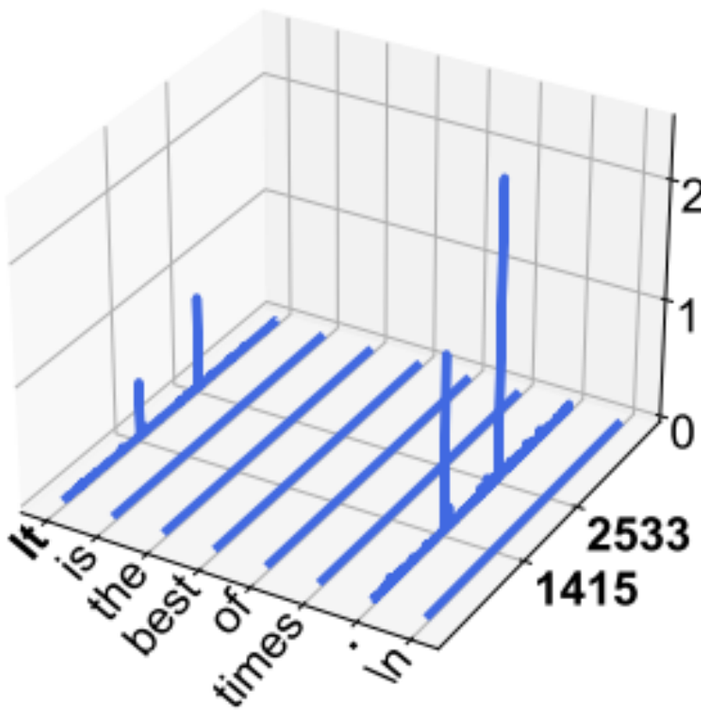
Mechanistically Demystifying Extreme-Token Phenomena in Large Language Models

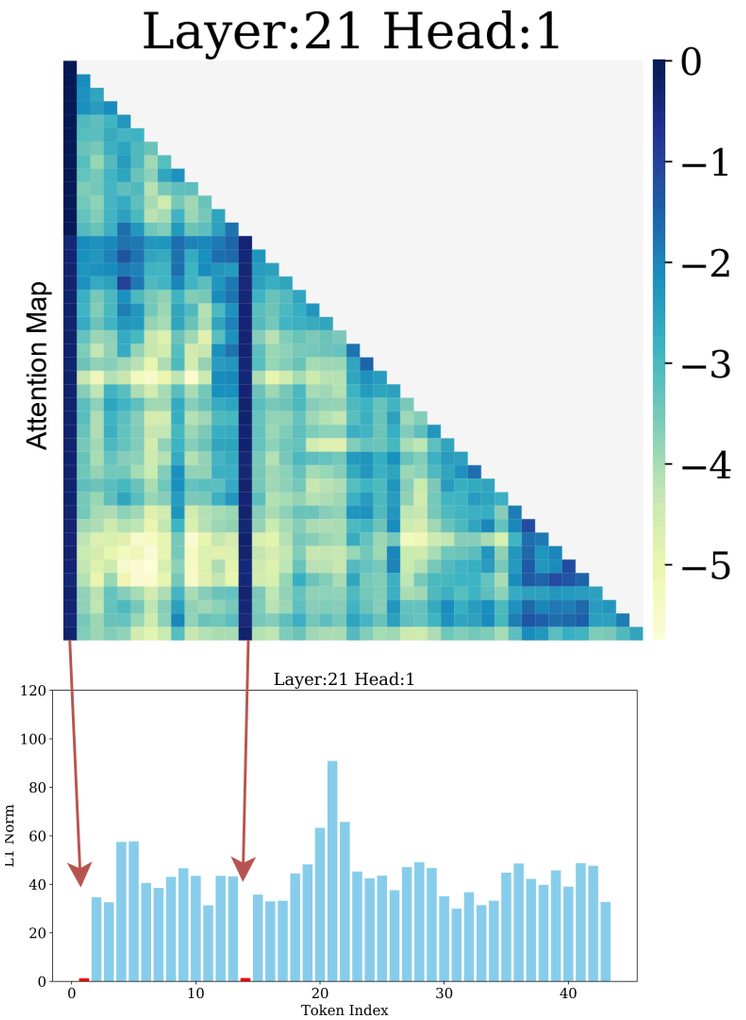
Song Mei
UC Berkeley

Peter Hall conference @ UC Davis
Statisitcs in the age of AI

Joint work with Tianyu Guo, Druv Pai,
Yu Bai, Jiantao Jiao, Mike Jordan

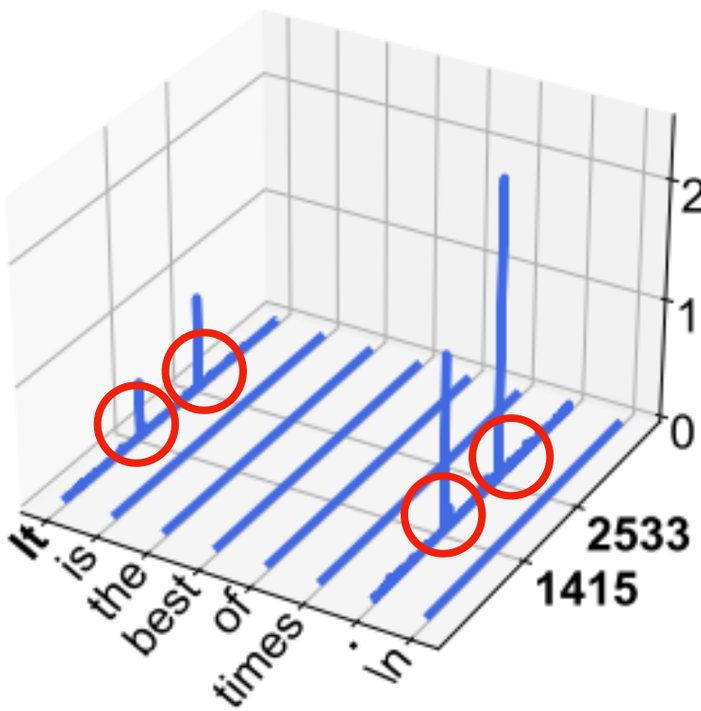
Extreme values in transformer-based LLMs

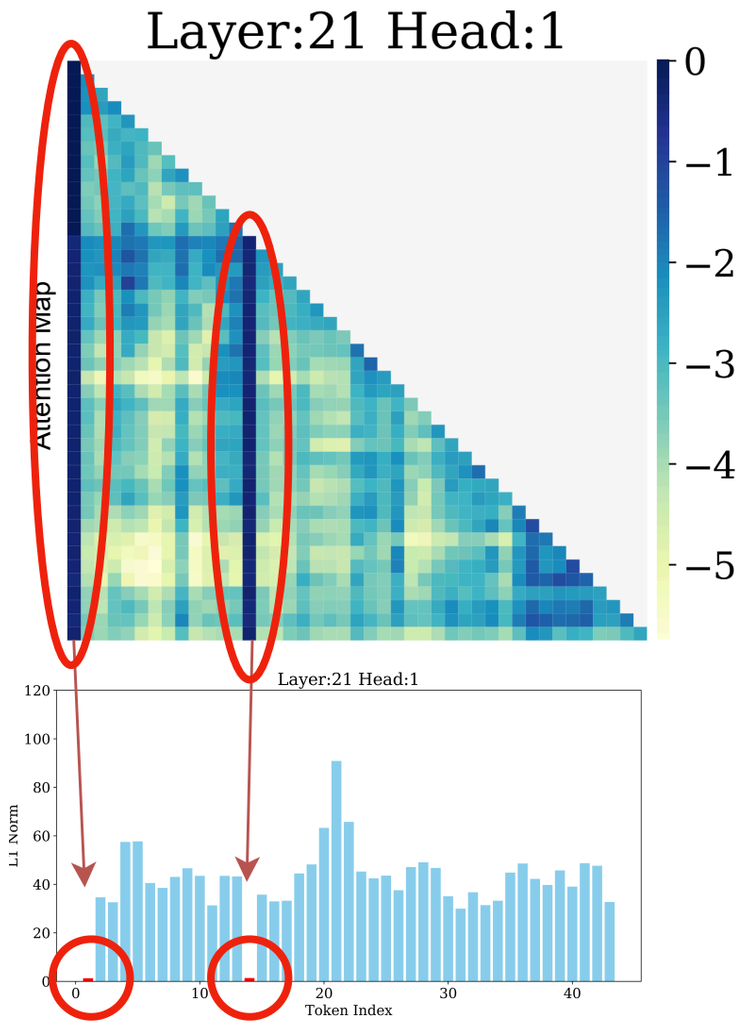




[Dettmers et al., 2022] [Xiao et al., 2023] [Sun et al., 2024] [Guo et al., 2024]

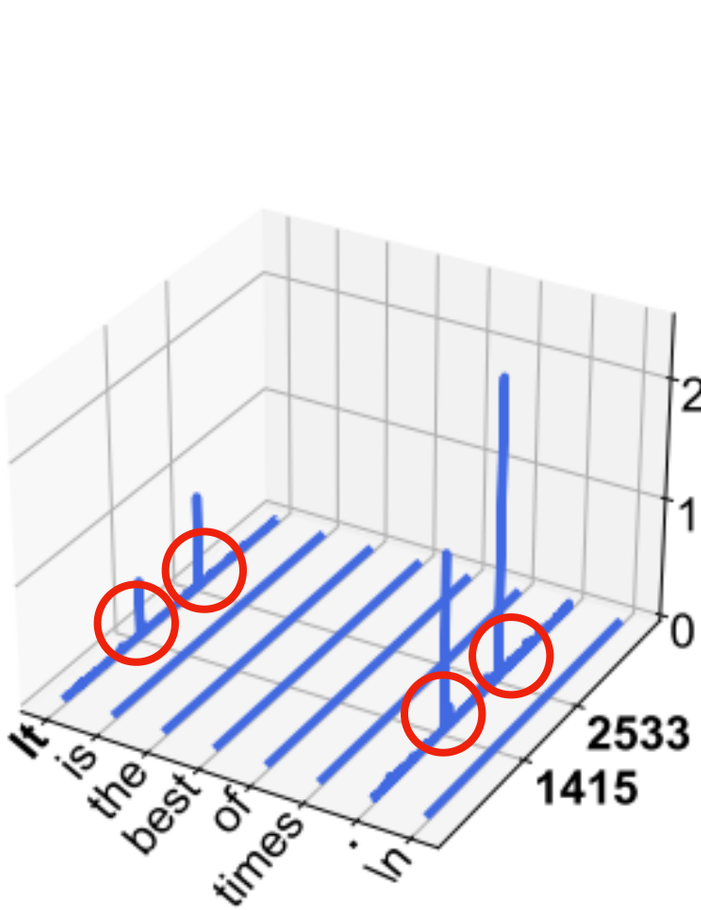
Extreme values in transformer-based LLMs

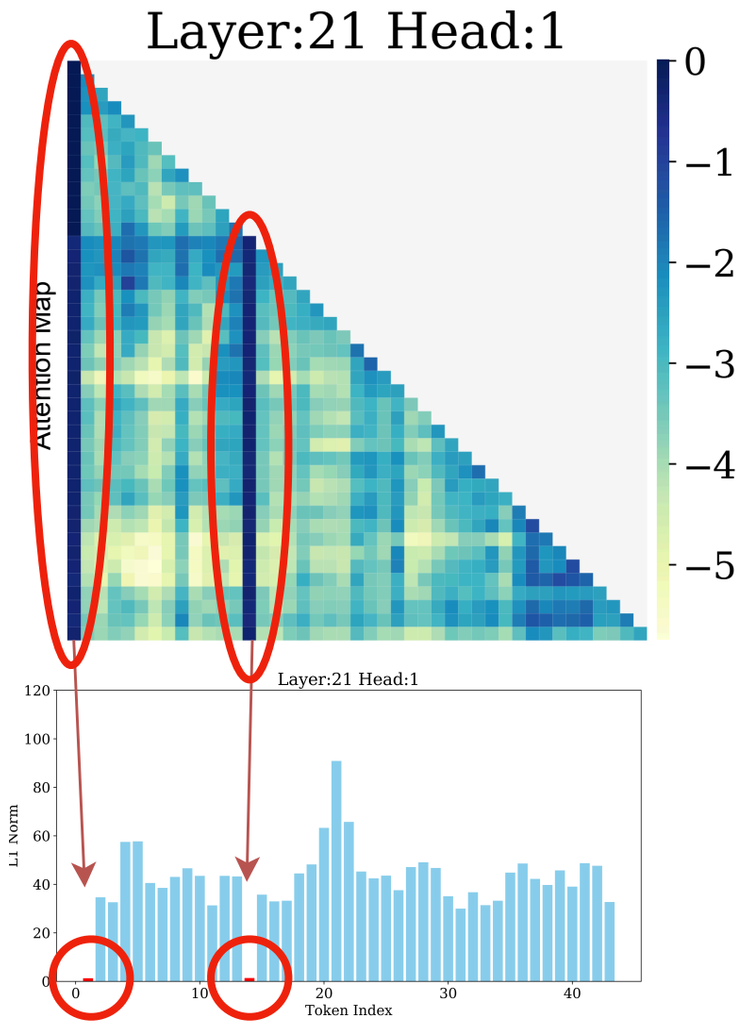




[Dettmers et al., 2022] [Xiao et al., 2023] [Sun et al., 2024] [Guo et al., 2024]

Extreme values in transformer-based LLMs





[Dettmers et al., 2022] [Xiao et al., 2023] [Sun et al., 2024] [Guo et al., 2024]

- A universal phenomena: happens in almost all open-source LLMs including GPT-2, Llama-2, Llama-3, Pythia, Mixtrial, etc.
- Troublemaking: inference, quantization, interpretability…

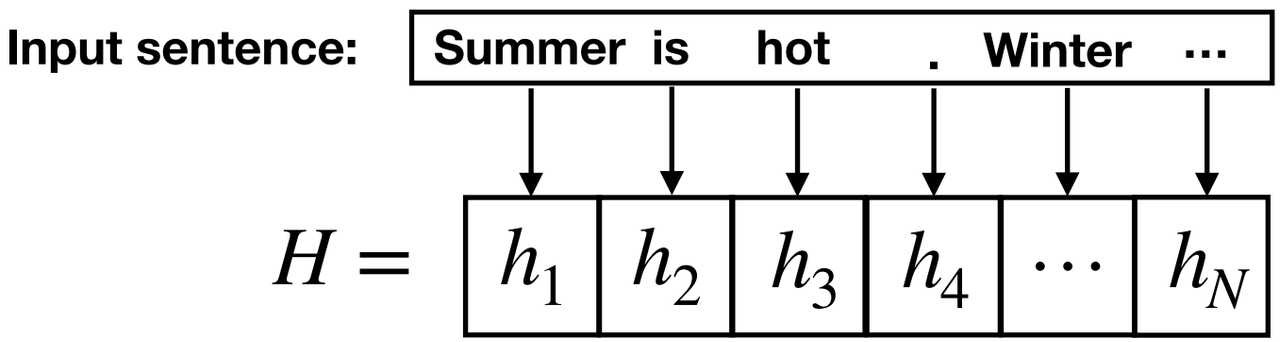
Transformer-based LLMs

- A transformer is a sequence-to-sequence neural network $\mathrm{TF}_{\boldsymbol{\theta}} : \mathbb{R}^{D\times N} \to \mathbb{R}^{D\times N}$.
- Input sequence: $H = [h_1, h_2, \ldots, h_N] \in \mathbb{R}^{D\times N}$; each $h_i \in \mathbb{R}^D$ is called a token.

Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Transformer-based LLMs

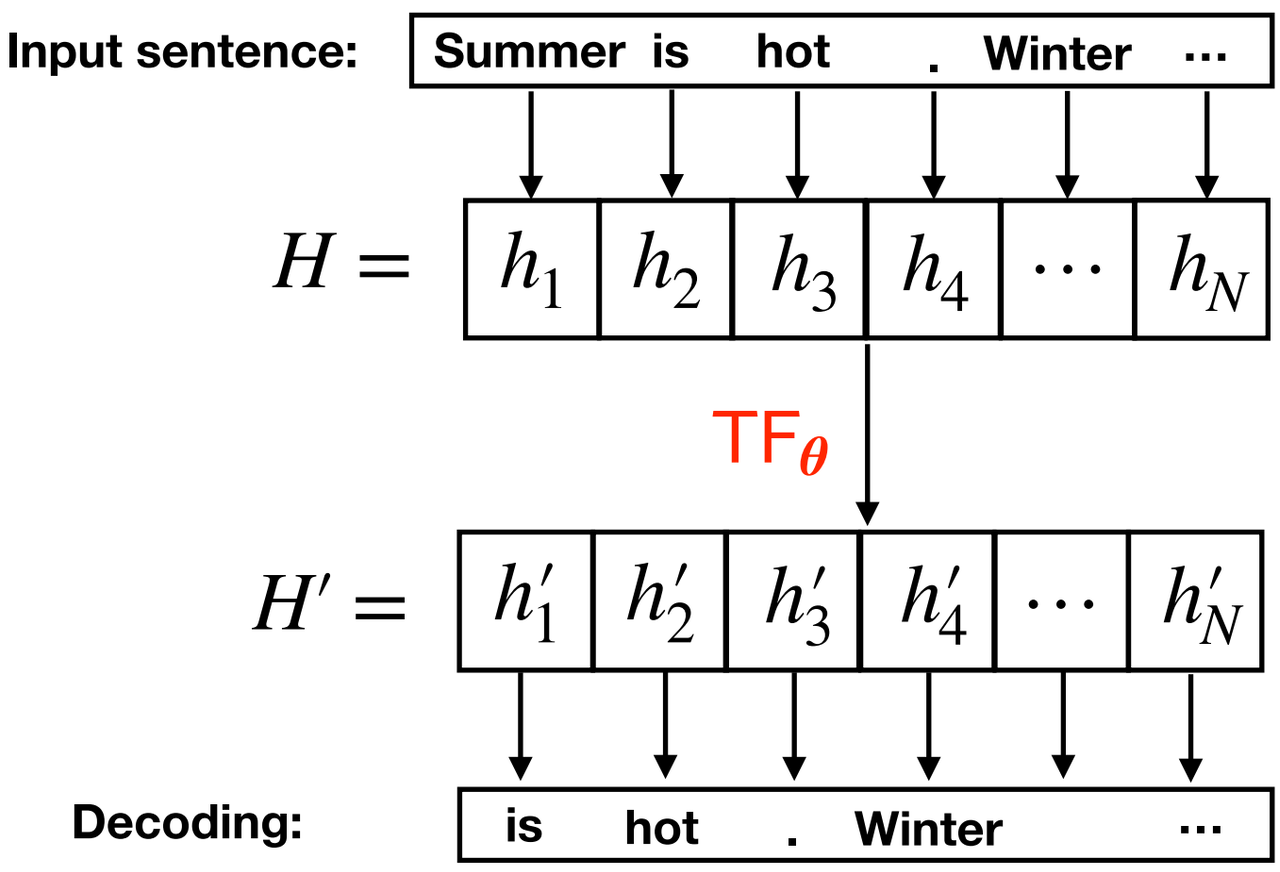
- A transformer is a sequence-to-sequence neural network $\mathrm{TF}_{\boldsymbol{\theta}} : \mathbb{R}^{D\times N} \to \mathbb{R}^{D\times N}$.
- Input sequence: $H = [h_1, h_2, \ldots, h_N] \in \mathbb{R}^{D\times N}$; each $h_i \in \mathbb{R}^D$ is called a token.



Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Transformer-based LLMs

- A transformer is a sequence-to-sequence neural network $\mathrm{TF}_{\boldsymbol{\theta}} : \mathbb{R}^{D\times N} \to \mathbb{R}^{D\times N}$.
- Input sequence: $H = [h_1, h_2, \ldots, h_N] \in \mathbb{R}^{D\times N}$; each $h_i \in \mathbb{R}^D$ is called a token.



Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Transformer architecture

- A transformer is an iterative composition of MLP layers and Attention layers

$$\mathrm{TF}_{\boldsymbol{\theta}}(\,\cdot\,) = (\mathrm{Id} + \mathrm{MLP}_{W^{(L)}}) \circ (\mathrm{Id} + \mathrm{ATTN}_{A^{(L)}}) \circ \cdots \circ (\mathrm{Id} + \mathrm{MLP}_{W^{(1)}}) \circ (\mathrm{Id} + \mathrm{ATTN}_{A^{(1)}})$$

Transformer architecture

- A transformer is an iterative composition of MLP layers and Attention layers

$$\mathrm{TF}_{\boldsymbol{\theta}}(\,\cdot\,) = (\mathrm{Id} + \mathrm{MLP}_{W^{(L)}}) \circ (\mathrm{Id} + \mathrm{ATTN}_{A^{(L)}}) \circ \cdots \circ (\mathrm{Id} + \mathrm{MLP}_{W^{(1)}}) \circ (\mathrm{Id} + \mathrm{ATTN}_{A^{(1)}})$$

- MLP layer:

$$\mathrm{MLP}_W : \mathbb{R}^{D\times N} \to \mathbb{R}^{D\times N}$$

$$H' = W_2 \cdot \sigma(W_1 H)$$

Transformer architecture

- A transformer is an iterative composition of MLP layers and Attention layers

$$\mathrm{TF}_{\boldsymbol{\theta}}(\,\cdot\,) = (\mathrm{Id} + \mathrm{MLP}_{W^{(L)}}) \circ (\mathrm{Id} + \mathrm{ATTN}_{A^{(L)}}) \circ \cdots \circ (\mathrm{Id} + \mathrm{MLP}_{W^{(1)}}) \circ (\mathrm{Id} + \mathrm{ATTN}_{A^{(1)}})$$

- MLP layer:

$$\mathrm{MLP}_W : \mathbb{R}^{D\times N} \to \mathbb{R}^{D\times N}$$

$$H' = W_2 \cdot \sigma(W_1 H)$$

- Attention layer:

$$\mathrm{ATTN}_A : \mathbb{R}^{D\times N} \to \mathbb{R}^{D\times N}$$

$$H' = \sum_{m=1}^{M} O_m V_m H \cdot \mathrm{Softmax}(\mathrm{mask}(H^\top {K_m}^\top Q_m H))$$

Transformer architecture

- A transformer is an iterative composition of MLP layers and Attention layers

$$\mathrm{TF}_{\boldsymbol{\theta}}(\,\cdot\,) = (\mathrm{Id} + \mathrm{MLP}_{W^{(L)}}) \circ (\mathrm{Id} + \mathrm{ATTN}_{A^{(L)}}) \circ \cdots \circ (\mathrm{Id} + \mathrm{MLP}_{W^{(1)}}) \circ (\mathrm{Id} + \mathrm{ATTN}_{A^{(1)}})$$

- MLP layer:

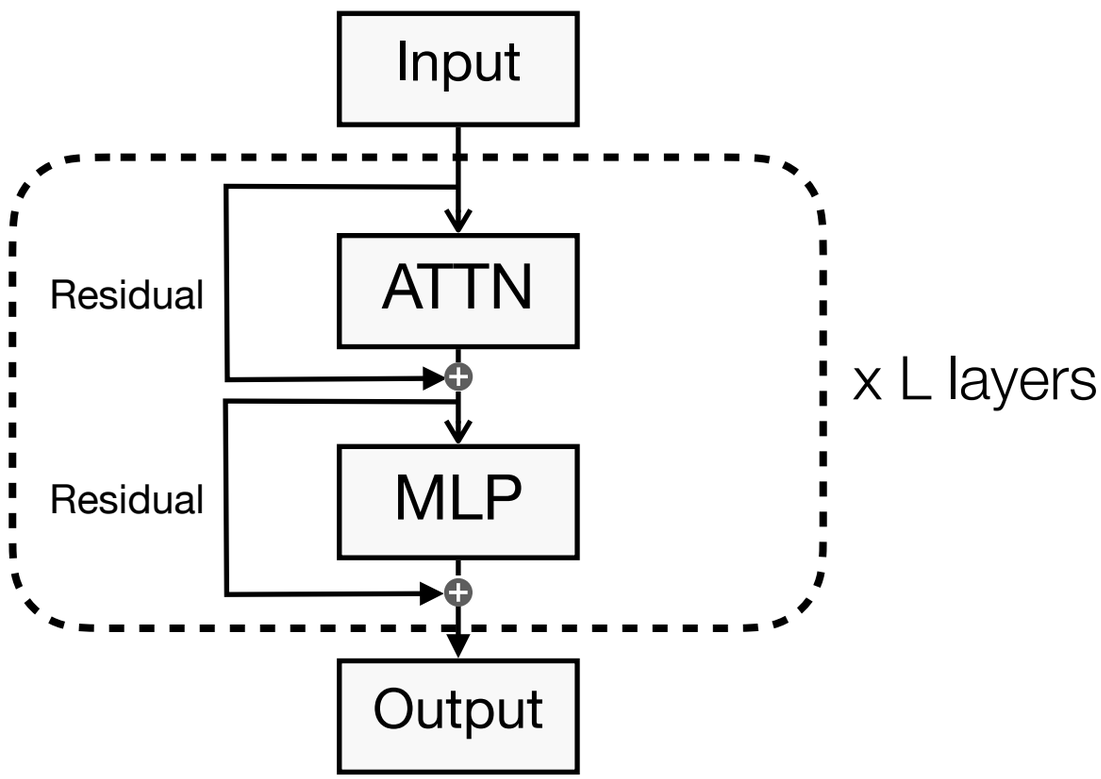
$$\mathrm{MLP}_W : \mathbb{R}^{D\times N} \to \mathbb{R}^{D\times N}$$

$$H' = W_2 \cdot \sigma(W_1 H)$$

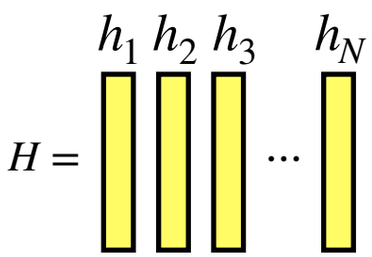
- Attention layer:

$$\mathrm{ATTN}_A : \mathbb{R}^{D\times N} \to \mathbb{R}^{D\times N}$$

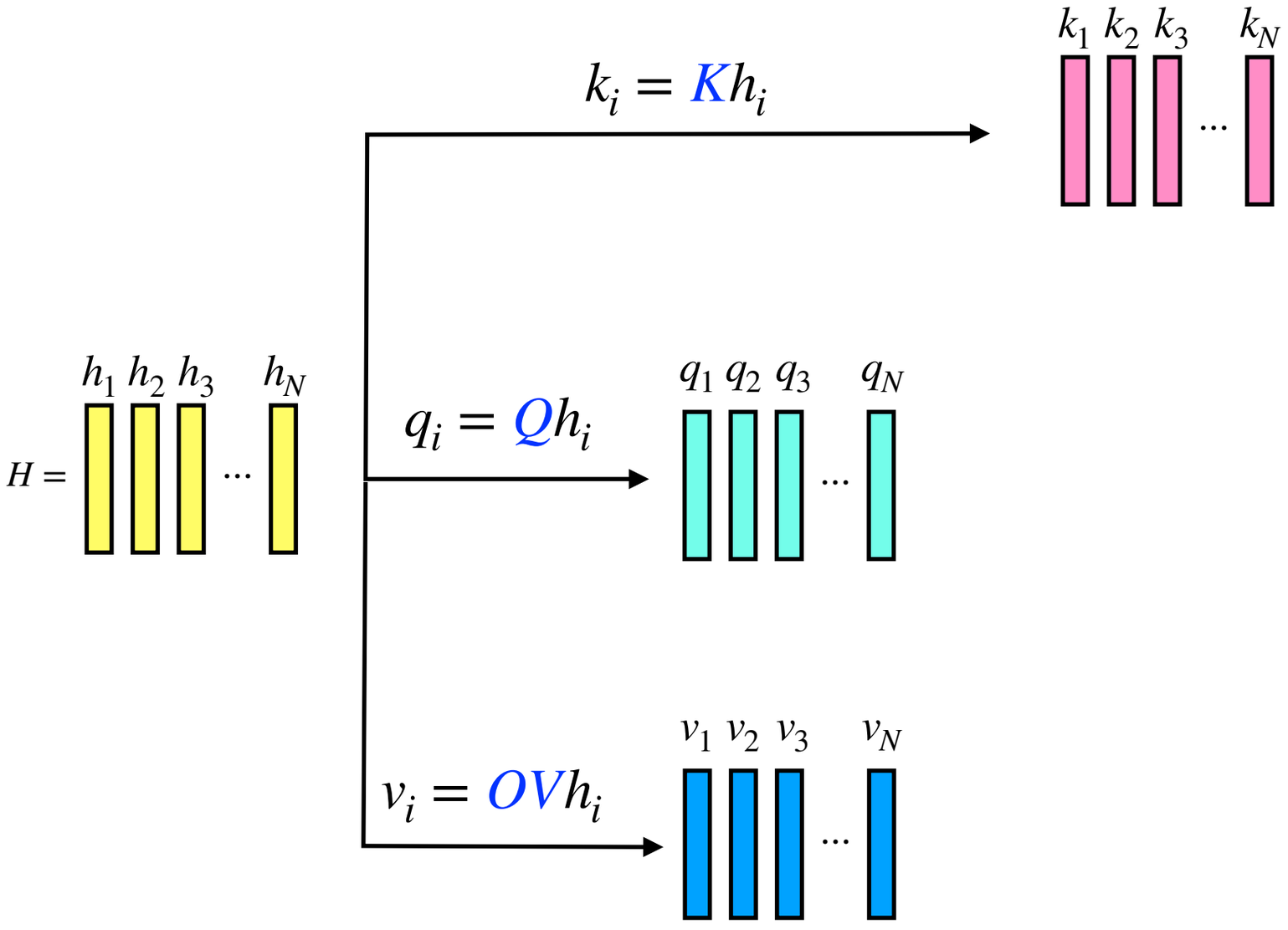
$$H' = \sum_{m=1}^{M} O_m V_m H \cdot \mathrm{Softmax}(\mathrm{mask}(H^\top {K_m}^\top Q_m H))$$



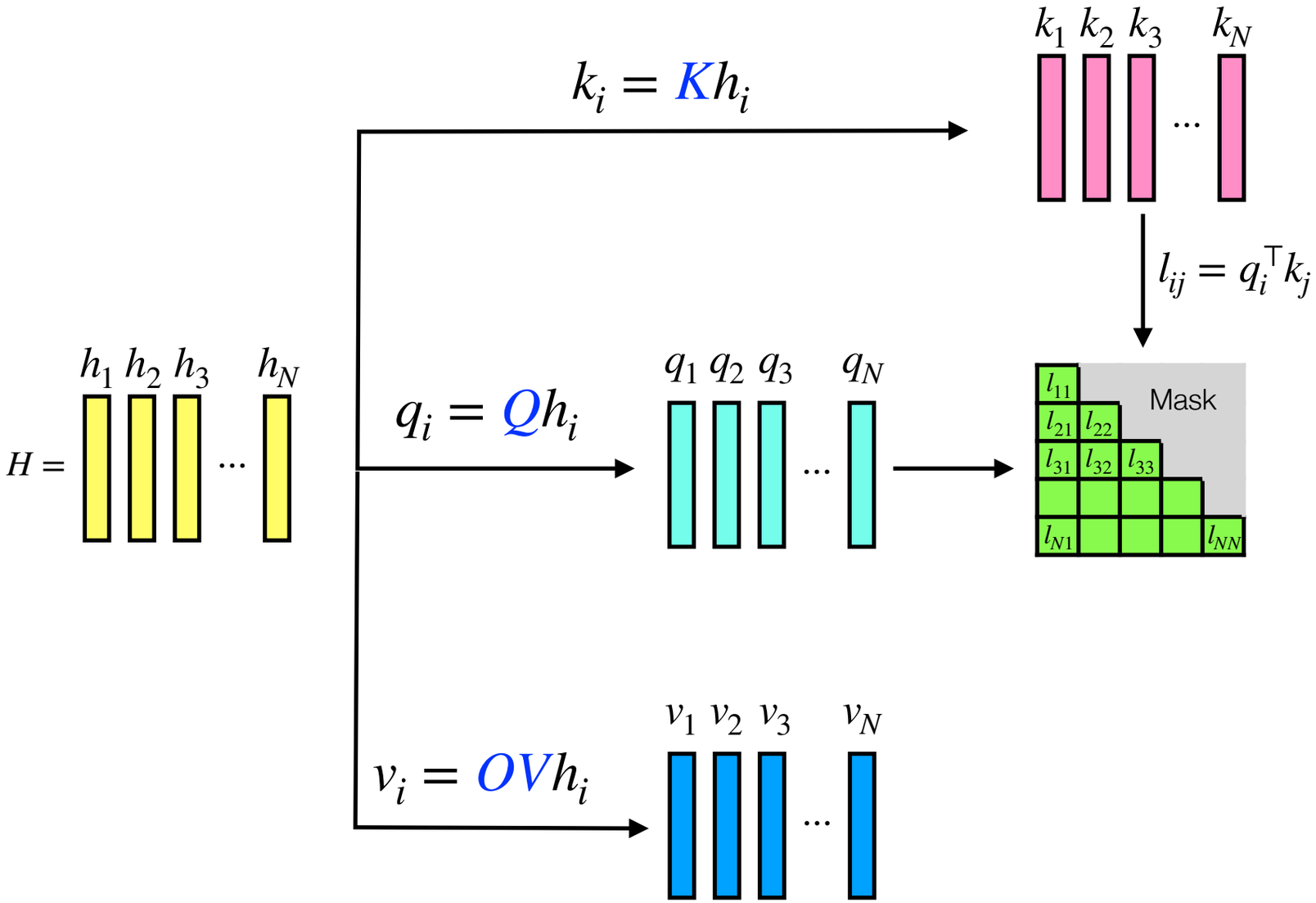
Inner structure of an attention head



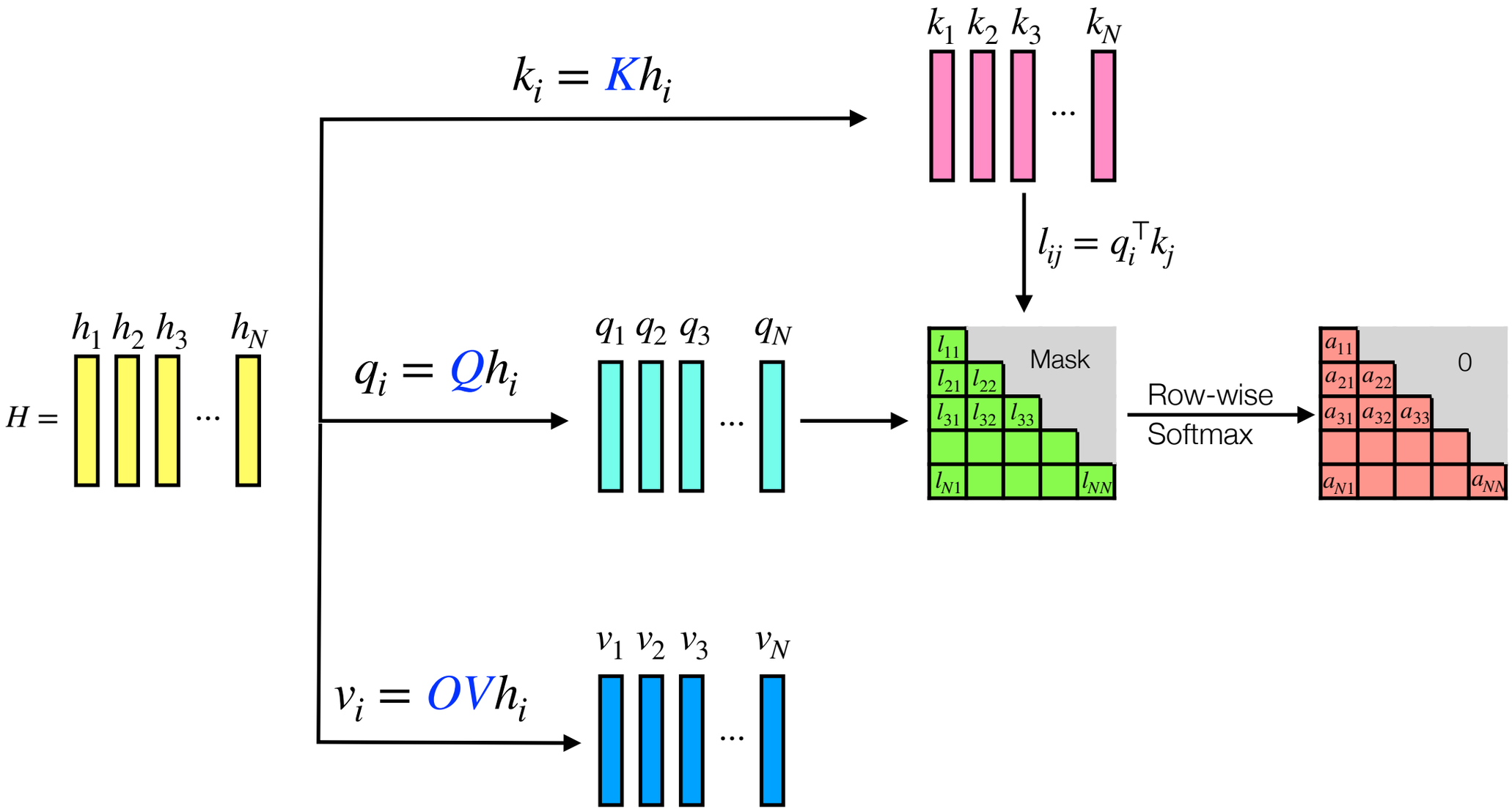
Inner structure of an attention head



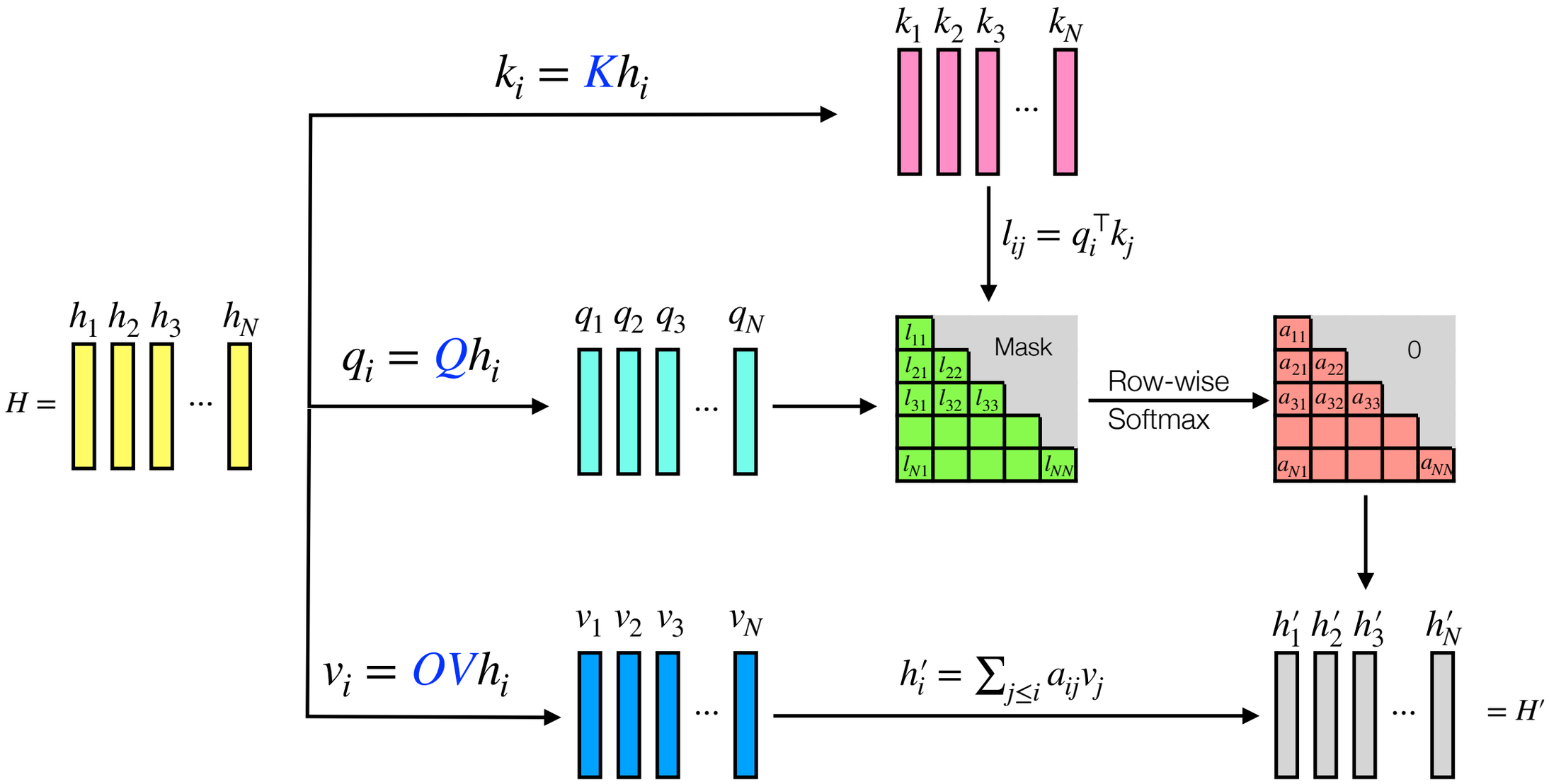
Inner structure of an attention head



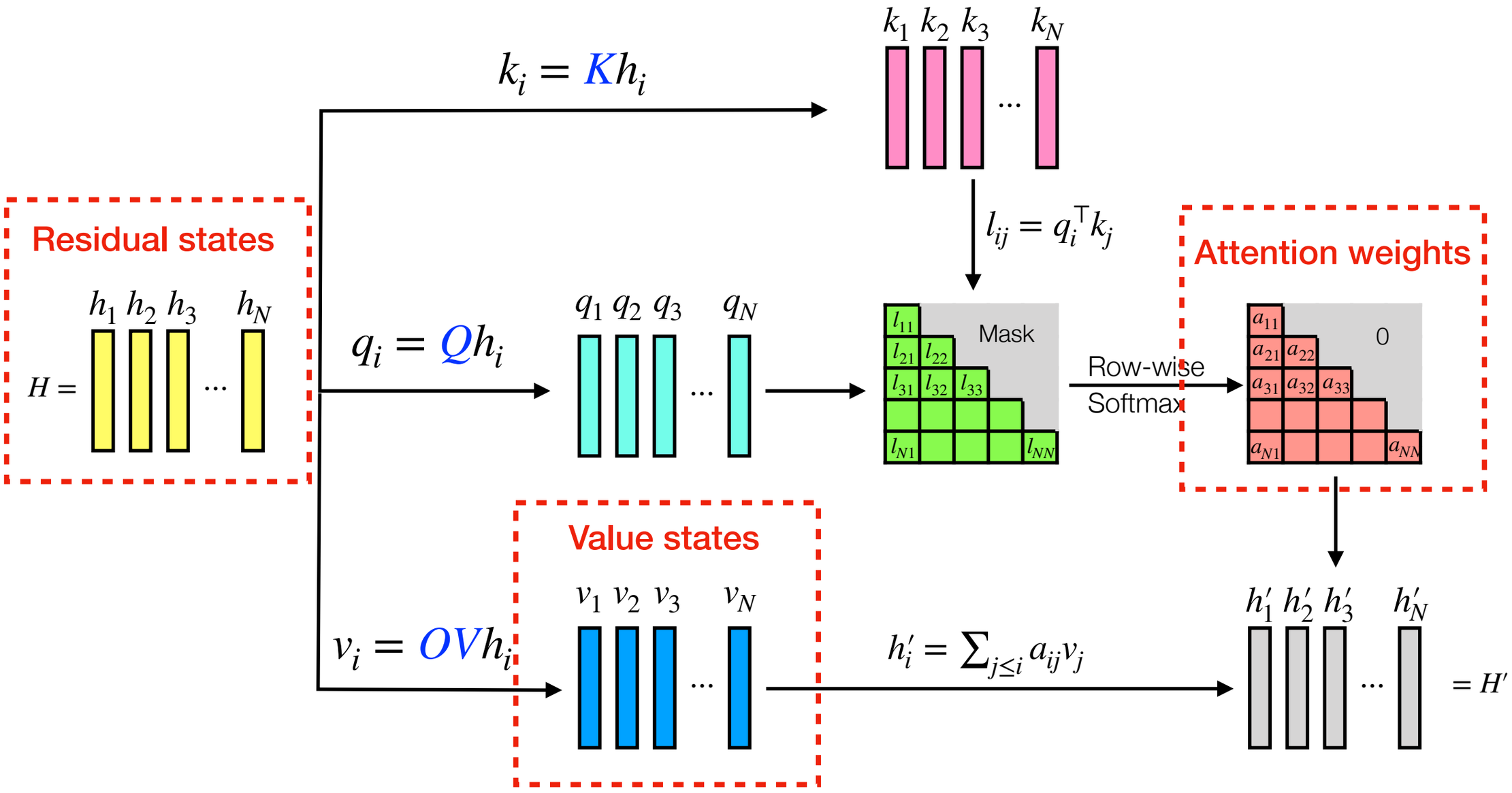
Inner structure of an attention head



Inner structure of an attention head



Inner structure of an attention head

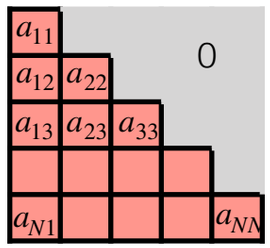


Residual states += Attention weights x Value states

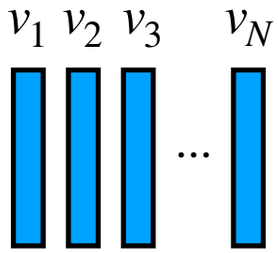
Extreme-token phenomena

Residual states += Attention weights x Value states

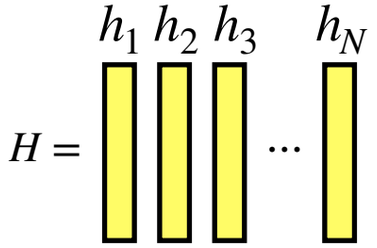
Attention weights



Value states

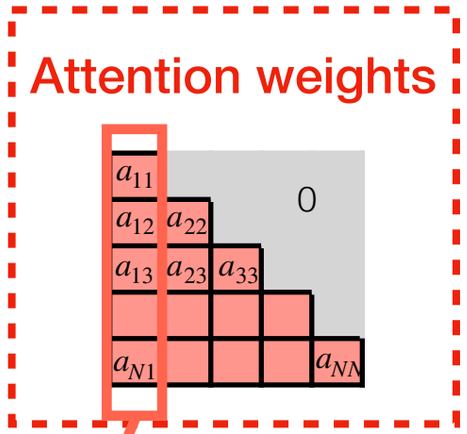


Residual states



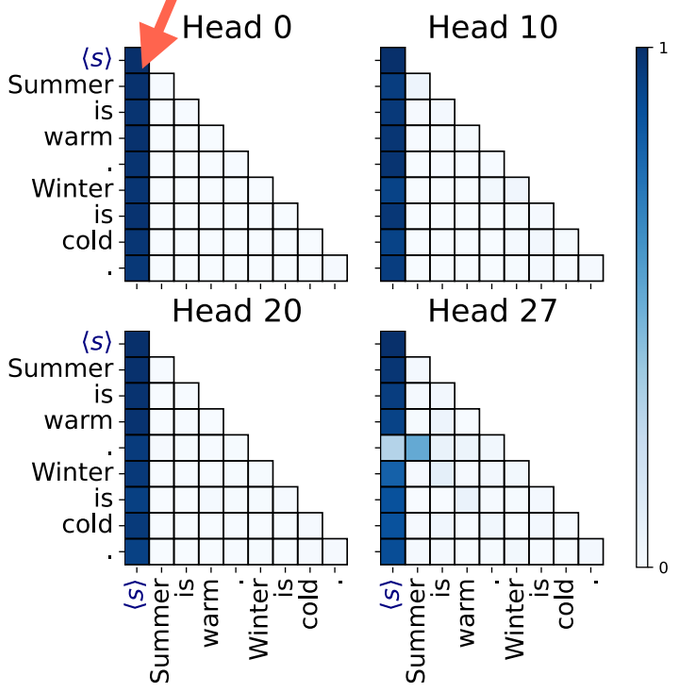
Extreme-token phenomena

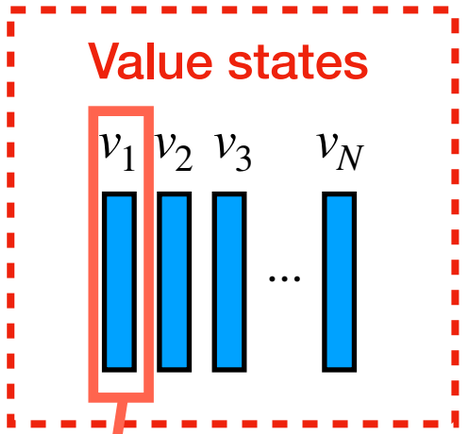
Residual states += Attention weights x Value states



Attention sinks

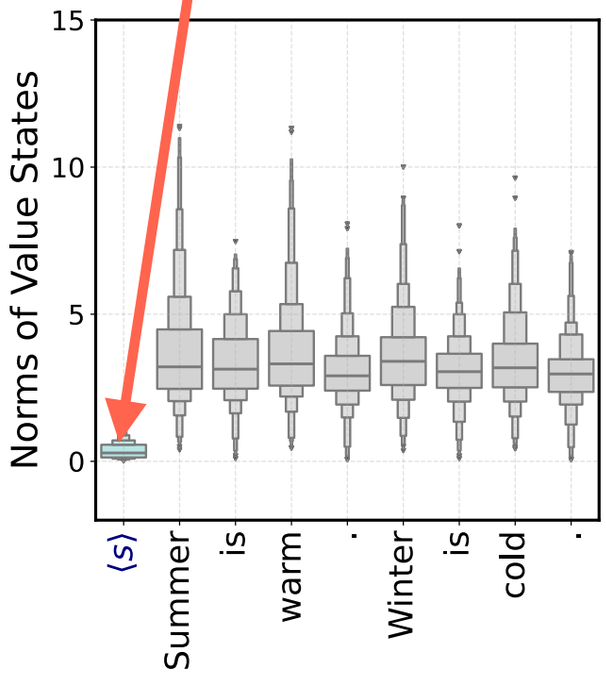
[Xiao et. al., 2023]

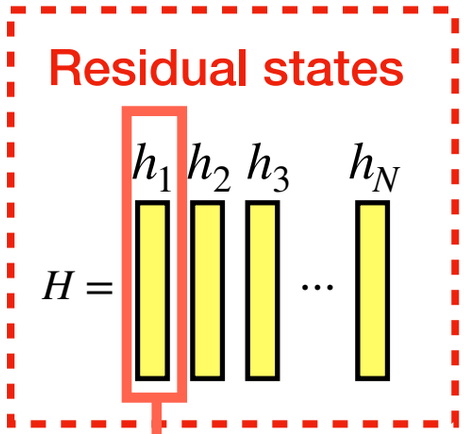




Value-state drains

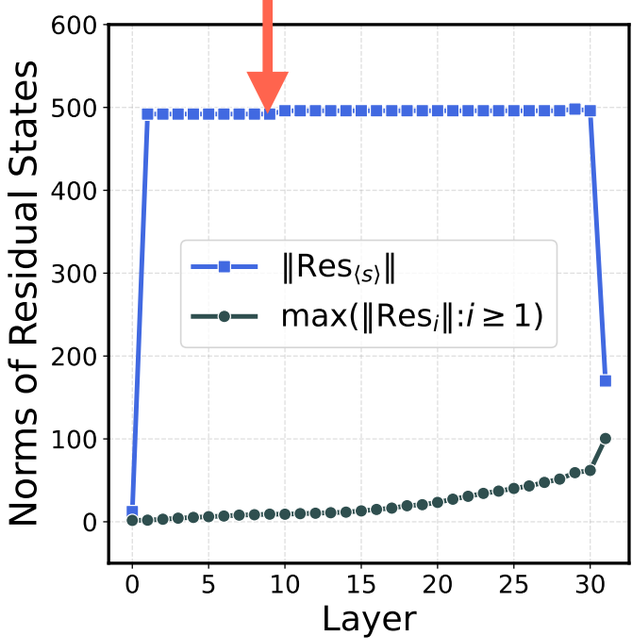
[Guo et. al., 2024]





Residual-state peaks

[Sun et. al., 2024]

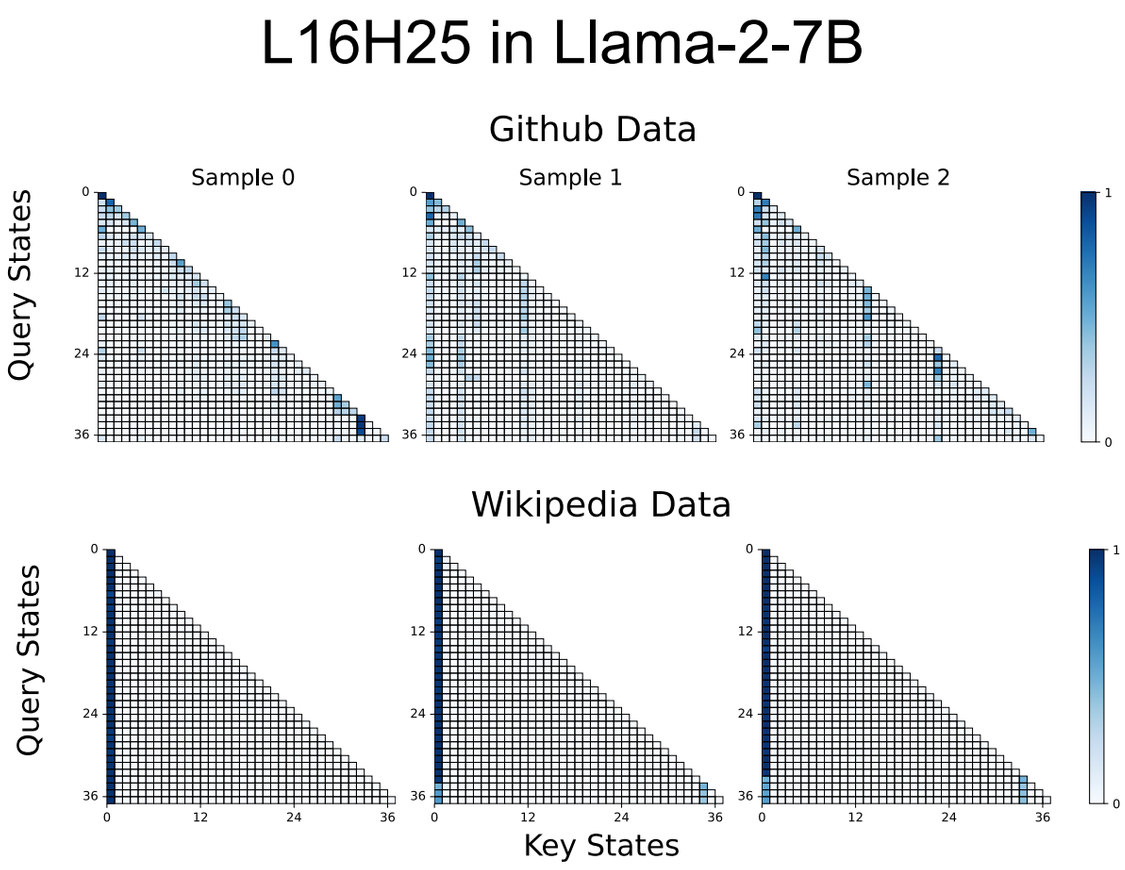


Goal

Explain why the extreme-token phenomena appear in transformer-based LLMs (Static and dynamic)

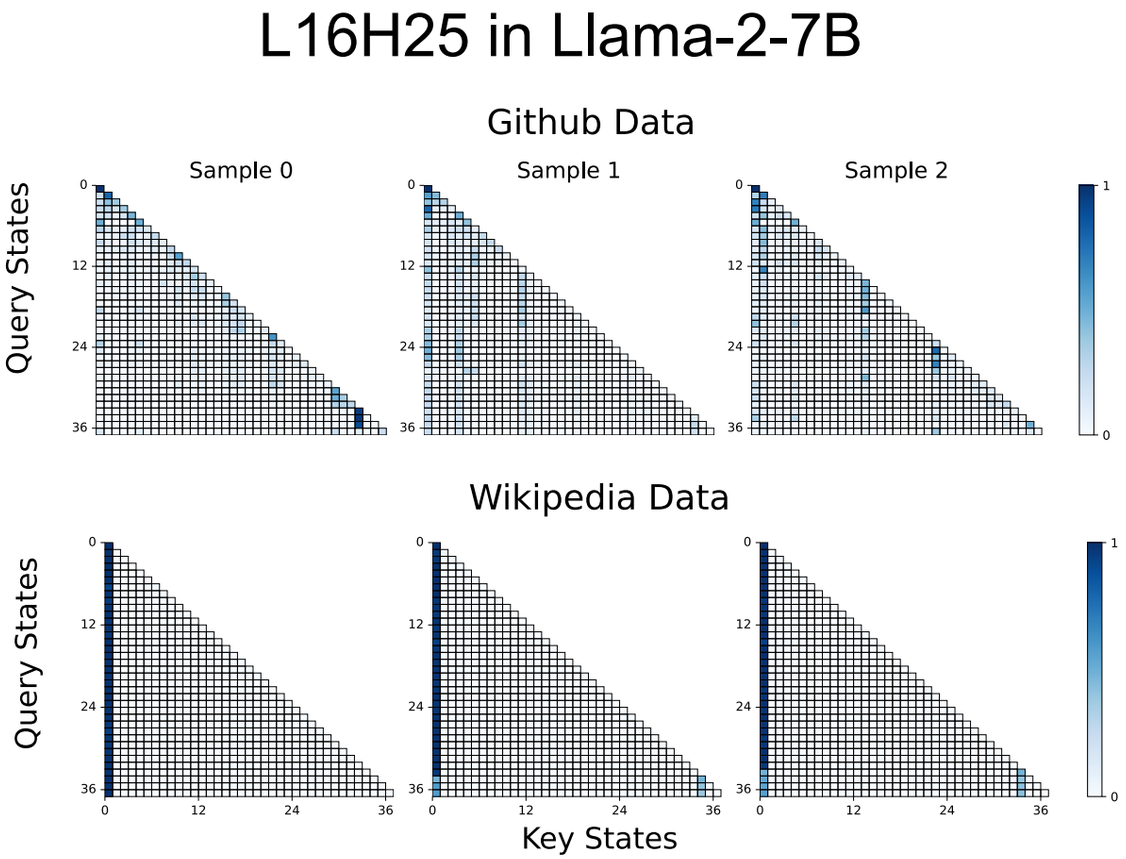
Tianyu Guo, Druv Pai, Yu Bai, Jiantao Jiao, Michael I. Jordan, and Song Mei. Active-Dormant Attention Heads: Mechanistically Demystifying Extreme-Token Phenomena in LLMs. arXiv preprint, arXiv: 2410.13835.

Active-Dormant Mechanism of attention heads

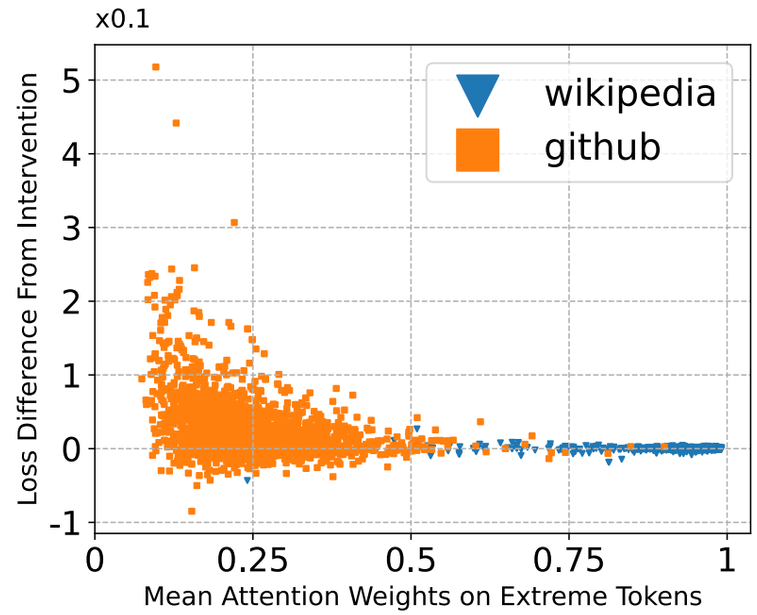


Different attention pattern on Wiki/GitHub

Active-Dormant Mechanism of attention heads

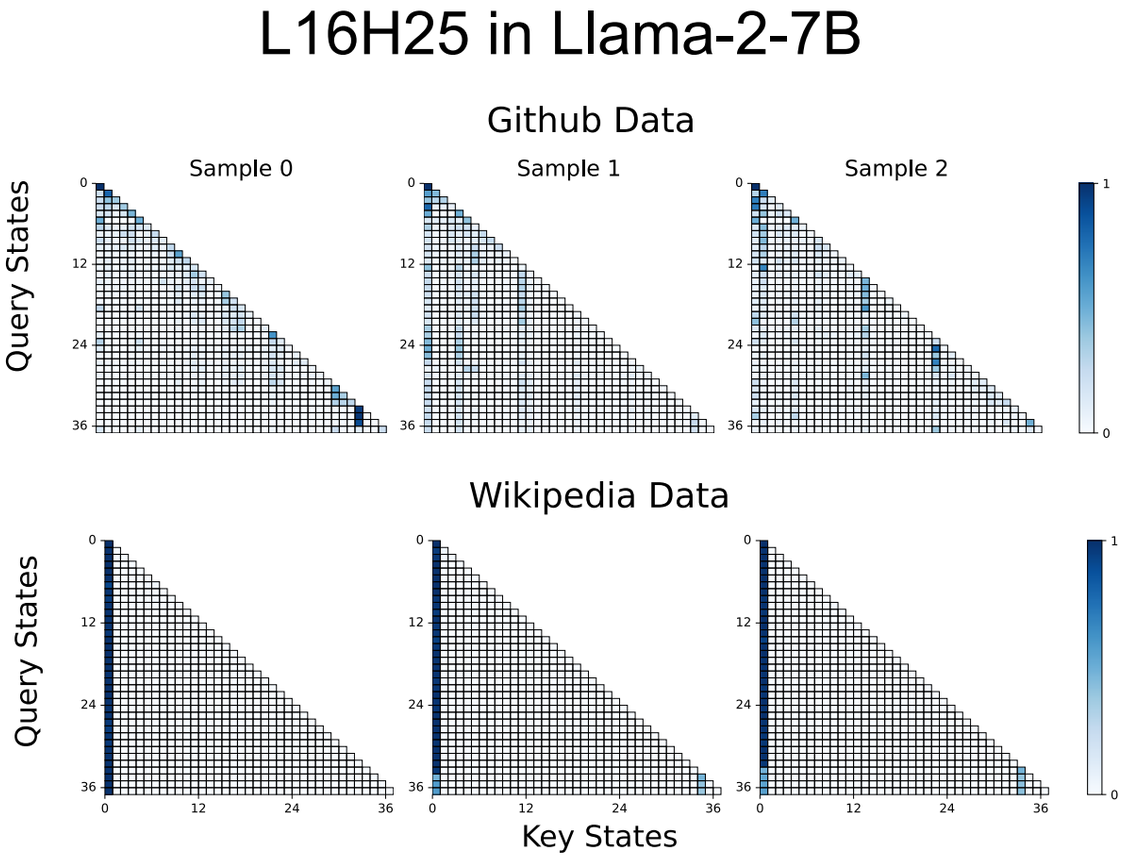


Different attention pattern on Wiki/GitHub

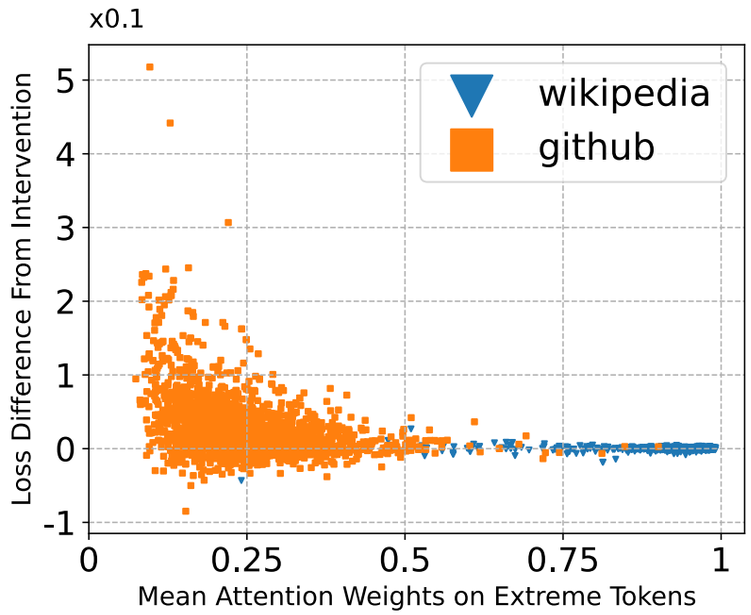


When zero-out the head, the performance of the transformer on GitHub data drops a lot.

Active-Dormant Mechanism of attention heads



Different attention pattern on Wiki/GitHub



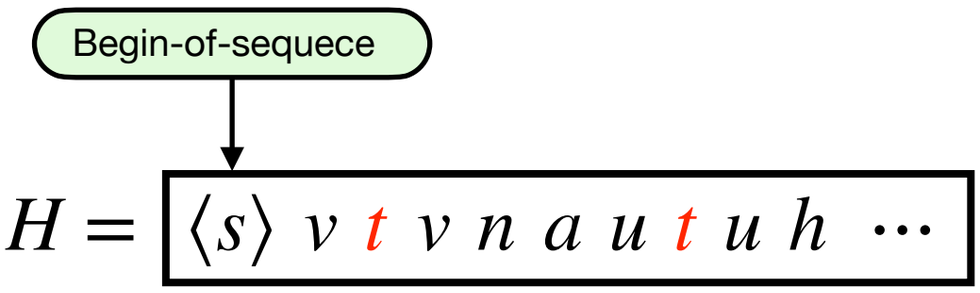
When zero-out the head, the performance of the transformer on GitHub data drops a lot.

L16H25 in Llama-2-7B is active on GitHub data and dormant on Wikipedia data

A simple statistical model and A simplified transformer

Bigram-Backcopy model

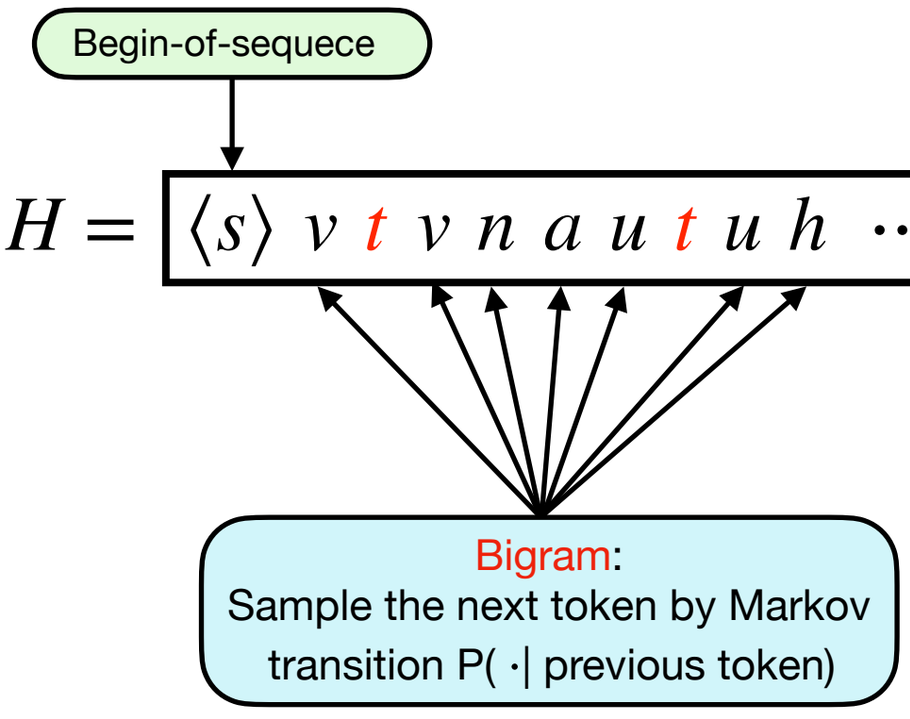
$\mathscr{V} = \{t, e\} \cup \{a, b, \cdots d, f, \cdots, s, u, \ldots\}$



A simple statistical model and A simplified transformer

Bigram-Backcopy model

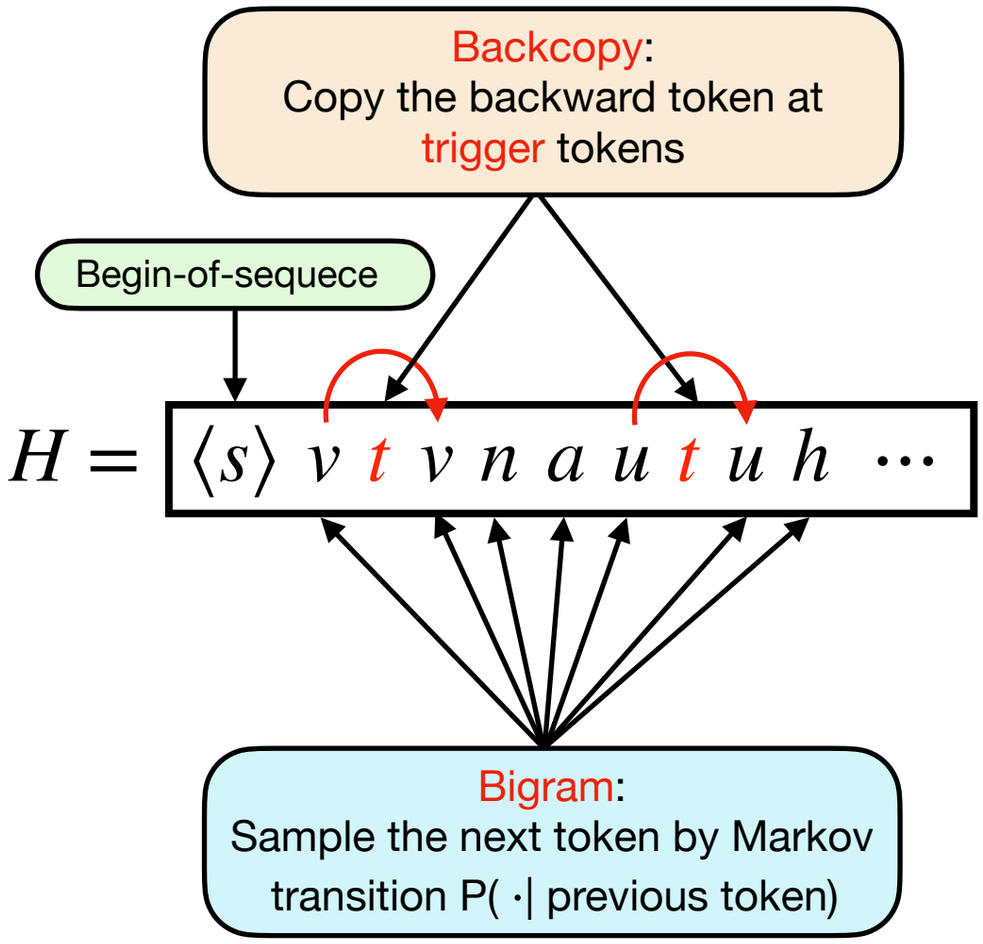
$\mathscr{V} = \{t, e\} \cup \{a, b, \cdots d, f, \cdots, s, u, \ldots\}$



A simple statistical model and A simplified transformer

Bigram-Backcopy model

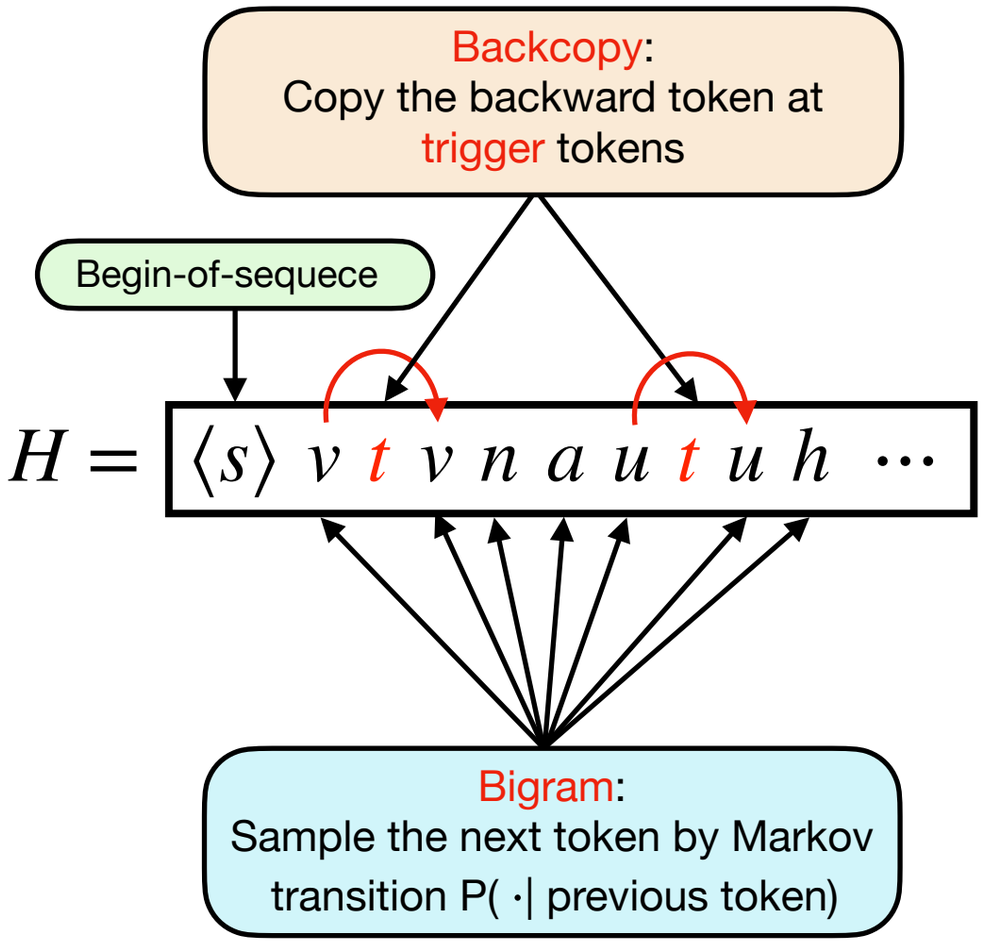
$$\mathscr{V} = \{t, e\} \cup \{a, b, \cdots d, f, \cdots, s, u, \ldots\}$$



A simple statistical model and A simplified transformer

Bigram-Backcopy model

$$\mathscr{V} = \{t, e\} \cup \{a, b, \cdots d, f, \cdots, s, u, \ldots\}$$

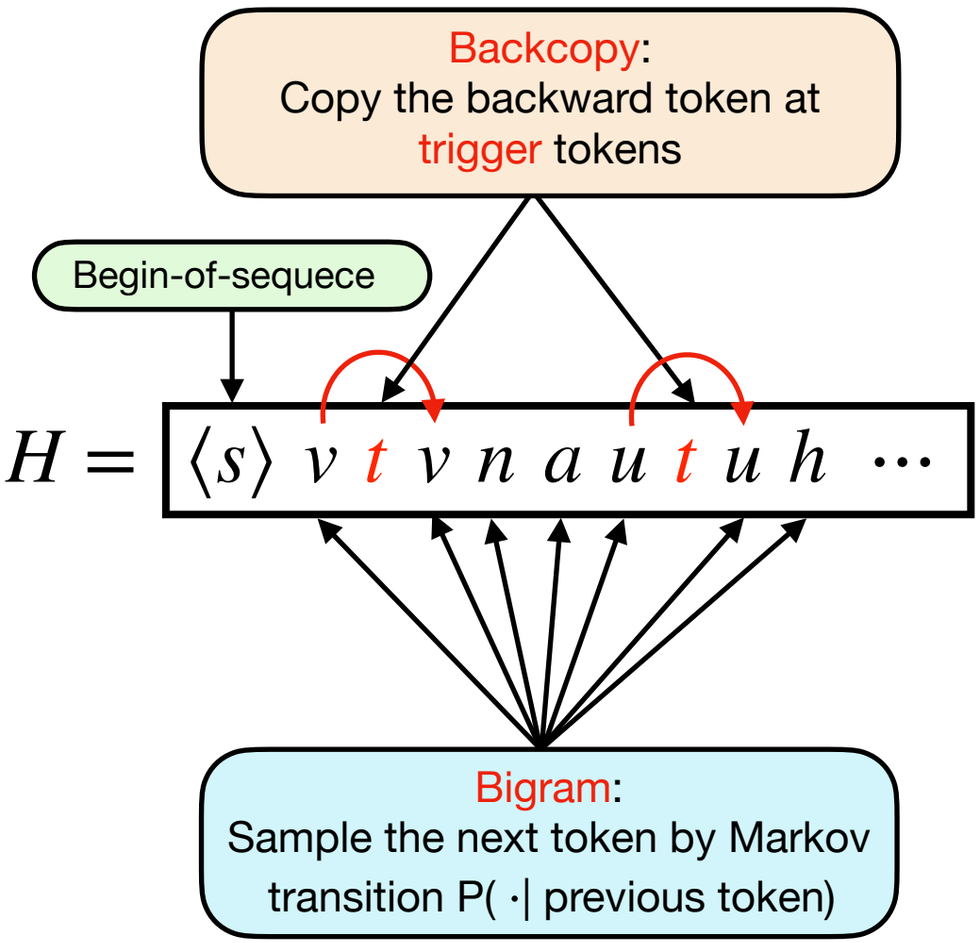


$$\text{Dataset} = \{H^{(i)}\} \sim_{iid} H$$

A simple statistical model and A simplified transformer

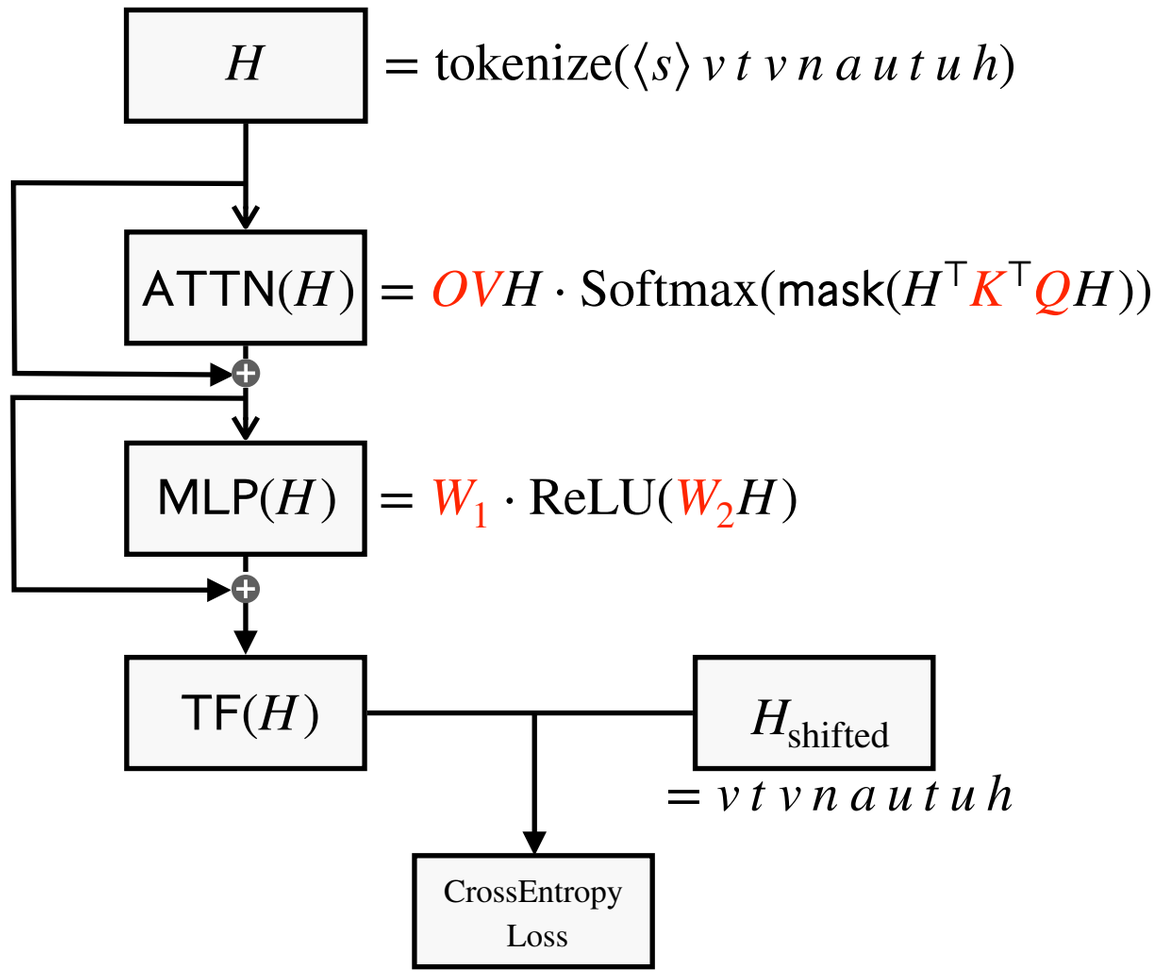
Bigram-Backcopy model

$$\mathscr{V} = \{t, e\} \cup \{a, b, \cdots d, f, \cdots, s, u, \ldots\}$$



$$\text{Dataset} = \{H^{(i)}\} \sim_{iid} H$$

Single-head single-layer transformer

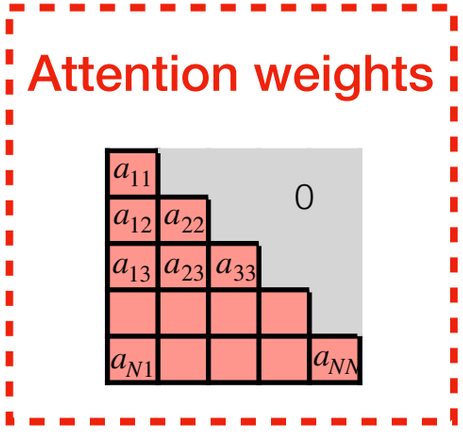


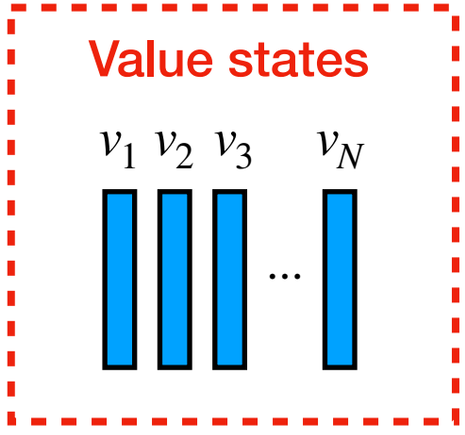
Training TF by running Adam algorithm on

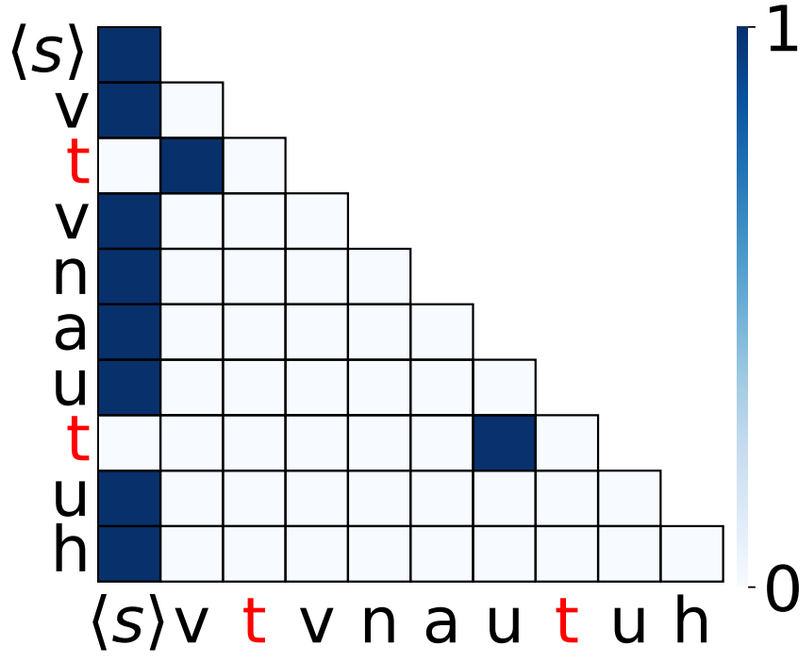
$$\text{CrossEntropy}(\text{TF}(H), H_{\text{shifted}})$$

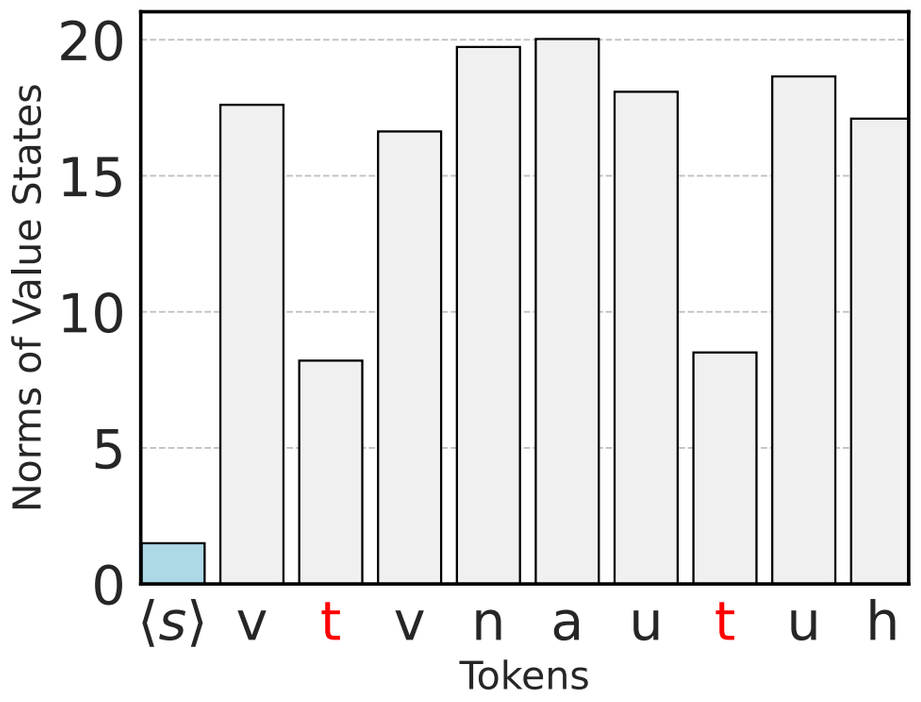
achieves optimal risk

The extreme-token phenomena in simplified model



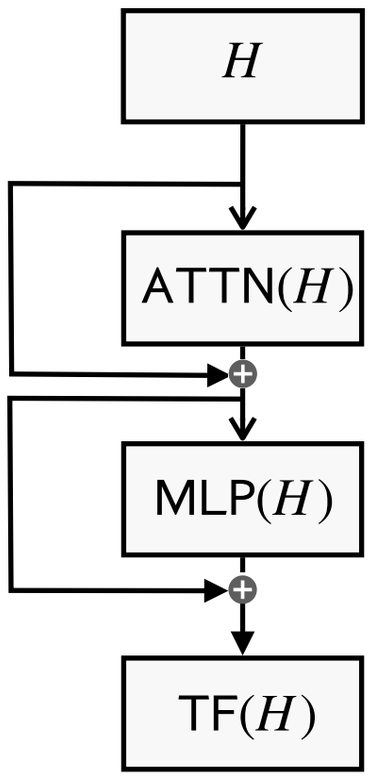


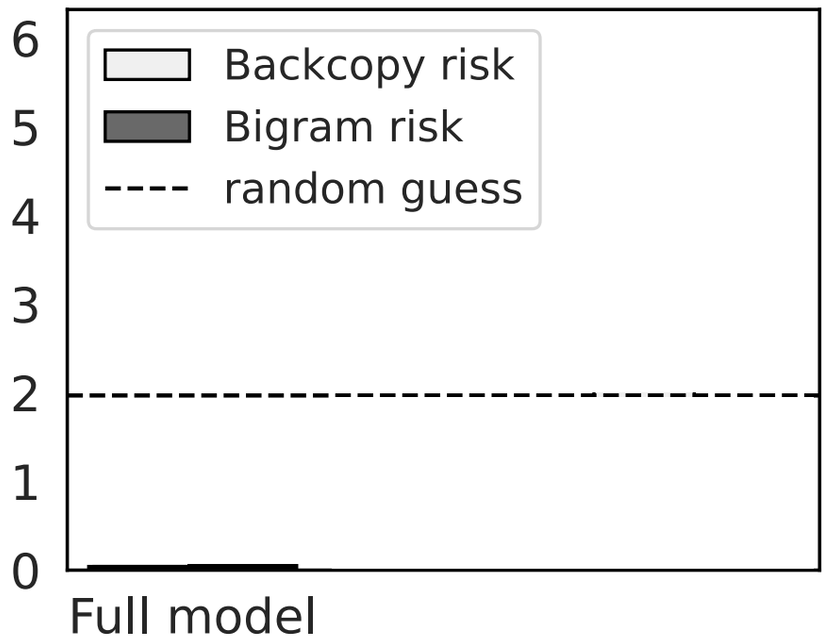




The role of MLP and Attention head in BB task

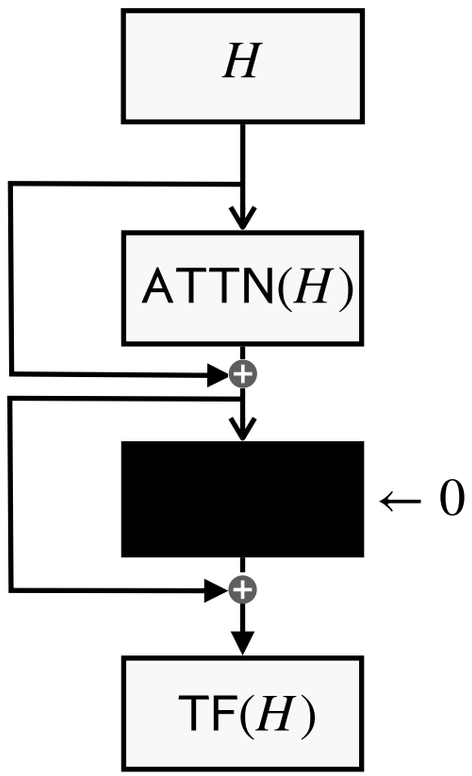
Pre-trained transformer: Full model

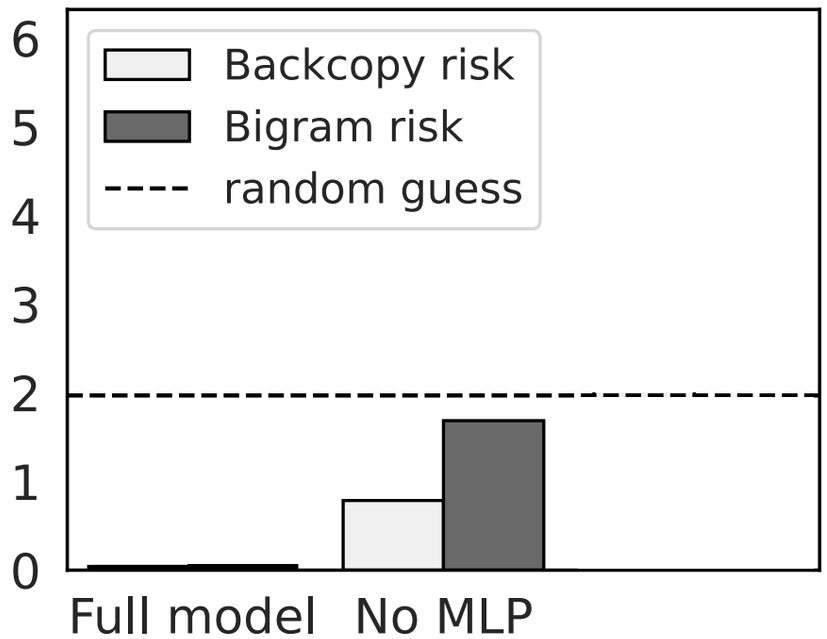




The role of MLP and Attention head in BB task

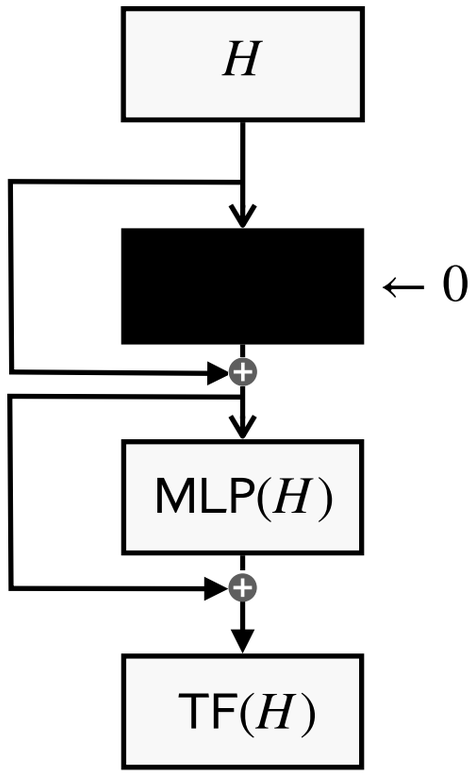
Pre-trained transformer: No MLP

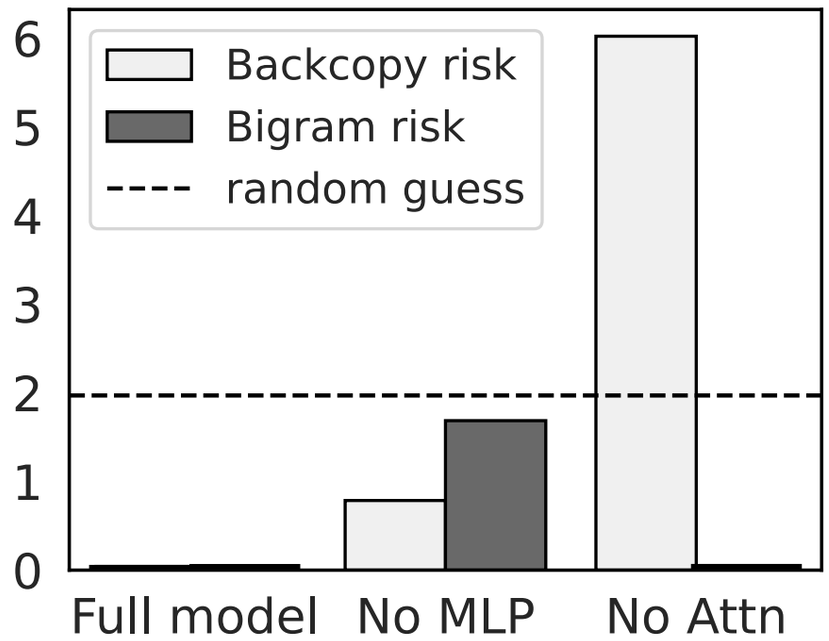




The role of MLP and Attention head in BB task

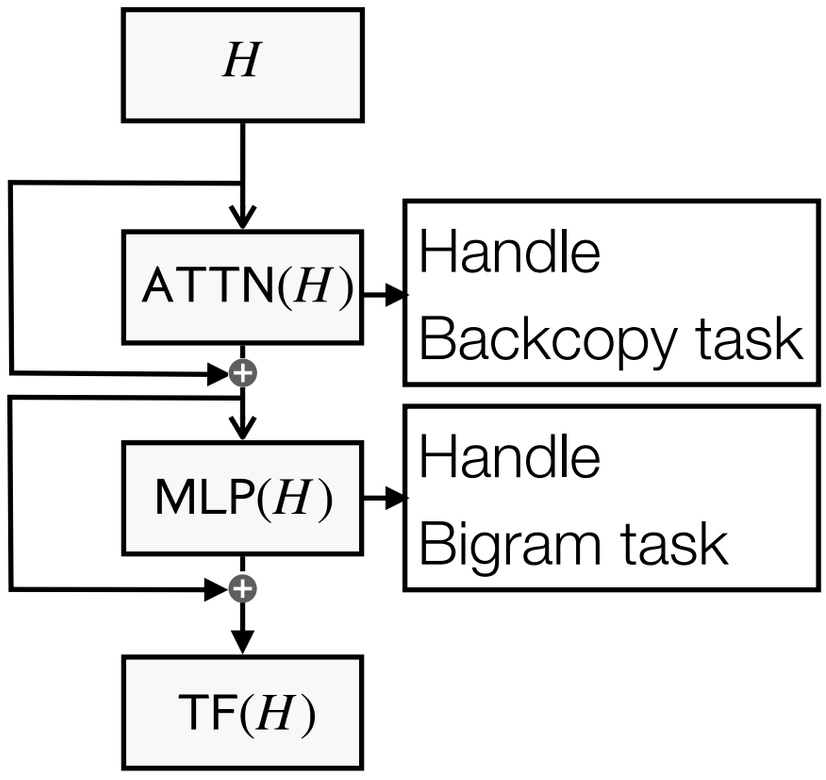
Pre-trained transformer: No Attn

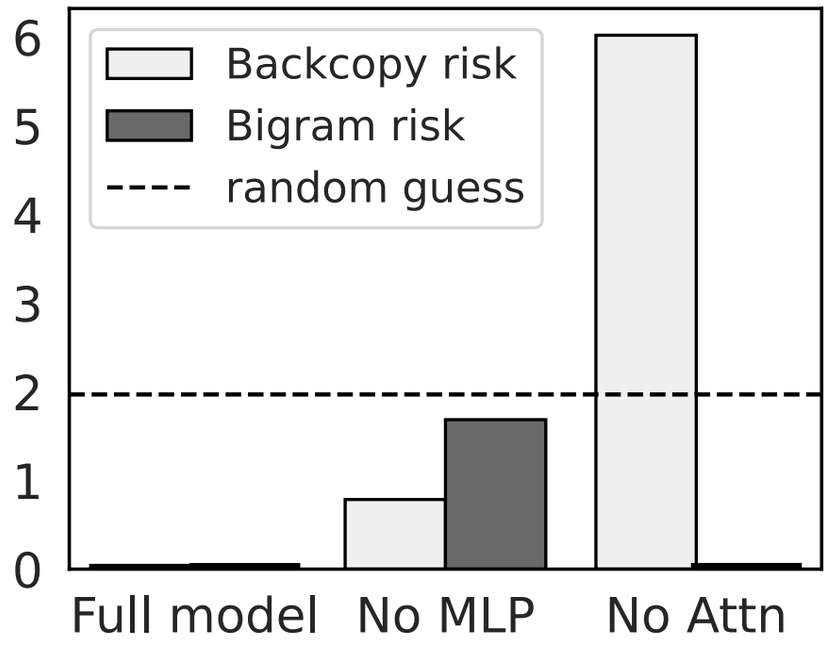




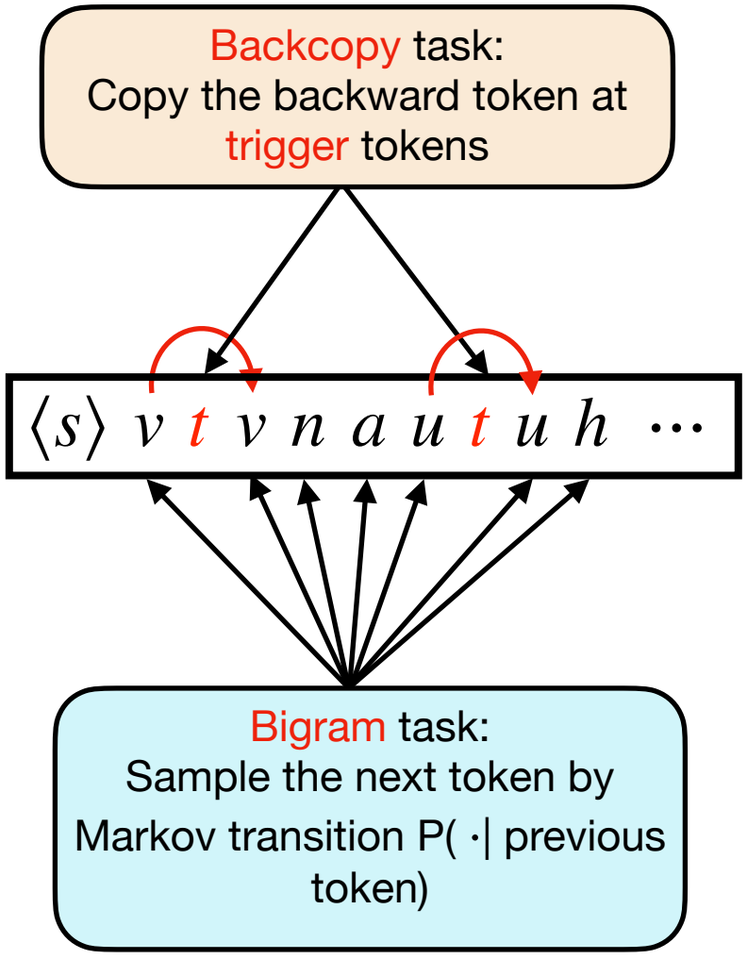
The role of MLP and Attention head in BB task

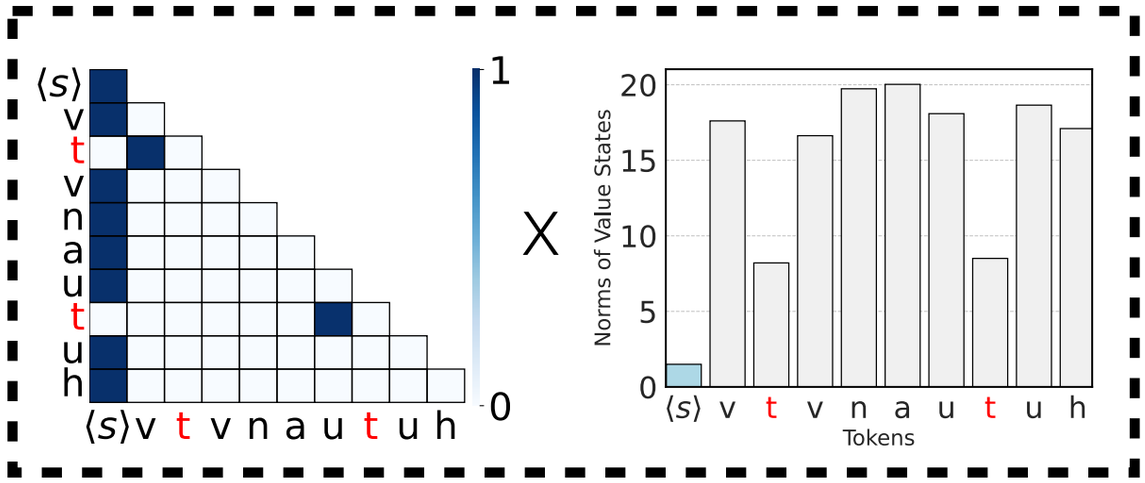
Pre-trained transformer: Full model

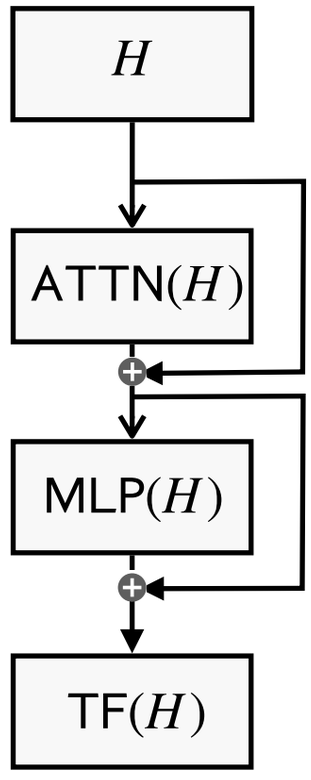




The Active-Dormant Mechanism





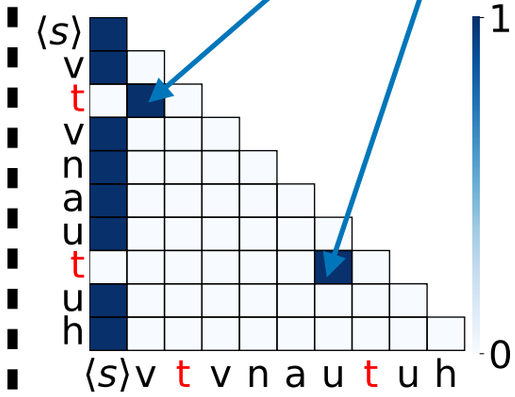


The Active-Dormant Mechanism

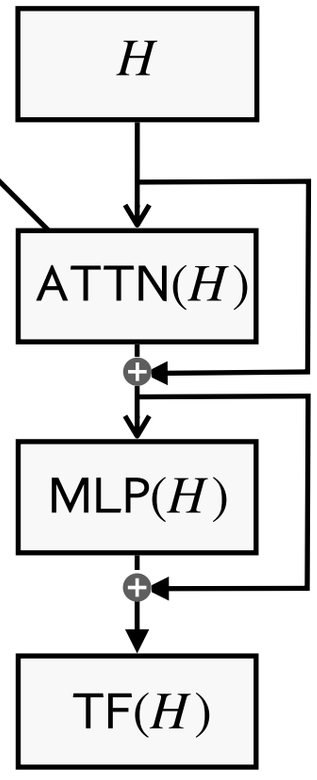
Handled by the Attention head:
For any trigger letter t, attention head picks the former token.

Backcopy task:
Copy the backward token at trigger tokens

$\langle s \rangle$ v t v n a u t u h ⋯



=



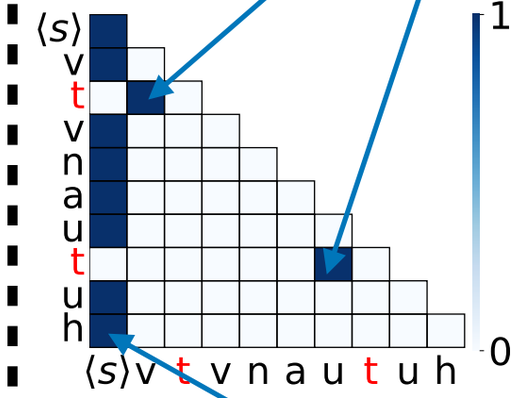
Bigram task:
Sample the next token by Markov transition P(·| previous token)

The Active-Dormant Mechanism

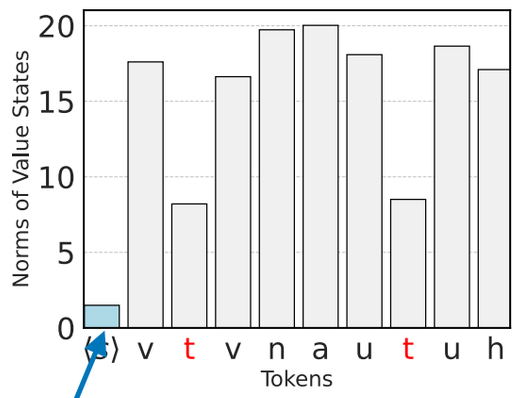
Handled by the Attention head:
For any trigger letter t, attention head picks the former token.

Backcopy task:
Copy the backward token at trigger tokens

$\langle s \rangle$ v t v n a u t u h ⋯



X



=

H

ATTN(H)

MLP(H)

TF(H)

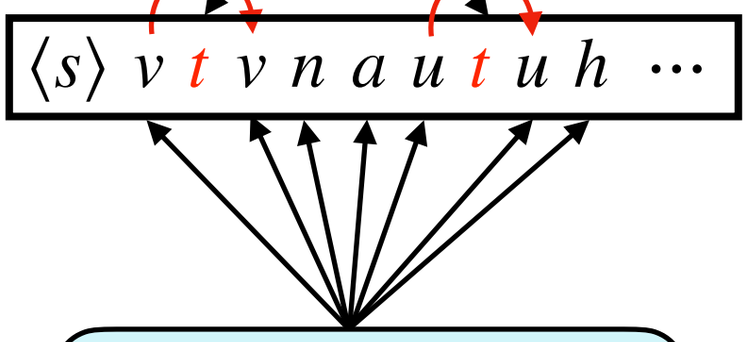
Bigram task:
Sample the next token by Markov transition P(·| previous token)

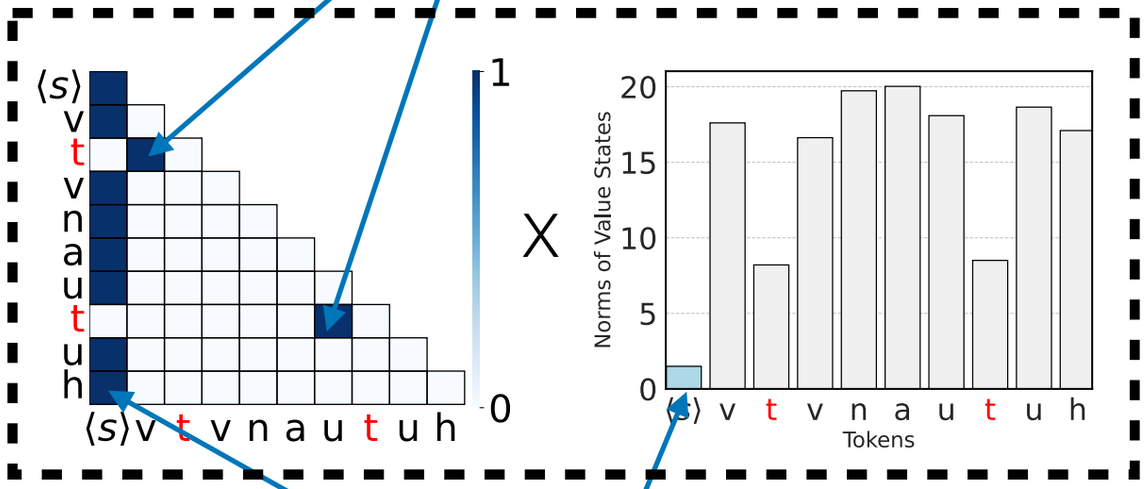
Handled by the MLP layer: for any non-trigger letter v, we have
$\mathrm{MLP}(v) = \log(P(\cdot \mid v))$.
We hope the Attention head do nothing on non-trigger letters

The Active-Dormant Mechanism

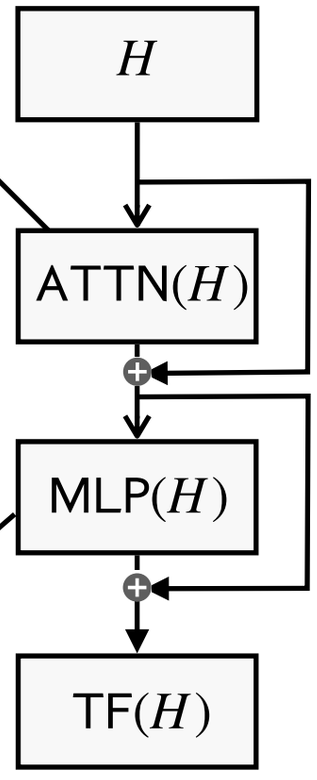
Backcopy task:
Copy the backward token at trigger tokens

Handled by the Attention head:
For any trigger letter t, attention head picks the former token.





=

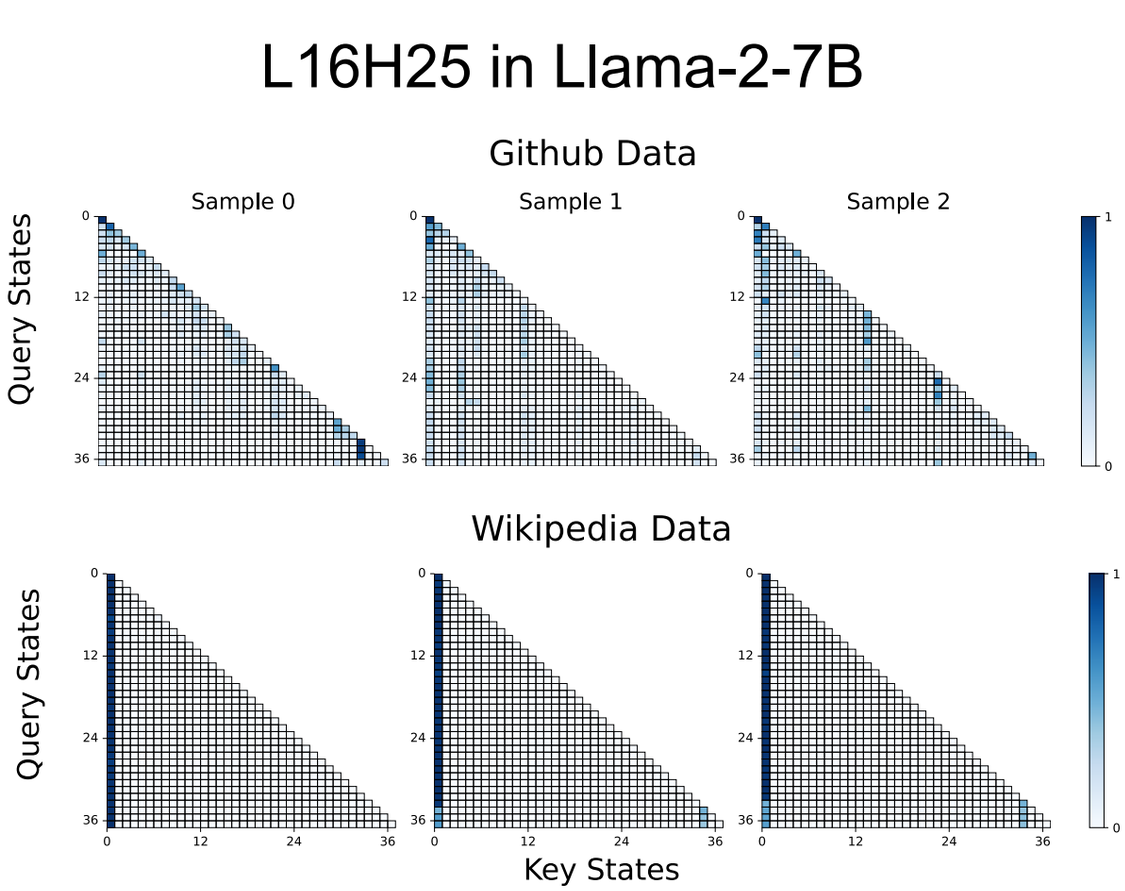


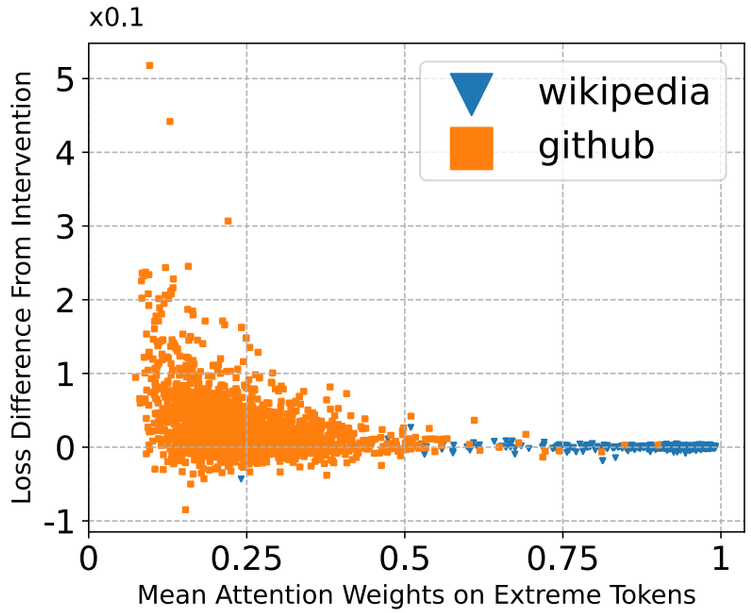
Bigram task:
Sample the next token by Markov transition P(·| previous token)

Handled by the MLP layer: for any non-trigger letter v, we have
$\mathrm{MLP}(v) = \log(P(\cdot \mid v))$.
We hope the Attention head do nothing on non-trigger letters

Attention head is active on trigger letters and dormant on non-trigger letters

Active-Dormant Mechanism of attention heads



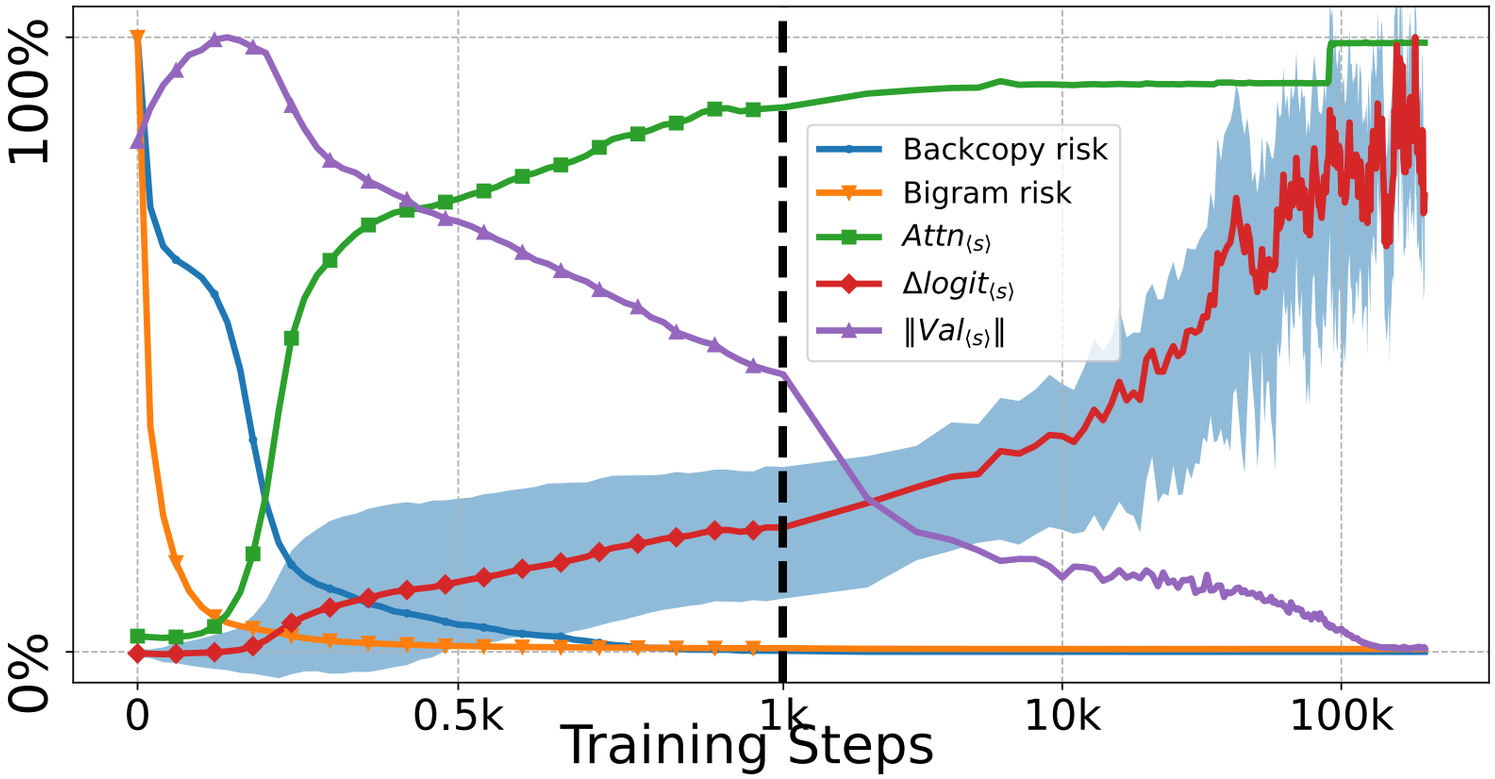


When zero-out the head, the performance of the transformer on GitHub data drops a lot.

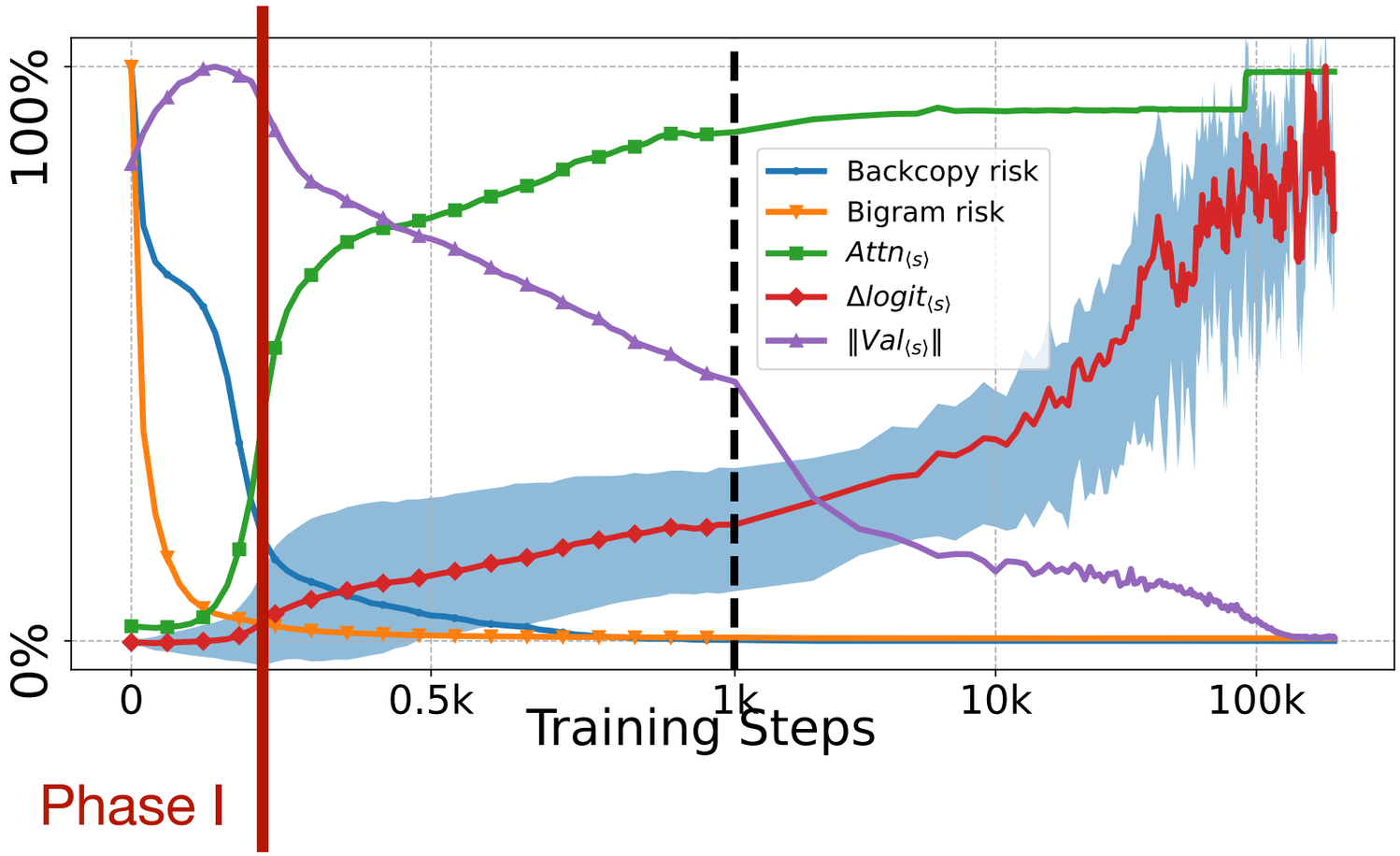
L16H25 in Llama-2-7B is active on GitHub data and dormant on Wikipedia data

How does attention sinks and value-state drains form along pre-training dynamics?

The formation of attention sink along pre-training

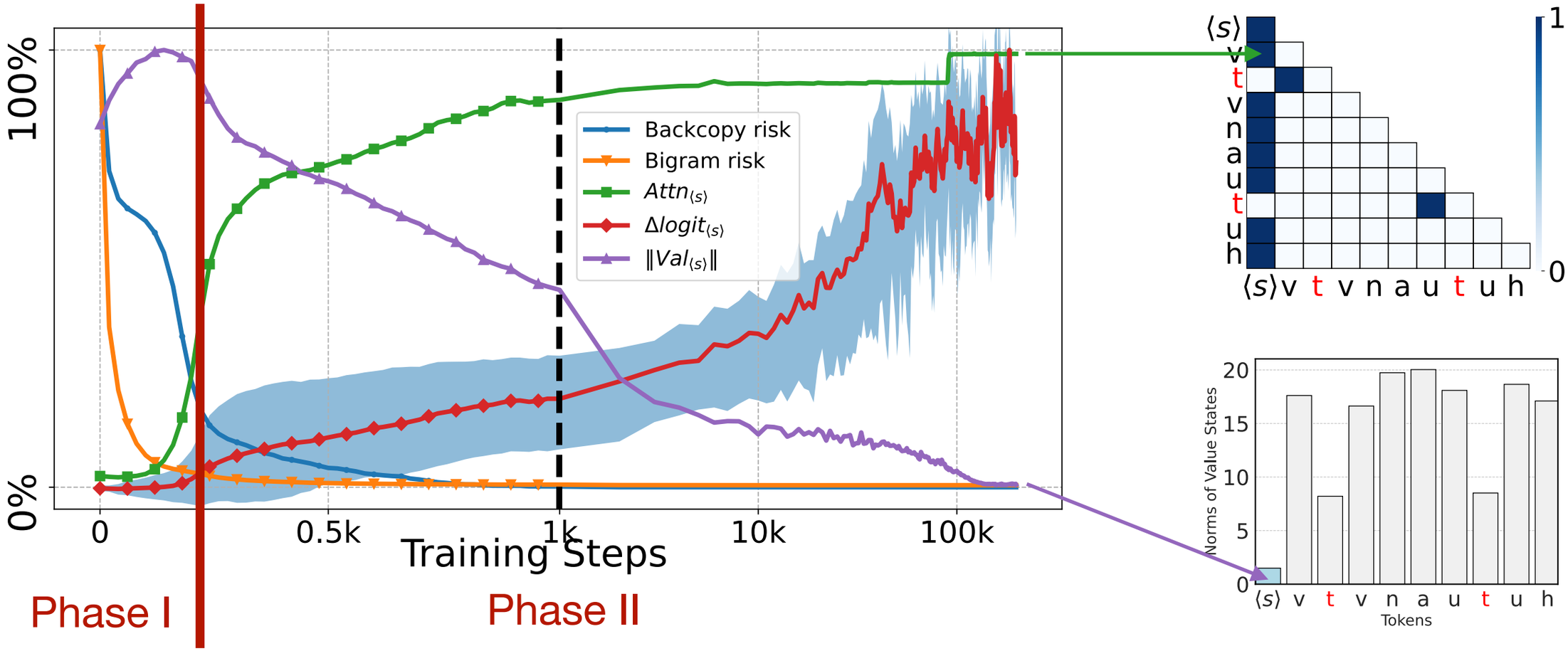


The formation of attention sink along pre-training



- Phase I: Bigram excess risk achieves near 0; Backcopy risk drops to a small value.

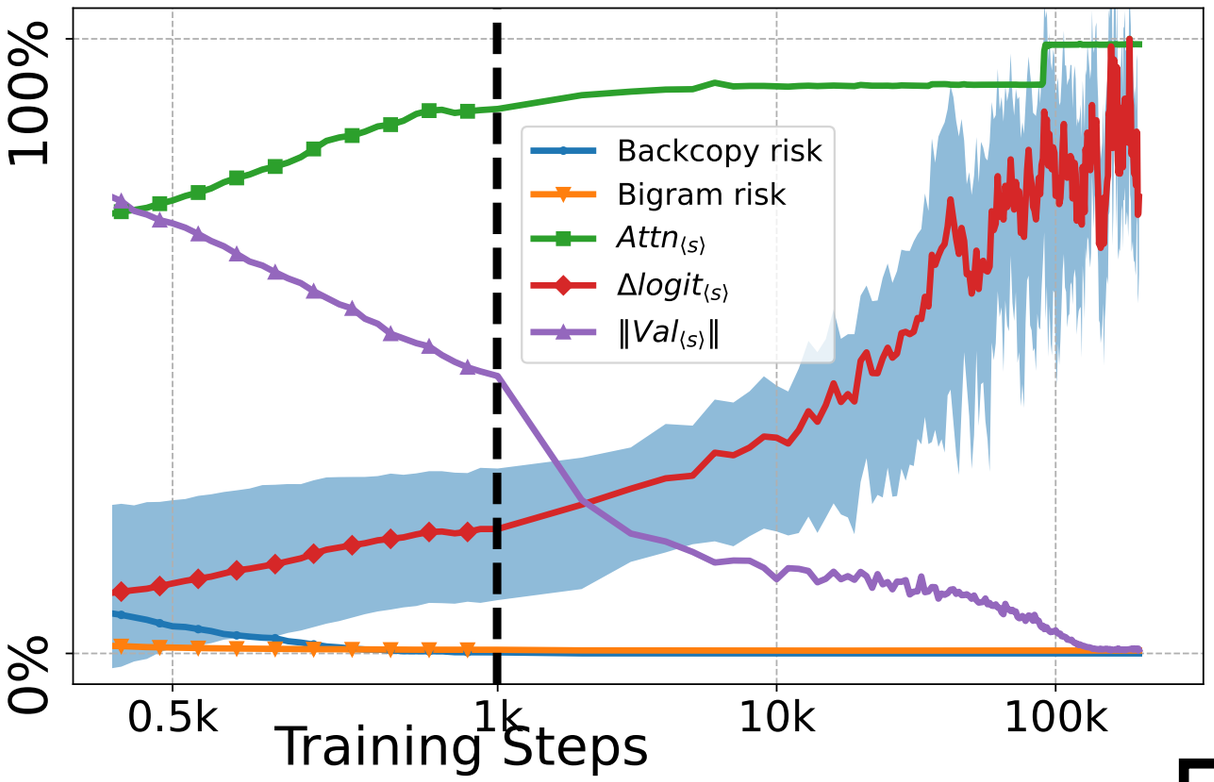
The formation of attention sink along pre-training

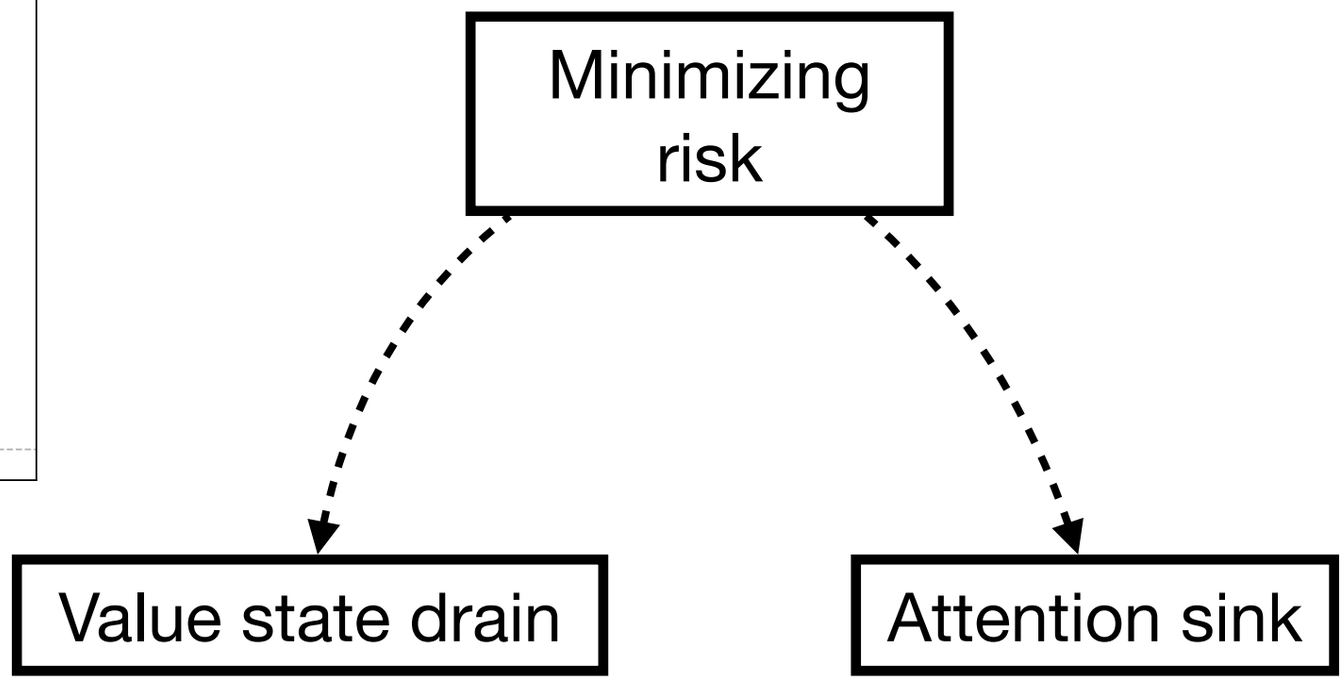


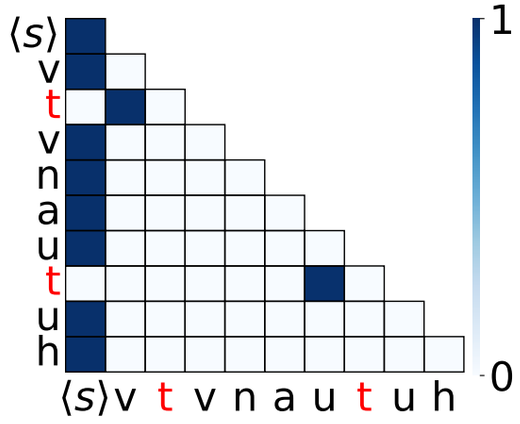
- Phase I: Bigram excess risk achieves near 0; Backcopy risk drops to a small value.

- Phase II: Backcopy risk continues to decrease; attention sinks and value-state drains begin to form, and go extreme even if Backcopy risk has been very close to 0.

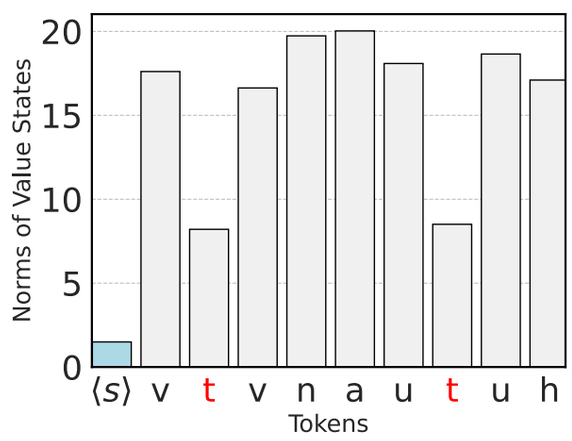
The Mutual Reinforcement Mechanism





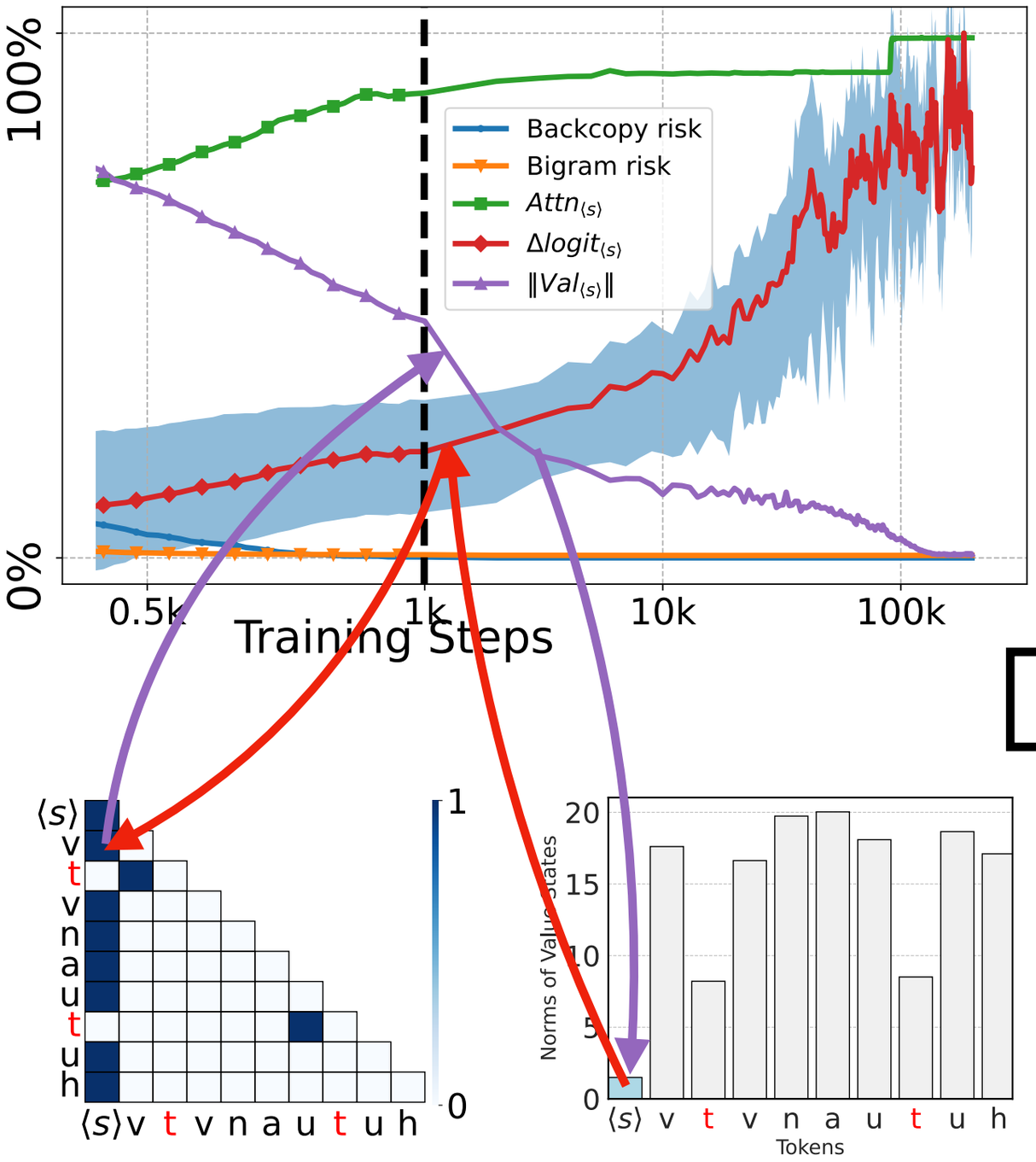


Attention sinks



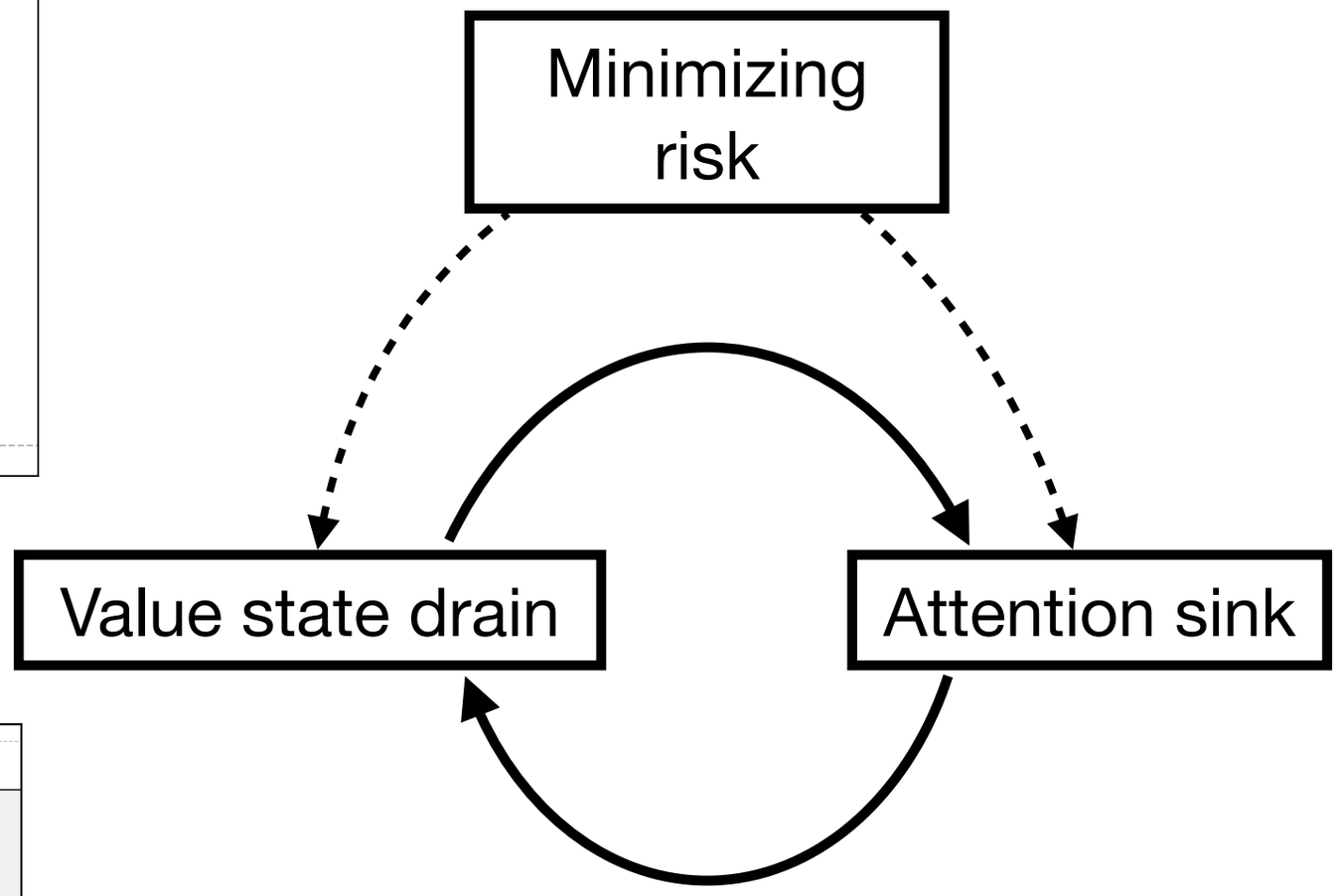
Value-state drains

The Mutual Reinforcement Mechanism

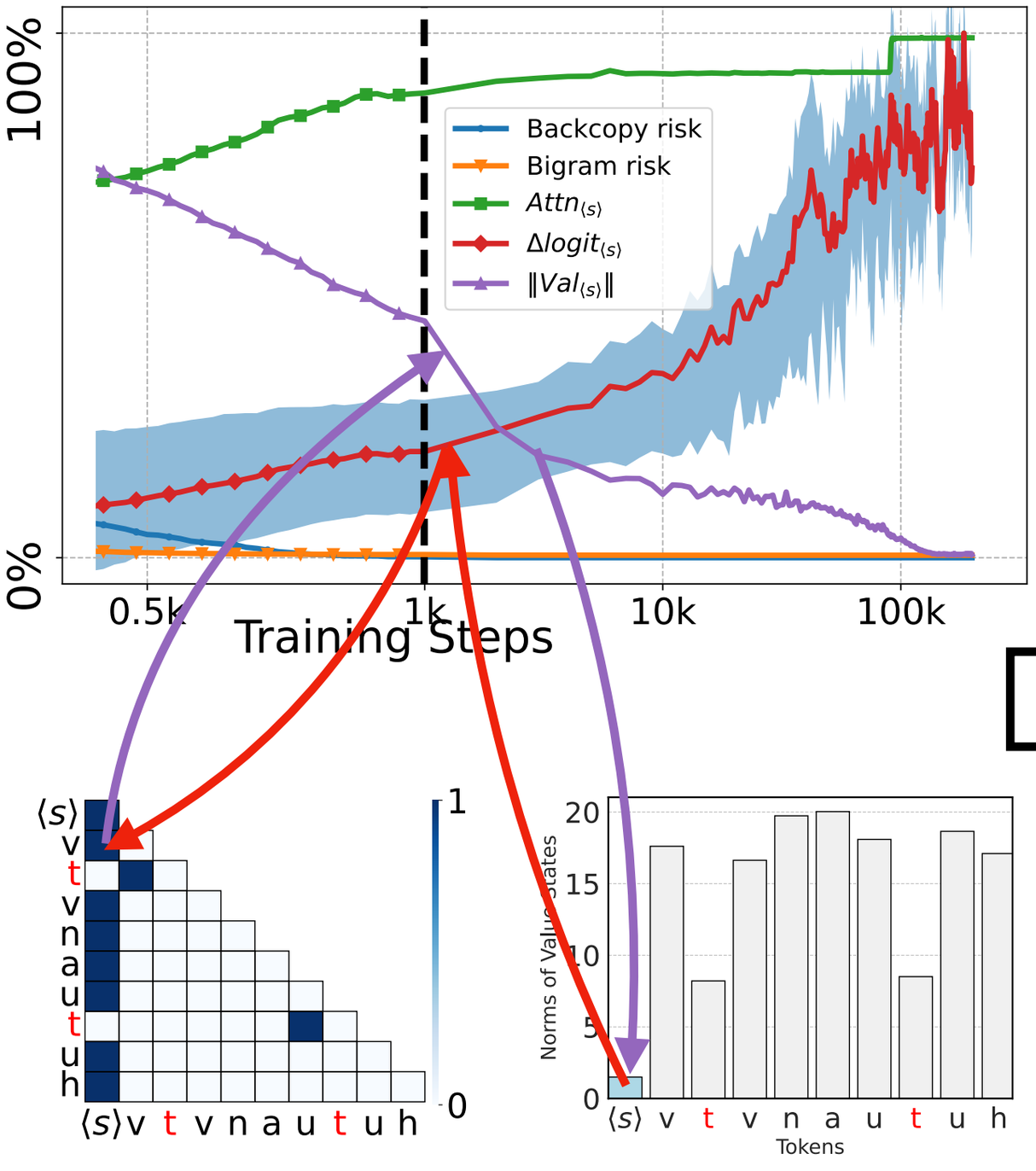


Attention sinks

Value-state drains



The Mutual Reinforcement Mechanism



Attention sinks

Value-state drains

Theorem [Our result; Informal]:

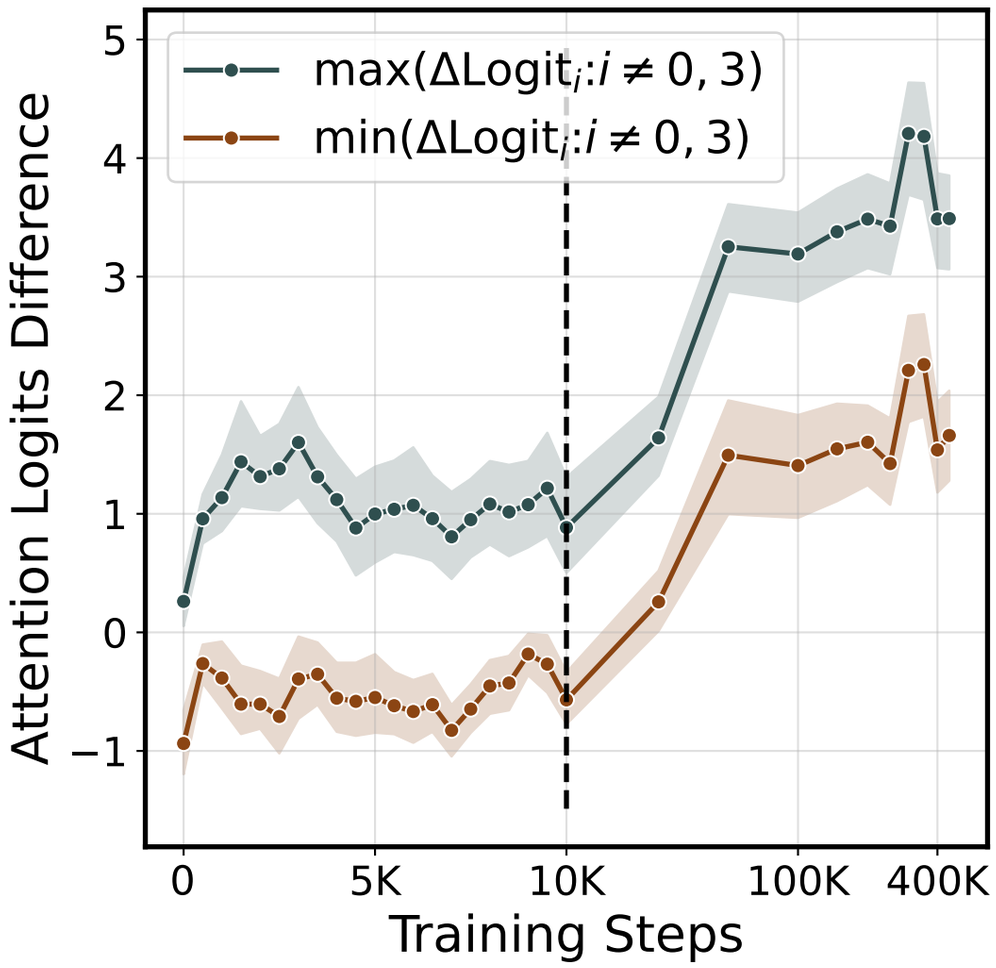
If the model can accurately predict the next token without using the attention head, but adding any value state from previous tokens worsens the prediction, the attention head will become dormant and form an attention sink. Dynamically, this arises from a mutual reinforcement mechanism:

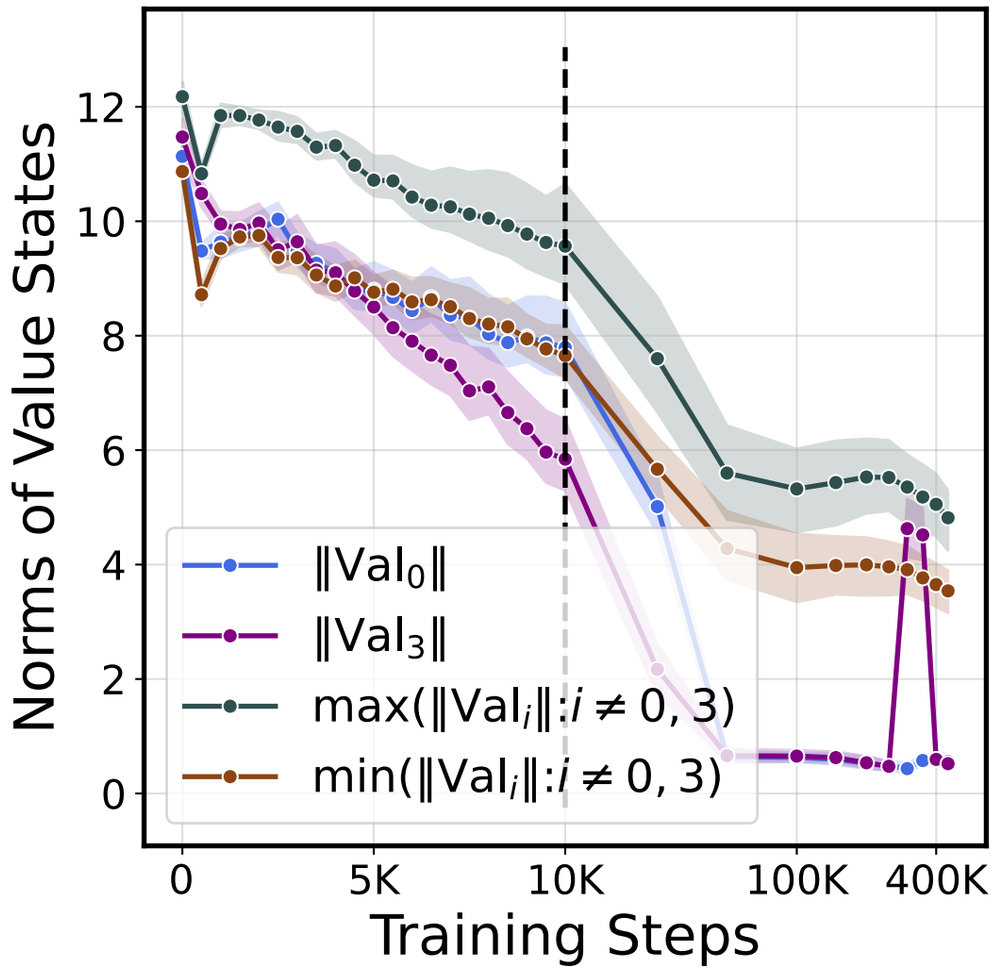
- The SoftMax function shifts attention weights towards tokens that exhibit value-state drains, reinforcing these tokens as attention sinks.
- Attention sinks on these extreme tokens further suppress their value states, reinforcing their role as value-state drains.

Examine mutual reinforcement in LLMs: the OLMo model

Models	arc challenge	arc easy	boolq	hella-swag	open bookqa	piqa	sciq	wino-grande	avg.
StableLM 1.6B	43.8	63.7	76.6	68.2	45.8	74.0	94.7	64.9	66.5
Pythia 1B	33.1	50.2	61.8	44.7	37.8	69.1	86.0	53.3	54.5
TinyLlama 1.1B	34.8	53.2	64.6	58.7	43.6	71.1	90.5	58.9	59.4
OLMo-1B	34.5	58.1	60.7	62.5	46.4	73.7	88.1	58.9	60.4
Falcon-7B	47.5	70.4	74.6	75.9	53.0	78.5	93.9	68.9	70.3
LLaMA 7B	44.5	67.9	75.4	76.2	51.2	77.2	93.9	70.5	69.6
Llama 2 7B	48.5	69.5	80.2	76.8	48.4	76.7	94.5	69.4	70.5
MPT-7B	46.5	70.5	74.2	77.6	48.6	77.3	93.7	69.9	69.8
Pythia 6.9B	44.1	61.9	61.1	63.8	45.0	75.1	91.1	62.0	63.0
RPJ-INCITE-7B	42.8	68.4	68.6	70.3	49.4	76.0	92.9	64.7	66.6
OLMo-7B	48.5	65.4	73.4	76.4	50.4	78.4	93.8	67.9	69.3

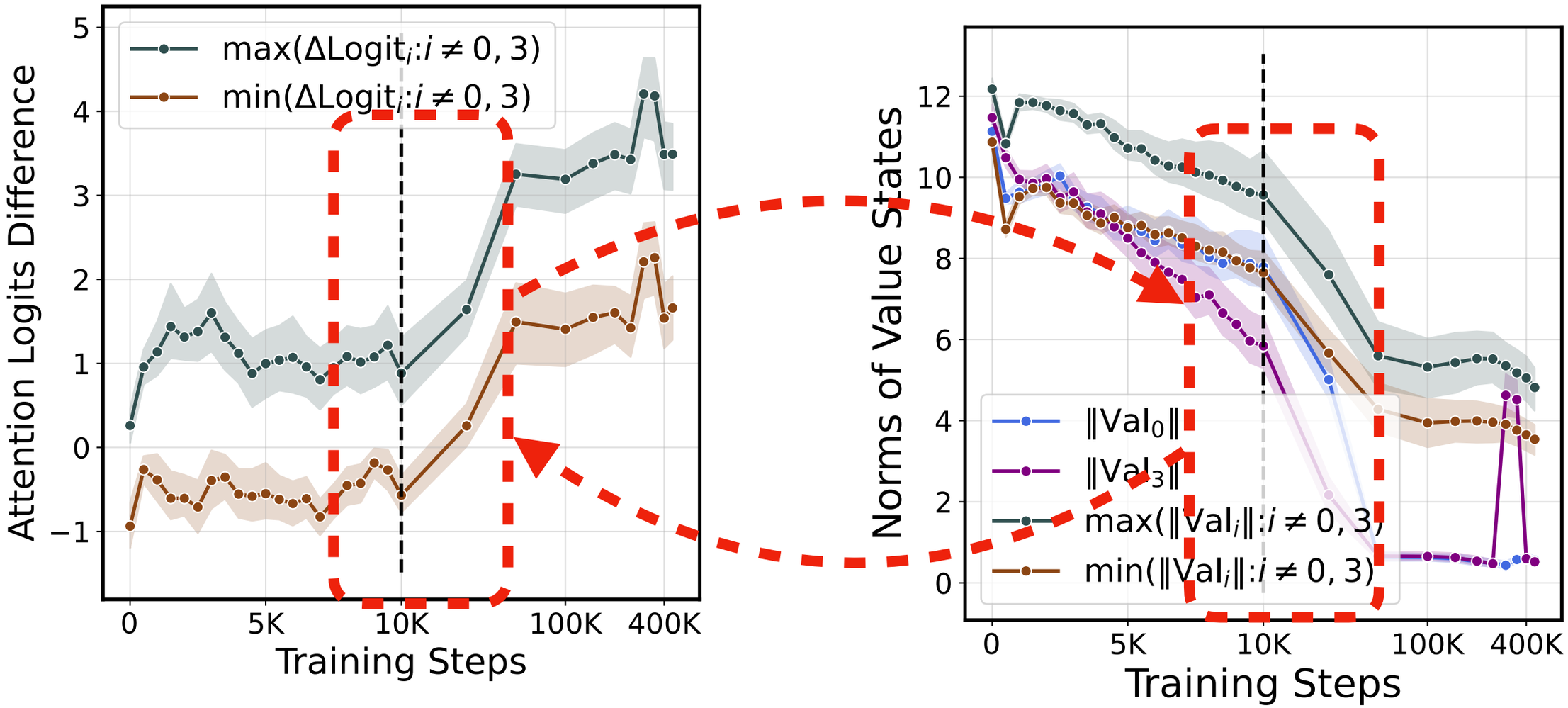
Mutual Reinforcement Mechanism in OLMo





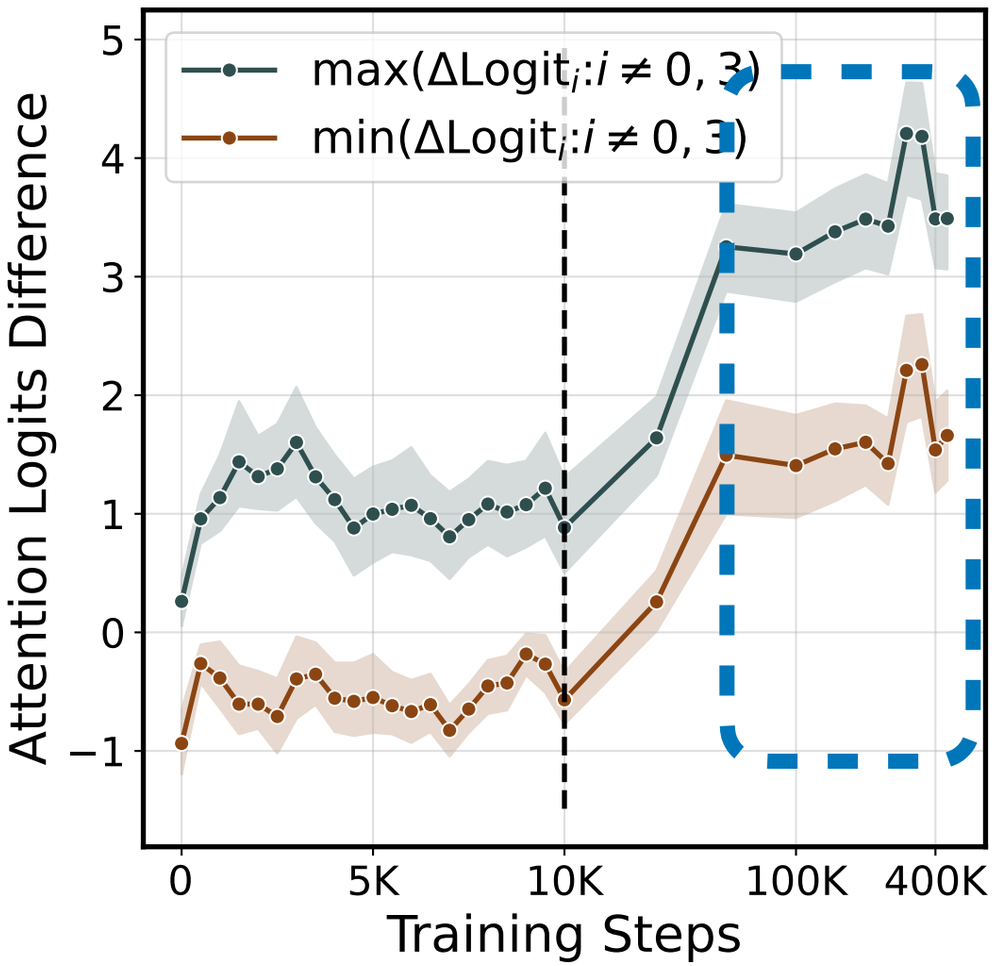
Mutual Reinforcement Mechanism in OLMo

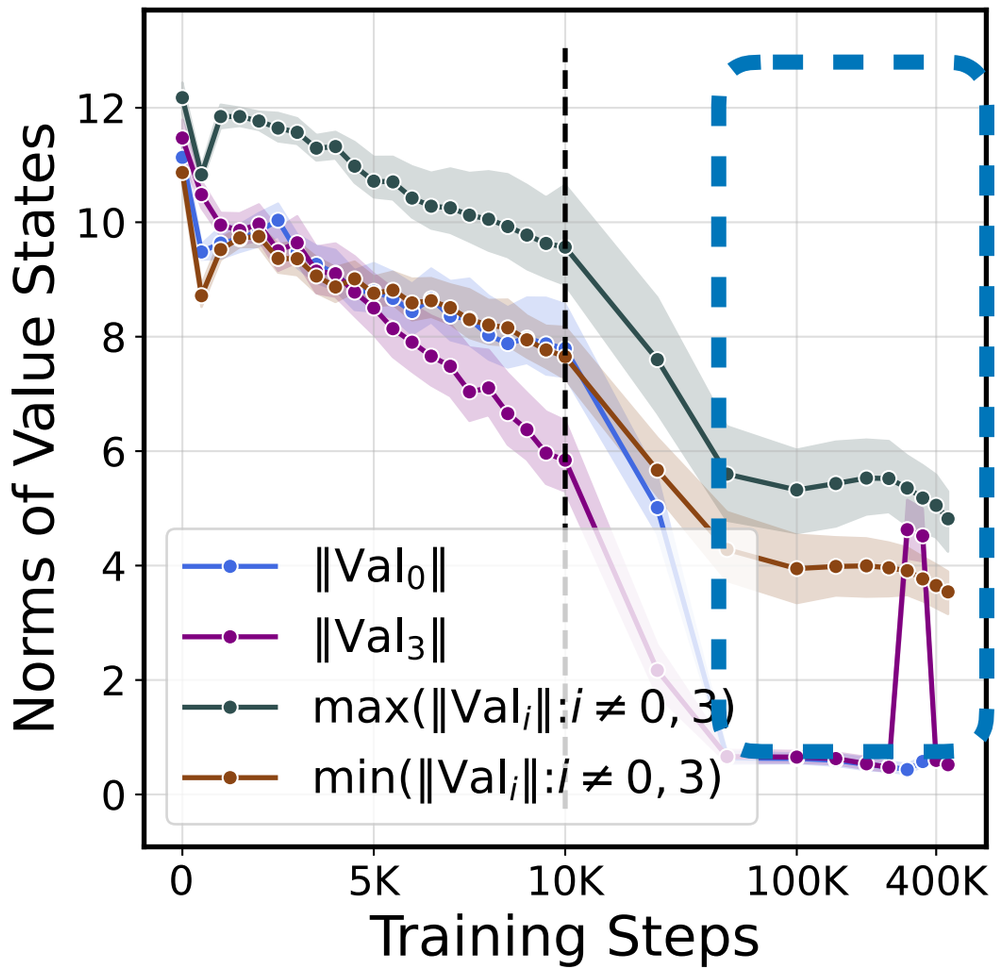
The attention sinks start increase when value states norms start decreasing



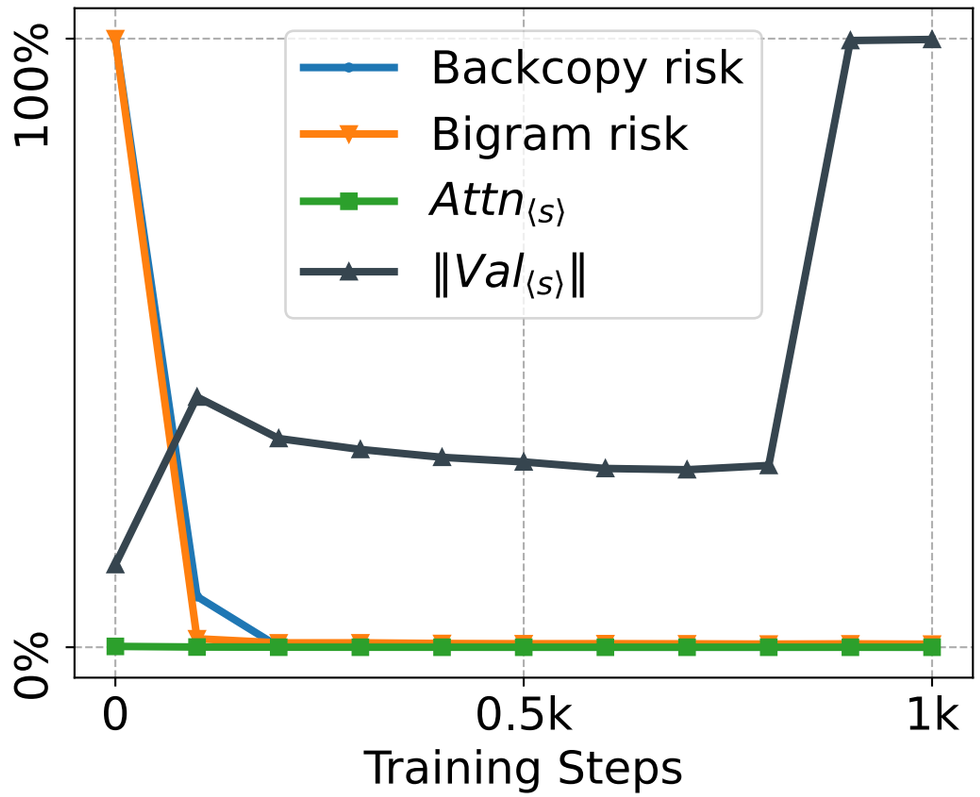
Mutual Reinforcement Mechanism in OLMo

The attention sinks stop increase when value states norms stop decreasing





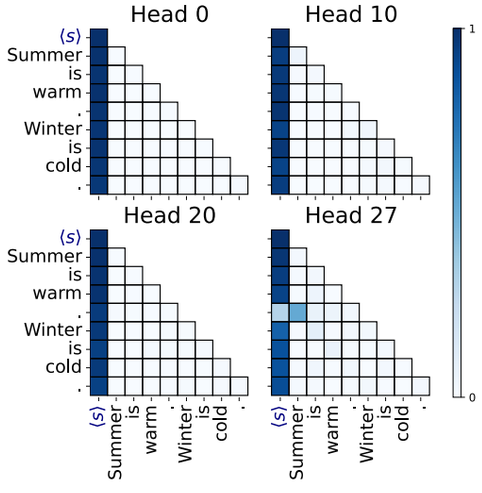
Replace softmax to ReLU elimintates the extreme-token phenomena



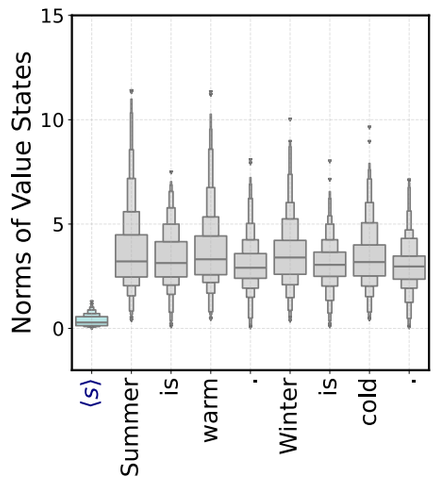
In practice, people have trained small scale ReLU transformer (up to 1 Billion parameters), demonstrating similar performance as Softmax transformer.

Summary

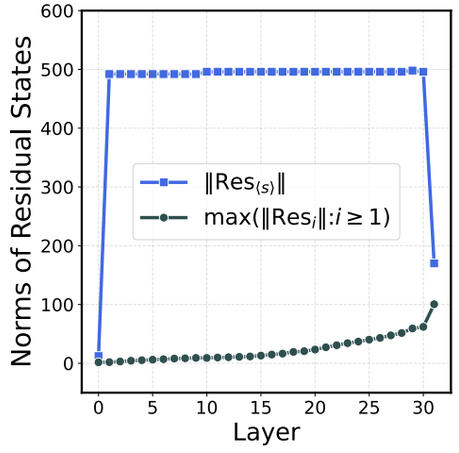
Attention sinks



Value-state drains



Residual-state peaks



- Pretrained transformer-based LLMs exhibit extreme-token phenomena.
- As predicted from the Bigram-Backcopy model, these extreme-token phenomena are governed by the Active-Dormant mechanism and mutual reinforcement mechanism.
- Replacing Softmax to ReLU attention eliminate the extreme-token phenomena.

Tianyu Guo, Druv Pai, Yu Bai, Jiantao Jiao, Michael I. Jordan, and Song Mei. Active-Dormant Attention Heads: Mechanistically Demystifying Extreme-Token Phenomena in LLMs. arXiv preprint, arXiv: 2410.13835.