# 8
# *The Physics of Representations: Superposition*

Some previous chapters (e.g., Chapter 5) viewed learning through the lens of a **Concept Class**. We assumed a fixed task ("is the input in the concept class?") and asked if a hypothesis class $\mathcal{H}$ was rich enough to contain the target function that solves the task.

This conceptual framework does not suffice for **Foundation Models**. These systems do not learn a single concept; they learn all the knowledge from a vast dataset, which we think of as learning a "World Model."

In this chapter, we treat the terms **concept** and **feature** interchangeably. A concept (e.g., "The Eiffel Tower" or "Recursive Logic") is an abstract idea in the world; a feature is the corresponding axis $x_i$ in the high-dimensional theoretical space $\mathbb{R}^N$. You can think of it as activation vector in a layer of the LLM.

The fundamental resource constraint is that a network with width $d$ must represent $N$ features/concepts, where often $N \gg d$. This chapter explores the internal physics of this compression. The core mechanism is **Superposition**: a geometric strategy where neural networks exploit high-dimensional geometry to defy the chaos that one would naively expect.

## *8.1 From Single Tasks to World Models*

The defining characteristic of foundation models is **polysemanticity**. In small, specialized networks, neurons often align with specific concepts (e.g., a "curve detector"). In large language models, neurons are routinely polysemantic: a single unit might activate for "jazz music," "syntax errors," and "finance."

### *8.1.1 The Entanglement Puzzle*

Historically, mixed selectivity appeared to be a failure to "disentangle" latent factors. Ideally, we want a basis where each neuron

represents one independent factor of variation. However, if the world is sparse (few features active at once), the optimal representation is **overcomplete**: there should be more basis vectors than dimensions.

### 8.1.2 *Superposition as Compression*

We view polysemanticity not as a bug, but as a compression artifact.

- **The Goal:** Represent $N$ features $x_1, \dots, x_N$.
- **The Constraint:** Store them in $\mathbb{R}^d$ where $d \ll N$.
- **The Solution:** Assign each feature a direction $W_i \in \mathbb{R}^d$. Since $N > d$, these directions cannot be orthogonal. They must overlap (i.e. have a nonzero inner product).

When feature $x_i$ activates, this activates direction $W_i$ in the layer's output. This creates "interference" on any feature $W_j$ with a non-zero dot product with $W_i$. This chapter studies how such interference can remain manageable.

## 8.2 *The Precursor: Classical Sparse Coding*

To understand how deep networks achieve this, we look to the 1990s. Olshausen and Field [1] proposed that the primary visual cortex maximizes the sparsity of the representation.

[1] Emergence of simple-cell receptive field properties by learning a sparse code for natural images. B.A. Olshausen and D.J. Fields, *Nature* 1996.

### 8.2.1 *The Inverse Problem*

Let $I \in \mathbb{R}^d$ be an observed signal (e.g., an image patch). We assume $I$ is generated by a linear combination of dictionary atoms $\Phi \in \mathbb{R}^{d \times N}$ with coefficients $a \in \mathbb{R}^N$:

$$I = \Phi a + \epsilon \tag{8.1}$$

where $N \gg d$. Here, the columns of $\Phi$ represent the **concepts** available to describe the world. Since there are more concepts ($N$) than observed dimensions ($d$), the system is underdetermined.

A foundational result in Compressed Sensing (Candès, Romberg, Tao [2]) is that we do not need to carefully engineer these concepts to be distinguishable. Random matrices work remarkably well. If $\Phi$ consists of random Gaussian vectors, it satisfies so-called **Restricted Isometry Property (RIP)** with high probability, allowing for perfect recovery of the coefficients $a$ provided the signal is sufficiently sparse.

[2] Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. E. Candès, J. Romberg, and T. Tao, *IEEE Trans. Inf. Theory* 2006.

To recover $a$, we impose a **sparsity prior**: the scene should be describable by very few active concepts. This leads to the Lasso objective (Basis Pursuit Denoising):

$$\min_a \frac{1}{2}\|I - \Phi a\|_2^2 + \lambda \|a\|_1 \tag{8.2}$$

### 8.2.2 Inference via ISTA

Finding the code $a^*$ requires solving an optimization problem *for every input*. A standard algorithm is Iterative Shrinkage-Thresholding (ISTA). The update rule is:

$$a_{t+1} = \mathcal{S}_{\lambda\eta}\left(a_t - \eta\Phi^T(\Phi a_t - I)\right) \tag{8.3}$$

where $\mathcal{S}_\theta(x) = \text{sign}(x)\max(|x|-\theta, 0)$ is the soft-thresholding operator. Notice the structure:

1. Compute residual error ($\Phi a_t - I$).
2. Project error back to code space via $\Phi^T$.
3. Apply a non-linearity ($\mathcal{S}$).

### 8.2.3 Amortized Inference

Gregor and LeCun [3] observed that unfolding the ISTA iterations for $k$ steps resembles a $k$-layer neural network. Crucially, a standard feedforward layer $h = \text{ReLU}(Wx)$ can be viewed as a **single step of amortized sparse coding**. The ReLU is simply a one-sided soft-thresholding operator. The weight matrix $W$ learns to approximate the "inverse" of the dictionary $\Phi$.

[3] K. Gregor and Y. LeCun. Learning Fast Approximations of Sparse Coding, ICML2010.

Note that deep networks do not need to solve the optimization problem at runtime; they learn a direct mapping that approximates the solution.

## 8.3 The Geometry of High Dimensions

Allowing neural nets to work with superposition of a large number of concepts relies on the counter-intuitive geometry of high-dimensional spaces.

### 8.3.1 The Orthogonality Limit

In $\mathbb{R}^d$, there are at most $d$ mutually orthogonal unit vectors. If we require features to be perfectly distinct, $N \leq d$.

### 8.3.2 Almost Orthogonality

Neural networks are robust to noise. We require only that interference is below a threshold $\epsilon$: $|\langle W_i, W_j\rangle| \leq \epsilon$ for $i \neq j$. The **Johnson-Lindenstrauss (JL) Lemma** states that for any set of $m$ points, we can project them into $d = O(\epsilon^{-2}\log m)$ dimensions while preserving distances. Inverting this: in dimension $d$, we can pack $m \approx \exp(d\epsilon^2)$

almost-orthogonal vectors, each of which can represent a concept. Increasing layer width $d$ linearly allows exponentially more concepts.

## 8.4 The Mechanism: ReLU Dynamics

The JL Lemma provides the *capacity*, but we must explain the *efficiency*. If the network had to disentangle $N$ features to process them, it would require $O(N)$ compute, which is prohibitive.

We use the toy model from Elhage et al. [4]. Let $x \in \mathbb{R}^N$ be a sparse input vector. The network stores it as a linear superposition in the residual stream:

[4] Toy models of Superposition. Elhage et al. 2022

$$h = \sum_{k=1}^{N} x_k W_k \tag{8.4}$$

where $W_k \in \mathbb{R}^d$ is the direction representing feature $k$.

### 8.4.1 The Static Query Heuristic

How does a downstream neuron check for the presence of feature $i$? It does not run a complex optimization to "explain away" the other features. It uses a **fixed** readout direction $u_i$ (typically aligned with $W_i$) and a learned bias $b$.

The neuron performs a simple $O(d)$ projection:

$$y_i = \langle u_i, h \rangle + b \tag{8.5}$$

The computational cost depends only on the dimension $d$, not the total number of concepts $N$. The network applies this static filter regardless of what other features are active.

### 8.4.2 Bias as a High-Pass Filter

We can analyze the effectiveness of this static query by expanding $h$:

$$y_i = \underbrace{x_i \langle u_i, W_i \rangle}_{\text{Signal}} + \underbrace{\sum_{j \neq i} x_j \langle u_i, W_j \rangle}_{\text{Interference}} + b \tag{8.6}$$

The "Chicken and Egg" paradox (how do we know the interference without knowing which $x_j$ are active?) is resolved by treating the interference as random noise. The network does not solve for $x_j$; it simply tolerates the interference.

The ReLU activation $\max(0, y_i)$ functions as a noise gate. For the static query to work, the bias $b$ must be set such that:

$$\max(\text{Interference}) < -b < \min(\text{Signal}) \tag{8.7}$$

If the input is sufficiently sparse, the sum of random dot products (interference) remains low. The ReLU filters it out, recovering $x_i$

cleanly. If the sparsity constraint is violated (too many active features), the interference floor rises above $-b$, and the static query fails. The neuron fires on the noise, causing an **interference hallucination**.

## 8.5 Optimization and Phase Transitions

Gradient Descent (GD) does not find these configurations smoothly.

### 8.5.1 The Geometry of Solutions

Consider packing $N = 3$ features into $d = 2$.

- **Orthogonal (PCA):** Align two features with axes, ignore the third.
- **Mercedes-Benz:** Arrange vectors at 120°.

The Mercedes-Benz solution represents all features but accepts non-zero interference.

### 8.5.2 Phase Transitions

As we vary sparsity $S$, the model undergoes a **phase transition**.

1. **Memorization Phase:** High sparsity. Model learns orthogonal features.
2. **Superposition Phase:** Sparsity drops. Weights snap into a high-dimensional polytope (e.g., a tetrahedron).

This implies that "smooth" loss curves in LLMs actually aggregate thousands of discrete phase transitions.

## 8.6 Adversarial Geometry and the Search for Truth

Superposition introduces a vulnerability: **Feature Crosstalk**. The interference term $\sum_{j \neq i} x_j \langle W_i, W_j \rangle$ is usually small for random inputs. However, an adversary can select features $J$ where $\langle W_i, W_j \rangle > 0$, maximizing constructive interference.

$$\sum_{j \in J} \langle W_i, W_j \rangle > -b \tag{8.8}$$

This causes the model to "hallucinate" feature $i$.

### 8.6.1 Contrastive Consistency Search (CCS)

If representations are noisy superpositions, how do we extract "truth"? Burns et al. [5] propose a geometric solution called **Con-**

[5] C. Burns, H. Ye, D. Klein, and J. Steinhardt. *Discovering Latent Knowledge in Language Models Without Supervision.* ICLR 2023.

**trastive Consistency Search**. Given any statement $x$ found on the internet (e.g., "The capital of France is Paris"), we construct a statement that contradicts it (e.g., "The capital of France is Brussels") to act as a negative prompt $x^-$. We seek a direction $\theta \in \mathbb{R}^d$ in the model's activation space such that the projected probabilities are consistent:

$$p(x^+) + p(x^-) \approx 1 \tag{8.9}$$

where $p(x) = \sigma(\theta^T \phi(x))$. Importantly, the optimization does not require ground truth labels. It minimizes a **consistency loss** (probabilities sum to 1) and maximizes **confidence** (probabilities are near 0 or 1). Geometrically, CCS finds a hyperplane that separates logical negations, cutting through the superposition noise. This suggests that while individual neurons are polysemantic, linear subspaces may still encode robust truth values.

## 8.7 Unpacking the Black Box: Sparse Autoencoders

To decrypt the superposition, we must map the compressed state $h \in \mathbb{R}^d$ back to the sparse basis $\mathbb{R}^N$. This is the domain of **Sparse Autoencoders (SAEs)**.

### 8.7.1 Dictionary Learning Redux

We treat the activation $h$ as the signal and train an autoencoder to recover the "true" features $c \in \mathbb{R}^M$ ($M \gg d$).

$$c = \text{ReLU}(W_{enc}h + b_{enc}), \quad \widehat{h} = W_{dec}c + b_{dec} \tag{8.10}$$

Objective: $\|h - \widehat{h}\|^2 + \lambda \|c\|_1$. This is exactly the Lasso formulation from Section 8.2, but trained on the model's internals rather than raw data.

### 8.7.2 Top-K SAEs

The $L_1$ penalty causes "shrinkage" (suppressed activation magnitudes). A modern alternative is the **Top-K SAE**, which enforces sparsity directly:

$$c = \text{TopK}(W_{enc}h + b_{enc}, k) \tag{8.11}$$

This keeps only the $k$ largest activations, setting the rest to zero. This connects directly to the theoretical intuition that the concept class is $k$-sparse.

### 8.7.3 Empirical Results

- **Monosemanticity:** SAE features are often pure. A famous example is the "Golden Gate Bridge" feature in Claude 3; clamping it high compels the model to mention the bridge in unrelated contexts.

- **Dead Neurons:** A major optimization challenge. If a latent unit never activates, its gradient is zero.

SAEs support the chapter's hypothesis: the "messy" weights of deep networks are highly precise, compressed representations of a vast, sparse concept class. Of course, this does not preclude that locating more complicated concepts (e.g., validity of reasoning chains) within the model may require more complicated methods.

## 8.8 From Features to Circuits: The Algebra of Attention

We established that the residual stream $x \in \mathbb{R}^d$ carries a superposition of features. We now ask: how does the network manipulate these features? If features are the *nodes* of the computation, **Circuits** are the edges. In Transformers, these edges are formed by Attention Heads.

### 8.8.1 The Residual Stream as a Communication Channel

A Transformer layer does not overwrite the hidden state; it adds to it:

$$x_{l+1} = x_l + \text{Attn}(x_l) + \text{MLP}(x_l) \tag{8.12}$$

This implies the residual stream acts as a shared communication channel (or "blackboard"). Layers read from the stream, perform a computation, and write the result back. Mathematically, this allows us to decompose the network into end-to-end linear paths. A feature computed at Layer 1 can be read directly by Layer 10, bypassing intermediate processing.

### 8.8.2 OV and QK Circuits

Elhage et al. [6] decompose an attention head into two independent operations acting on the residual stream. Let $W_Q, W_K, W_V, W_O$ be the query, key, value, and output matrices of a head.

[6] A Mathematical Framework for Transformer Circuits. Elhage et al. *Transformer Circuits Thread*, 2021.

1. **The OV Circuit ($W_{OV} = W_O W_V$):** This matrix determines *what* information is moved. If a token attends to a source token, $W_{OV}$ describes the linear map from the source's residual stream to the destination's residual stream.
2. **The QK Circuit ($W_{QK} = W_Q^T W_K$):** This matrix determines *where* to move information. It acts as a bilinear form on the residual stream to compute attention scores: $A = \text{softmax}(x^T W_{QK} x)$.

If we understand the features (via SAEs), we can analyze $W_{OV}$ directly. If the "Golden Gate Bridge" feature is an eigenvector of

$W_{OV}$ with a large eigenvalue, that head specializes in preserving or amplifying that concept.

## 8.9 The Mechanism of In-Context Learning: Induction Heads

How does a model learn from context without weight updates? The primary mechanism appears to be the **Induction Head** [7].

[7] In-context Learning and Induction Heads. Olsson et al. 2022.

### 8.9.1 The Copying Problem

Consider the sequence: `[A] [B] ... [A]`. The model must predict `[B]`. To do this, it must:

1. Identify the current token is `[A]`.
2. Look back to the *previous* occurrence of `[A]`.
3. Shift attention one step forward to `[B]`.
4. Copy `[B]` to the current position.

This operation—"look at where I am now, find where I was before, and copy what came next"—is the definition of **Induction**.

### 8.9.2 Composition of Heads

A single attention head cannot perform induction because attention is computed on the *current* token. It requires information about the *previous* token. This necessitates **Head Composition**.

- **Head 1 (Previous Token Head):** Writes the content of token $t-1$ into the residual stream of token $t$.
- **Head 2 (Induction Head):** Uses the information written by Head 1 as a Query. It searches for a Key in the past that matches the current token.

### 8.9.3 K-Composition

Mathematically, this corresponds to the composition of the OV circuit of Layer $L_1$ with the QK circuit of Layer $L_2$:

$$W_{QK}^{eff} = W_Q^{L2}(W_{OV}^{L1})W_K^{L2} \tag{8.13}$$

If the subspace of "token identity" is preserved through this multiplication, the model implements a content-addressable memory lookup. Empirically, we observe a phase transition during training. The sudden emergence of induction heads (where $W_{QK}^{eff}$ acquires specific

singular values) correlates perfectly with the onset of the model's ability to reduce loss on in-context learning tasks.

This provides a mechanistic bridge: **Superposition** stores the features; **Induction Heads** manipulate them to simulate a learning algorithm at inference time.

## 8.10 Exercises

1. **[Interference Variance]** Assume $W_i$ are random unit vectors in $\mathbb{R}^d$ and inputs $x$ are $k$-sparse. Show that the variance of the interference term scales as $O(k/d)$.

2. **[The Mercedes-Benz Optima]** Let $N = 3, d = 2$. Show that the symmetric configuration ($120^\circ$ separation) is a local minimum of the interference loss $\sum_{i \neq j} \langle W_i, W_j \rangle^2$, while the orthogonal configuration is a saddle point.

3. **[CCS Geometry]** Let $v_{true}$ be the direction of truth. Let $x^+ = v_{true} + \delta$ and $x^- = -v_{true} + \delta$, where $\delta$ is noise. Show that a linear probe $\theta$ minimizing $(1 - (\sigma(\theta^T x^+) + \sigma(\theta^T x^-)))^2$ must align with $v_{true}$.

4. **[Rank of the Copy Circuit]** Consider a "pure copy" head where $W_{OV} = I$. Suppose the input $x$ contains a superposition of $k$ features. Show that for the head to copy a specific feature $x_i$ without distorting the others, $W_{OV}$ need not be the identity, but must act as the identity on the subspace spanned by the active features. Connect this to the concept of "privileged basis" in Section 8.4.

5. **[Induction Arithmetic]** Let the residual stream at position $t$ be $x_t$. Assume Layer 1 writes the previous token embedding: $x_t^{(1)} \leftarrow x_t + E_{t-1}$. Assume Layer 2 computes attention scores based on $x_t^{(1)}$. Derive the condition on $W_Q, W_K$ in Layer 2 such that attention is maximized at position $j$ if and only if $E_j = E_t$ (i.e., the token at $j$ is the same as the current token $t$). (Hint: Expand the bilinear form $(x_t + E_{t-1})^T W_Q^T W_K (x_j + E_{j-1})$).