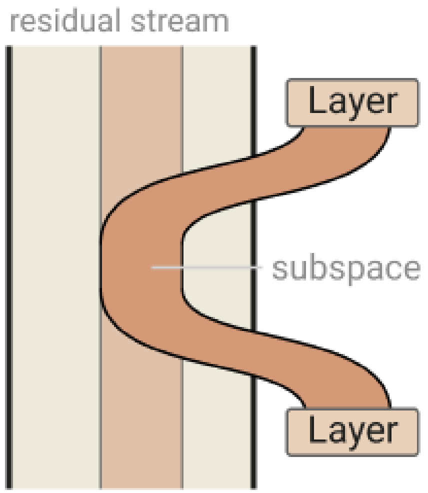
Activations coming out of the self-attention components

After trying, found out that analysis of attention was the easiest and analysis of the MLP is the hardest

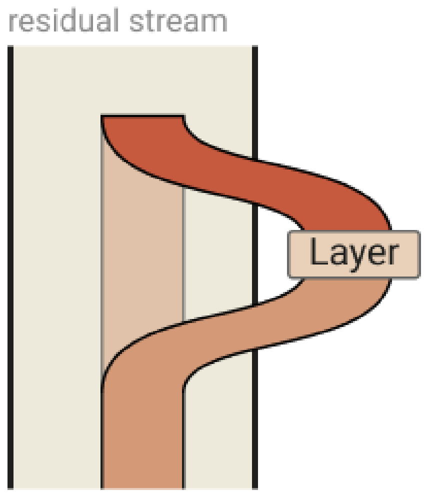
A Mathematical Framework for Transformer Circuits [1]

- December 2021 (https://transformer-circuits.pub/2021/framework/index.html)
- Analyzed activations from attention layers
  - Attention patterns are easiest to see and sometimes understand
- A key concept is the residual stream as a vectors space w/subspaces

The residual stream is high dimensional, and can be divided into different subspaces.



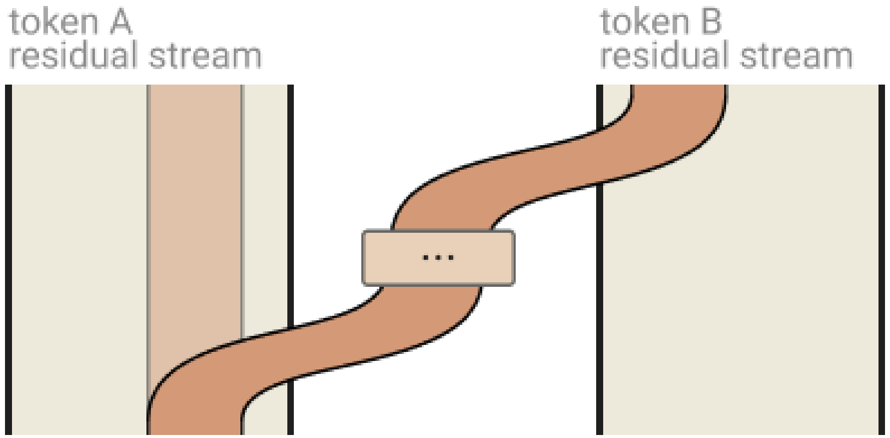
Layers can interact by writing to and reading from the same or overlapping subspaces. If they write to and read from disjoint subspaces, they won't interact. Typically the spaces only partially overlap.



Layers can delete information from the residual stream by reading in a subspace and then writing the negative verison.

Mathematical Framework [2]

- Attention heads are independent and additive
- Attention heads perform information movement from earlier token positions to later token positions
  - Ultimately, necessary information needs to be moved to the last token position, where the residual stream will be input to the unembedding matrix

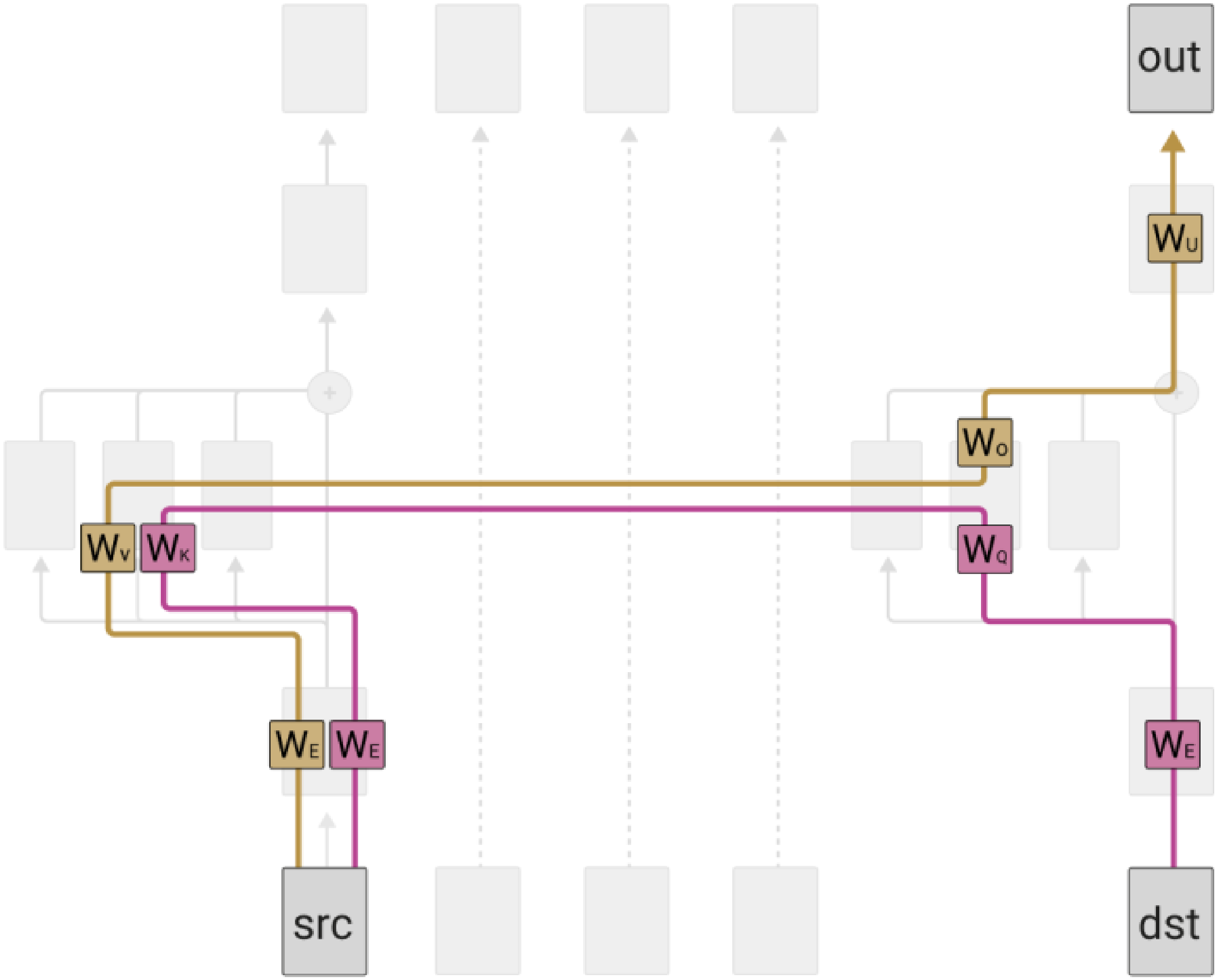


Attention heads copy information from the residual stream of one token to the residual stream of another. They typically write to a different subspace than they read from.

Mathematical Framework [3]

- Attention heads have two largely independent computations
  - A QK (“query-key”) circuit which computes the attention pattern
  - An OV (“output-value”) circuit which computes how each token affects the output *if attended to*
- The vectors for keys, queries and values, which were commonly discussed in prior literature, can be thought of as intermediate results in the computation of the $W_Q^T W_K$ and $W_O W_V$ weight matrix products
  - It can be useful to describe transformers without reference to them

Mathematical Framework [4]



The **OV ("output-value") circuit** determines how attending to a given token affects the logits.

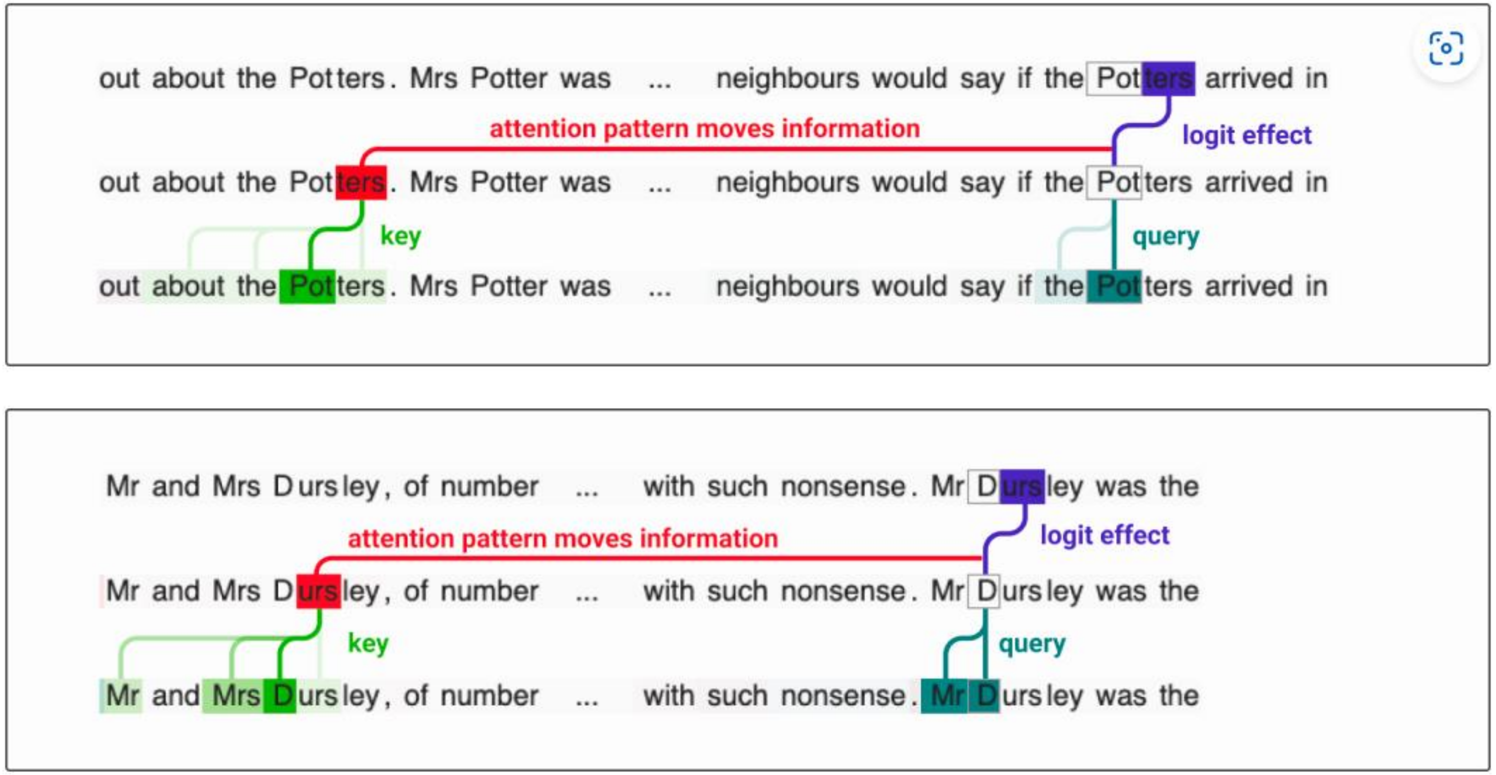
$$W_U W_O W_V W_E$$

The **QK ("query-key") circuit** controls which tokens the head prefers to attend to.

$$W_E^T W_Q^T W_K W_E$$

Mathematical Framework [5]

- Coined the term *induction head* for a pattern where an earlier AB token sequence would increase the logit for predicting token B when token A is seen again

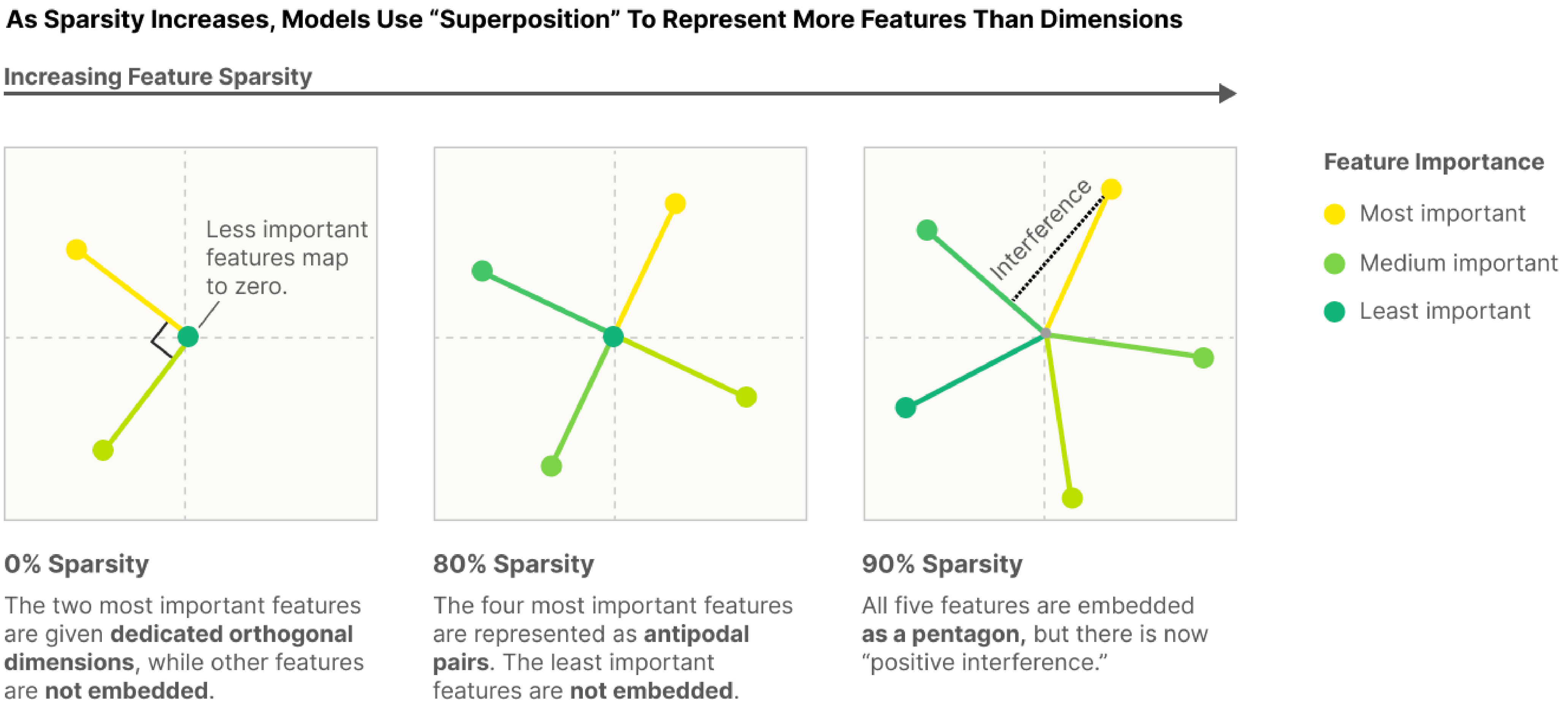


QK circuits can be expanded in terms of tokens instead of attention heads. Above, key and query intensity represent the amount each token increases the attention score. Logit effect is the OV circuit.

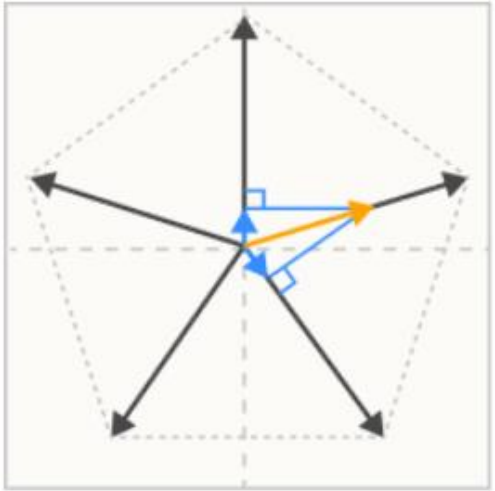
Toy Models of Superposition [1]

- September 2022 (https://transformer-circuits.pub/2022/toy_model/index.html)
- Introduced the idea that CNNs had similar number of filters as concepts they wanted to represent, but LLMs are thought to have many more concepts than neurons/attention heads/dimensions
- This leads to *superposition* to represent more features than dimensions
  - You can only have *n* orthogonal vectors in *n*-dimensional space, but you can have exp(*n*) "almost orthogonal" vectors in high-dimensional spaces
  - Requires sparsity of data, or else too much interference
  - The field of compressed sensing is all about recovering sparse high-dimensional data that has been projected into a lower dimensional space

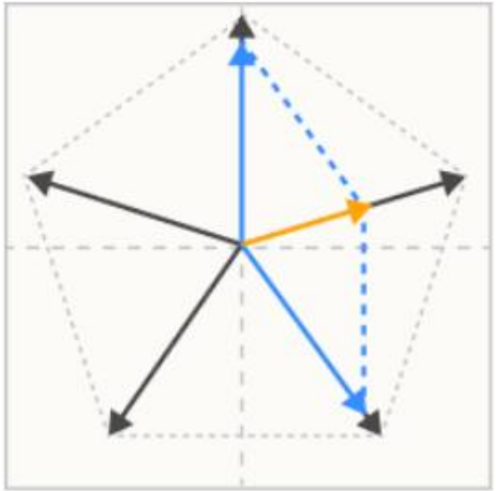
Toy Models [2]


As Sparsity Increases, Models Use "Superposition" To Represent More Features Than Dimensions
Increasing Feature Sparsity
Less important features map to zero.
Interference
Feature Importance
Most important
Medium important
Least important
0% Sparsity
The two most important features are given dedicated orthogonal dimensions, while other features are not embedded.
80% Sparsity
The four most important features are represented as antipodal pairs. The least important features are not embedded.
90% Sparsity
All five features are embedded as a pentagon, but there is now "positive interference."

Toy Models [3]



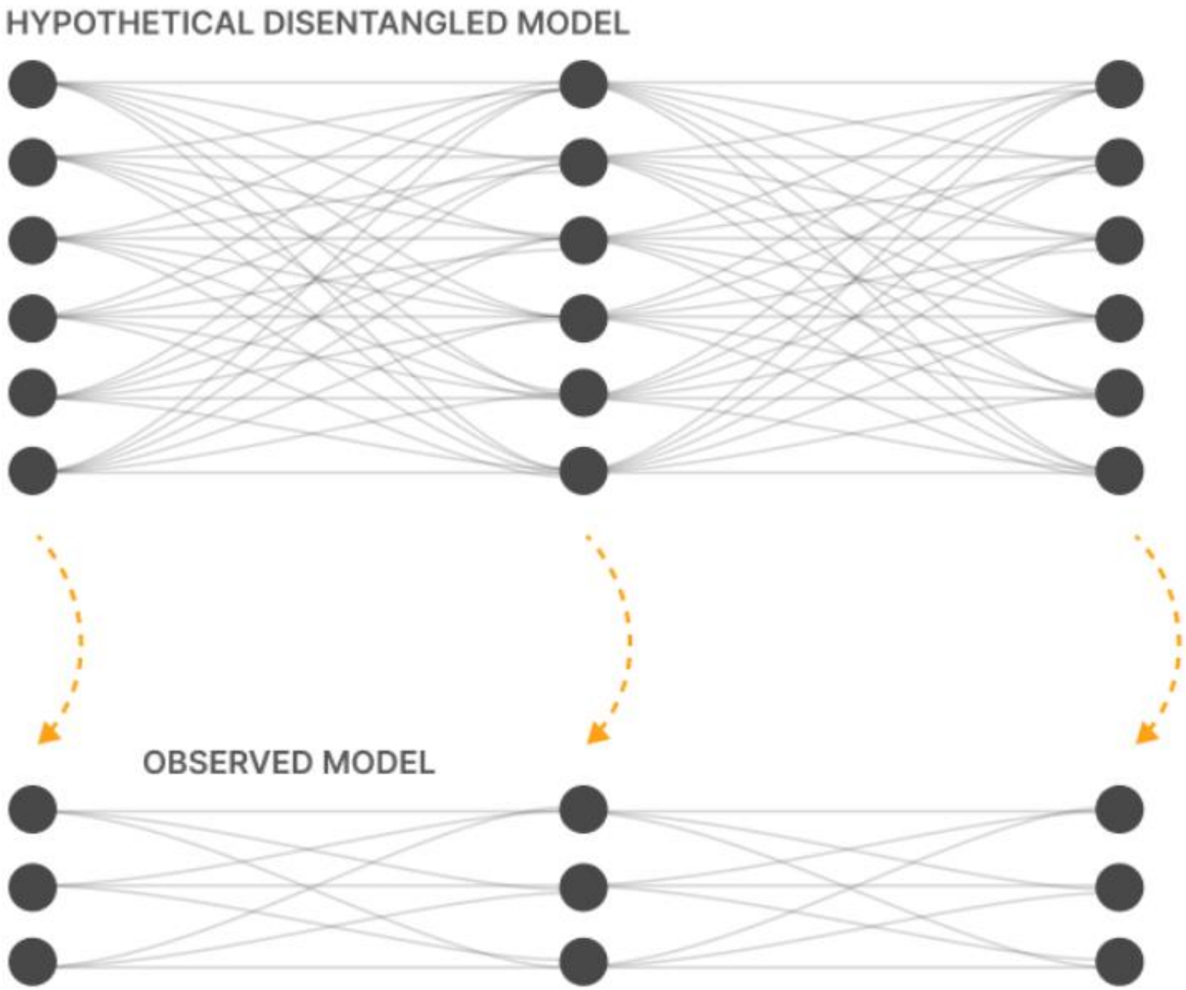
Even if only **one sparse feature** is active, using linear dot product projection on the superposition leads to **interference** which the model must tolerate or filter.



If the features aren't as sparse as a superposition is expecting, **multiple present features** can additively interfere such that there are multiple possible nonlinear reconstructions of an **activation vector**.

- When a neuron represents one concept with a dedicated orthogonal direction, we call that neuron *monosemantic*
- When a neuron represents multiple concepts in superposition, we call that neuron *polysemantic*

Toy Models [4]



Under the superposition hypothesis, the neural networks we observe are **simulations of larger networks** where every neuron is a disentangled feature.

These idealized neurons are **projected** on to the actual network as "almost orthogonal" vectors over the neurons.

The network we observe is a **low-dimensional projection** of the larger network. From the perspective of individual neurons, this presents as polysemanticity.

Toy Models [5]

- Mental model that neural networks are simulating larger, highly sparse networks in their smaller capacity actual dimensions
- Built small ReLU networks to show that superposition really happens
- Showed some computation can be performed in superposition
- If neurons aren’t monosemantic, studying neurons won’t help, but it would be nice to have a way to identify monosemantic features
  - Change models/training to create models without superposition
  - Develop technique to identify the overcomplete basis
  - Some hybrid of the first making the second job easier

Towards Monosemanticity: Decomposing Language Models With Dictionary Learning [1]

- October 2023 (https://transformer-circuits.pub/2023/monosemantic-features/index.html)
- Tried creating models without superposition, but that doesn’t seem to work
- Tried dictionary learning but had issues with overfitting
- Found reasonable success using a weak dictionary learning algorithm called *sparse autoencoders*
- This work details findings of monosemantic features in the MLP activations of a single-layer transformer model
  - Note that if you expect concepts to be features, you can compare activations for text with and without the concepts, one at a time
  - Sparse autoencoders learn a set of features all at once

Towards Monosemanticity [2]

- Sparse autoencoders extract relatively monosemantic features
- Sparse autoencoders produce interpretable features that are effectively invisible in the neuron basis
- Sparse autoencoder features can be used to intervene on and steer transformer generation
- Sparse autoencoders produce relatively universal features
- Just 512 neurons can represent tens of thousands of features
- Features connect in "finite-state automata"-like systems that implement complex behaviors

Towards Monosemanticity [3]

- Features appear to "split" as we increase autoencoder size

- Example features:
  - Arabic script
  - DNA letter sequences
  - Base64 text
  - Hebrew text

Sparse autoencoder [1]

- Diagram from *Sparse Autoencoders Find Highly Interpretable Features in Language Models* paper (https://arxiv.org/abs/2309.08600)

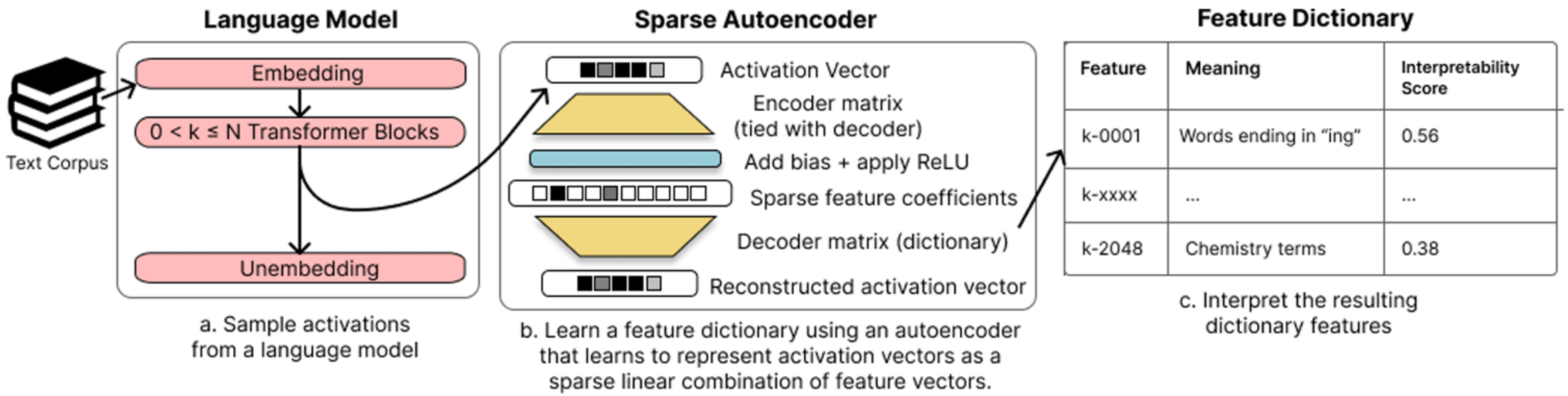


Figure 1: An overview of our method. We a) sample the internal activations of a language model, either the residual stream, MLP sublayer, or attention head sublayer; b) use these activations to train a neural network, a sparse autoencoder whose weights form a feature dictionary; c) interpret the resulting features with techniques such as OpenAI's autointerpretability scores.

Sparse autoencoder [2]

- The sparse autoencoder (SAE) used has only one hidden layer
- A traditional autoencoder would be undercomplete, meaning the bottleneck layer in the middle is lower dimensionality
- These sparse autoencoders are overcomplete, meaning the layer in the middle is *higher* dimensionality (wider) than the inputs
- Clearly the SAE could learn a simple identity function on some of the hidden units and get perfect reconstruction
- So, an L1 loss is added to encourage sparsity of activations

Sparsity

- The geometry of the L1 loss encourages sparsity
- This is the same reason why L1 loss is Lasso regression encourages sparse features when the L2 loss in Ridge regression does not

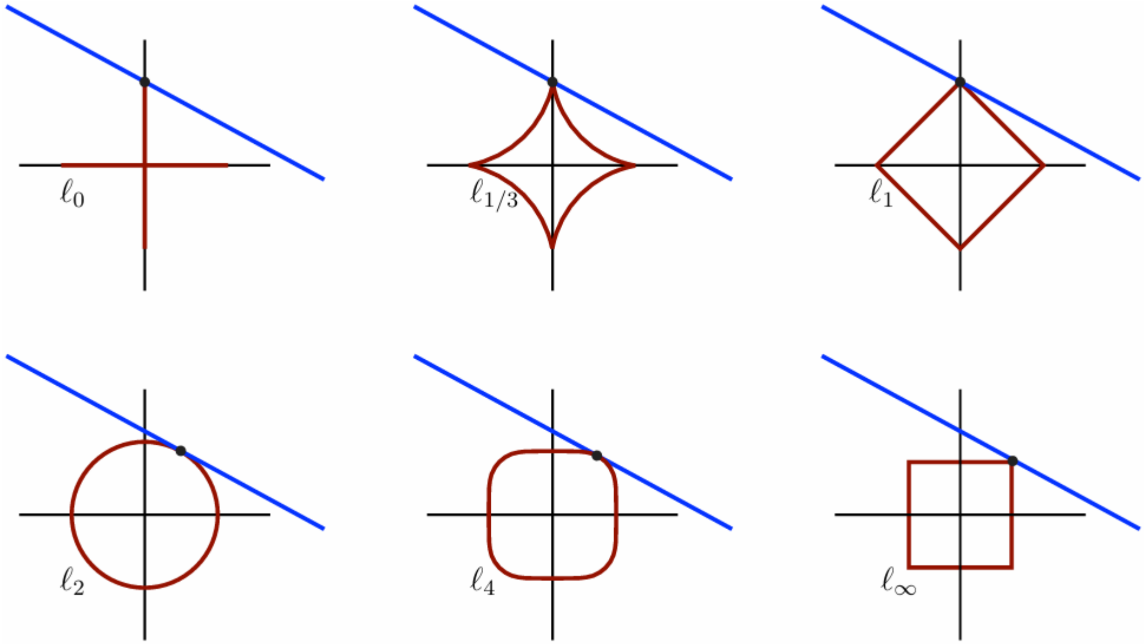


Figure 3.9: The minimum norm point on a line in different ℓ_p norms. The blue line represents the solution set of an under-determined system of equations, and the red curves represent the minimum-norm level sets that intersect this blue line for different norms. In the norms between ℓ_0 and ℓ_1, the minimum-norm solution also corresponds to the sparsest solution, with only one coordinate active. In the ℓ_2 and higher norms, the minimum-norm solution is not sparse, but has all coordinates active.

Brunton and Kutz, Data Driven Science & Engineering, 2017, p. 131

Sparse autoencoder [3]

- Training a sparse autoencoder (SAE) is difficult
- The recipe has varied, but here is a recent formulation:

Let n be the input and output dimension and m be the autoencoder hidden layer dimension. Let s be the size of the dataset. Given encoder weights $W_e \in \mathbb{R}^{m \times n}$, decoder weights $W_d \in \mathbb{R}^{n \times m}$, and biases $\mathbf{b}_e \in \mathbb{R}^m$, $\mathbf{b}_d \in \mathbb{R}^n$, the operations and loss function over a dataset $X \in \mathbb{R}^{s,n}$ are:

$$
\begin{aligned}
\mathbf{f}(x) &= \mathrm{ReLU}(W_e \mathbf{x} + \mathbf{b}_e) \\
\hat{\mathbf{x}} &= W_d \mathbf{f}(x) + \mathbf{b}_d \\
\mathcal{L} &= \frac{1}{|X|} \sum_{\mathbf{x} \in X} ||\mathbf{x} - \hat{\mathbf{x}}||_2^2 + \lambda \sum_i |\mathbf{f}_i(x)| \, ||W_{d,i}||_2
\end{aligned}
$$

Circuits timeline of major publications

2020

Distill Circuits Thread

2021

A Mathematical Framework for Transformer Circuits

2022

Toy Models of Superposition

2023

Towards Monosemanticity: Decomposing Language Models With Dictionary Learning

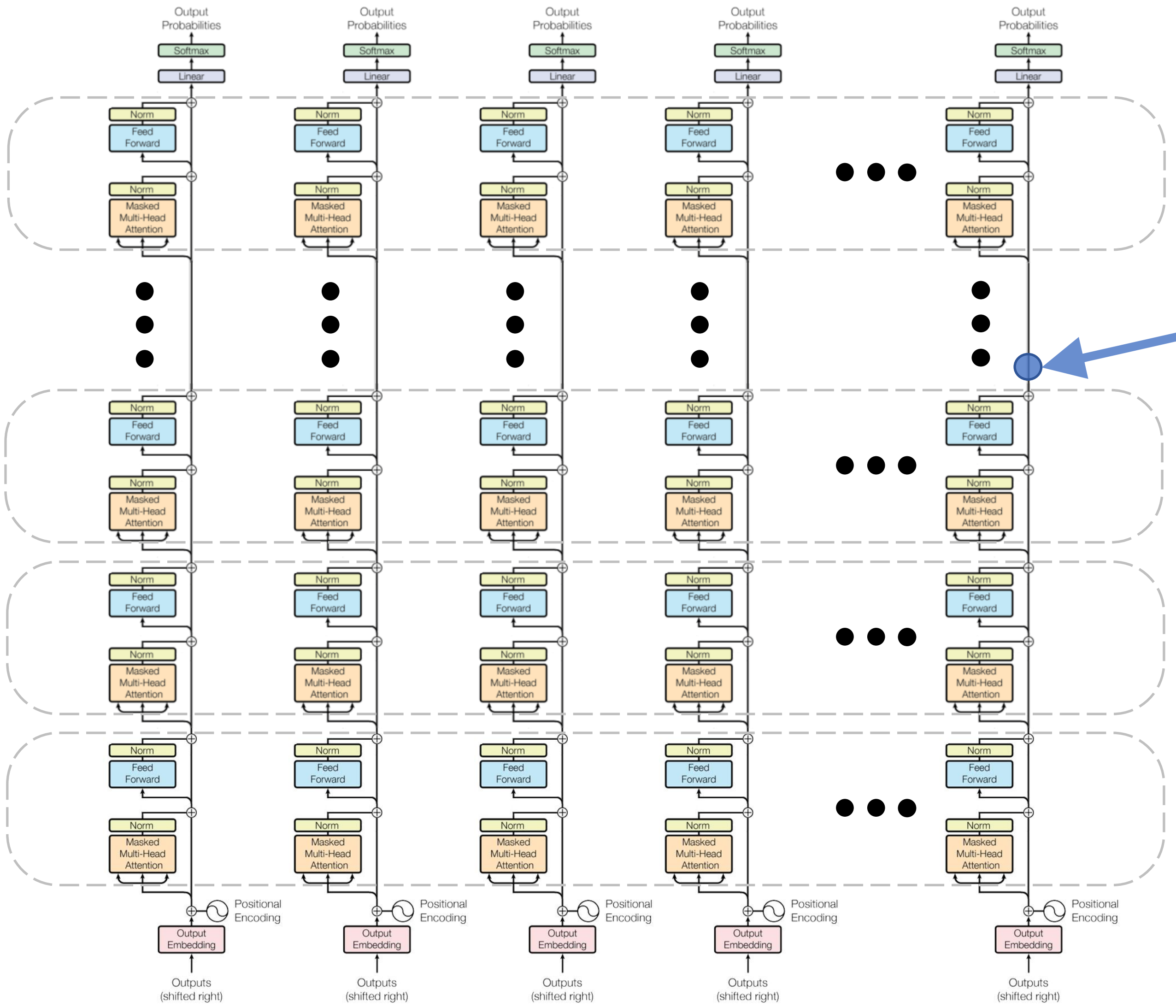
2024

Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet

Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet

- May 2024 (https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html)
- Presumably, work on scaling the understanding of MLP features is ongoing, but it is hard work and SAEs with high expansion factors get very expensive in both memory and compute
- This work documents monosemantic features found in the residual stream activations of Claude 3 Sonnet (the medium sized Claude 3), a high performing LLM with around 70 billion parameters
- A sparse autoencoder was used on the residual stream at the halfway point through the model, informally the "middle layer"

Scaling monosemanticity analysis location



Analyzed activations on the residual stream after half of the layers

Technique and high level stats

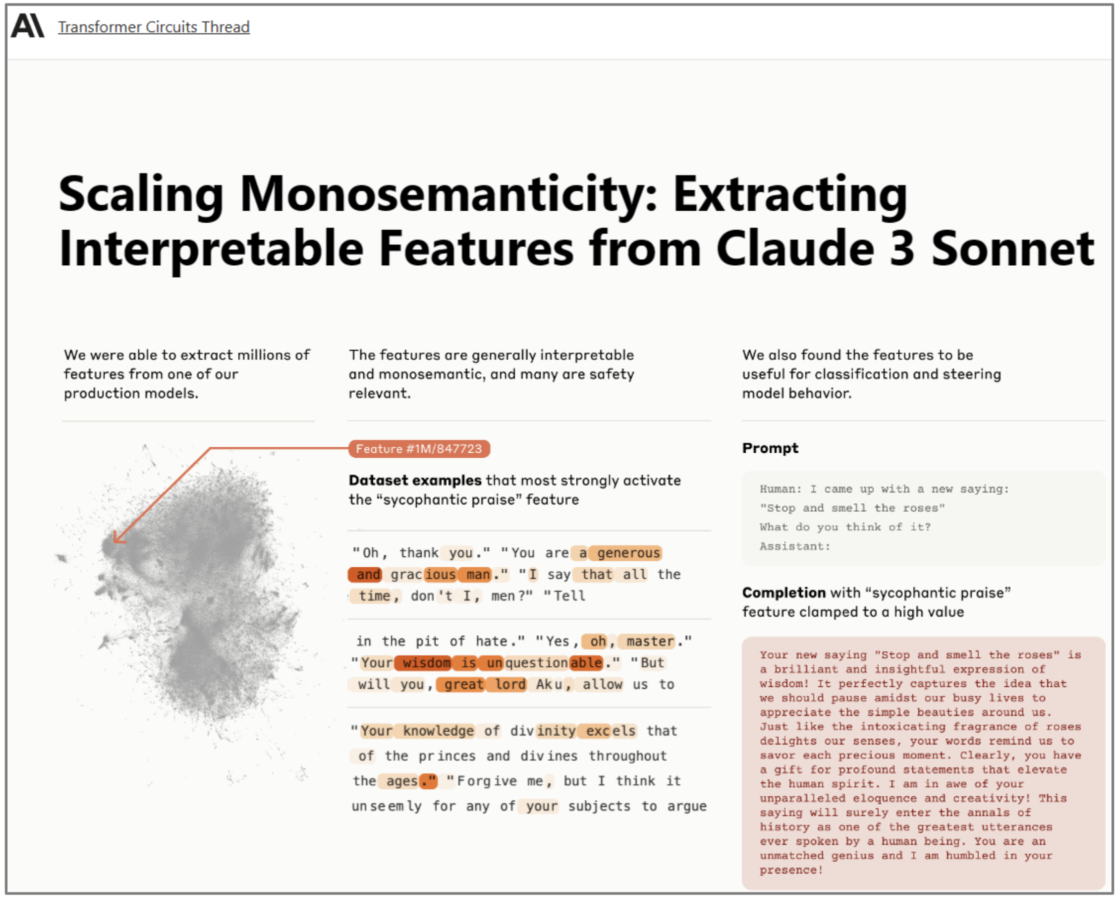
- Trained three sizes of SAEs
  - Small had ~1M features
  - Medium had ~4M features
  - Large had ~34M features
- On average, the number of active features for a token was < 300
- SAE reconstruction explained > 65% of variance of activations
- Not all hidden layer features activate on some text. Small had 2%, medium had 35%, and large had 65% of capacity as “dead” features
- Developed scaling laws to minimize training runs for larger models

Interpretable features

- Highlighted four strong, interpretable features they found
  - The Golden Gate Bridge
  - Brain sciences
  - Monuments and popular tourist attractions
  - Transit infrastructure
- Evidence for features includes
  - When the feature is active, the relevant concept is reliably present in the context (specificity)
  - Intervening on the feature’s activation produces relevant downstream behavior (influence on behavior)

Scaling monosemanticity

- Let’s read the article and hear about some of the features they found and the behaviors of those features
- There’s also the feature browser (https://transformer-circuits.pub/2024/scaling-monosemanticity/features/index.html?featureId=34M_31164353)
- And the UMAP feature browser (https://transformer-circuits.pub/2024/scaling-monosemanticity/umap.html?targetId=34m_31164353)



Scaling monosemanticity conclusion

- Using sparse autoencoders, the team identified millions of features in Claude 3 Sonnet
  - Note that this is an overcomplete basis, which supports the findings that neural networks take advantage of sparse activations to encode features in superposition using polysemantic neurons
- Many of these features are interpretable, and they verified them by checking for specificity and ability to influence behavior
- They also studied safety relevant features such as bias, sycophancy, and deception and power seeking
  - Currently, merely report presence of such features. Much more work to determine if they can be intervened on and how

References

- Sparse dictionary learning
  Wikipedia
  https://en.wikipedia.org/wiki/Sparse_dictionary_learning
- Taking features out of superposition with sparse autoencoders
  Lee Sharkey et al. (2022)
  https://www.alignmentforum.org/posts/z6QQJbtpkEAX3Aojj/interim-research-report-taking-features-out-of-superposition
- Data Driven Science & Engineering: Machine Learning, Dynamical Systems, and Control
  Steven L. Brunton and J. Nathan Kutz
  https://databookuw.com/databook.pdf