



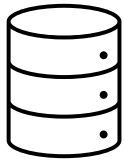
Mechanistic Interpretability

Explainable and Trustworthy AI

Gabriele Ciravegna

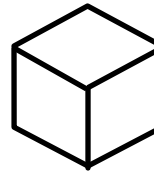
Stages of Explainability – Mechanistic

Explainability involves the entire AI development pipeline



Pre-modelling explainability

- Before building the model
- Data exploration
 - Data selection
 - Feature engineering



Explainable modeling

- Build inherently interpretable models
- Manage the accuracy and interpretability trade-off



Post-modelling explainability

- After model development
- Explaining predictions and behavior of trained models

Contents

1. Mechanistic Interpretability in CNN
 1. Features
 2. Circuits
 3. Universality
2. Mechanistic Interpretability in LLMs
 1. Analysis of Simplified Transformers
 2. The Logit Lens
 3. Sparse Dictionary Learning

Mechanistic Interpretability in CNN

How Neural Networks Globally Reason Internally?

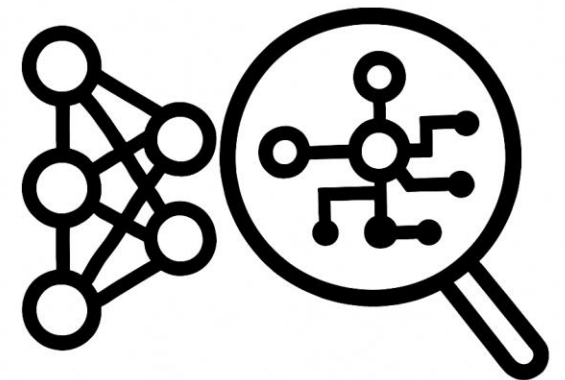
The decision making of NNs is seen often defined **indecipherable**:

- We are unable to interpret NNs inner workings

We can reveal meaningful patterns **zooming in** on:

- Individual neurons
- Their connections

Mechanistic Interpretability helps us break down NN decisions into **globally understandable structures**

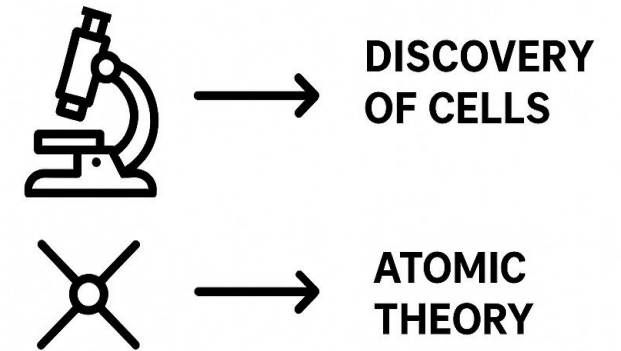


**MECHANISTIC
INTERPRETABILITY**

The power of “Zooming In”¹

Scientific progress is driven by the ability to zoom into finer details of a given field:

- Microscope --> Discovery of cells --> Cellular Biology
- X-ray cristallography --> DNA structure and molecules --> Atomic Theory

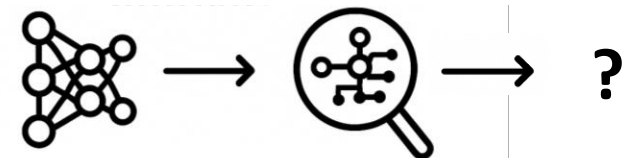


Why don't we try to do the same with NNs?

- NN visualization --> Computational circuits --> **NN interpretation?**

NNs often studied at a **macro level** but:

- Finer analysis may uncover universal processes within NNs



[1] <https://distill.pub/2020/circuits/zoom-in/>

Three Speculative Claims

1. Features

- NN representations are composed of **individual features** such as edges, textures, and object parts
- Each neuron (or combination of neurons) specializes in detecting specific characteristics

2. Circuits

- Neurons form **meaningful interactions**, creating circuits rather than working in isolation
- **Circuits connect features**, creating complex computations as shape recognition or object segmentation

3. Universality

- Similar **neurons and circuits** appear across **different architectures and tasks**
- Do certain structures emerge **naturally from training data?**

Feature Example - Curve Detectors

Neurons in vision models detect **curves and edges**, a crucial step in visual processing

Supporting evidence:

- **Feature visualization:** Visualizing neurons shows curve patterns
- **Dataset examples:** Neurons activate when encountering curve-related images
- **Synthetic testing:** Creating artificial inputs verifies neuron behavior



Feature Visualization²

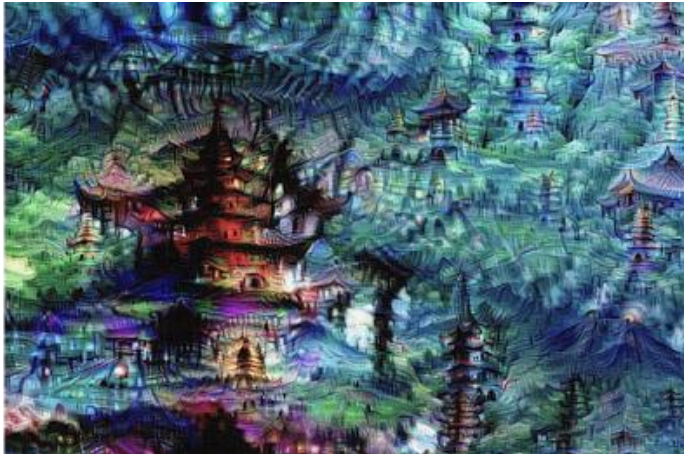


How?

- Optimizes an input image to strongly activate a specific neuron or layer
- Iteratively adjust the image until maximizing activation

What do we obtain?

- Generate synthetic representations of the features learnt by the network (neurons or layers)



Deep Dream

- Modifies an existing image to exaggerate the patterns that a NN detects
 - Instead of generating new images from scratch
- Artistic yet insightful look into how models perceive patterns and objects

Feature Example -Pose-Invariant Dog Head Detector

- Some neurons specialize in recognizing **dog heads from various angles**
- Networks generalize recognition by forming “**union over cases**”
- This allows models to **learn abstract concepts** beyond pixel patterns



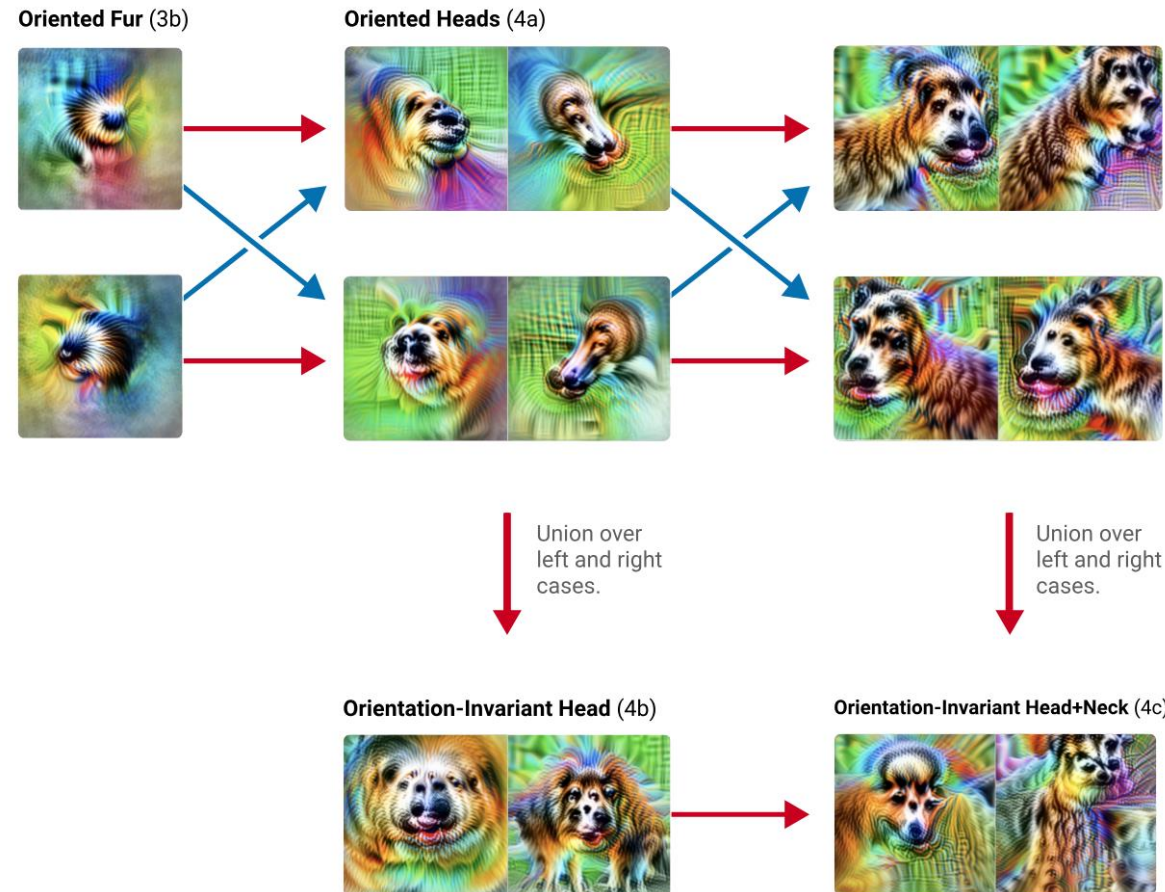
Circuits Motifs in Neural Networks

- Features are connected by weights, forming circuits
- Neural networks develop **consistent circuits**, known as **motifs**:
 - **Equivariance** – Rotation-invariant recognition.
 - **Unioning over cases** – Combining multiple perspectives.
 - **Superposition** – Using neurons efficiently to store information.

Circuit Example: Forming the Pose-Invariant Dog Head Detector

InceptionV1 has a **left-oriented** pathway detecting dogs facing left...

... and a symmetric **right-oriented** pathway detecting dogs facing right. At each step, the two pathways **inhibit** each other and **excite** the next stage.



Universality

NNs often develop similar features and circuits across **different models and architectures**

Why Universality Matters?

- Suggests neural networks may be converging toward **fundamental computational principles**
- Provides a foundation for **transfer learning**

Examples of Universality:

- **Edge and texture features** appear across all vision model
- **Curve detectors circuits** function similarly in different architectures

Curve detectors

ALEXNET

Krizhevsky et al. [34]



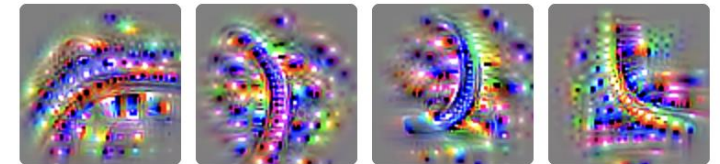
INCEPTIONV1

Szegedy et al. [26]



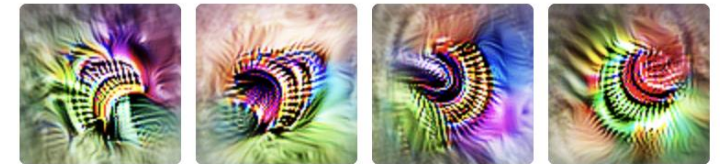
VGG19

Simonyan et al. [35]



RESNETV2-50

He et al. [36]



Challenges

Polisemanticity

- Some neurons respond **to multiple unrelated features**
- Complicates alignment to human decision-making
- **Superposition**: need to use neurons efficiently to store information



Required **human annotation to:**

- Inspect neurons
- Create visualizations
- Examine circuits



Dataset examples

Universality is not strictly required

- But if it does not hold future research can focus only on **individual models**

Mechanistic Interpretability in LLMs

Different Representation Structure

Lack of Spatial Structure

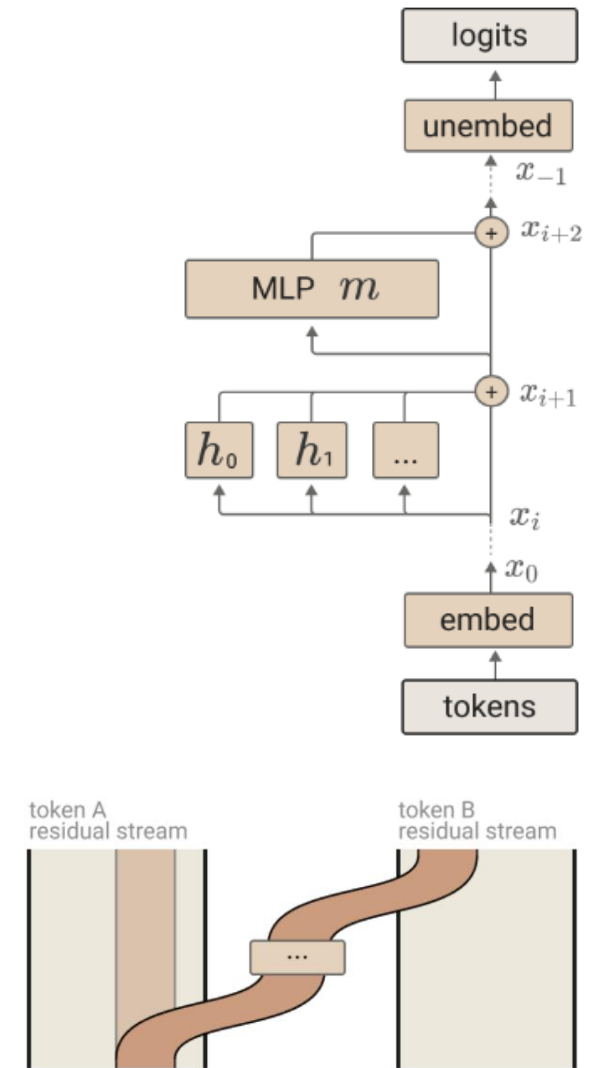
- Transformers process tokens in a sequence rather than pixels in a grid
 - > We do not have filters that learn patterns tied to specific locations

Dynamic, Complex Interactions

- Attention mechanisms dynamically shift focus based on context
- Transformers learn relationships between tokens
 - > Challenging to attribute a single neuron to a specific feature
 - > Challenging to visualize circuits

Reverse Engineering Transformers

- The Residual Stream
 - Each layer reads its input and writes its output on the *residual stream*
 - Deeper layers do not overwrite previous information but additively build upon it
- Attention heads are independent and additive
 - Attention heads move information from one token to the other (residual stream)
 - Their contribution can be considered quasi-linear and additive
 - only the attention score with the softmax is non linear



Analysis of Simplified Transformers³

- Zero-Layer Transformer

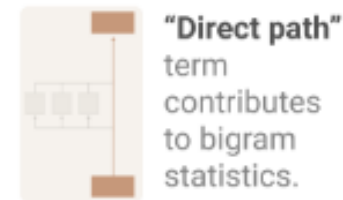
- No information moves across tokens

--> it predicts the next token purely based on bigram (i.e. «token pairs») statistics

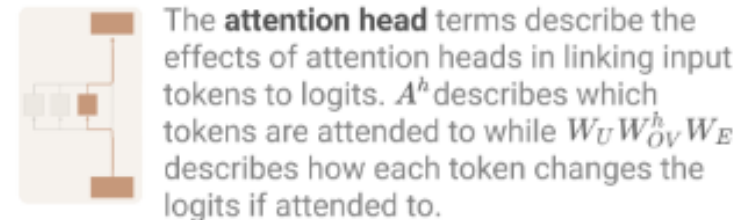
- One-Layer Transformer

- Keep bigram probabilities (from residual stream)
 - It introduces skip-trigrams
 - Patterns "A...BC": a token earlier in the sequence (A) influences a later prediction (C) despite being separated by (one or more) token B
 - Attention heads selectively attend to earlier tokens to modify predictions accordingly

$$T = \text{Id} \otimes W_U W_E$$

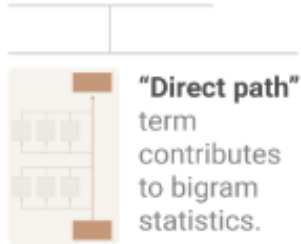


$$+ \sum_{h \in H} A^h \otimes (W_U W_{OV}^h W_E)$$

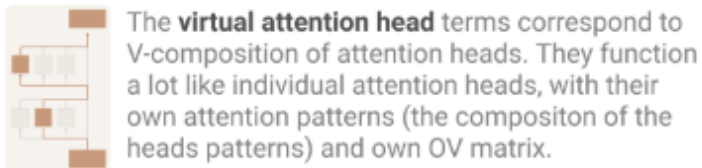


Analysis of Simplified Transformers (2)

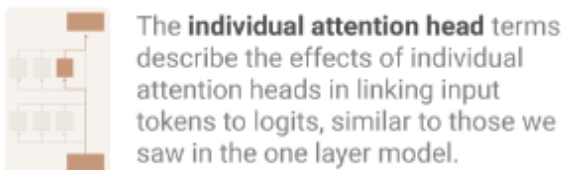
$$T = \text{Id} \otimes W_U W_E$$



$$+ \sum_{h_2 \in H_2} \sum_{h_1 \in H_1} (A^{h_2} A^{h_1}) \otimes (W_U W_{OV}^{h_2} W_{OV}^{h_1} W_E)$$



$$+ \sum_{h \in H_1 \cup H_2} A^h \otimes (W_U W_{OV}^h W_E)$$



- Two Layer Transformer are composed of:
 - Direct path contributing to next token statistics
 - Virtual and Individual attention heads behaving similarly
 - Virtual attention heads are the linear multiplication of several attention heads
- Much higher in-context learning by «Induction heads»

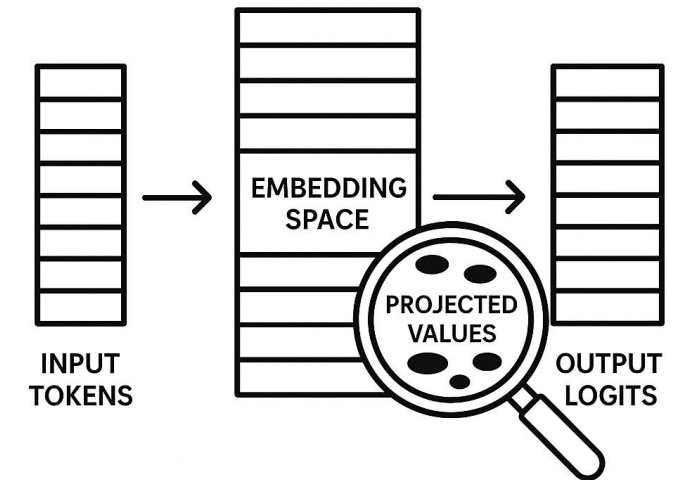
Induction Head - Example 1

Mr and Mrs Dursley, of ...	such nonsense.	Mr Dursley was the
Mr and Mrs Dursley, of ...	such nonsense.	Mr Dursley was the
Mr and Mrs Dursley, of ...	such nonsense.	Mr Dursley was the
Mr and Mrs Dursley, of ...	such nonsense.	Mr Dursley was the
Mr and Mrs Dursley, of ...	such nonsense.	Mr Dursley was the
Mr and Mrs Dursley, of ...	such nonsense.	Mr Dursley was the
Mr and Mrs Dursley, of ...	such nonsense.	Mr Dursley was the

☐ Present Token
☒ Attention
☒ Logit Effect

Transformer Matrix dimensions

- Embedding step:
 - Size: $N_v \times N_e$ (50257 \times 1600 for GPT-2 1558M)
 - Purpose: From the vocab space (N_v) into the embedding space (N_e)
- Transformer Blocks:
 - Processing: The 1600-dimensional vector moves through all transformer layers (blocks)
 - Each Block's Output: Another 1600-dimensional vector, progressively refining the representation
- Un-Embedding Step (Final Projection):
 - Size: $N_e \times N_v$ (1600 \times 50257)
 - Purpose: Transforms the processed 1600-dimensional vector back into vocab space (N_v)



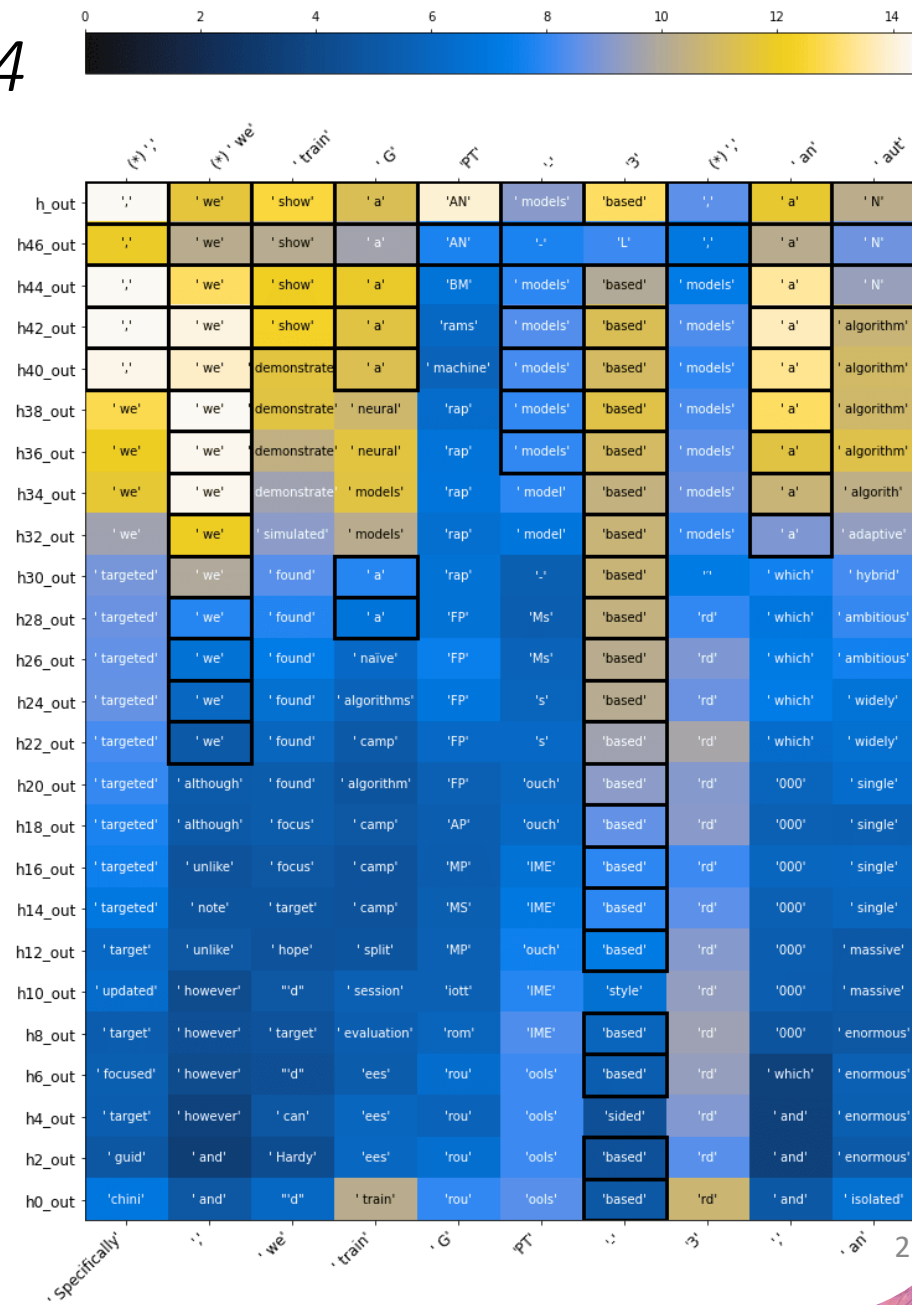
--> **We can analyze how the residual stream evolves through the layers**

- By applying the *Un-Embedding* step of the output layer to any intermediate layer!

Analysis under the *Logit Lens*⁴

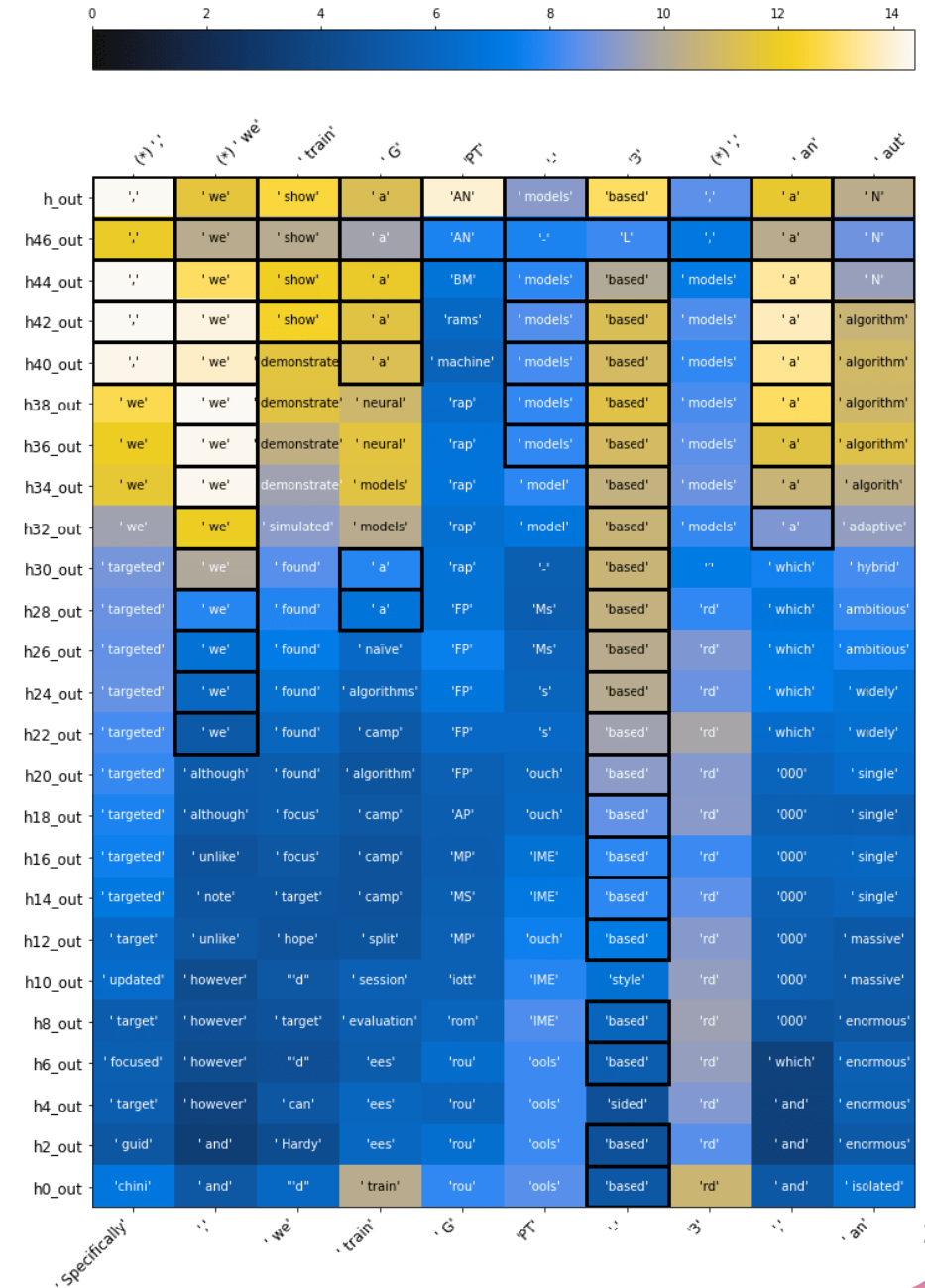
- **Input:** A segment of the GPT-3 paper's abstract
 - Preceding text available but not explicitly visualized
- **Output:** Token predictions (h_out) and correct labels:
 - Correct label: the token following the current input token
 - Correct predictions (*) where the model's top guess matches the expected output
 - DISCLAIMER: Even in output, many predictions are wrong. It does not matter, GPT2 was an early LLM, the underlying process is still valid
- **Intermediate values:** Color-coded logits
 - Logits increase as the model refines its predictions and improves its certainty

[4]<https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens>



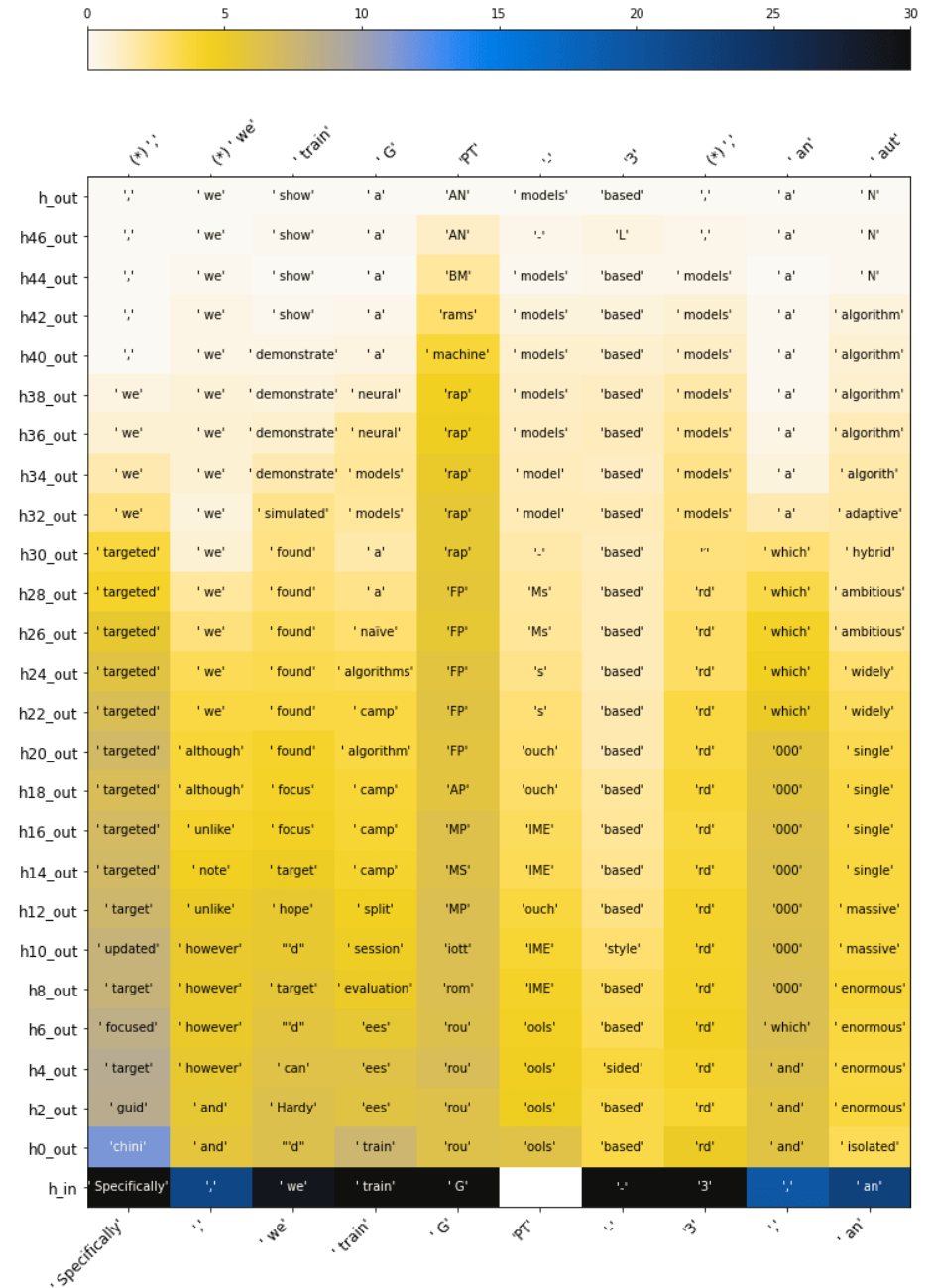
Observations

- GPT "early guesses" are generally wrong but often sensible enough in some way:
 - "We train GPT-3..." *000?* (someday!)
 - "GPT-3, an..." *enormous? massive?* (not wrong!)
- Some early predictions look noisy but gradually become coherent
 - "We train GPT-3, an aut..." *oreceptor?* (later converges to the correct *oregressive*)
- The logit lens reveals **how each step contributes** to the final output



KL divergence plot

- The KL Divergence measure the similarity between two distributions
 - The current predicted output distribution
 - The correct output label distribution
- Input token information is **quickly discarded** after the first layer
 - Inputs are transformed immediately rather than preserved for gradual processing
 - Later layers refine guesses **without keeping direct input reference**
- GPT works more like an iterative **predictive space** refiner rather than an input processing model



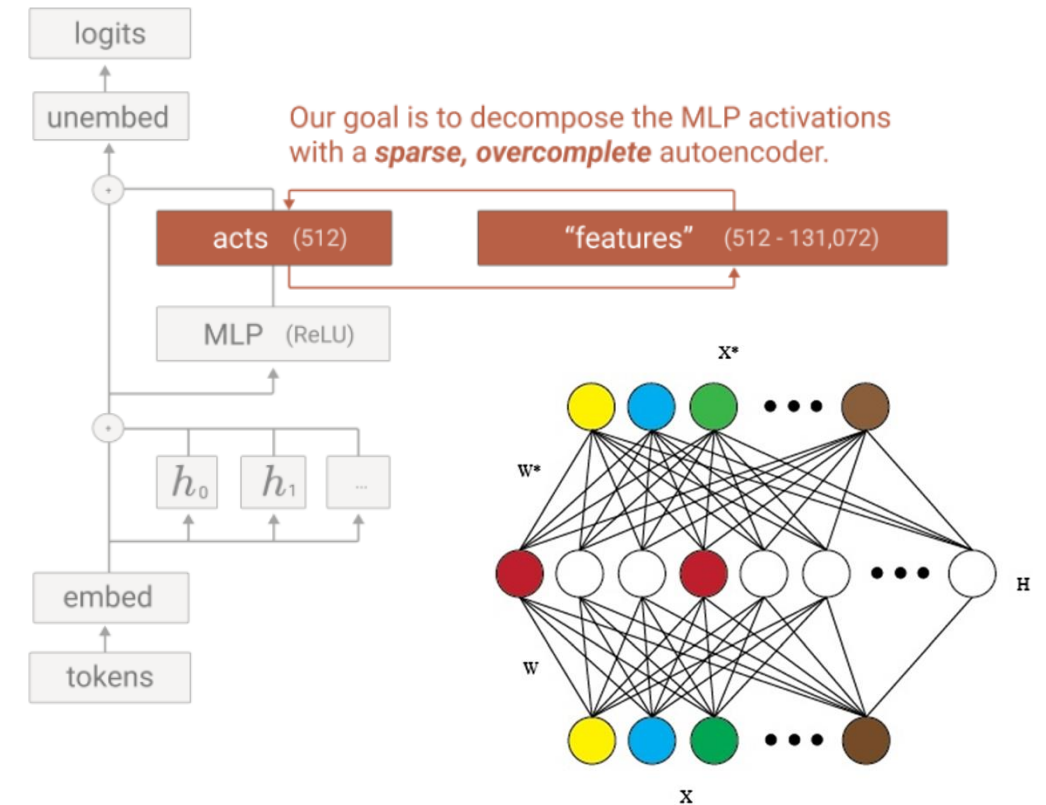
Addressing the Polisemanticity Issue

- Mechanistic interpretability seeks to break neural networks into simpler, understandable components
- **BUT:** Neurons are often polysemantic -- they activate for multiple unrelated concepts
 - E.g., The neuron in Inception v1 responding to both **cat faces and car fronts**.
 - In LLM a neuron can fire for **academic citations, HTTP requests, and Korean text**.
- **Polysemanticity complicates interpretability**, making it hard to assign clear functions to neurons.
- Researchers believe **dictionary learning** may allow extract **monosemantic features**, improving transparency

Towards Monosemanticity: Decomposing Language Models With Dictionary Learning⁵

- Feature-Based Decomposition:
 - Instead of analyzing neurons
 - We analyze the entire activations
 - We decompose it into single general **features**
 - It can be applied on **top of MLP** as well!
- Sparse AutoEncoders (SAE)
 - Represent activations as a combination of distinct features instead of single neuron responses
 - Feature expansion: SAE hidden dimensions \gg input dimension (8x)
 - Allows to decompose MLP representation and **avoid superposition**

$$\bullet \text{Loss}_{SAE} = |W_{DEC}(MLP(x)W_{ENC}) - MLP(x)|$$



[5] <https://transformer-circuits.pub/2023/monosemantic-features/index.html>

Let's find features within an LLM!

Feature Number
(click for hyperlink)

Human explanation

Histogram of randomly
sampled non-zero
activations

Top 10 negative and
positive output logits
of the feature

Top 20 max
activating examples

Ten evenly spaced intervals
spanning the full range of
activation values

Autointerp
explanation and
prediction score

Top 3 neurons
by how much the
feature activates them

Top 3 neurons
by token correlation

Top 3 features from
the parallel run with
a different random
seed

#3923 Latin: ?

AUTOINTERP. (SCORE = 0.305)
The neuron attends to common Latin words and short phrases.

NEURON ALIGNMENT		
Index	Value	% of L ₁
138	+0.40	3.6%
86	+0.27	2.5%
317	+0.27	2.4%

CORRELATED NEURONS		
Index	Pearson Corr.	Cosine Sim.
257	+0.09	+0.09
99	+0.05	+0.06
138	+0.05	+0.04

CORRELATED B FEATURES		
Index	Pearson Corr.	Cosine Sim.
2535	+0.75	+0.75
624	+0.10	+0.10
3431	+0.08	+0.08

ACTIVATIONS (DENSITY = 0.1440%)

NEGATIVE LOGITS	
Rangers	-0.37
Center	-0.33
NBA	-0.32
CW	-0.31
WF	-0.31
League	-0.30
watch	-0.29
Hockey	-0.29
West	-0.29
s ville	-0.29

POSITIVE LOGITS	
ibus	+0.48
orem	+0.43
ips	+0.43
aliqu	+0.40
ea	+0.38
imus	+0.38
orum	+0.37
ulum	+0.37
itor	+0.37
nostr	+0.36

TOP ACTIVATIONS
TRAIN TOKEN MAX ACT = 11.54

amus elementum semper nisi. A
posuere semper felis placer
ero, at semper nulla finibus
a eligendi obcaecati
quibusdam corrupti officia consequ
imentis utuntur magna corpor
am provident reiciendis unde vit
Nulla facilisi. Quisque
ea, deserunt reiciendis
unde omnis iste natus error sit
, nascetur ridiculus mus.
sapiente ullam alias quaerat
illo, ipsam saepe e
veritatis itaque, placeat ven
non nisi semper tellus malesu
ore id, magni provident, error
In eget semper nibh. F
, ut aut reiciendis volupt
um lugubri complet, ut aec
met bibendum nullam, massa lac

SUBSAMPLE INTERVAL 0
TRAIN TOKEN MAX ACT = 10.04

ricies enim et mi gravida vari
itur fermentum, nibh ac
. Pellentesque in vehicula
nisi non sceler
ut primum eos adprehendit,

SUBSAMPLE INTERVAL 1
TRAIN TOKEN MAX ACT = 9.168

ut primum eos adprehendit,
us, et dictum nulla tempus
, ciuitatis dei presidis
tabulas (conventuales) iter
hominem, et unde multo pl

SUBSAMPLE INTERVAL 2
TRAIN TOKEN MAX ACT = 8.342

hominem, et unde multo pl
ut aliquip ex ea commodo consequ
tineremus, lege recitata
si facere nequirem, nec
am vero senatus non sententiis

Blue underline means a lower ablation loss (better token prediction); red means a higher loss

Bold token comes from the training data (used to select each example). Surrounding 4 tokens give context

Hover over any token (for 2+ seconds) to see its activation value and ablations

means this specific example has already appeared in another interval

26

Let's find features within an LLM!

https://transformer-circuits.pub/2023/monosemantic-features/vis/a1.html?ordering=count&search_text=food

Let's see how to use these features to analyze a text

<https://transformer-circuits.pub/2023/monosemantic-features/vis/a1-en.html>