Linear Representation Hypothesis & Superposition

Senem Işık

Transformers

1. Attention layer
   - Building blocks to leverage connection with other tokens for good **embeddings**

2. MLP (Multi-layer perceptron)
   - Feed Forward
   - Building blocks to extract relevant features + information from the embeddings

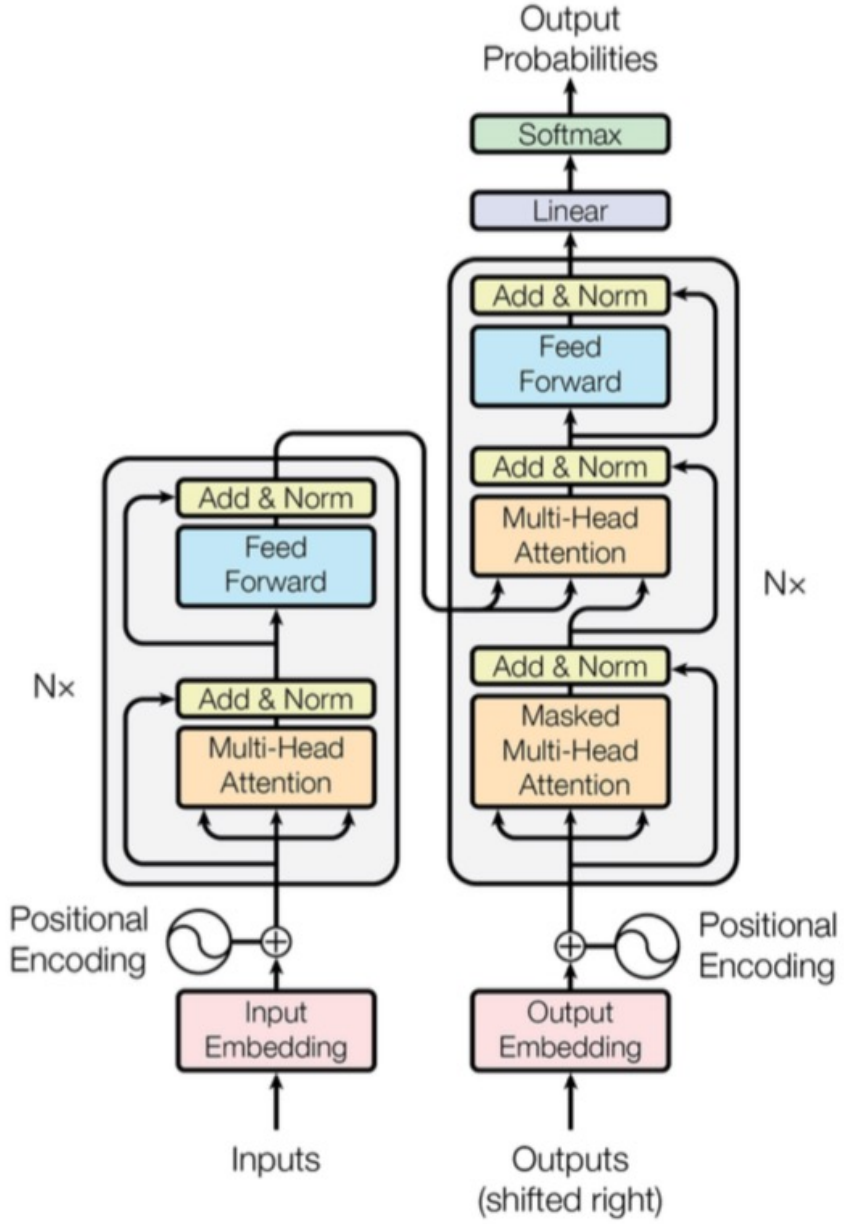
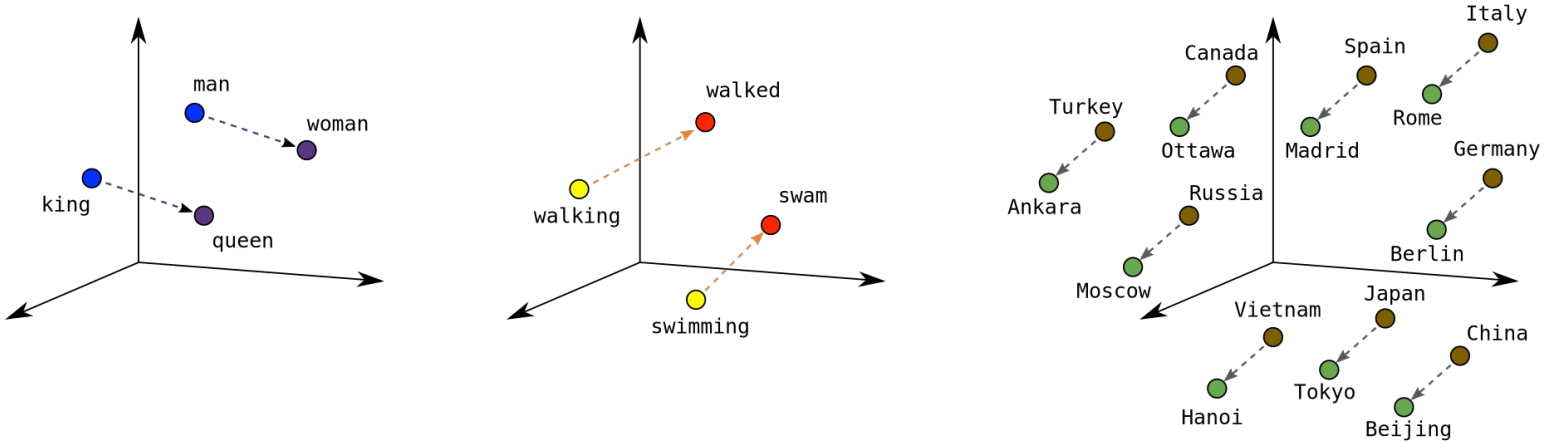


Figure 1: The Transformer - model architecture.

Linear Representation Hypothesis

LRH states that intermediate layers of language models store features linearly.



Male-Female Verb Tense Country-Capital

2-layer Perceptron

Input: Micheal Jordan …?
Embedding: x

$z_1 = W_1 x + b_1$

Is feature $W_1^{i,:}$ present? (Michael)
Is feature $W_1^{j,:}$ present? (Jordan)
Is feature $W_1^{k,:}$ present? (Michael Jordan)

$h = \phi(z_1)$ (say RELU)

Yes/No (yes: 1) (otherwise: 0)
(yes: 1)
(yes: 1)

$z_2 = W_2 h + b_2$

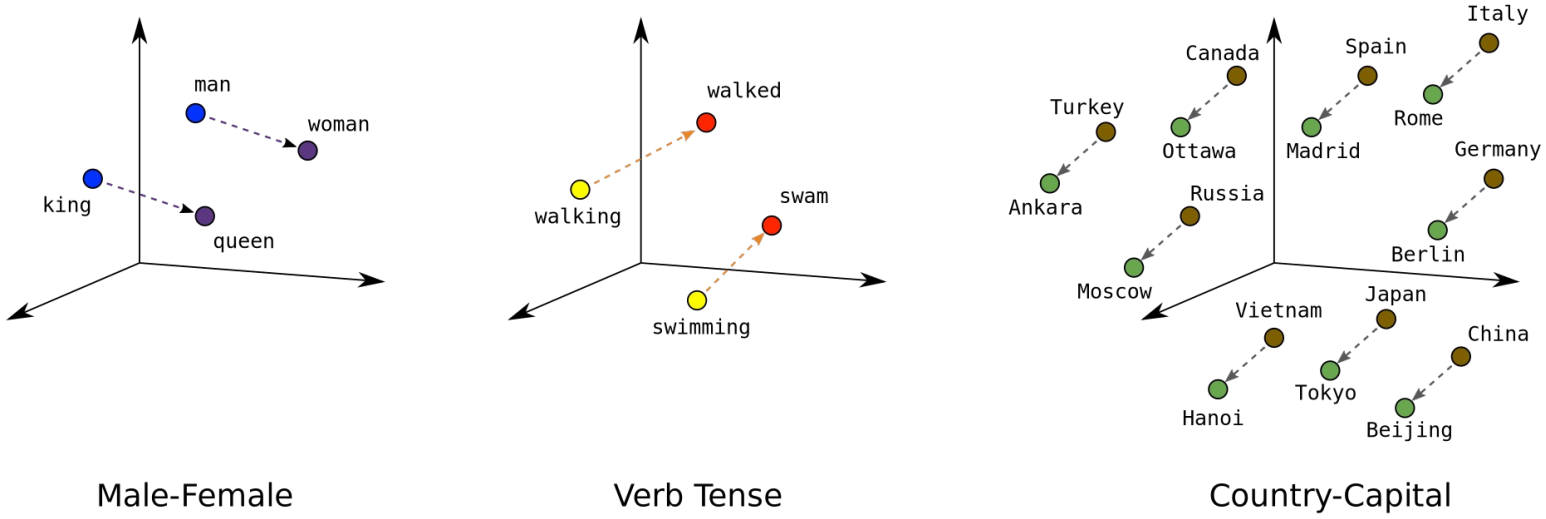
Get $W_2^{:,i} + W_1^{:,j} + W_1^{:,k}$ (plays basketball)

$y = x + z_2$

New embedding! (Michael Jordan plays basketball)

Linear Representation Hypothesis

LRH states that intermediate layers of language models store features linearly.



Male-Female Verb Tense Country-Capital

Features (e.g., the presence of cats or dogs) could increasingly be extracted using linear classifiers (**linear probes)** trained on activations, suggesting that deep neural networks work to arrange features linearly.

What if you cannot ask so many questions??? (just m Qs)

$$z_1 = W_1 x + b_1$$

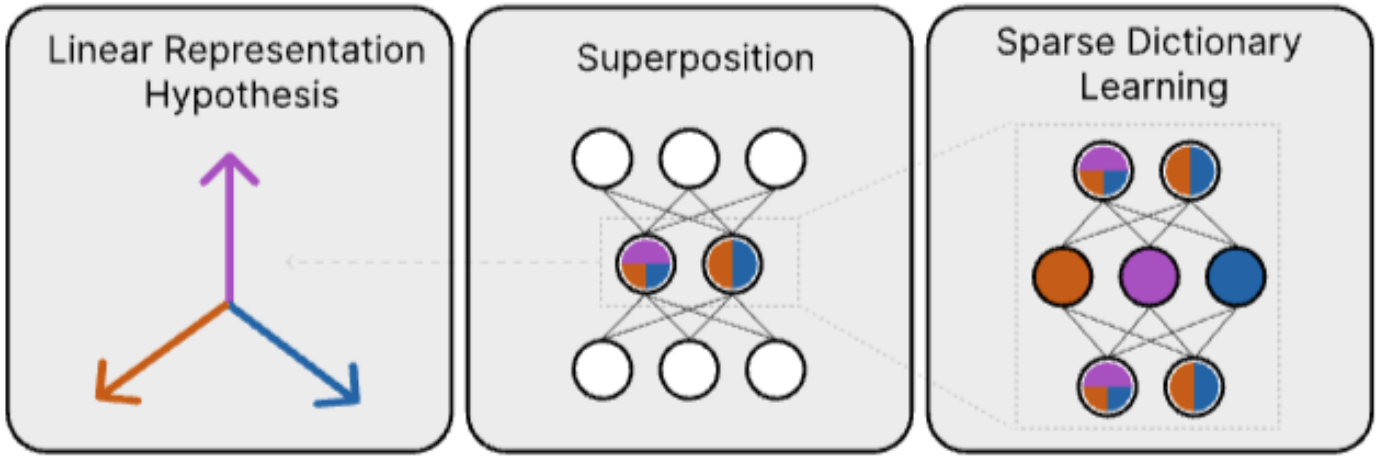
Is feature $W_1^{i,:}$ present? (Michael)

Is feature $W_1^{j,:}$ present? (Jordan)

Is feature $W_1^{k,:}$ present? (Michael Jordan)

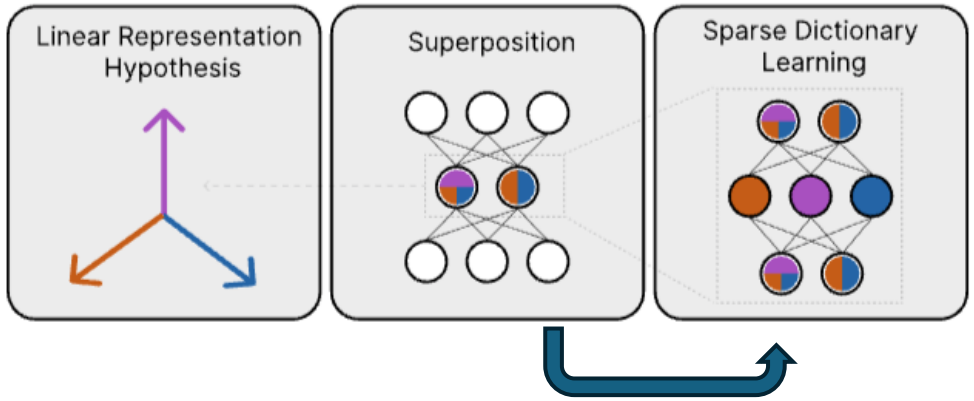
Superposition

d neurons can store m ≫ d features

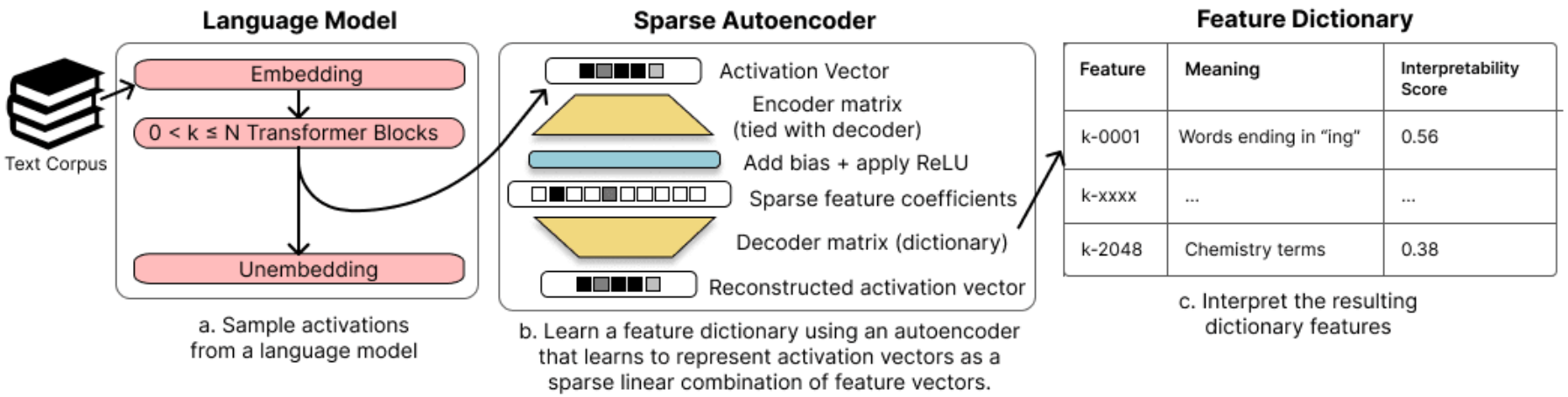


Interference

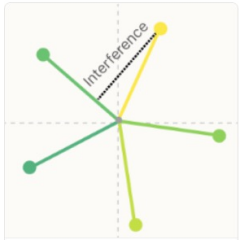
d neurons can store m ≫ d features



sparse autoencoders



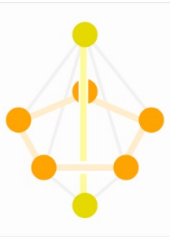
Toy Models of Superposition



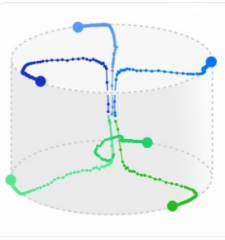
SECTION 1
Background & Motivation

SECTION 2
Demonstrating Superposition

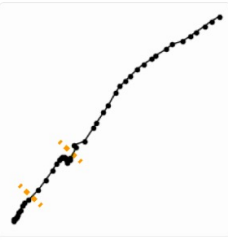
SECTION 3
Superposition as a Phase Change



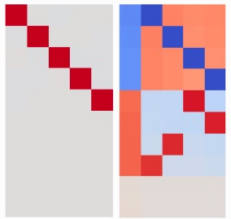
SECTION 4
The Geometry of Superposition



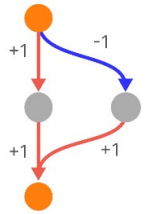
SECTION 5
Learning Dynamics



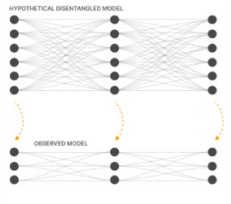
SECTION 6
Relationship to Adversarial Examples



SECTION 7
Superposition in a Privileged Basis



SECTION 8
Computation in Superposition



SECTION 9
The Strategic Picture

Discussion

Does this occur in real models?

Open Questions

SECTION 10
Discussion

Related Work

SECTION 11
Related Work

Comments & Replications

SECTION 12
Comments & Replications

AUTHORS

Nelson Elhage*, Tristan Hume*, Catherine Olsson*, Nicholas Schiefer*, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg*, Christopher Olah‡

AFFILIATIONS

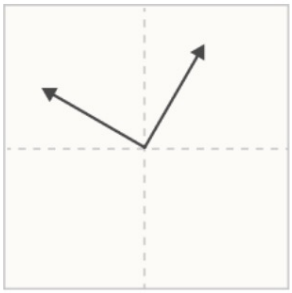
Anthropic, Harvard

PUBLISHED

Sept 14, 2022

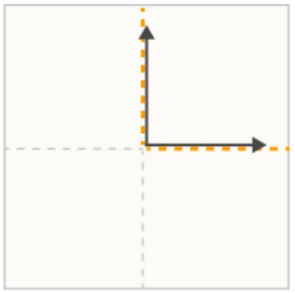
* Core Research Contributor; ‡ Correspondence to colah@anthropic.com; Author contributions statement below.

Prelims + recap



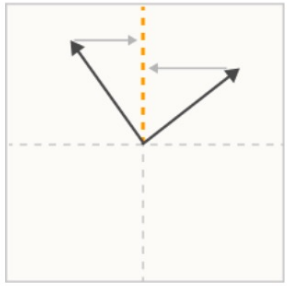
In a **non-privileged basis**, features can be embedded in any direction. There is no reason to expect basis dimensions to be special.

Examples: word embeddings, transformer residual stream

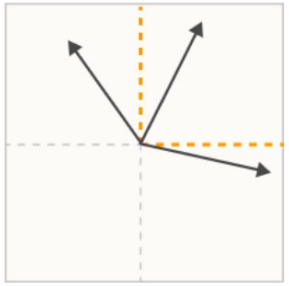


In a **privileged basis**, there is an incentive for features to align with basis dimensions. This doesn't necessarily mean they will.

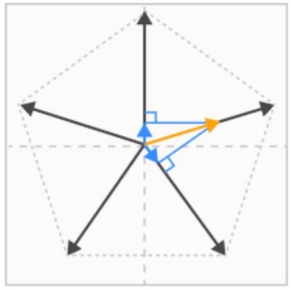
Examples: conv net neurons, transformer MLPs



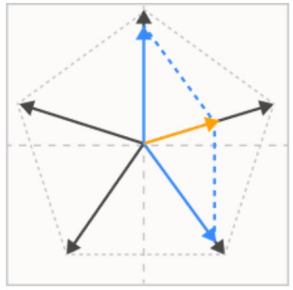
Polysemanticity is what we'd expect to observe if features were not aligned with a neuron, despite incentives to align with the privileged basis.



In the **superposition hypothesis**, features can't align with the basis because the model embeds more features than there are neurons. Polysemanticity is inevitable if this happens.



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**.

Setup

x_i is a feature (we're imagining features to be perfectly aligned with neurons)

- Feature Sparsity
  - For example, in vision, most positions in an image don't contain a horizontal edge, or a curve, or a dog head.
  - In language, most tokens don't refer to Martin Luther King or aren't part of a clause describing music.
- More Features Than Neurons
- Features Vary in Importance

$$x_i = \begin{cases} 0 & \text{w. p. } \mathrm{S_i} \\ \mathrm{U}[0,1] & \text{w. p. ty } 1 - \mathrm{S_i} \end{cases}$$

$\mathrm{S_i}$ is sparsity. Say all $\mathrm{S_i} = \mathrm{S}$ for all *I*

x_i *has importance* $\mathrm{I_i}$

Setup

$$L = \sum_x \sum_i I_i (x_i - x_i')^2$$

x_i is a feature (we're imagining features to be perfectly aligned with neurons)

Linear Model	**ReLU Output Model**
$h = Wx$	$h = Wx$
$x' = W^T h + b$	$x' = \mathrm{ReLU}(W^T h + b)$
$x' = W^T W x + b$	$x' = \mathrm{ReLU}(W^T W x + b)$

Recall that each column W_i corresponds to the direction in the lower-dimensional space that represents a feature x_i.

Setup

$$x' = \mathrm{ReLU}(W^T W x + b)$$

It tends to be easier to visualize $W^T W$ than W.

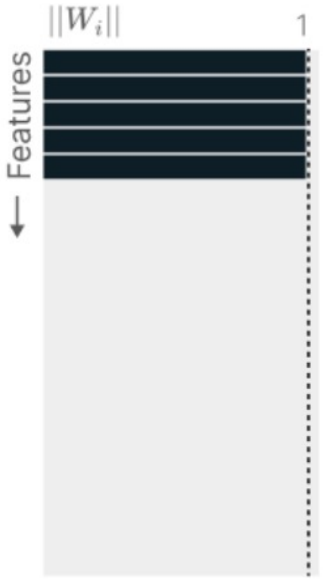
Here we see that $W^T W$ is an **identity matrix** for the most important features and **0** for less important ones.

We can also look at the bias, b.

The bias is **zero** for features learned to pass though, and the **expected value** (a positive number) for others.

Weight / Bias Element Values

-1 0 1



We want to understand which features the model chooses to represent in its hidden representation, and whether they're orthogonal to each other.

To do this, we visualize the norm of each feature's direction vector, $||W_i||$. This will be ~1 if a feature is fully represented, and zero if it is not. For each feature, we also use color to visualize whether it is orthogonal to other features (i.e. in superposition).

This model simply dedicates one dimension to each of the most important features, representing them orthogonally.

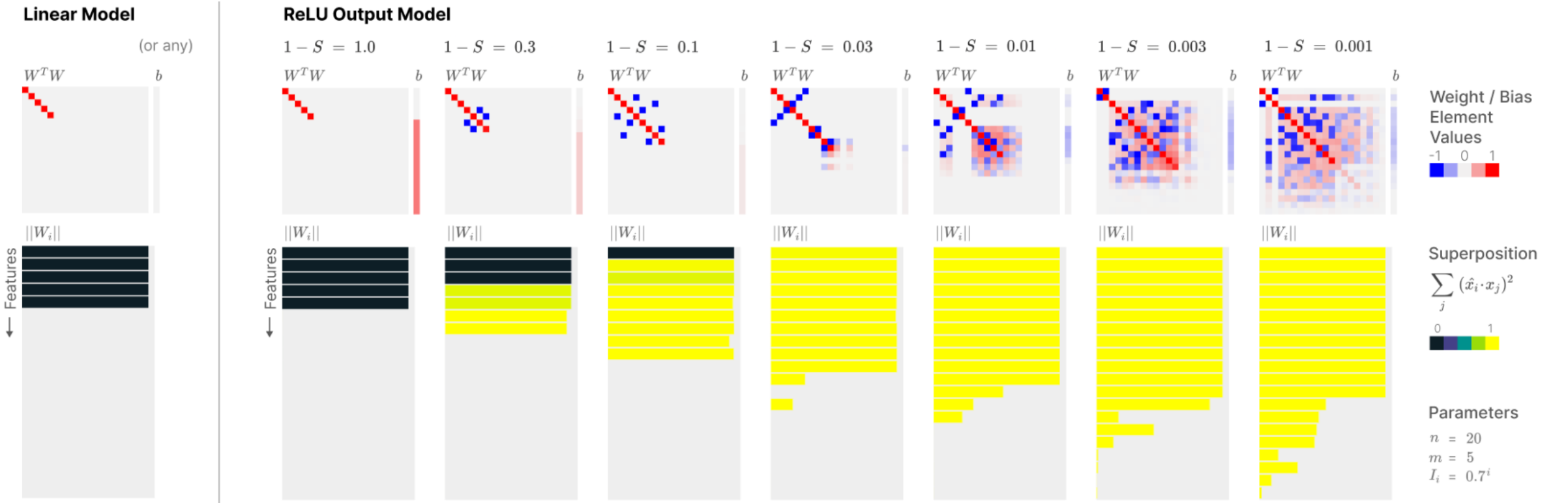
Superposition

$$\sum_j (\hat{x}_i \cdot x_j)^2$$

0 1

Results

$$x' = \mathrm{ReLU}(W^T W x + b)$$



Linear models learn the top m features. $1 - S$ = 0.001 is shown, but others are similar.

In the **dense** regime, ReLU output models also learn the top m features.

As **sparsity increases**, superposition allows models to represent more features. The most important features are initially untouched. This early superposition is organized in antipodal pairs (more on this later).

In the **high sparsity** regime, models put all features in superposition, and continue packing more. Note that at this point we begin to see positive interference and negative biases. We'll talk about this more later.

n features
m hidden

Mathematical understanding

Linear loss

$$L \sim \sum_i I_i(1 - ||W_i||^2)^2 + \sum_{i \neq j} I_j(W_j \cdot W_i)^2$$

Feature benefit is the value a model attains from representing a feature. In a real neural network, this would be analagous to the potential of a feature to improve predictions if represented accurately.

Interference betwen x_i and x_j occurs when two features are embedded non-orthogonally and, as a result, affect each other's predictions. This prevents superposition in linear models.

RELU loss

$$L_1 = \sum_i \int_{0 \leq x_i \leq 1} I_i(x_i - \mathrm{ReLU}(||W_i||^2 x_i + b_i))^2 + \sum_{i \neq j} \int_{0 \leq x_i \leq 1} I_j \mathrm{ReLU}(W_j \cdot W_i x_i + b_j)^2$$

If we focus on the case $x_i = 1$, we get something which looks even more analagous to the linear case:

$$= \sum_i I_i(1 - \mathrm{ReLU}(||W_i||^2 + b_i))^2 + \sum_{i \neq j} I_j \mathrm{ReLU}(W_j \cdot W_i + b_j)^2$$

Feature benefit is similar to before. Note that ReLU never makes things worse, and that the bias can help when the model doesn't represent a feature by taking on the expected value.

Interference is similar to before but ReLU means that negative interference, or interference where a negative bias pushes it below zero, is "free" in the 1-sparse case.

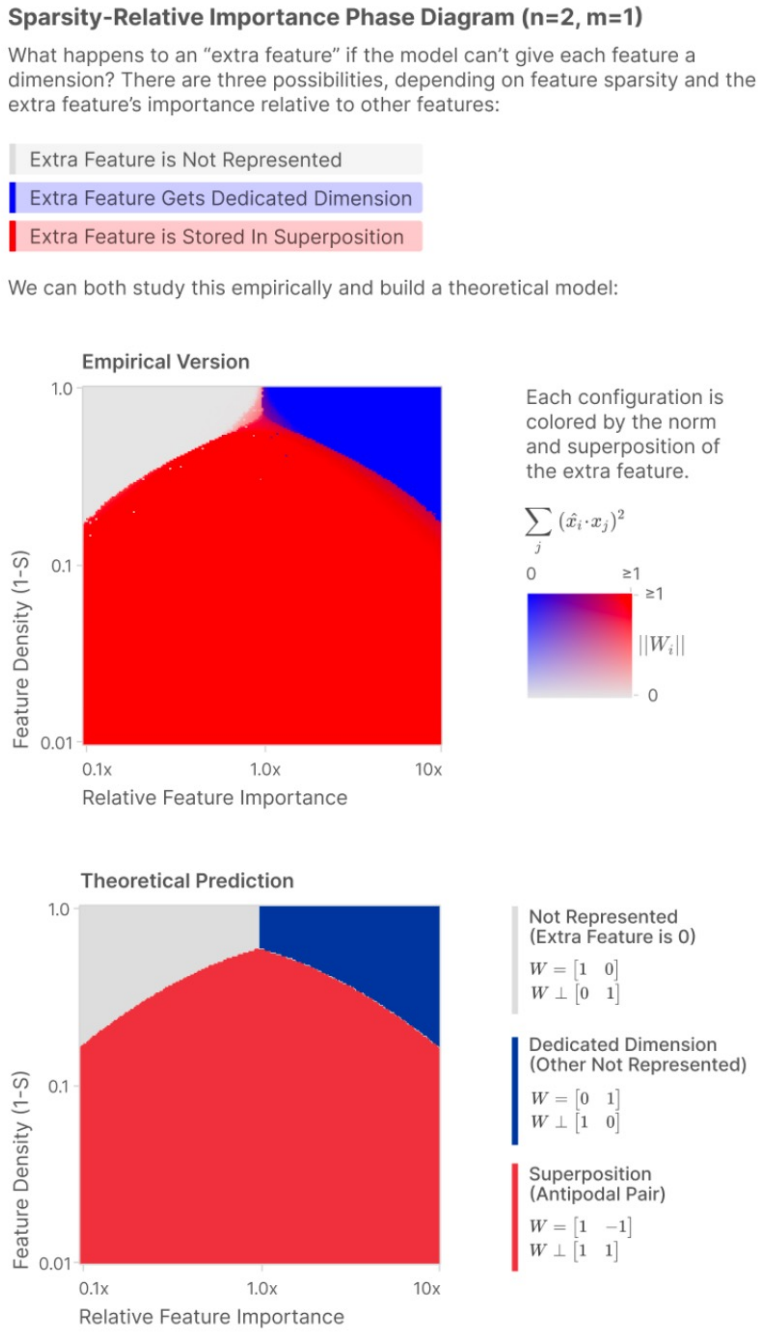
Outcomes

(1) the feature may simply not be learned

(2) the feature may be learned, and represented in superposition

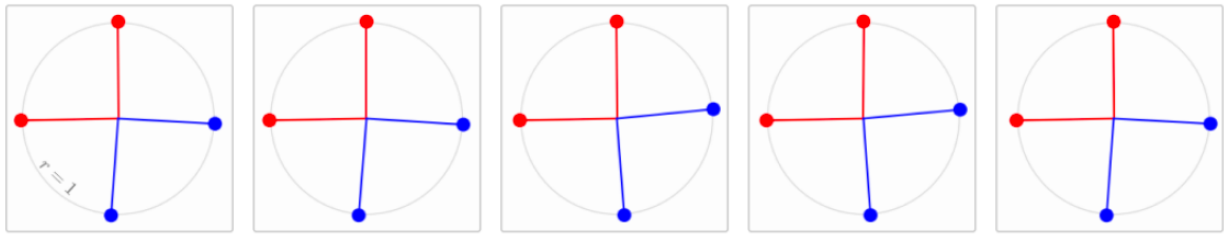
(3) the model may represent a feature with a dedicated dimension

Say 2 features but 1 hidden dimension



Models prefer to represent correlated features in orthogonal dimensions.

We train several models with 2 sets of 2 correlated features (n=4 total) and a m=2 hidden dimensions. We then visualize the weight column for each feature. For ease of comparison, we rotate and flip solutions to have a consistent orientation.

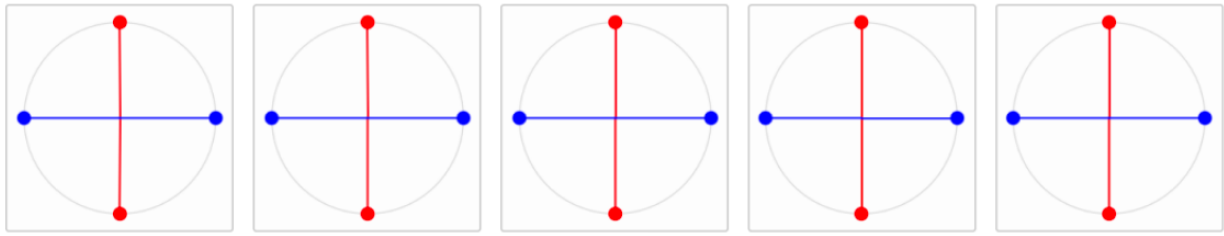


●● and ●● denote **correlated** feature sets.

Correlated feature sets are constructed by having them always co-occur (ie. be zero or not) at the same time.

Models prefer to represent anticorrelated features in opposite directions.

We train several models with 2 sets of 2 anticorrelated features (n=4 total) and a m=2 hidden dimensions. We then visualize the weight column for each feature. For ease of comparison, we rotate and flip solutions to have a consistent orientation.

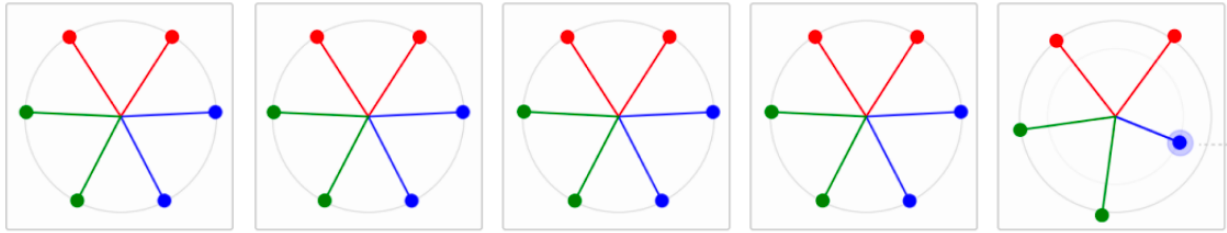


●● and ●● denote **anticorrelated** feature sets.

Anticorrelated feature sets are constructed by having them never co-occur (ie. be zero or not) at the same time.

Models prefer to arrange correlated features side by side if they can't be orthogonal.

We train several models with 3 sets of 2 correlated features (n=6 total) and a m=2 hidden dimensions. We then visualize the weight column for each feature. For ease of comparison, we rotate and flip solutions to have a consistent orientation. (Note that models will not embed 6 independent features as a hexagon like this.)



●●, ●●, and ●● denote **correlated** feature sets.

Sometimes correlated feature sets "collapse". In this case it's an optimization failure, but we'll return to it shortly as an important phennomenon.

PCA ↔ Superposition

- **Experiment:** Six features were grouped into three correlated pairs, with each pair always activating together while their non-zero values remained independent.
- Sparse features produce superposition patterns, but as features become denser, representations collapse into principal components, eventually behaving like PCA.

How many features can one layer of d neurons *store*?

There is a lack of clarity around…

How Many Features Can a Language Model Store Under the Linear Representation Hypothesis?

Nikhil Garg Jon Kleinberg Kenny Peng

Cornell University

represented linearly ≈ accessible linearly

Linear representation → model behavior can be altered/steered by modifying activation space along linear directions

Linear accessibility → features can be extracted with linear probes

+ next layer of a model can use it!

Mathematical Framework

Activations

$$f: L \to \mathbb{R}^d$$

Features

$$z_i: L \to \mathbb{R}$$

Feature accessibility. A feature z_i is (ϵ, S)**-recovered** by a **probe** $g_i : \mathbb{R}^d \to \mathbb{R}$ if

$$|g_i(f(\ell)) - z_i(\ell)| < \epsilon \tag{1}$$

for all $\ell \in S$. Intuitively, this means that the value of a feature can be accessed from a text's activations $f(\ell)$, at least up to some approximation error.

LRH formalized

Activations $f: L \to \mathbb{R}^d$

Features $z_i: L \to \mathbb{R}$

Linear representation for a set of features $z_1, \cdots, z_m : L \to \mathbb{R}$ implies that there exists a set of **representation vectors** $a_1, \cdots, a_m \in \mathbb{R}^d$, such that

$$f(\ell) = \sum_{i=1}^{m} z_i(\ell) a_i. \tag{2}$$

Equivalently, there exists $A \in \mathbb{R}^{d \times m}$ such that

$$f(\ell) = Az(\ell), \tag{3}$$

$f(z) = Az$

where $z(\ell) = [z_1(\ell), z_2(\ell), \cdots, z_m(\ell)] \in \mathbb{R}^m$. We refer to $z(\ell)$ as a text ℓ's **feature representation**. Under linear representation, activations are a function only of an input's feature representation.

Linear access formalized

Activations $f: L \to \mathbb{R}^d$

Features $z_i: L \to \mathbb{R}$

Feature accessibility. A feature z_i is (ϵ, S)**-recovered** by a **probe** $g_i : \mathbb{R}^d \to \mathbb{R}$ if

$$|g_i(f(\ell)) - z_i(\ell)| < \epsilon \tag{1}$$

for all $\ell \in S$. Intuitively, this means that the value of a feature can be accessed from a text's activations $f(\ell)$, at least up to some approximation error.

$g_i(x) = \langle b_i, x \rangle$ for some $b_i \in \mathbb{R}^d$

If all features are recoverable, there exists

$B = [b_1\ b_2 \ldots b_m]$ such that

$$\left|B^{\mathrm{T}} f(\ell) - z(\ell)\right| < \varepsilon$$

Questions

There is a quantitative gap in the answer to (1) and (2)

Q1. (General accessibility.) For a fixed m, how many dimensions d are required for there to exist $A \in \mathbb{R}^{d \times m}$ and $g : \mathbb{R}^d \to \mathbb{R}^m$ such that

$$\|g(Az) - z\|_\infty < \epsilon \tag{7}$$

for all $z \in S$.

Q2. (Linear accessibility.) For a fixed m, how many dimensions d are required for there to exist $A, B \in \mathbb{R}^{d \times m}$ such that

$$\left\|B^T Az - z\right\|_\infty < \epsilon \tag{8}$$

for all $z \in S$.

In both settings, we will generally assume S to be the set of k-sparse vectors (i.e., having at most k non-zero entries), and answer in terms of m, k, ϵ. In other words, we study settings where only a small number of features tend to be "active" for any particular input (intuitively, this is matches our intuition about language—that any given piece of text only expresses a small fraction of possible concepts).

Past Results: Compressed Sensing

Theorem 1 (Compressed Sensing). *There exists a matrix* $A \in \mathbb{R}^{d \times m}$ *with* $d = O\left(k \log \frac{m}{k}\right)$ *such that for all* k*-sparse* $z \in \mathbb{R}^m$,

$$g(x) := \operatorname{argmin}_{z' \in \mathbb{R}^m : x = Az'} \left\|z'\right\|_1 \tag{9}$$

satisfies

$$g(Az) = z. \tag{10}$$

Non-linear g

New Results: Linear Compressed Sensing

Let $d(m, k, \epsilon)$ be the smallest choice of d such that there exists $A, B \in \mathbb{R}^{d \times m}$ such that

$$\left\|B^T A z - z\right\|_\infty < \epsilon \tag{12}$$

for all k-sparse $z \in [-1, 1]^m$. (Here, any bounded interval would suffice.)

Theorem 2 (Upper Bound).

$$d(m, k, \epsilon) = O_\epsilon\left(k^2 \log m\right). \tag{13}$$

superposition

Theorem 3 (Lower Bound). *For* $\epsilon > \frac{k^{3/2}\sqrt{5}}{\sqrt{m}}$,

$$d(m, k, \epsilon) = \Omega_\epsilon\left(\frac{k^2}{\log k} \log \frac{m}{k}\right). \tag{14}$$

Intuition

$z :=$ features

Let $a_1, \ldots, a_m \in \mathbb{R}^d$ be the columns of A (representation vectors)

Let $b_1, \ldots, b_m \in \mathbb{R}^d$ be the rows of B (probe vectors)

$$\hat{z} = B^{\mathrm{T}} A z$$

$$\|\hat{z} - z\|_\infty < \epsilon \iff |\hat{z}_i - z_i| < \epsilon, \quad \forall i \in [m].$$

$$\hat{z}_i = \langle b_i, Az \rangle = \sum_{j=1}^{m} z_j \langle b_i, a_j \rangle.$$

Interference

Thank you ☺