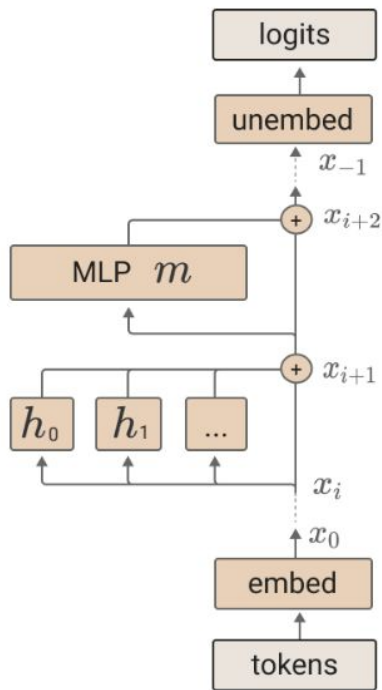


AI Alignment Cohort
Session 8

Introduction to Mechanistic Interpretability

TransformerLens & Induction circuits

Transformer Architecture



The final logits are produced by applying the unembedding.

$$T(t) = W_U x_{-1}$$

An MLP layer, m , is run and added to the residual stream.

$$x_{i+2} = x_{i+1} + m(x_{i+1})$$

One residual block

Each attention head, h , is run and added to the residual stream.

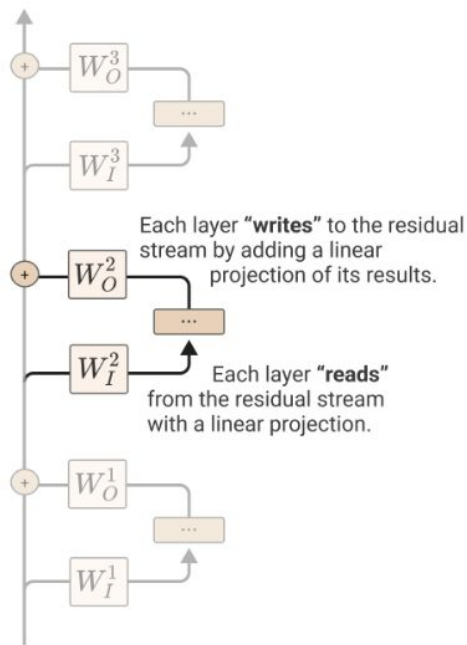
$$x_{i+1} = x_i + \sum_{h \in H_i} h(x_i)$$

Token embedding.

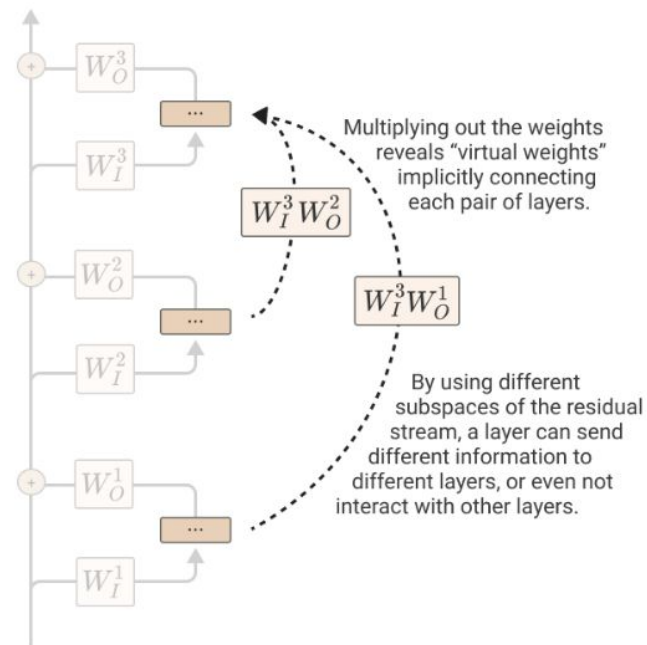
$$x_0 = W_E t$$

Residual Stream

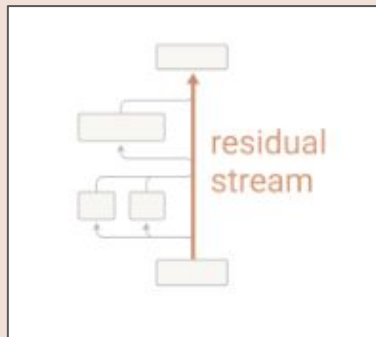
The residual stream is modified by a sequence of MLP and attention layers "reading from" and "writing to" it with linear operations.



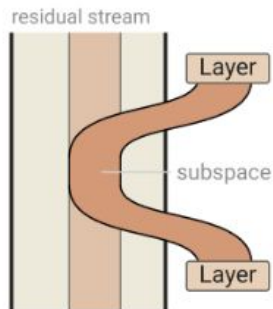
Because all these operations are linear, we can "multiply through" the residual stream.



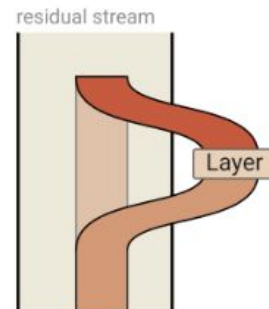
Residual Stream



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



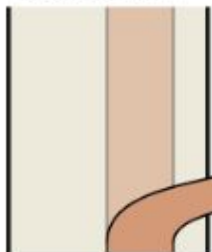
Layers can interact by writing to and reading from the same or overlapping subspaces. If they write to and read from disjoint subspaces, they won't interact. Typically the spaces only partially overlap.



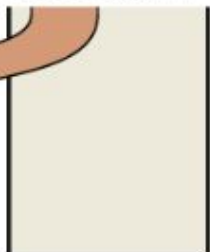
Layers can delete information from the residual stream by reading in a subspace and then writing the negative version.

Attention Heads as Information Movement

token A
residual stream

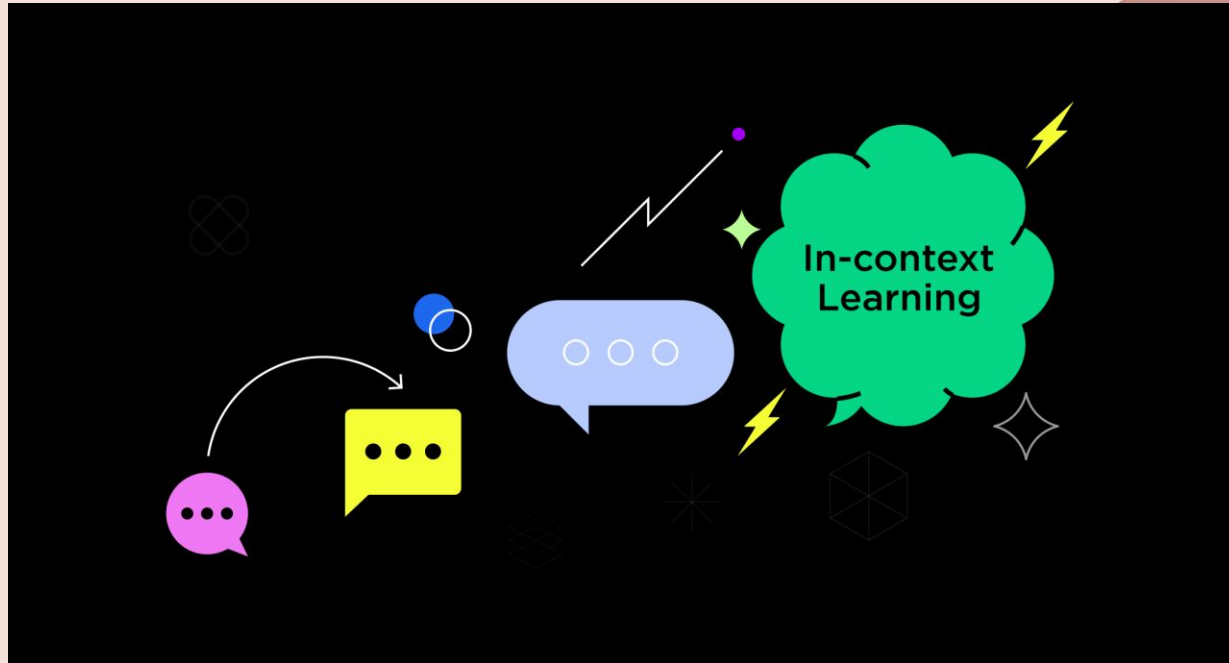


token B
residual stream



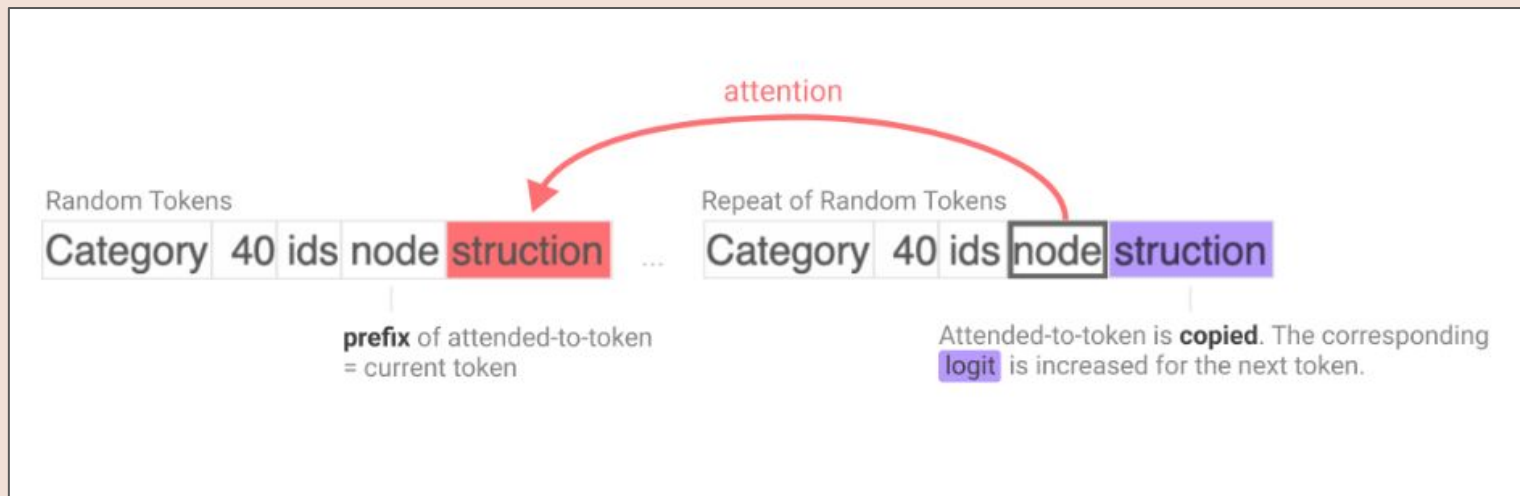
Attention heads copy information from the residual stream of one token to the residual stream of another. They typically write to a different subspace than they read from.

In-Context Learning



Induction Heads

Induction heads are implemented by a circuit consisting of a pair of attention heads in different layers that work together to copy or complete patterns



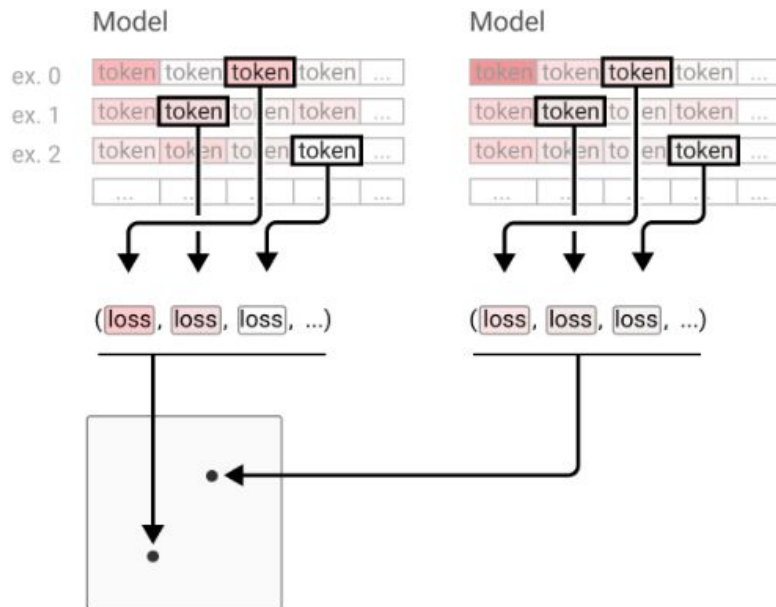
ICL Score

In-context learning score: the loss of the 500th token in the context minus the average loss of the 50th token in the context, averaged over dataset examples

Step 1: Run each model / snapshot over the same set of multiple dataset examples, collecting one token's loss per example.

Step 2: For each sample, extract the loss of a consistent token. Combine these to make a vector of losses per model / snapshot.

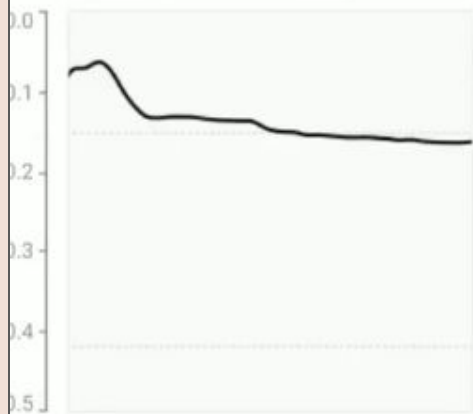
Step 3: The vectors are jointly reduced with principal component analysis to project them into a shared 2D space.



ONE LAYER (ATTENTION-ONLY)

Elapsed Training Tokens

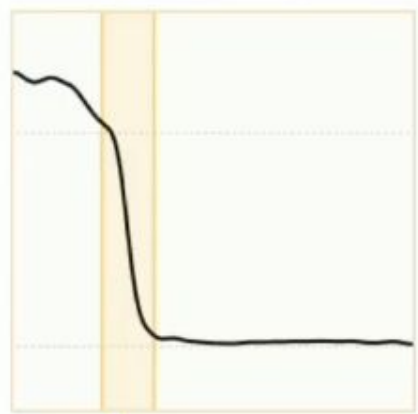
0 2.5e9 5.0e9 7.5e9 1e10



TWO LAYER (ATTENTION-ONLY)

Elapsed Training Tokens

0 2.5e9 5.0e9 7.5e9 1e10



ONE LAYER (ATTENTION-ONLY)

Elapsed Training Tokens

0 2.5e9 5.0e9 7.5e9 1e10



TWO LAYER (ATTENTION-ONLY)

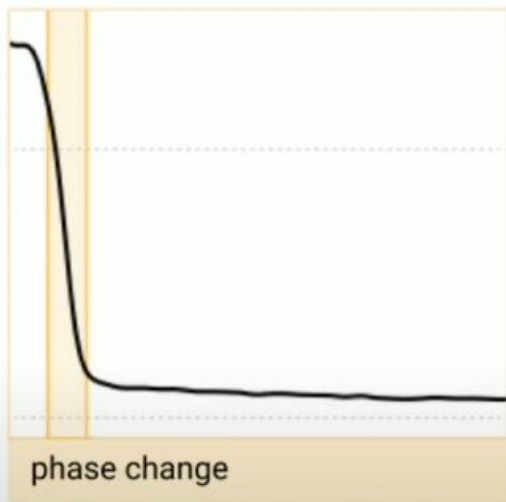
Elapsed Training Tokens

0 2.5e9 5.0e9 7.5e9 1e10

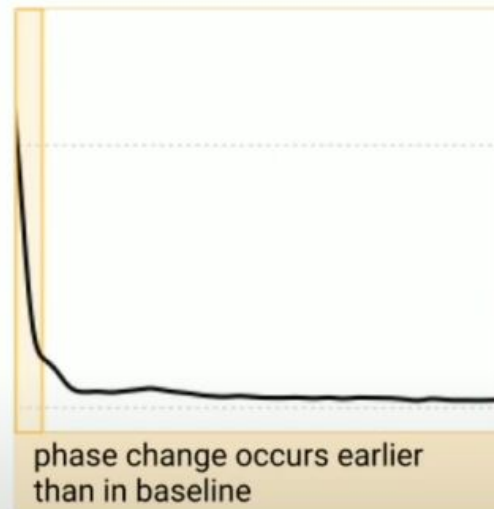


Elapsed Training Tokens

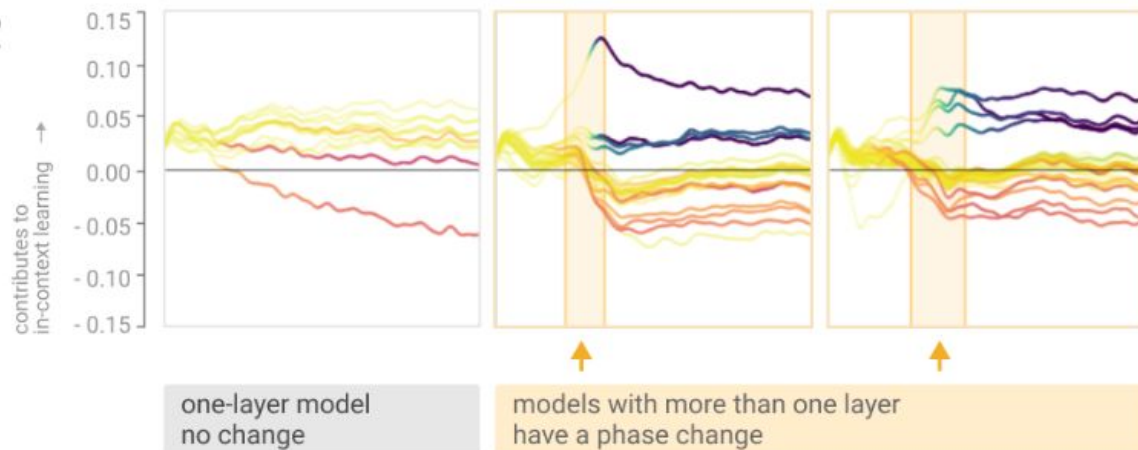
0 2.5e9 5.0e9 7.5e9 1e10



Change architecture to promote induction heads => phase change happens earlier

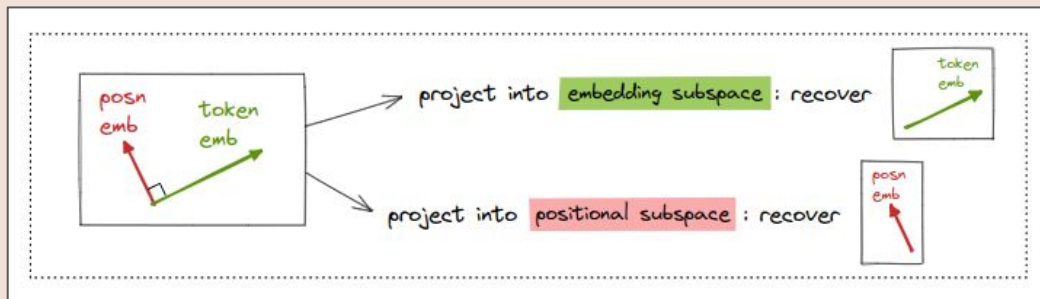


ABLATION TO "IN-CONTEXT LEARNING" SCORE

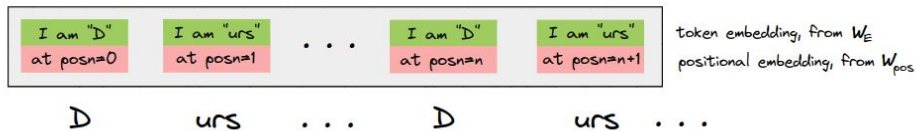


The attention heads which increase in-context learning are almost entirely **induction heads**. They form as in-context learning increases and drive its increase.

Subspaces in the Residual Stream & Embedding



Embedding



token encoding subspace (i.e. "this token is x")

= rows of W_E

positional encoding subspace (i.e. "this token is at position x")

= rows of W_{pos}

decoding subspace (i.e. "the next token will be x")

= cols of W_U

prev token subspace (i.e. "the previous token was x")

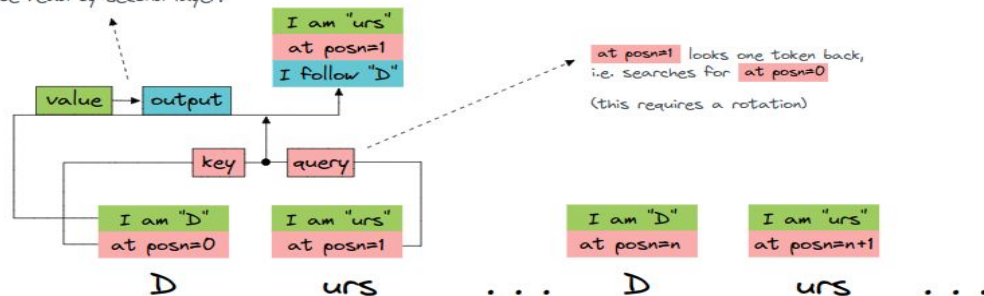
= "intermediate information"

Layer 0 Attention Head

Layer 0 attention head

Summary: each token looks one position backwards, and gets the information about which token preceded it.

"I am 'D'" is converted to "I follow 'D'" and moved to the destination token, ready to be read by second layer.



QK circuit

Query vector is $\begin{bmatrix} \text{I am "urs"} \\ \text{at posn=1} \end{bmatrix}^T W_Q = \begin{bmatrix} \text{"I'm looking for posn 1 - 1 = 0"} \end{bmatrix}$

Key vector is $\begin{bmatrix} \text{I am "D"} \\ \text{at posn=0} \end{bmatrix}^T W_K = \begin{bmatrix} \text{"I'm at posn 0"} \end{bmatrix}$

The attention score (the dot product of the key and query vectors) is large, because the key is a good match for the query.

i.e. we subtract one from the current position (this requires a rotation!)

OV circuit

The OV circuit reads token information, and moves it to a different subspace (so it doesn't erase the token embedding information which is already stored at the destination token).

We have:

$\begin{bmatrix} \text{I am "D"} \\ \text{at posn=0} \end{bmatrix}^T W_V W_O = \begin{bmatrix} \text{I follow "D"} \end{bmatrix}$

which gets added to the residual stream for the first "urs" token.

Layer 0 Attention Head Mathematically

Mathematically, the attention scores are:

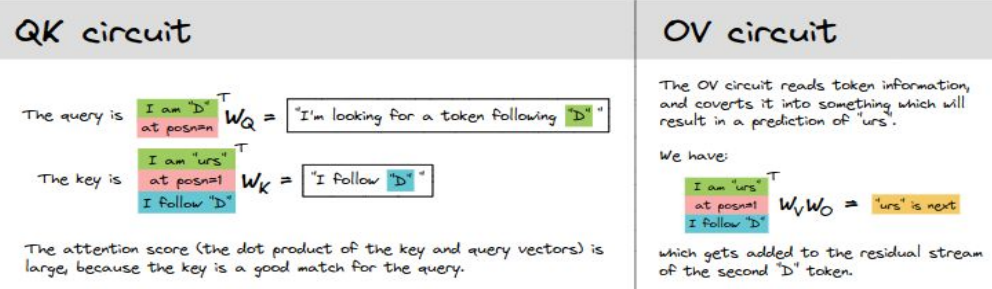
$$\begin{matrix} \text{I am 'X'} \\ \text{at posn } i \end{matrix}^T W_Q W_K^T \begin{matrix} \text{I am 'Y'} \\ \text{at posn } j \end{matrix} \approx \begin{matrix} \text{at posn } i \end{matrix}^T W_Q W_K^T \begin{matrix} \text{at posn } j \end{matrix} = \begin{cases} \text{large} & \text{if } j = i - 1 \\ \text{small} & \text{if else} \end{cases}$$

How can we prove this?

Define the "full QK circuit" $W_{\text{pos}} W_Q W_K^T W_{\text{pos}}$

The (i, j) th element of this matrix is $\begin{matrix} \text{at posn } i \end{matrix}^T W_Q W_K^T \begin{matrix} \text{at posn } j \end{matrix}$, so we can verify this.

Summary: the 2nd "D" token looks back for the token following the 1st "D" (which is "urs"), and uses that as its prediction.



Layer 1 Attention Head Mathematically

Mathematically, the attention scores are:

$$\begin{aligned}
 & \begin{bmatrix} \text{I am 'x'} \\ \text{at posna} \\ \text{I follow 'a'} \end{bmatrix}^T W_Q W_K^T \begin{bmatrix} \text{I am 'y'} \\ \text{at posna} \\ \text{I follow 'a'} \end{bmatrix} \approx \begin{bmatrix} \text{I am 'x'} \end{bmatrix}^T W_Q W_K^T \begin{bmatrix} \text{I follow 'a'} \end{bmatrix} \\
 &= \underbrace{\left(\begin{bmatrix} \text{I am 'x'} \end{bmatrix}^T W_Q \right)}_{\text{"I am looking for something which follows 'x' "}} \cdot \underbrace{\left(\begin{bmatrix} \text{I follow 'a'} \end{bmatrix}^T W_K \right)}_{\text{"I follow 'a' "}} = \begin{cases} \text{large} & \text{if "x" = "a"} \\ \text{small} & \text{if else} \end{cases}
 \end{aligned}$$

How can we prove this?

Define the "full K-composition circuit" $W_E W_{QK}^1 (W_{OV}^0)^T W_E^T$ ↗ Superscript 0 means the head in layer 0, and 1 to means the head in layer 1.

If X and Q are tokens, then the (X, Q) th element of this matrix is:

$$\begin{aligned}
 \begin{bmatrix} \text{'x'} \end{bmatrix}^T W_E W_{QK}^1 (W_{OV}^0)^T W_E^T \begin{bmatrix} \text{'a'} \end{bmatrix} &= \begin{bmatrix} \text{I am 'x'} \end{bmatrix}^T W_{QK}^1 (W_{OV}^0)^T \begin{bmatrix} \text{I am 'a'} \end{bmatrix} \\
 &= \begin{bmatrix} \text{I am 'x'} \end{bmatrix}^T W_{QK}^1 \begin{bmatrix} \text{I follow 'a'} \end{bmatrix}
 \end{aligned}$$

So we just need to verify that this is $\begin{cases} \text{large} & \text{if "x" = "a"} \\ \text{small} & \text{if else} \end{cases}$

quAIdditch

In the context of transformer models, which specific circuit is responsible for writing information into the residual stream that helps subsequent layers make predictions about token sequences?

OV (Output-Value) circuit