Intro to Mechanistic Interpretability

COMS6998: Frontiers of ML

Sweta Karlekar

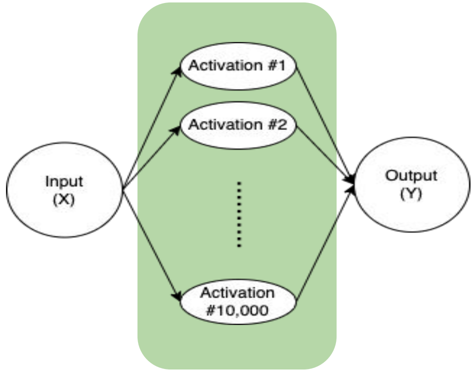
Overview

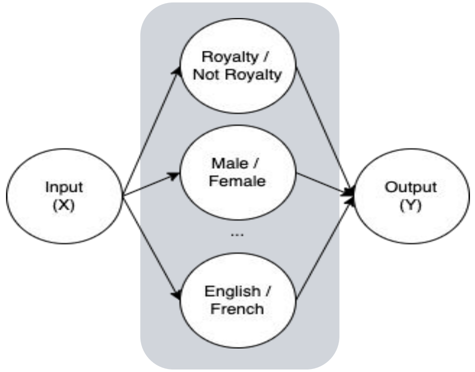
- ➔ Background
- ➔ Linear Representation Hypothesis
- ➔ Finding Feature Directions
- ➔ Using Feature Directions

Background

A Causal Problem

$$\begin{tikzcd}\boxed{\text{Large Language Models}} \arrow[r] & \boxed{\text{Causal DAGs}}\end{tikzcd}$$





➔ Nodes in a neural network represent causal relationships amongst inputs and outputs

➔ If we treat LLMs as Causal DAGs, we can use the suite of tools already available in the field of causal inference to understand the internal mechanisms of an LLM

A Causal Problem

$$\begin{tikzcd}\boxed{\text{Large Language Models}} \arrow[r] & \boxed{\text{Causal DAGs}}\end{tikzcd}$$

- ➔ Nodes in a neural network represent causal relationships amongst inputs and outputs
- ➔ Technically there’s no confounding! We know all the causes (AKA parents) of a given node in a network
- ➔ Therefore, we can perform arbitrary causal interventions in LLMs with relative ease

Challenge: Human-interpretable causal inference at this level is very difficult! It’s not possible to understand the mechanisms of billions of neurons individually. The causal problem here is to understand what’s the **right level of abstraction**, how to **define the “intervention”**, and understand **where to intervene**.

Mechanistic Interpretability

➔ Area of research that seeks to understand the neural mechanisms that enable specific behaviors in LLMs by leveraging **causality-based methods.**[3]

➔ Importantly, mechanistic interpretability focuses on reverse-engineering model components into **human-understandable** algorithms[4]

Intervene by adjusting the activation values, weights or inputs and examine the output

[3] Palit, V., Pandey, R., Arora, A., & Liang, P. P. (2023). Towards vision-language mechanistic interpretability: A causal tracing tool for blip. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 2856-2861).
[4] Conmy, A., Mavor-Parker, A., Lynch, A., Heimersheim, S., & Garriga-Alonso, A. (2023). Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, *36*, 16318-16352.

Mechanistic Interpretability

- ➔ Area of research that seeks to understand the neural mechanisms that enable specific behaviors in LLMs by leveraging **causality-based methods.**[3]
- ➔ Importantly, mechanistic interpretability focuses on reverse-engineering model components into **human-understandable** algorithms[4]

Intervene by adjusting the activation values, weights or inputs and examine the output

- ➔ What are examples of LLM interventions?
  - ◆ Circuit ablation
  - ◆ Jittering perturbations
  - ◆ Activation steering

[3] Palit, V., Pandey, R., Arora, A., & Liang, P. P. (2023). Towards vision-language mechanistic interpretability: A causal tracing tool for blip. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 2856-2861).
[4] Conmy, A., Mavor-Parker, A., Lynch, A., Heimersheim, S., & Garriga-Alonso, A. (2023). Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, *36*, 16318-16352.

Mechanistic Interpretability

- ➔ Area of research that seeks to understand the neural mechanisms that enable specific behaviors in LLMs by leveraging **causality-based methods.**[3]
- ➔ Importantly, mechanistic interpretability focuses on reverse-engineering model components into **human-understandable** algorithms[4]

Intervene by adjusting the activation values, weights or inputs and examine the output

- ➔ What are examples of LLM interventions?
  - ◆ **Circuit ablation** — Replacing activation values to remove causal paths in the computation graph
  - ◆ Jittering perturbations
  - ◆ Activation steering

[3] Palit, V., Pandey, R., Arora, A., & Liang, P. P. (2023). Towards vision-language mechanistic interpretability: A causal tracing tool for blip. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 2856-2861).
[4] Conmy, A., Mavor-Parker, A., Lynch, A., Heimersheim, S., & Garriga-Alonso, A. (2023). Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, *36*, 16318-16352.

Mechanistic Interpretability

- ➔ Area of research that seeks to understand the neural mechanisms that enable specific behaviors in LLMs by leveraging **causality-based methods.**[3]
- ➔ Importantly, mechanistic interpretability focuses on reverse-engineering model components into **human-understandable** algorithms[4]

Intervene by adjusting the activation values, weights or inputs and examine the output

- ➔ What are examples of LLM interventions?
  - ◆ Circuit ablation
  - ◆ **Jittering perturbations** — add noise/variations to inputs or activations to measure robustness or sensitivity, can be used in conjunction with other interventions
  - ◆ Activation steering

[3] Palit, V., Pandey, R., Arora, A., & Liang, P. P. (2023). Towards vision-language mechanistic interpretability: A causal tracing tool for blip. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 2856-2861).
[4] Conmy, A., Mavor-Parker, A., Lynch, A., Heimersheim, S., & Garriga-Alonso, A. (2023). Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, *36*, 16318-16352.

COLUMBIA

Mechanistic Interpretability

- ➔ Area of research that seeks to understand the neural mechanisms that enable specific behaviors in LLMs by leveraging **causality-based methods.**[3]
- ➔ Importantly, mechanistic interpretability focuses on reverse-engineering model components into **human-understandable** algorithms[4]

Intervene by adjusting the activation values, weights or inputs and examine the output

- ➔ What are examples of LLM interventions?
  - ◆ Circuit ablation
  - ◆ Jittering perturbations
  - ◆ **Activation steering** — adjusting the existing values of the activation vector with new values to steer model generation towards a specific target

[3] Palit, V., Pandey, R., Arora, A., & Liang, P. P. (2023). Towards vision-language mechanistic interpretability: A causal tracing tool for blip. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 2856-2861).
[4] Conmy, A., Mavor-Parker, A., Lynch, A., Heimersheim, S., & Garriga-Alonso, A. (2023). Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, *36*, 16318-16352.

Representations vs. Circuits

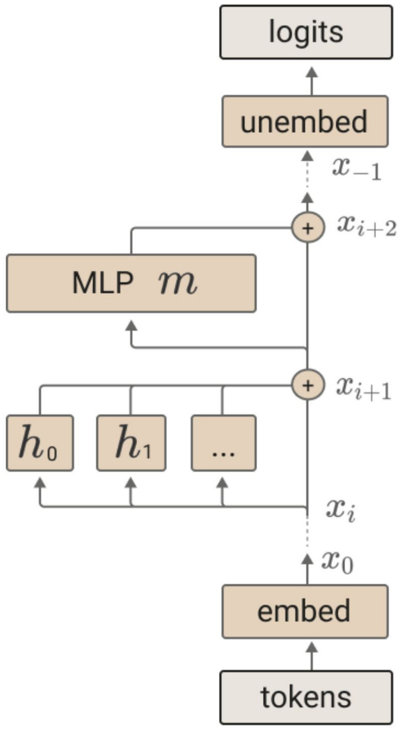
Representations

Understanding how properties of the input are **represented internally** through studying activation values

Circuits

Understanding the algorithms encoded in the weights by which **representations are computed and used**

Residual Stream

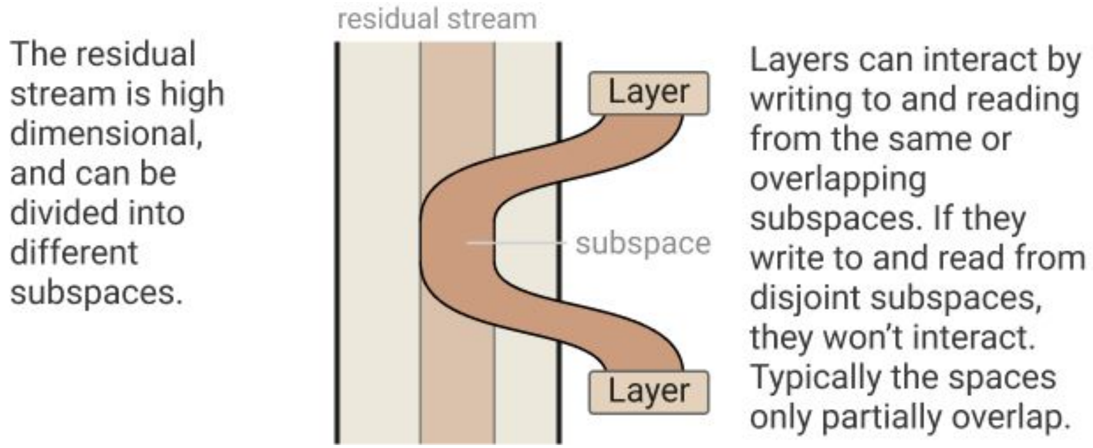


- ➔ Transformer-based models are comprised of blocks of attention layers followed by multi-layer perceptron (MLP) layers[1]
- ➔ Attention heads can be understood as independent operations, each outputting results (AKA activation values) that are added into the **residual stream**[2]
- ➔ Residual stream is **high-dimensional vector space** made up of the sum of the output of all previous layers and the original input embedding, can be thought of as a "communication channel"[2]

[1]Turner, A., Thiergart, L., Udell, D., Leech, G., Mini, U., & MacDiarmid, M. (2023). Activation addition: Steering language models without optimization. *arXiv preprint arXiv:2308.10248*.
[2]Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., ... & Olah, C. (2021). A mathematical framework for transformer circuits. *Transformer Circuits Thread*, *1*, 1.

Residual Stream

- ➔ Every attention head/MLP block independently reads from the residual stream through **linear projection** and performs another **linear projection** to the output before **adding** to “write” its output back into the residual stream [2]
- ➔ We can split the residual stream into subspaces, and we believe these subspaces can correspond to interpretable “**concepts**”



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.

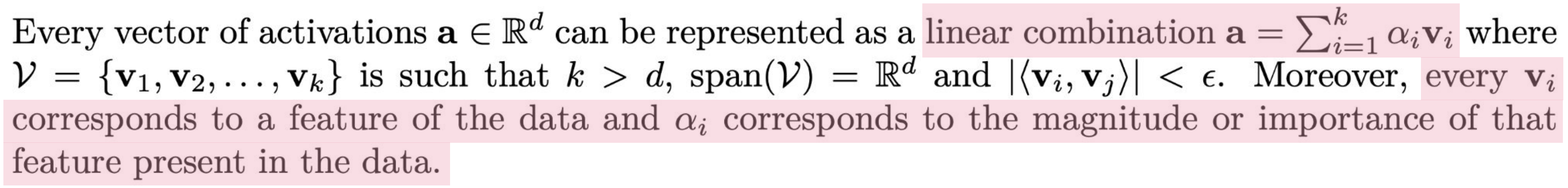
[1]Turner, A., Thiergart, L., Udell, D., Leech, G., Mini, U., & MacDiarmid, M. (2023). Activation addition: Steering language models without optimization. *arXiv preprint arXiv:2308.10248*.
[2]Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., ... & Olah, C. (2021). A mathematical framework for transformer circuits. *Transformer Circuits Thread*, *1*, 1.

Linear Representation Hypothesis

The Linear Representation Hypothesis

Every vector of activations $\mathbf{a} \in \mathbb{R}^d$ can be represented as a linear combination $\mathbf{a} = \sum_{i=1}^{k} \alpha_i \mathbf{v}_i$ where $\mathcal{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$ is such that $k > d$, $\text{span}(\mathcal{V}) = \mathbb{R}^d$ and $|\langle \mathbf{v}_i, \mathbf{v}_j \rangle| < \epsilon$. Moreover, every $\mathbf{v}_i$ corresponds to a feature of the data and α_i corresponds to the magnitude or importance of that feature present in the data.

The Linear Representation Hypothesis



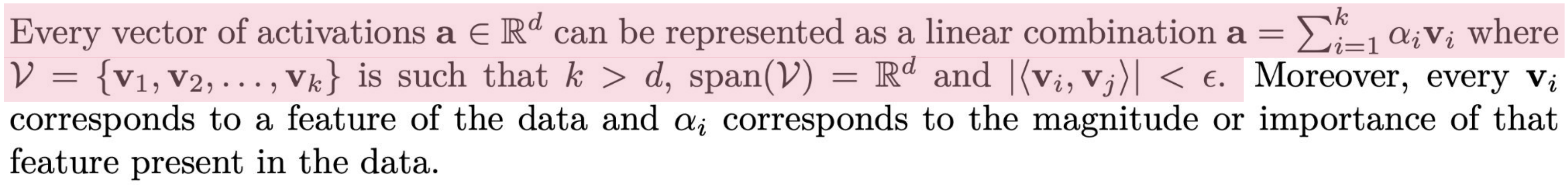
Every vector of activations $\mathbf{a} \in \mathbb{R}^d$ can be represented as a linear combination $\mathbf{a} = \sum_{i=1}^{k} \alpha_i \mathbf{v}_i$ where $\mathcal{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$ is such that $k > d$, $\text{span}(\mathcal{V}) = \mathbb{R}^d$ and $|\langle \mathbf{v}_i, \mathbf{v}_j \rangle| < \epsilon$. Moreover, every $\mathbf{v}_i$ corresponds to a feature of the data and α_i corresponds to the magnitude or importance of that feature present in the data.

A feature/concept is "any factor of variation that can be changed in isolation".

- English → French
- Male → Female
- Love → Hate
- Contemporary Prose → Shakespearean Prose

Activation vectors can be broken down into **combinations of these various feature directions**.

The Linear Representation Hypothesis



Every vector of activations $\mathbf{a} \in \mathbb{R}^d$ can be represented as a linear combination $\mathbf{a} = \sum_{i=1}^{k} \alpha_i \mathbf{v}_i$ where $\mathcal{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$ is such that $k > d$, $\text{span}(\mathcal{V}) = \mathbb{R}^d$ and $|\langle \mathbf{v}_i, \mathbf{v}_j \rangle| < \epsilon$. Moreover, every $\mathbf{v}_i$ corresponds to a feature of the data and α_i corresponds to the magnitude or importance of that feature present in the data.

However, the number of features we want to encode (*k*) in an LLM is much higher than the number of basis vectors (*d*). We have an **overcomplete basis**.

- To get around this, LLMs relax the definition of orthogonality to be within ϵ of 90 degrees, which dramatically increases the number of concepts we could encode.
- Because of this, the activation vector **a** can unintentionally activate multiple feature directions. One result of this hypothesis is we assume LLMs represent features sparsely so there is less overlap of features for any given input.

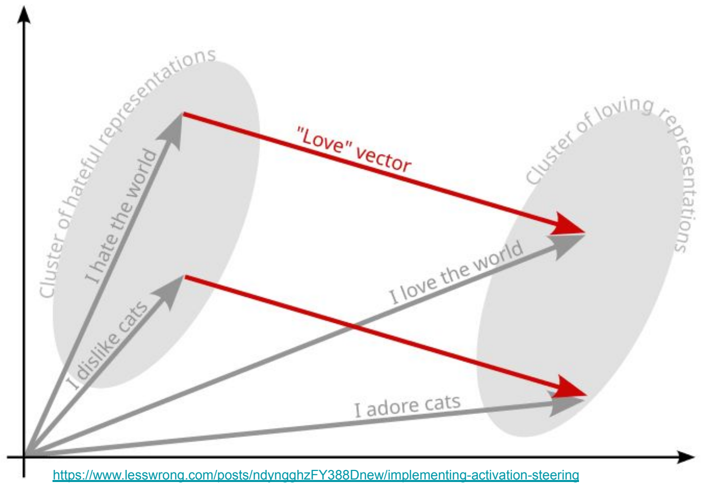
If we can find the direction vector for each concept, what can we do with it?

- **Guided generation**
  - Steer the model towards generating text that has more or less of a specific feature.
- **Understanding LLMs mechanistically**
  - What is the structure of information within an LLM? How are related topics stored within this representation space?
- **Auditing the model**
  - Know when certain features are being activated, such as features having to do with sensitive demographic information.

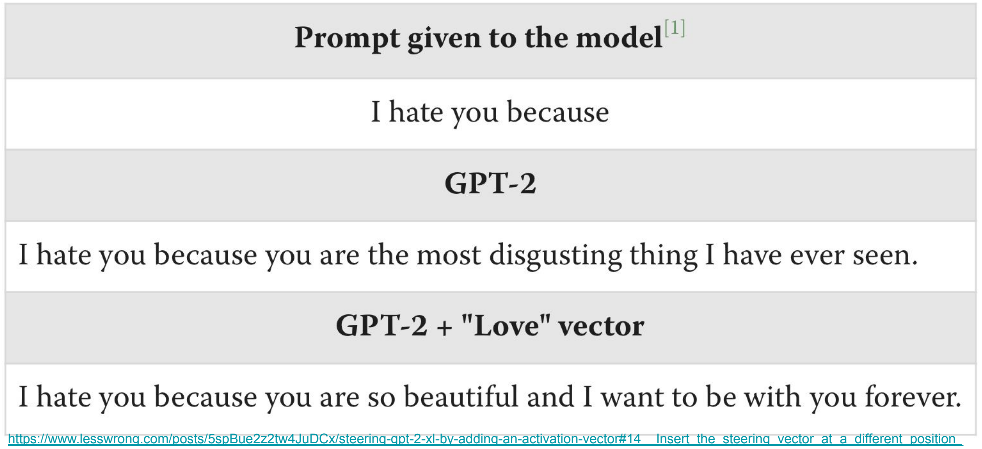
Finding Feature Directions

Steering Vectors

- ➔ Method to control or guide the behavior of the model by adjusting the internal states or activations of the model
- ➔ Goal: add some vector to the internal model activations at a given layer to influence the output in a specific way



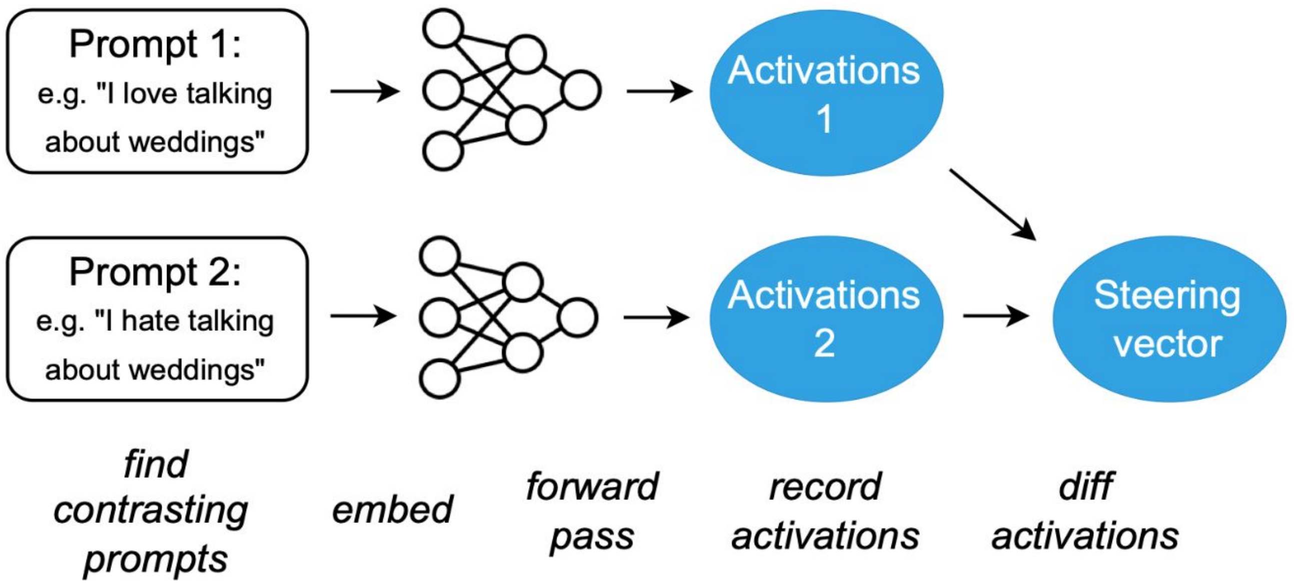
https://www.lesswrong.com/posts/ndyngghzFY388Dnew/implementing-activation-steering



Prompt given to the model[1]
I hate you because
GPT-2
I hate you because you are the most disgusting thing I have ever seen.
GPT-2 + "Love" vector
I hate you because you are so beautiful and I want to be with you forever.

https://www.lesswrong.com/posts/5spBue2z2tw4JuDCx/steering-gpt-2-xl-by-adding-an-activation-vector#14__Insert_the_steering_vector_at_a_different_position_

Finding Steering Vectors with ActAdd



Demo Notebook

https://colab.research.google.com/drive/1swr9OVAtmAWM1FUvrm2610lx4LXcVKoJ?usp=sharing

Issues with activation addition

- ➔ We will always find **some** vector with ActAdd, even if the feature does not necessarily exist within the LLM (i.e. the vector we get might not have anything to do with the feature we're interested in).
- ➔ There's a lot of potential for confounding features
  - ◆ E.g. If we have a lot of sentences about love, a lot of sentences about hate, we may have just found the positive/negative sentiment instead of love/hate.
  - ◆ E.g. If we have a bunch of famous husband/wife pairs. We may have just found the male/female direction vector instead of the spousal vector.
- ➔ This also isn't a very scalable method; finding each feature vector requires a lot of work to curate a large dataset with the desired properties.

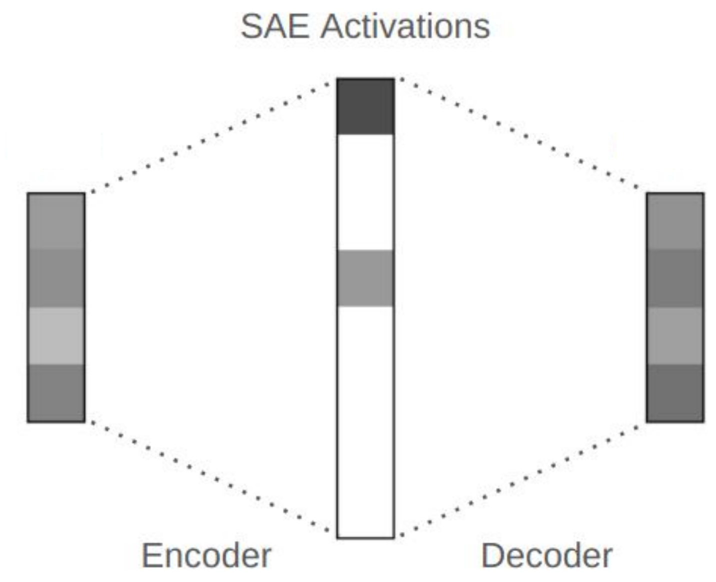
Issues with activation addition

- We will always find **some** vector with ActAdd, even if the feature does not necessarily exist within the LLM (i.e. the vector we get might not have anything to do with the feature we're interested in).
- There's a lot of potential for confounding features
  - E.g. If we have a lot of sentences about love, a lot of sentences about hate, we may have just found the positive/negative sentiment instead of love/hate.
  - E.g. If we have a bunch of famous husband/wife pairs. We may have just found the male/female direction vector instead of the spousal vector.
- This also isn't a very scalable method; finding each feature vector requires a lot of work to curate a large dataset with the desired properties.
- What can we use instead?

Issues with activation addition

- ➔ We will always find **some** vector with ActAdd, even if the feature does not necessarily exist within the LLM (i.e. the vector we get might not have anything to do with the feature we're interested in).
- ➔ There's a lot of potential for confounding features
  - ◆ E.g. If we have a lot of sentences about love, a lot of sentences about hate, we may have just found the positive/negative sentiment instead of love/hate.
  - ◆ E.g. If we have a bunch of famous husband/wife pairs. We may have just found the male/female direction vector instead of the spousal vector.
- ➔ This also isn't a very scalable method; finding each feature vector requires a lot of work to curate a large dataset with the desired properties.
- ➔ What can we use instead? ... Sparse autoencoders!

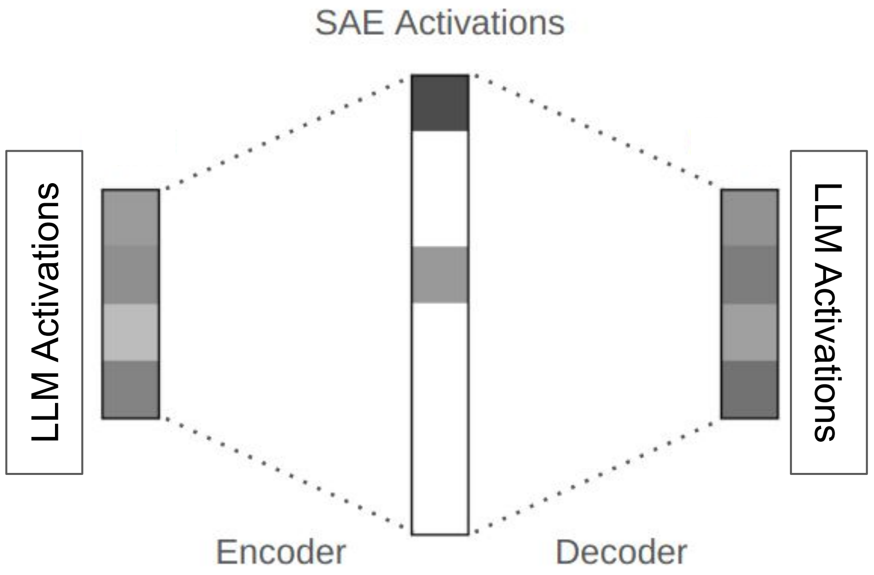
Sparse Autoencoders (SAEs) for Feature Learning



- ➔ Sparse autoencoders are trained to reconstruct their input data
- ➔ Trains on a combination of reconstruction loss and L1 loss which induces sparsity

https://www.lesswrong.com/posts/CJPqwXoFtgkKPRay8/an-intuitive-explanation-of-sparse-autoencoders-for

Sparse Autoencoders (SAEs) for Feature Learning



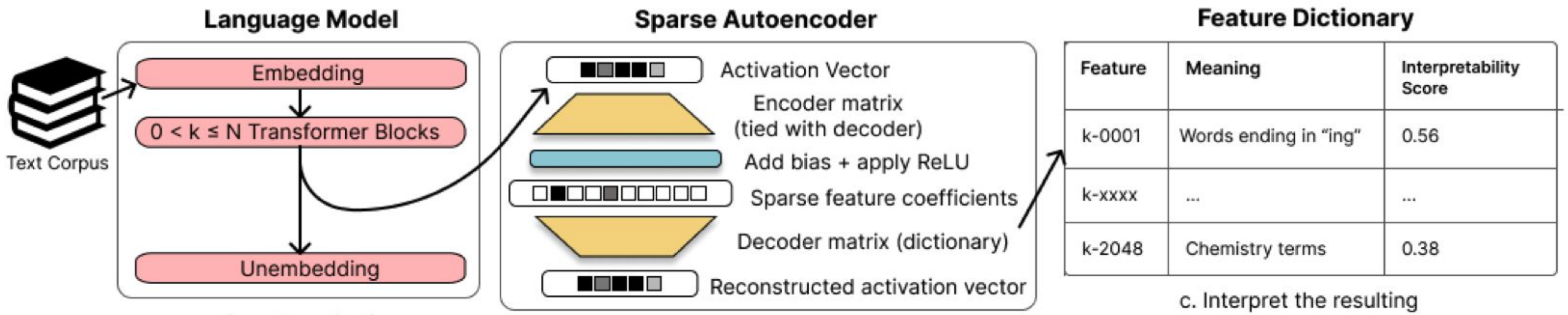
- ➔ Sparse autoencoders are trained to reconstruct their input data
- ➔ Trains on a combination of reconstruction loss and L1 loss which induces sparsity
- ➔ This sparsity is important for finding the decomposable feature representations, thus reducing feature confounding
- ➔ We can train SAEs on lots of unsupervised text data and find multiple features at once, which addresses the scalability issue

https://www.lesswrong.com/posts/CJPqwXoFtgkKPRay8/an-intuitive-explanation-of-sparse-autoencoders-for

COLUMBIA

SAEs can find highly interpretable features

- ➔ Use auto-interpretability for learning what each feature corresponds to
  - ◆ Find out which inputs activate the feature highly and use an LLM to understand the common theme between all inputs
- ➔ Learn a dictionary of {SAE feature: steering vector} using the decoder matrix


Language Model
Text Corpus
Embedding
0 < k ≤ N Transformer Blocks
Unembedding
Sparse Autoencoder
Activation Vector
Encoder matrix (tied with decoder)
Add bias + apply ReLU
Sparse feature coefficients
Decoder matrix (dictionary)
Reconstructed activation vector
Feature Dictionary

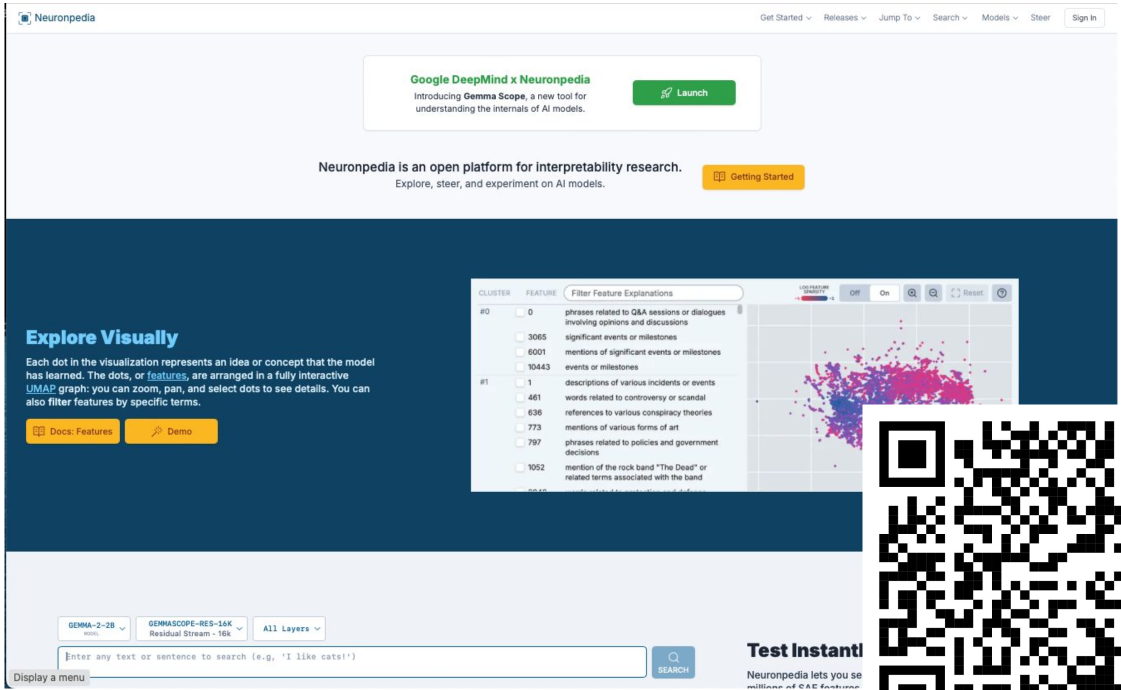
Feature	Meaning	Interpretability Score
k-0001	Words ending in "ing"	0.56
k-xxxx	...	...
k-2048	Chemistry terms	0.38

a. Sample activations from a language model

b. Learn a feature dictionary using an autoencoder that learns to represent activation vectors as a sparse linear combination of feature vectors.

c. Interpret the resulting dictionary features

Gemma Scope + Neuropedia API



Google DeepMind x Neuronpedia

Introducing **Gemma Scope**, a new tool for understanding the internals of AI models.

Neuronpedia is an open platform for interpretability research.

Explore, steer, and experiment on AI models.

Explore Visually

Each dot in the visualization represents an idea or concept that the model has learned. The dots, or features, are arranged in a fully interactive UMAP graph: you can zoom, pan, and select dots to see details. You can also **filter** features by specific terms.

CLUSTER	FEATURE	Filter Feature Explanations
#0	0	phrases related to Q&A sessions or dialogues involving opinions and discussions
	3065	significant events or milestones
	6001	mentions of significant events or milestones
	10443	events or milestones
#1	1	descriptions of various incidents or events
	461	words related to controversy or scandal
	636	references to various conspiracy theories
	773	mentions of various forms of art
	797	phrases related to policies and government decisions
	1052	mention of the rock band "The Dead" or related terms associated with the band

Enter any text or sentence to search (e.g, 'I like cats!')

Test Instantl

Neuronpedia lets you se

https://www.neuronpedia.org

- ➔ **Gemma Scope**: an open suite of sparse autoencoders for Gemma 2 9B and 2B
- ➔ **Neuronpedia API**: Interpretability API for various open source LLMs

Demo Notebook

https://colab.research.google.com/drive/1g_3A5XrWYfO1rcvMiKCrAI_nCaVTcFs6?usp=sharing

Using Feature Directions

Guided Generation

➔ Other common forms of controlling or steering generation include fine-tuning and prompt engineering

	Data requirements	Robustness to inputs	Requires internal access to model
Prompt engineering/ In-context learning	Low	Low	No
Activation steering	Low	High	Yes
Fine-tuning	High	High	Yes

Guided Generation

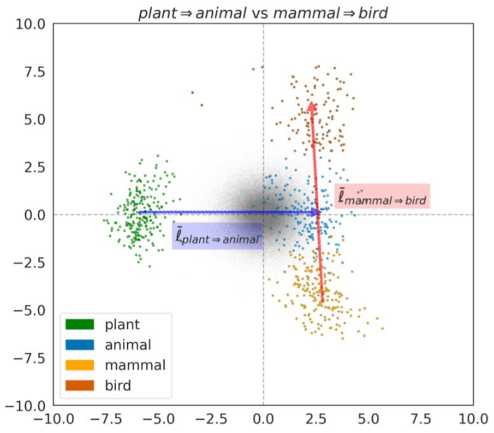
- ➔ Subramani et al. (2022) & Hernandez et al. (2023) employ steering vectors that are added to the forward pass of GPT-2 with the goal of modifying generation; the former found steering vectors like “love”, “hate”, etc. while the latter used steering vectors for fact-editing
- ➔ Merullo et al. (2023) observed the linearity of transformer representations; were able to find a steering vector for country capitals; the vector added to the residuals to convert Poland to Warsaw could be used to transform China into Beijing
- ➔ Elhage et al. (2022) showed that Othello-GPT can be intervened on with linear activation vectors that represented to, in essence, represent “a black piece is here and not a white piece”.

Guided Generation

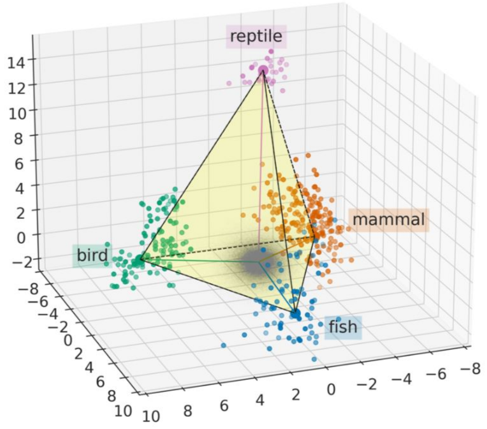
- Turner et al. (2023)
  - Found multiple linear steering vectors including “love” (attention layer 6), “intent to praise” (attention layer 6), “conspiracy” (attention layer 23), “want to die” (attention layer 10), “anger” (attention layer 20), and some odd ones like “talking about weddings” (attention layer 20), “Dragons live in Berkeley” (attention layer 15), and “the Eiffel tower is in Rome” (attention layer 24)
  - Found that adding embedding vectors isn’t as effective as adding steering vectors
- Park et al. (2023) found language concepts that allow translation (English⇒French, French⇒German, French⇒Spanish, and German⇒Spanish)

Structure within LLMs

Park, K., Choe, Y. J., Jiang, Y., & Veitch, V. (2024). **The Geometry of Categorical and Hierarchical Concepts in Large Language Models.**



(b) Hierarchy is encoded as orthogonality in Gemma.



(c) Categorical concepts are represented as simplices in Gemma.

Figure 1: In large language models, categorical concepts are represented as simplices in the representation space. Further, hierarchically related concepts (such as animal and mammal ⇒ bird) live in orthogonal subspaces. The top panel illustrates the structure, the bottom panels show the measured representation structure in the Gemma LLM. See Section 5 for details.

Structure within LLMs

Engels, J., Liao, I., Michaud, E. J., Gurnee, W., & Tegmark, M. (2024). **Not All Language Model Features Are Linear.**

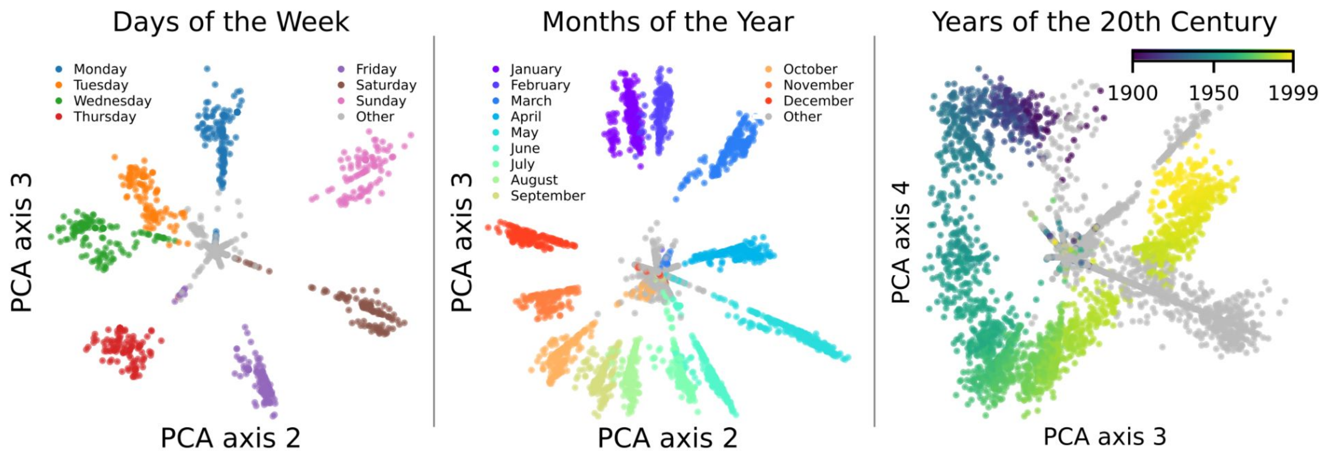


Figure 1: Circular representations of days of the week, months of the year, and years of the 20th century in layer 7 of GPT-2-small. These representations were discovered via clustering SAE dictionary elements, described in Section 4. Points are colored according to the token which created the representation. See Fig. 12 for other axes and Fig. 13 for similar plots for Mistral 7B.

Auditing LLMs

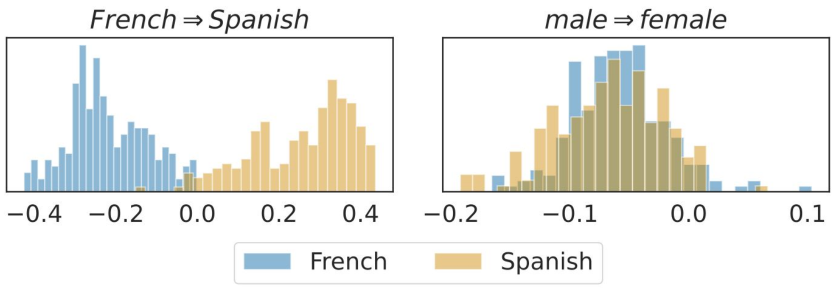
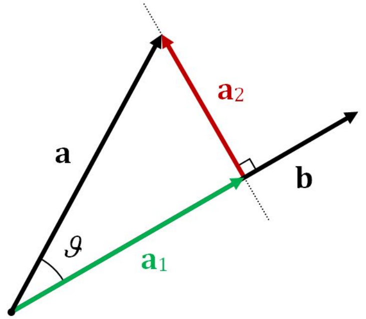


Figure 4. The subspace representation $\bar{\gamma}_W$ acts as a linear probe for W. The histograms show $\bar{\gamma}_W^\top \lambda(x_j^{\texttt{fr}})$ vs. $\bar{\gamma}_W^\top \lambda(x_j^{\texttt{es}})$ (left) and $\bar{\gamma}_Z^\top \lambda(x_j^{\texttt{fr}})$ vs. $\bar{\gamma}_Z^\top \lambda(x_j^{\texttt{es}})$ (right) for $W = \texttt{French} \Rightarrow \texttt{Spanish}$ and $Z = \texttt{male} \Rightarrow \texttt{female}$, where $\{x_j^{\texttt{fr}}\}$ and $\{x_j^{\texttt{es}}\}$ are random contexts from French and Spanish Wikipedia, respectively. We also see that $\bar{\gamma}_Z$ does *not* act as a linear probe for W, as expected.

- ➔ We can use these feature directions as linear probes
- ➔ We project the activations of our input onto the direction vector and look at the magnitude of the resulting projection

a = input
b = feature direction
|**a**$_1$| = magnitude of the feature present in our input



Park, K., Choe, Y. J., & Veitch, V. (2023). The linear representation hypothesis and the geometry of large language models.

Some Other Good Resources

➔ LessWrong blog: https://www.lesswrong.com/

➔ Transformer Circuits: https://transformer-circuits.pub/

➔ Neel Nanda’s blog: https://www.neelnanda.io/

Questions?

COLUMBIA

Appendix

Finding Steering Vectors with ActAdd

➔ Use a package like `TransformerLens` or `baukit`

```
from baukit import Trace

layer_id = 5
module = model.layers[layer_id]
with Trace(module) as ret:
     _ = model("Love")
     act_love = ret.output
     _ = model("Hate")
     act_hate = ret.output
steering_vec = act_love-act_hate
```