ReLU Strikes Back: Exploiting Activation Sparsity in Large Language Models

Iman Mirzadeh, Keivan Alizadeh, Sachin Mehta, Carlo C Del Mundo,

Oncel Tuzel, Golnoosh Samei, Mohammad Rastegari, Mehrdad Farajtabar

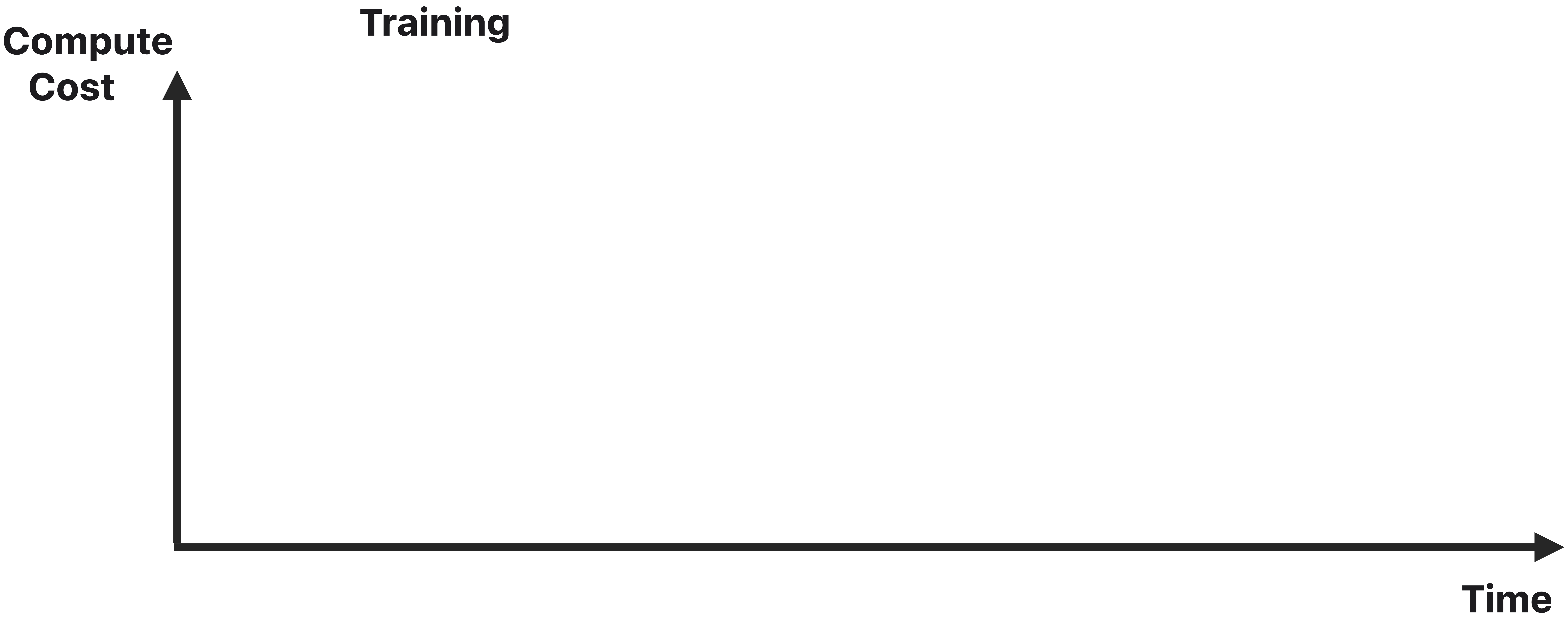
ICLR'24 | Apple

Motivation

Training vs Inference

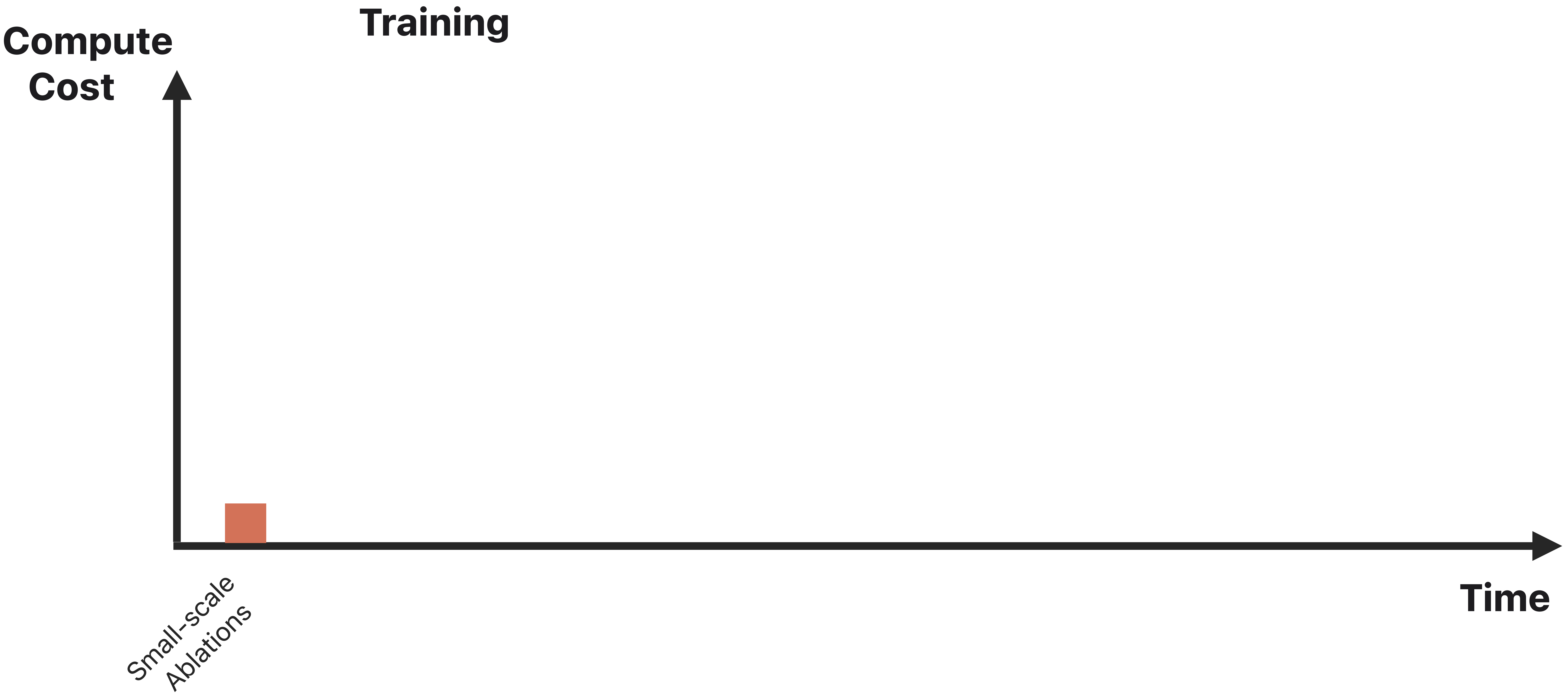
Motivation

Training vs Inference



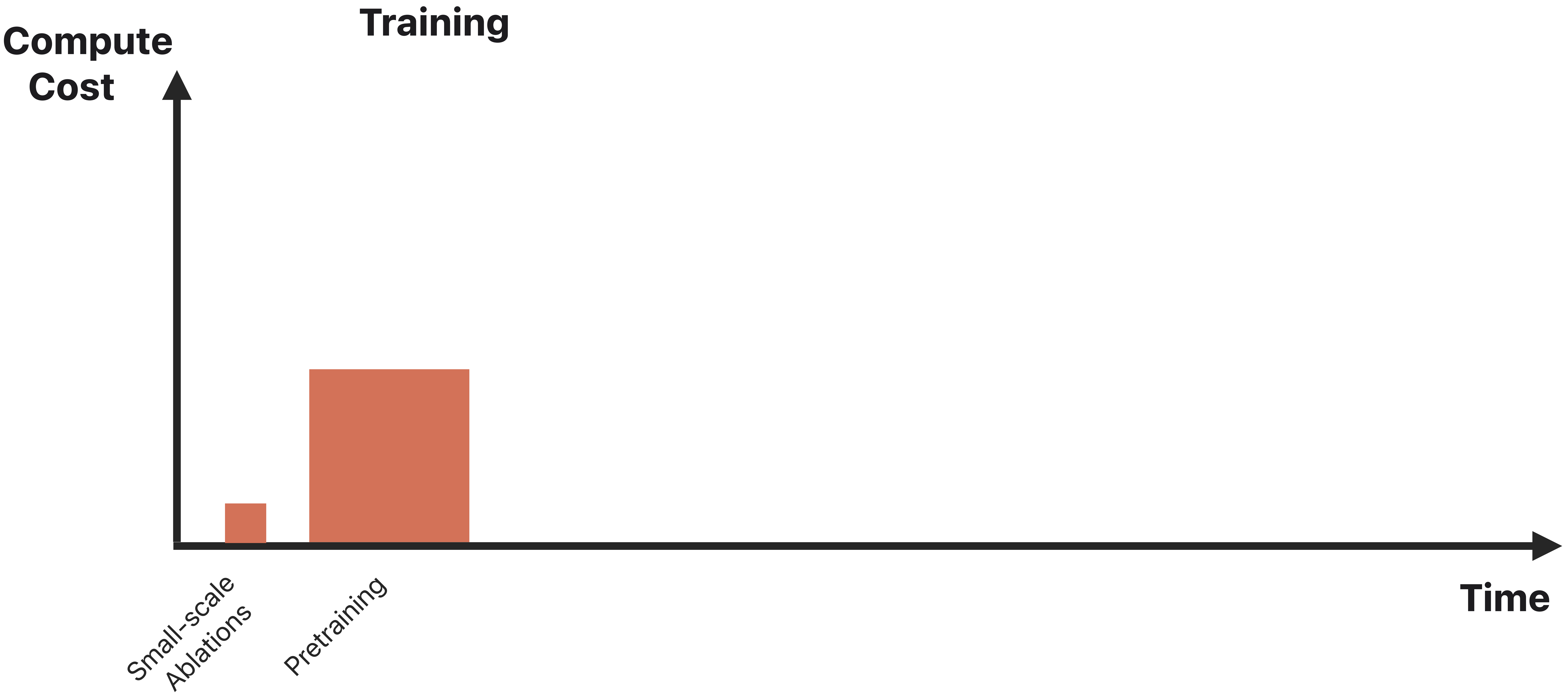
Motivation

Training vs Inference



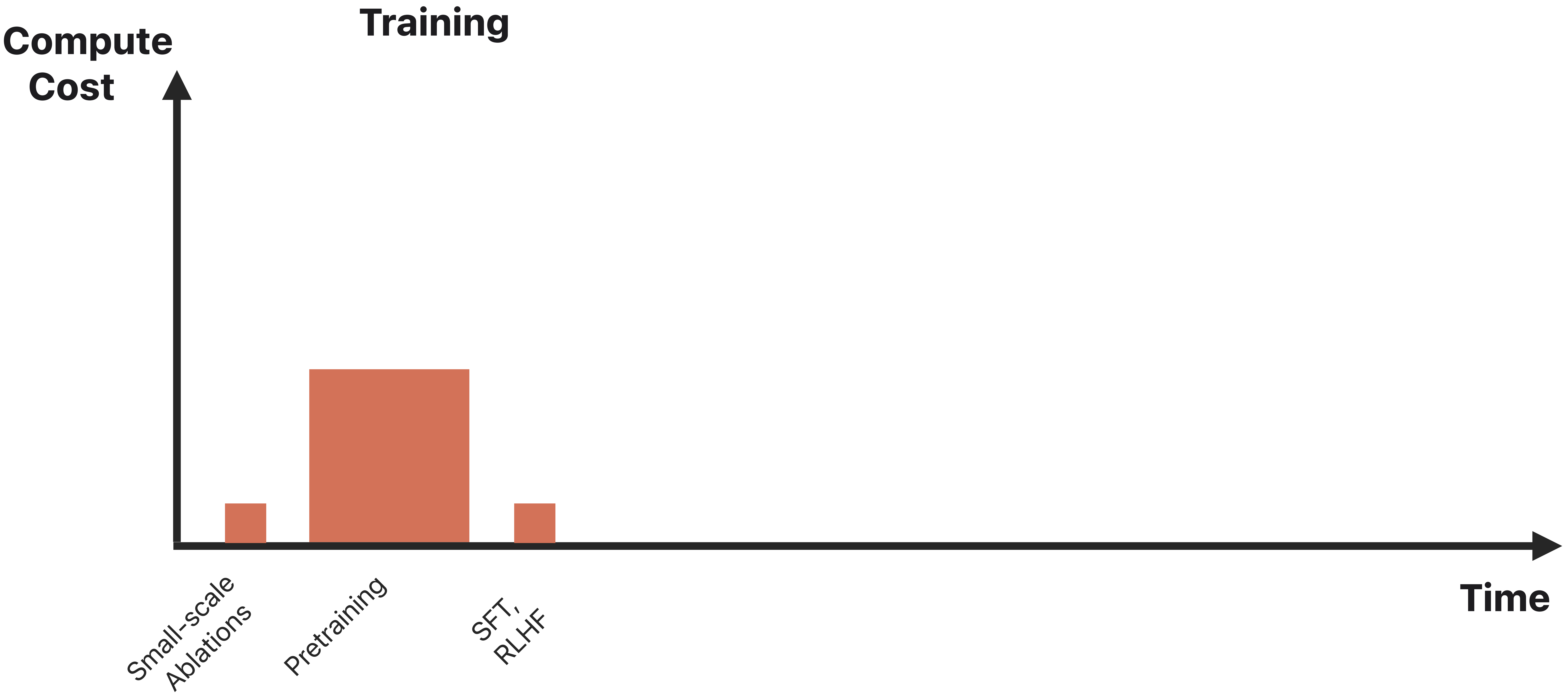
Motivation

Training vs Inference



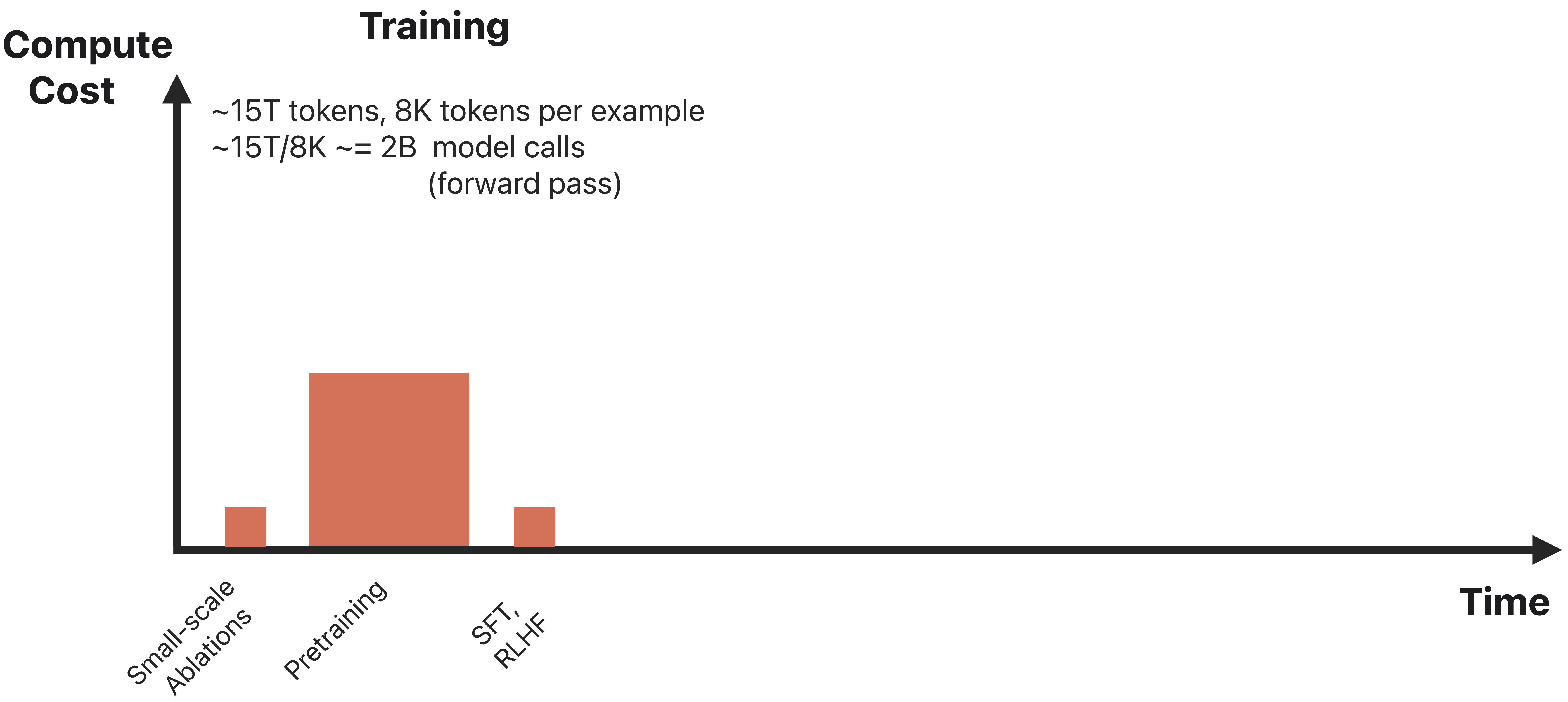
Motivation

Training vs Inference



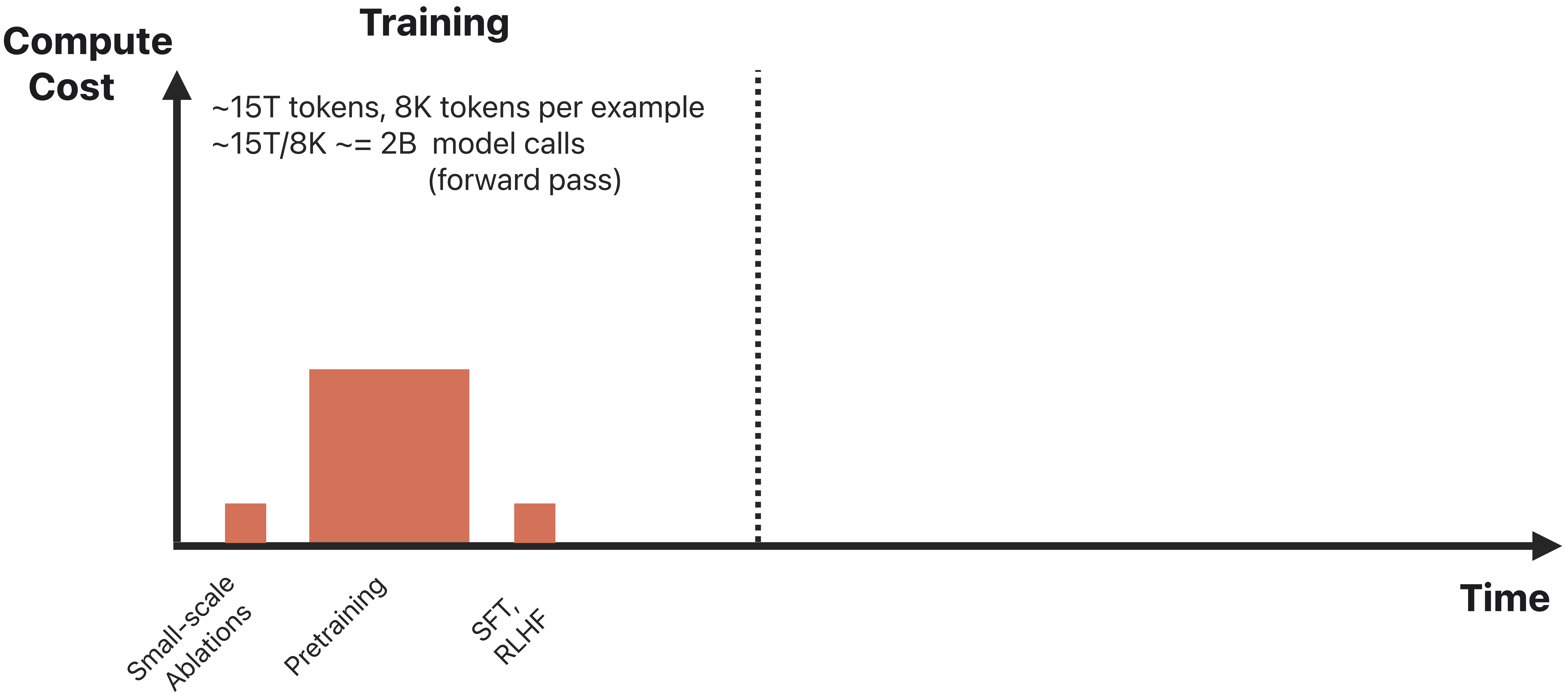
Motivation

Training vs Inference



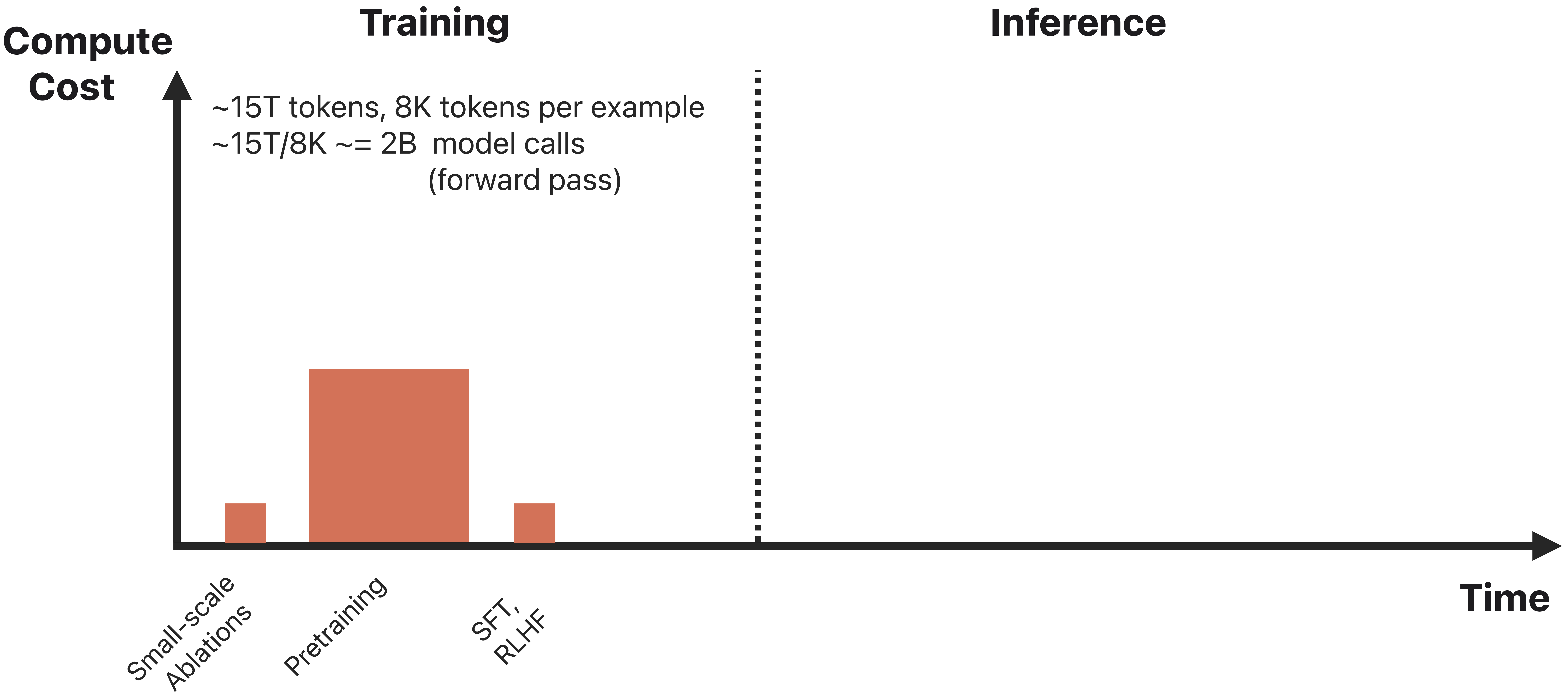
Motivation

Training vs Inference



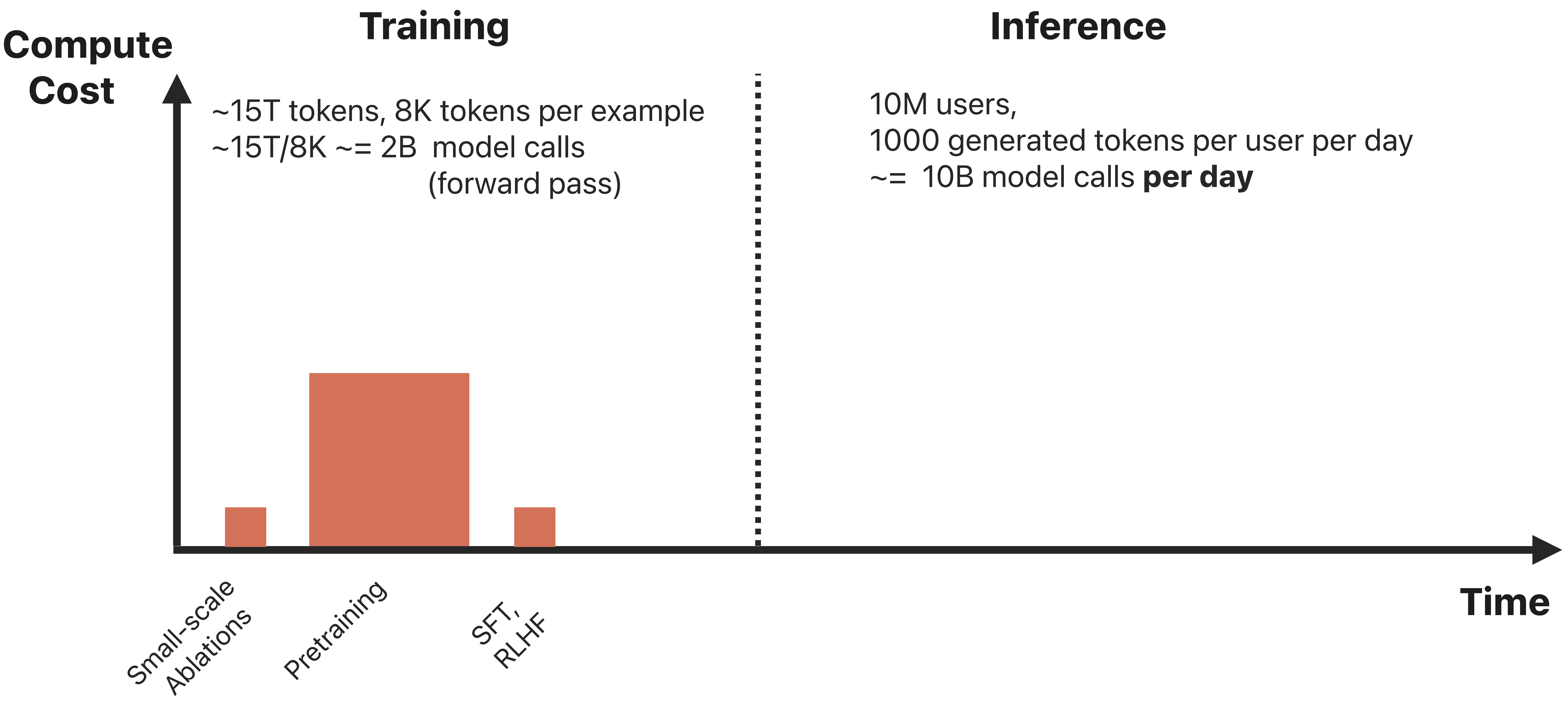
Motivation

Training vs Inference



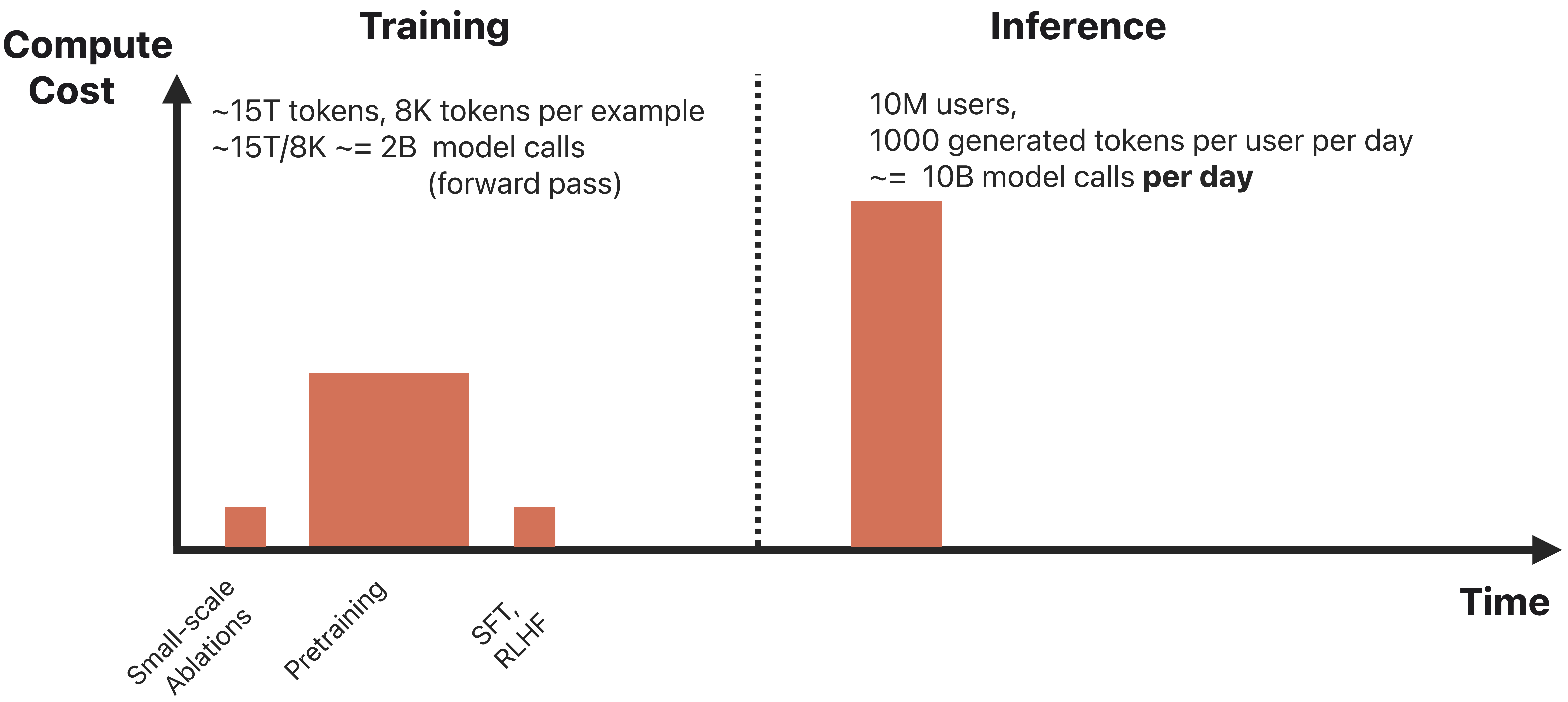
Motivation

Training vs Inference



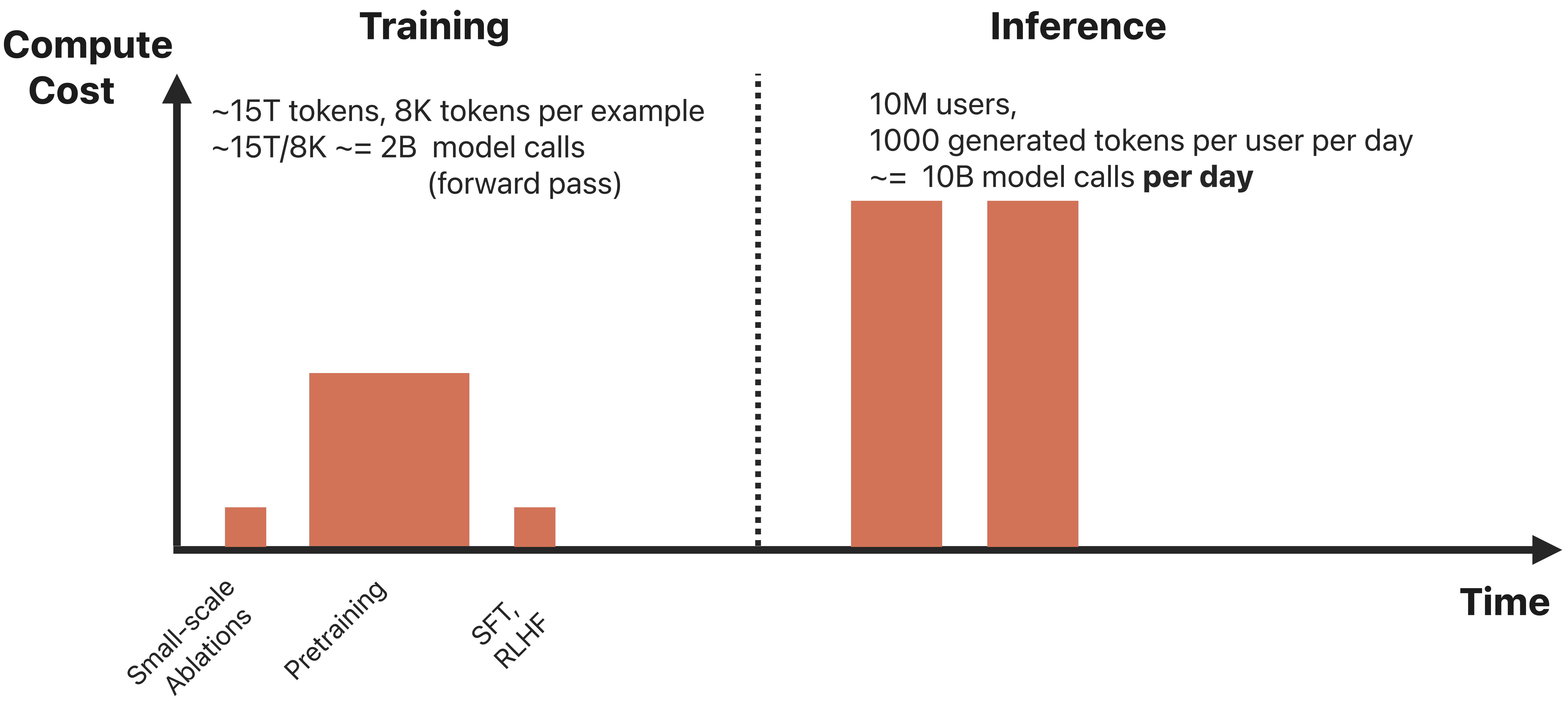
Motivation

Training vs Inference



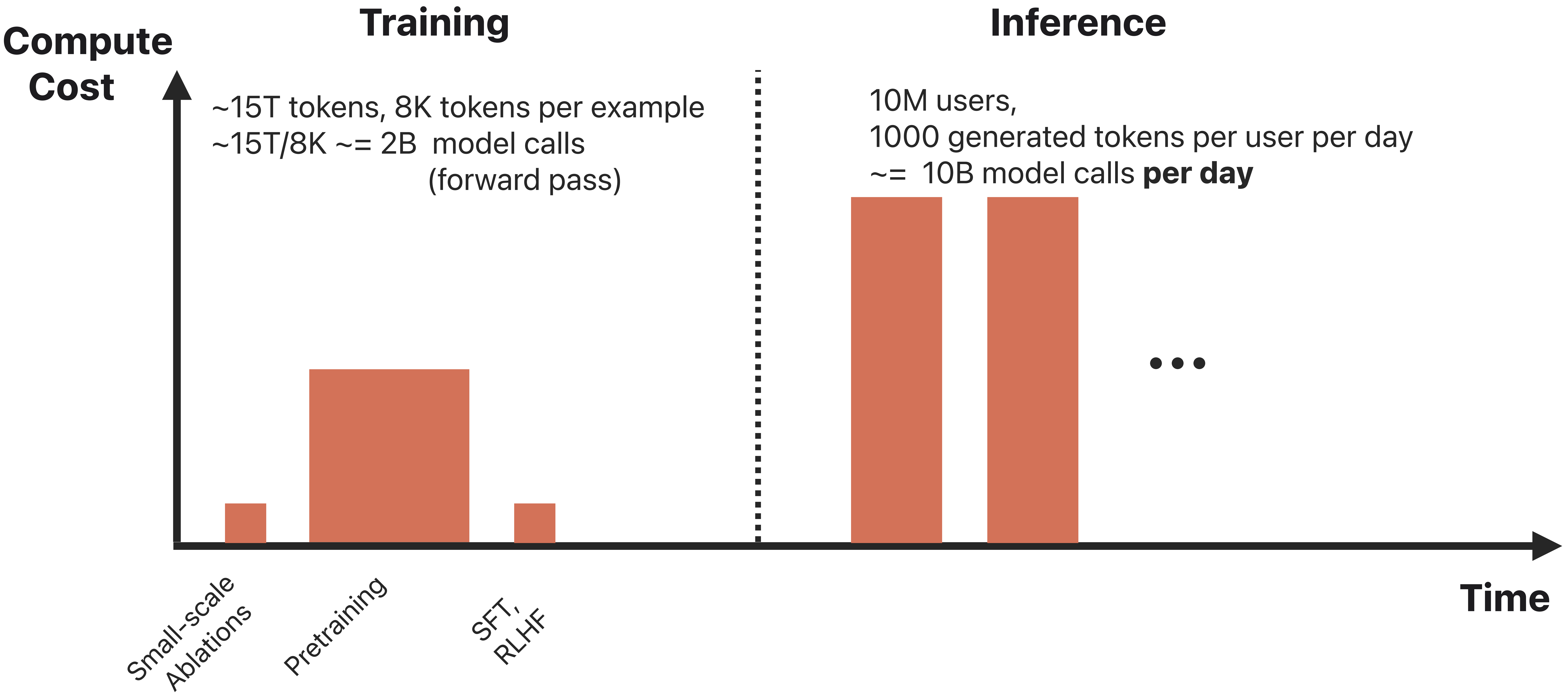
Motivation

Training vs Inference



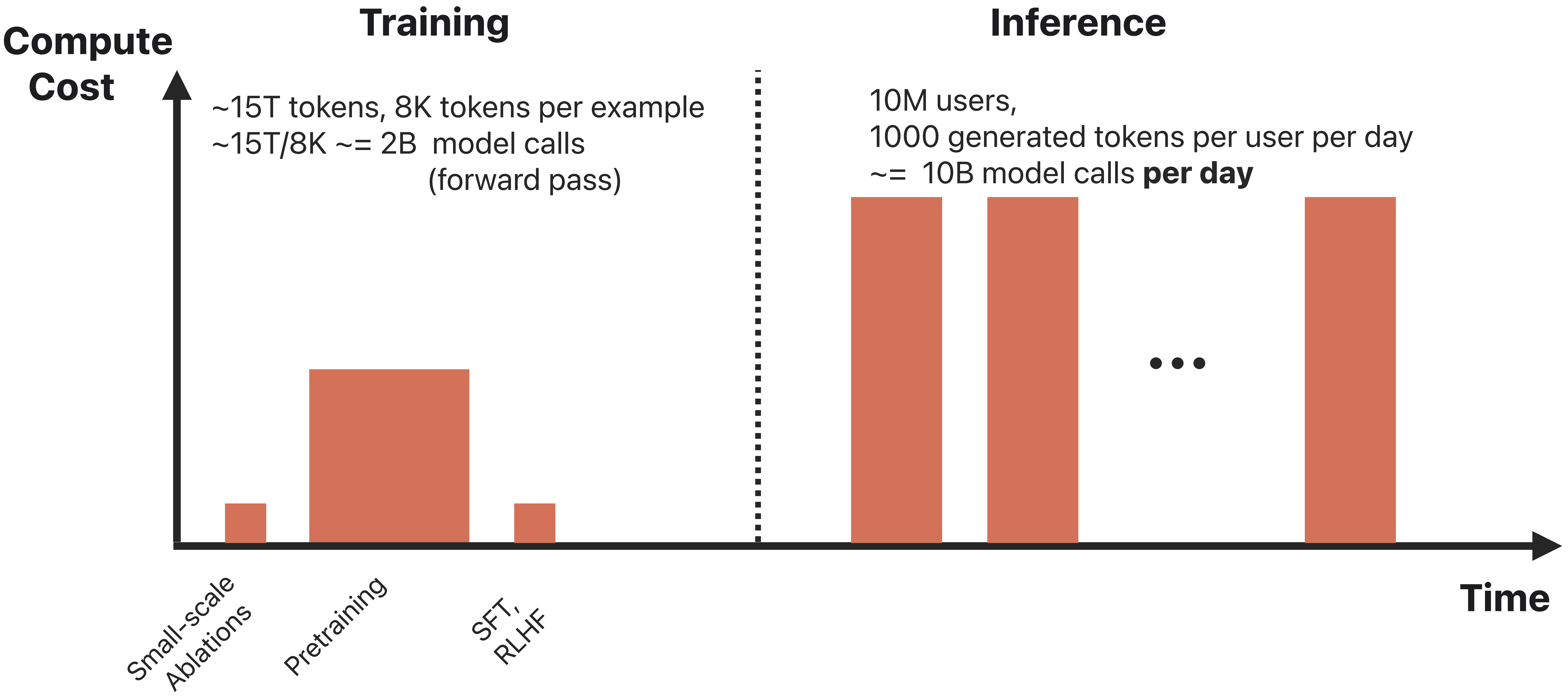
Motivation

Training vs Inference



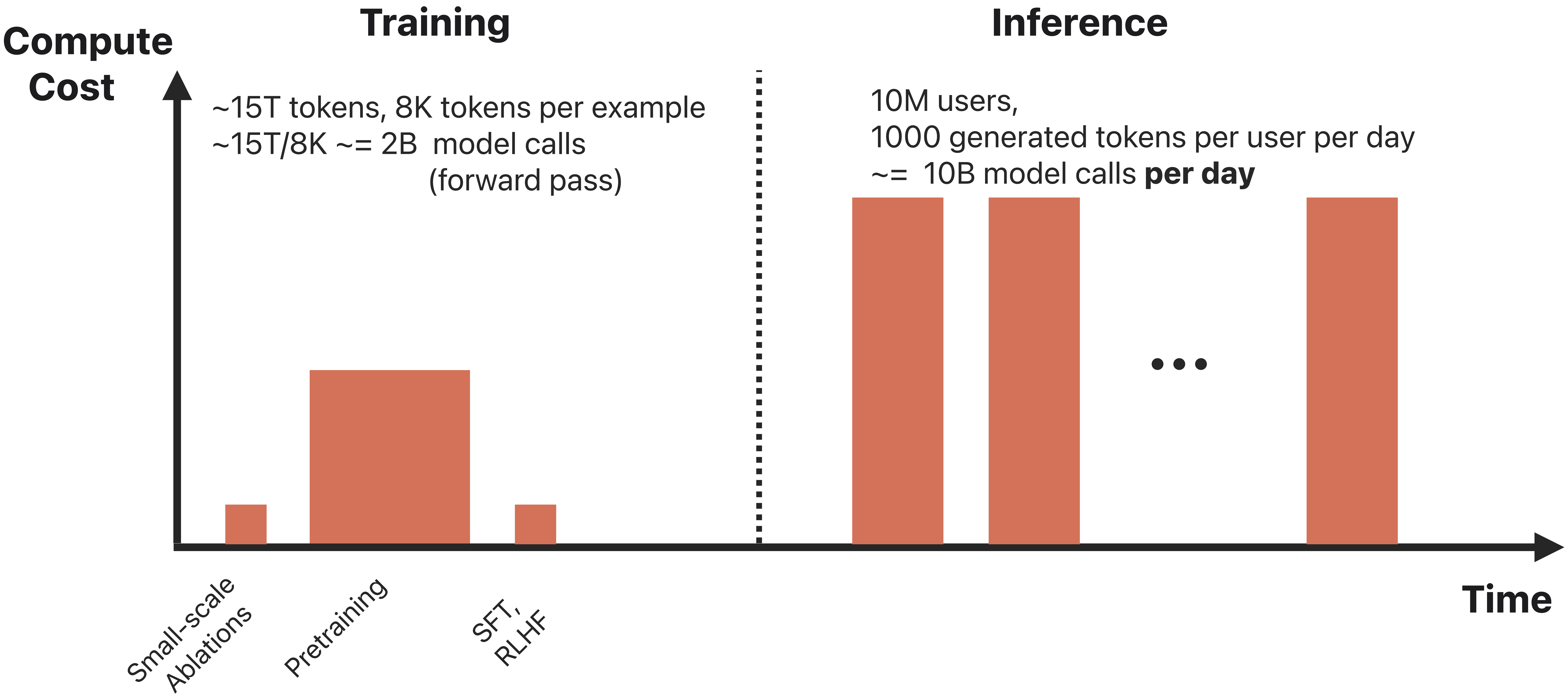
Motivation

Training vs Inference



Motivation

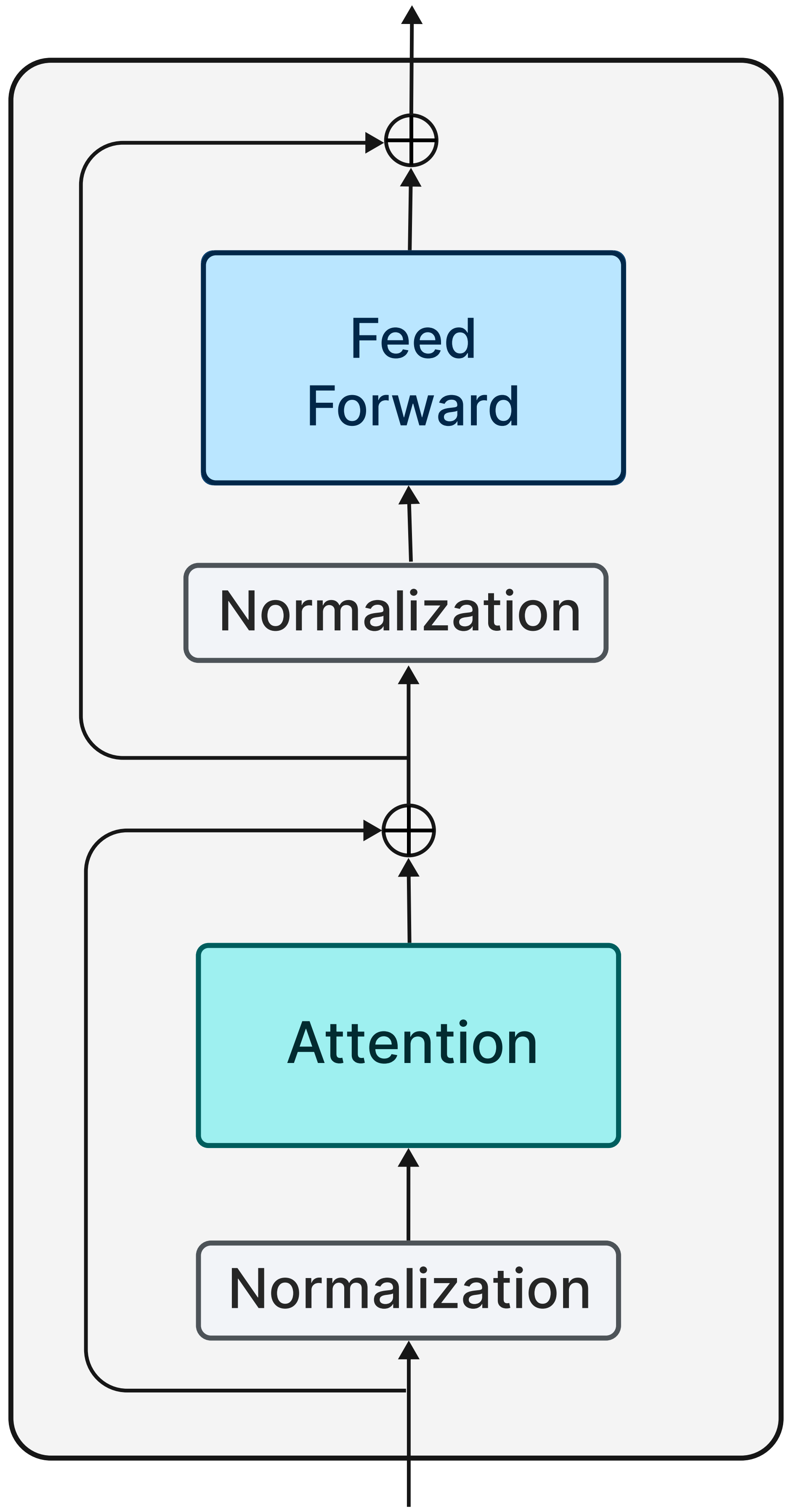
Training vs Inference



[1] OpenAI's ChatGPT now has 100 million weekly active users
https://techcrunch.com/2023/11/06/openais-chatgpt-now-has-100-million-weekly-active-users/

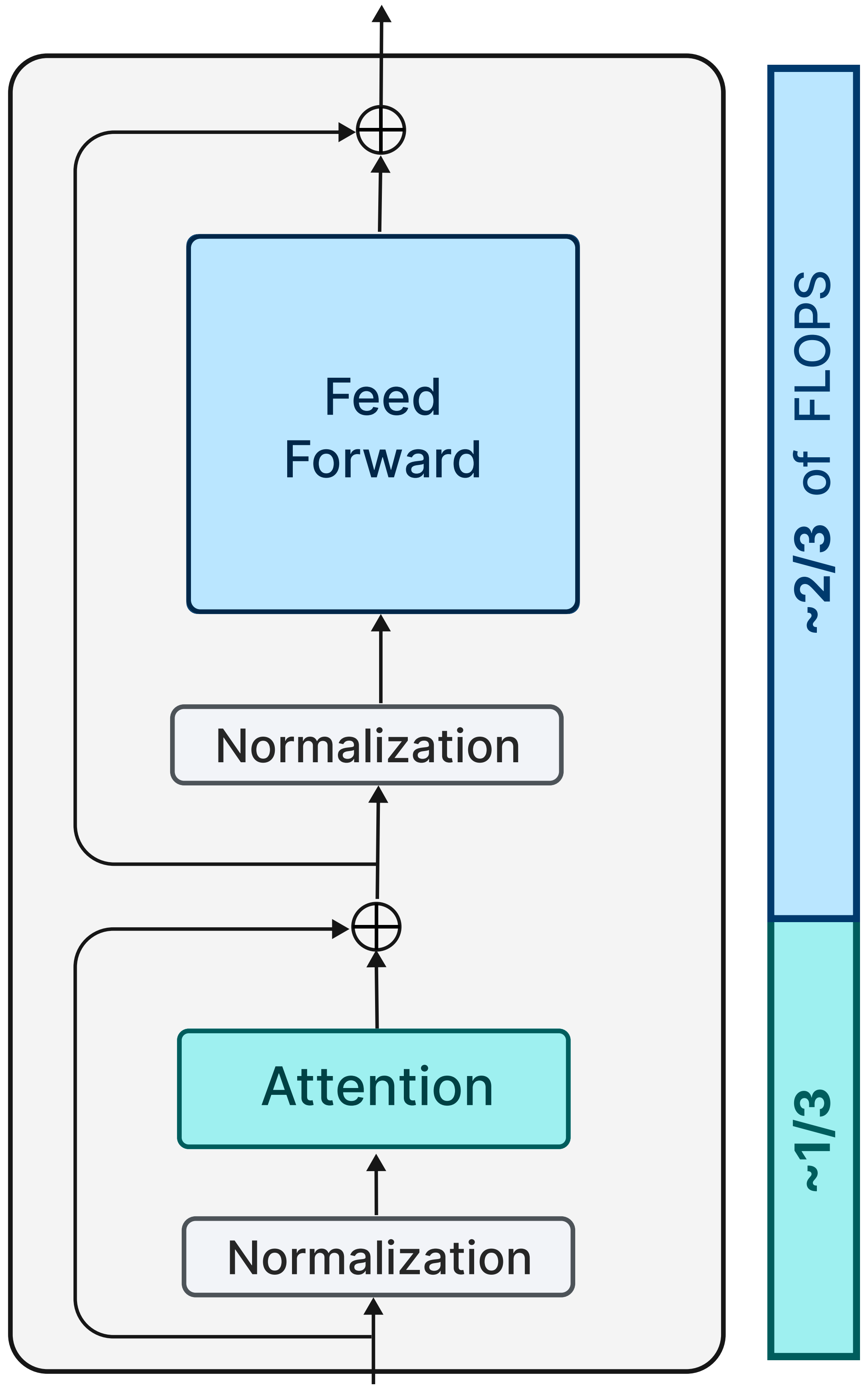
Inference costs almost always outweigh training costs.

Motivation: Where are we spending our compute?



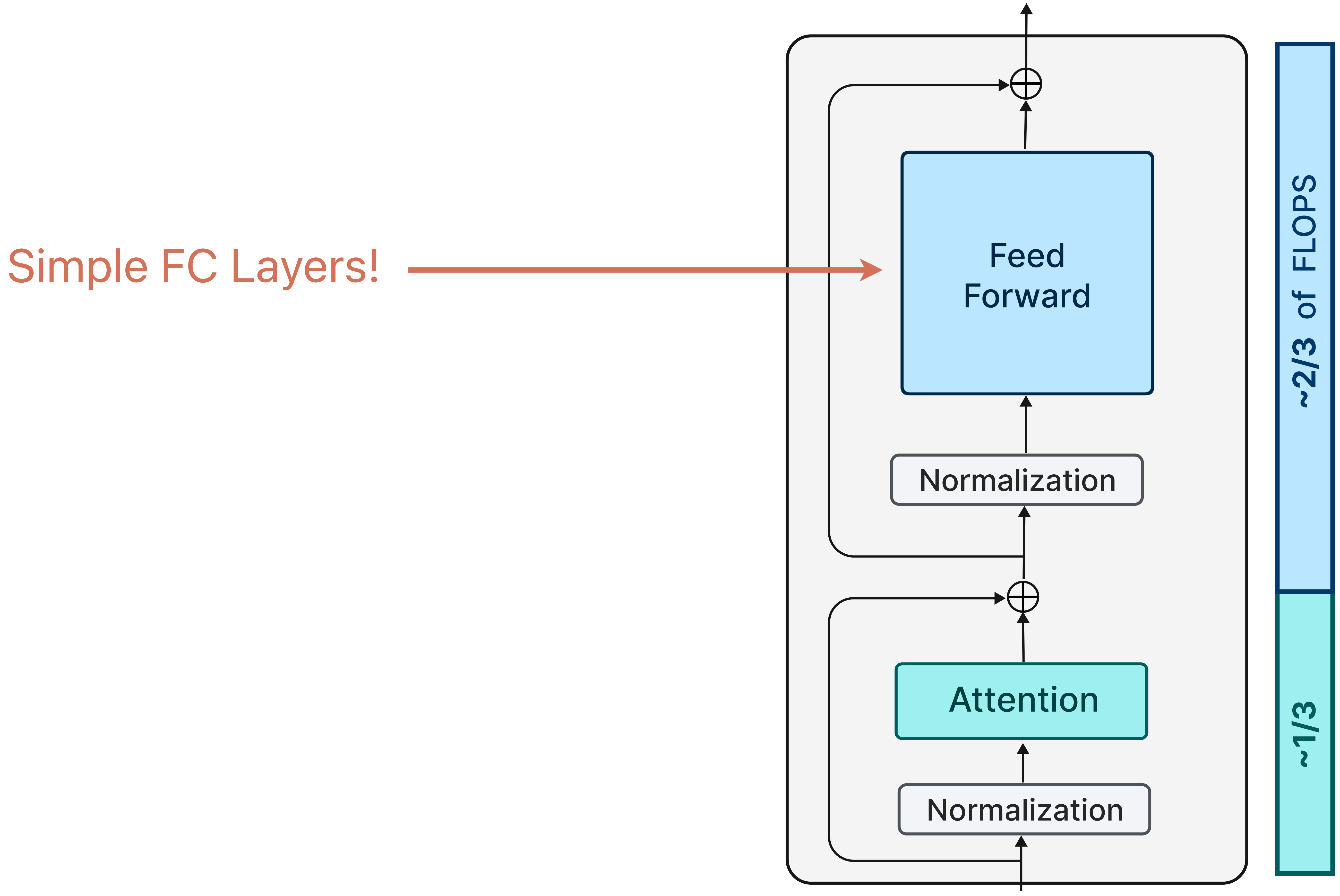
Transformer Block

Motivation: Where are we spending our compute?



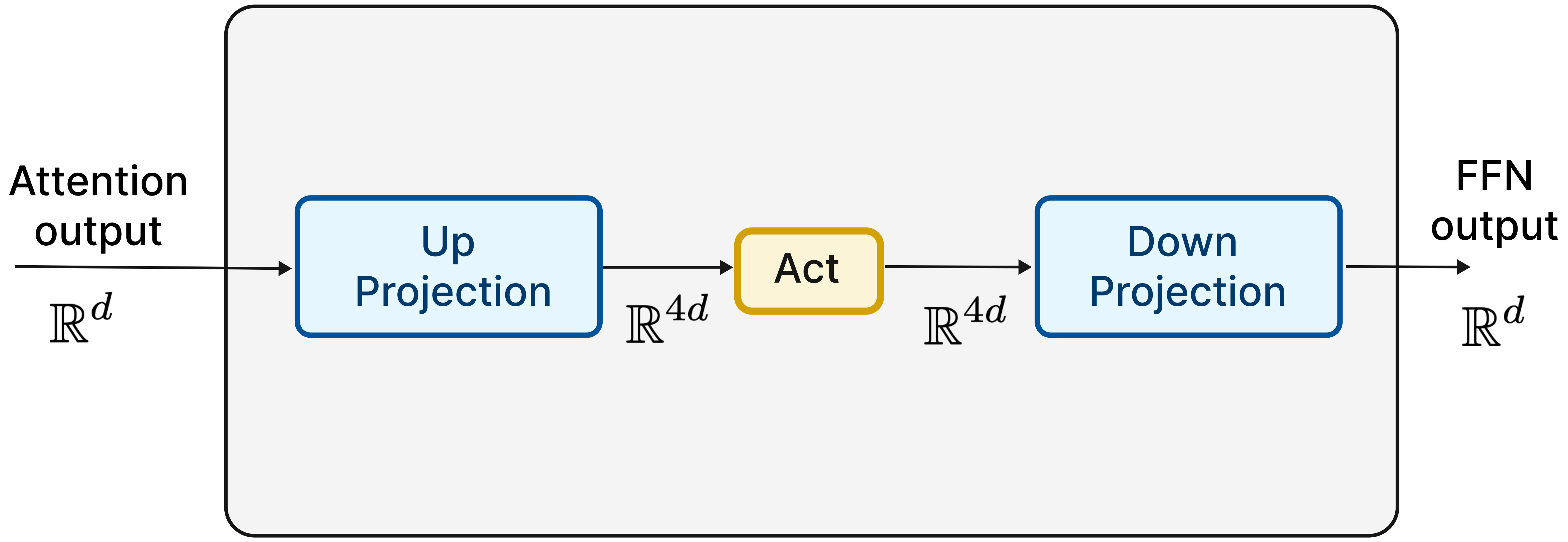
Transformer Block

Motivation: Where are we spending our compute?

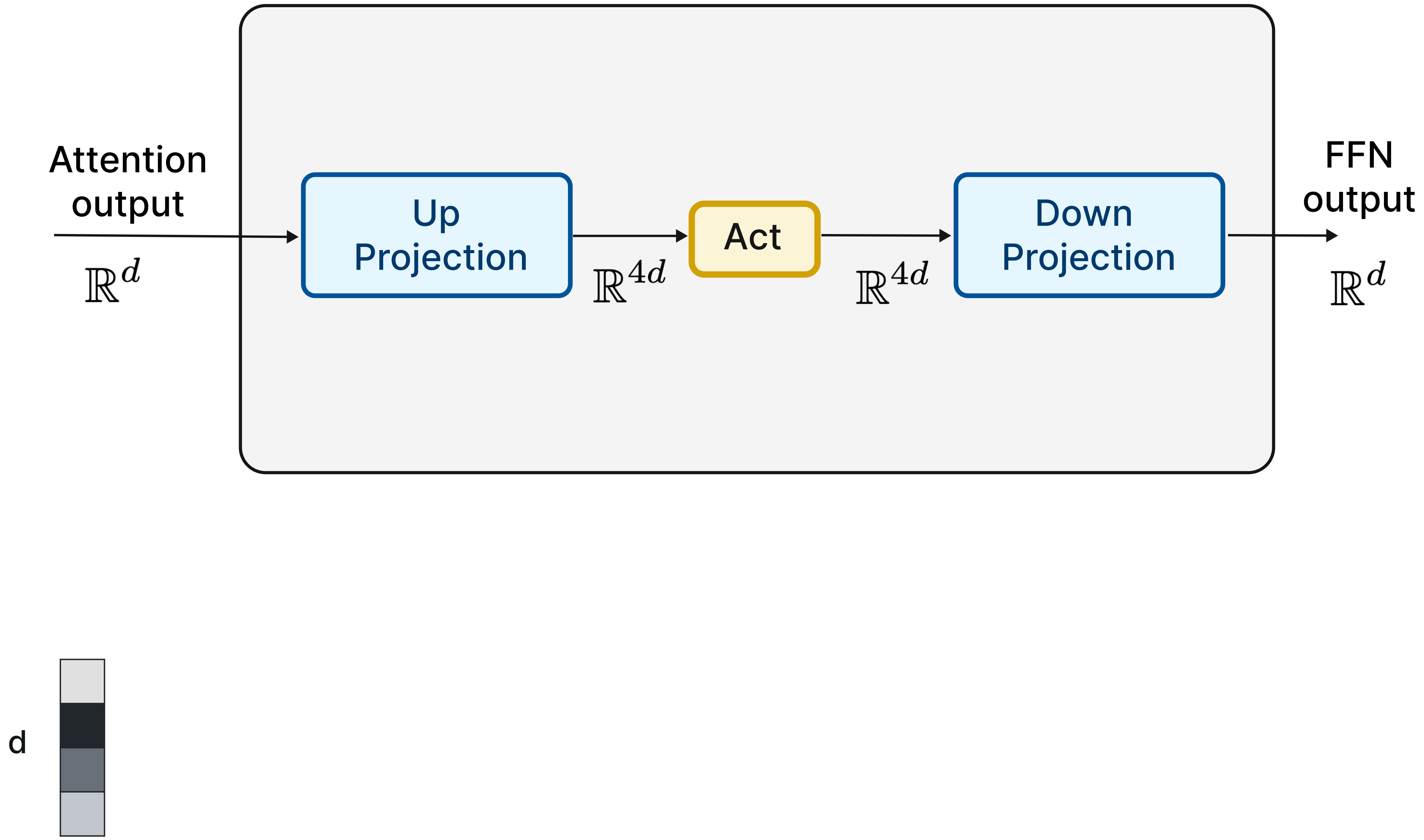


Transformer Block

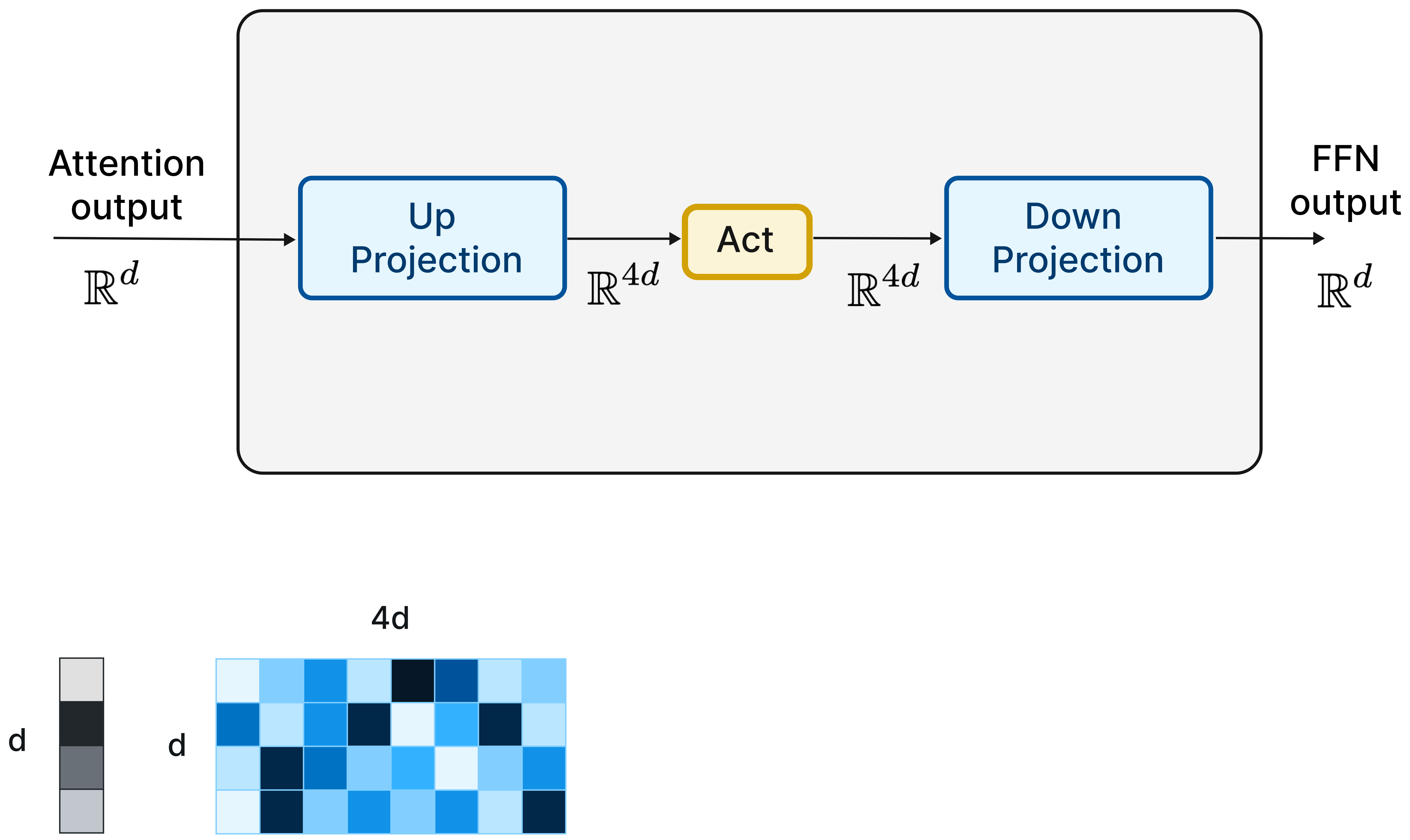
Motivation: Skipping FFN computation



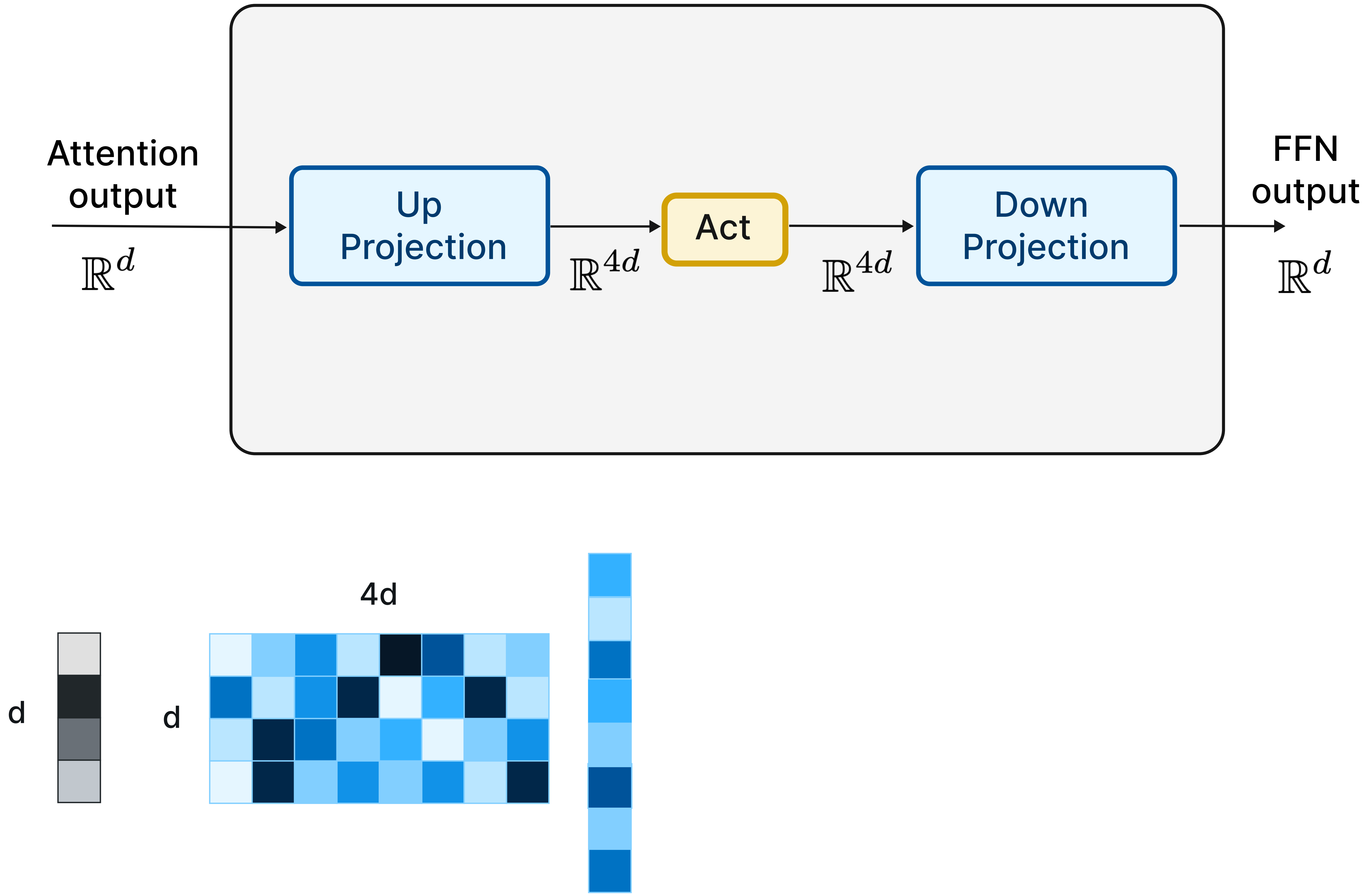
Motivation: Skipping FFN computation



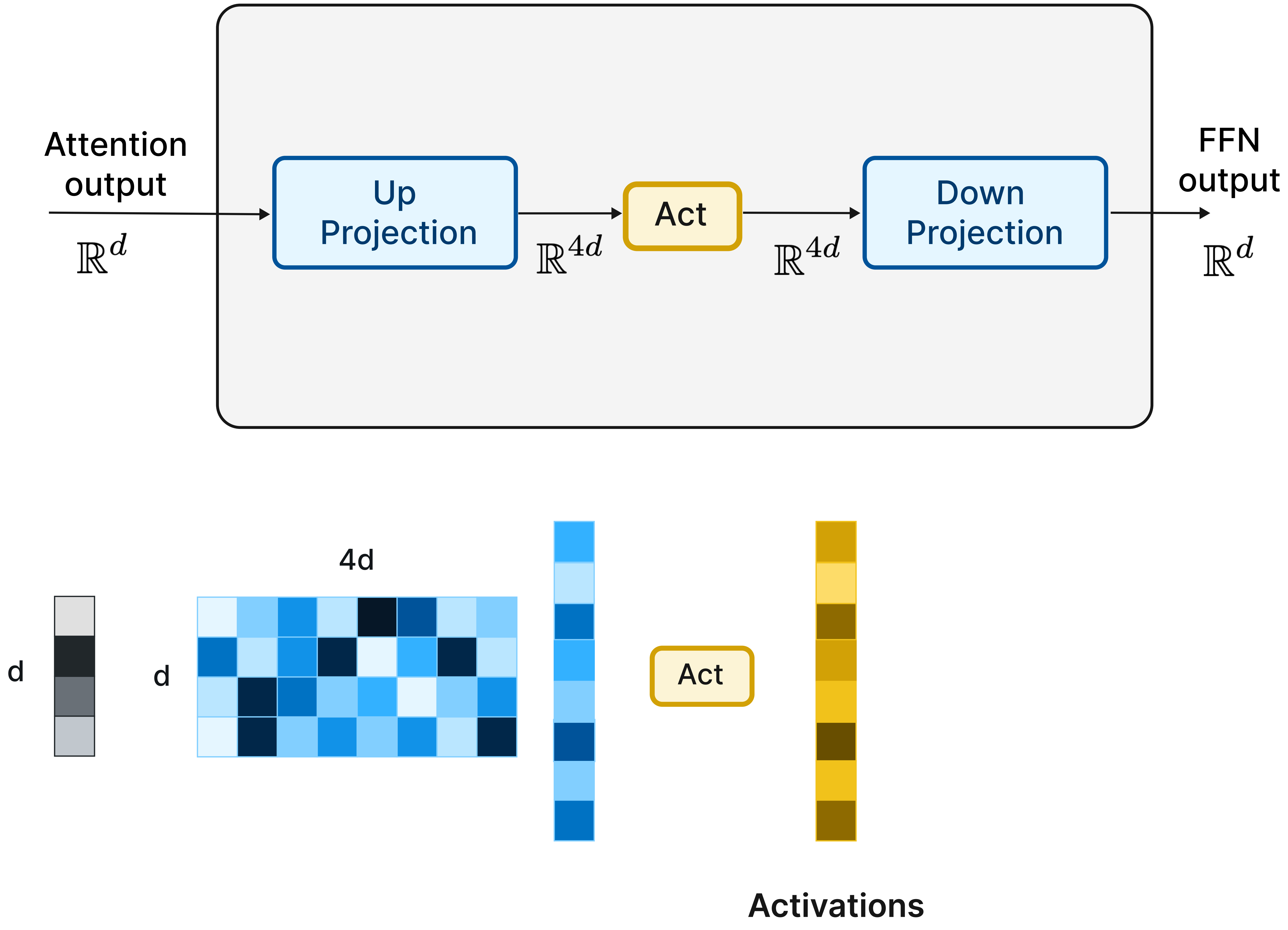
Motivation: Skipping FFN computation



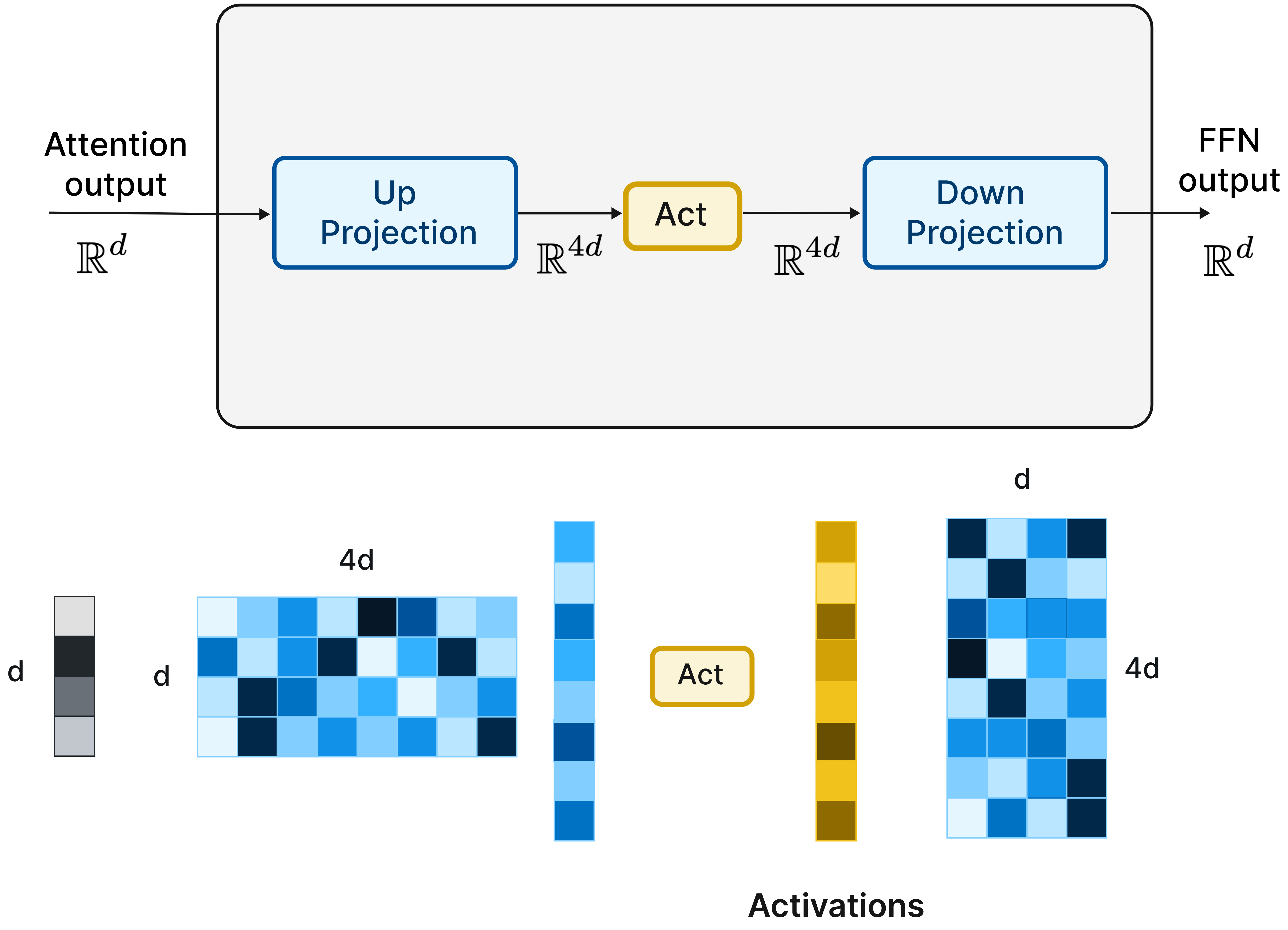
Motivation: Skipping FFN computation



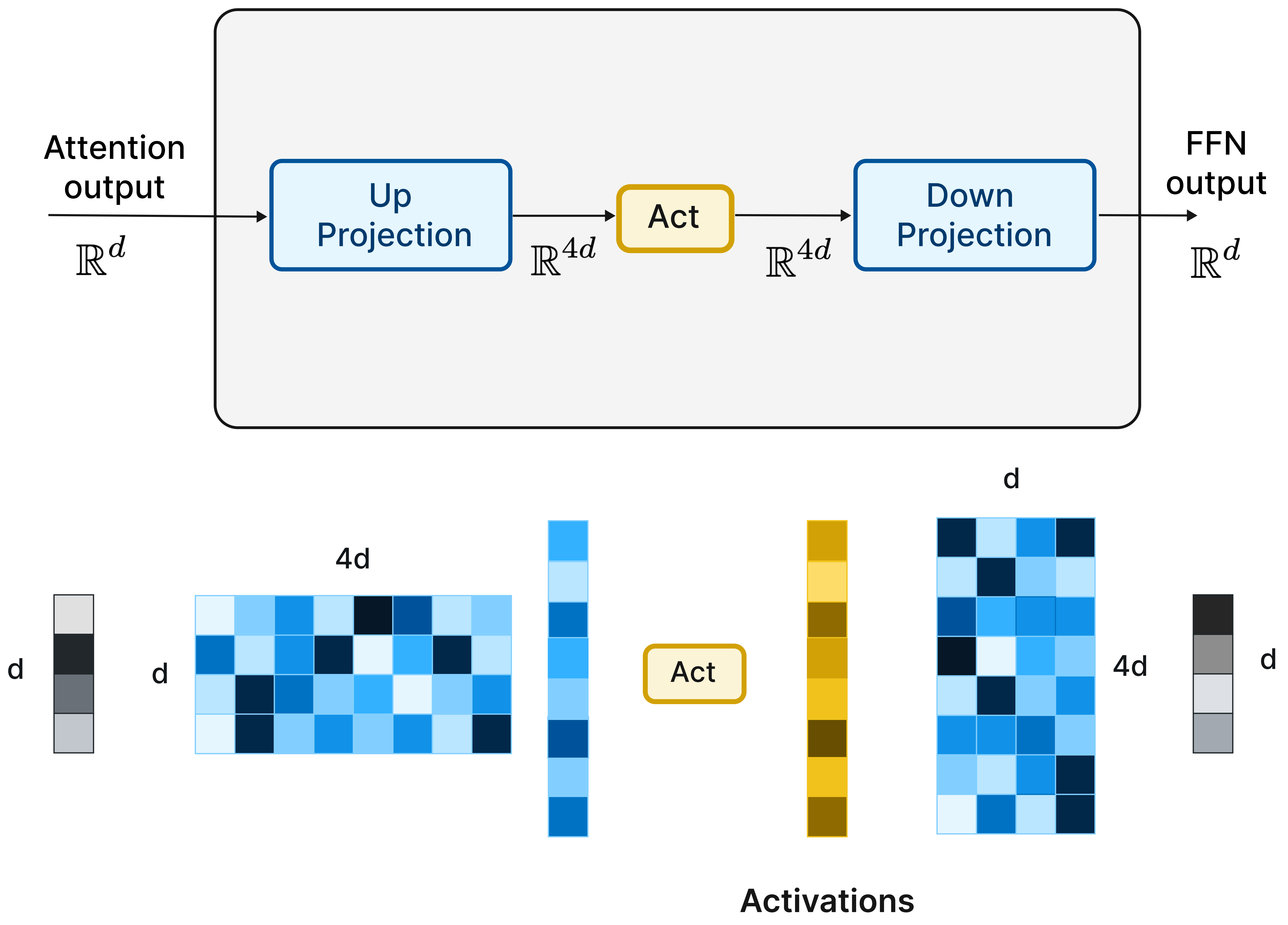
Motivation: Skipping FFN computation



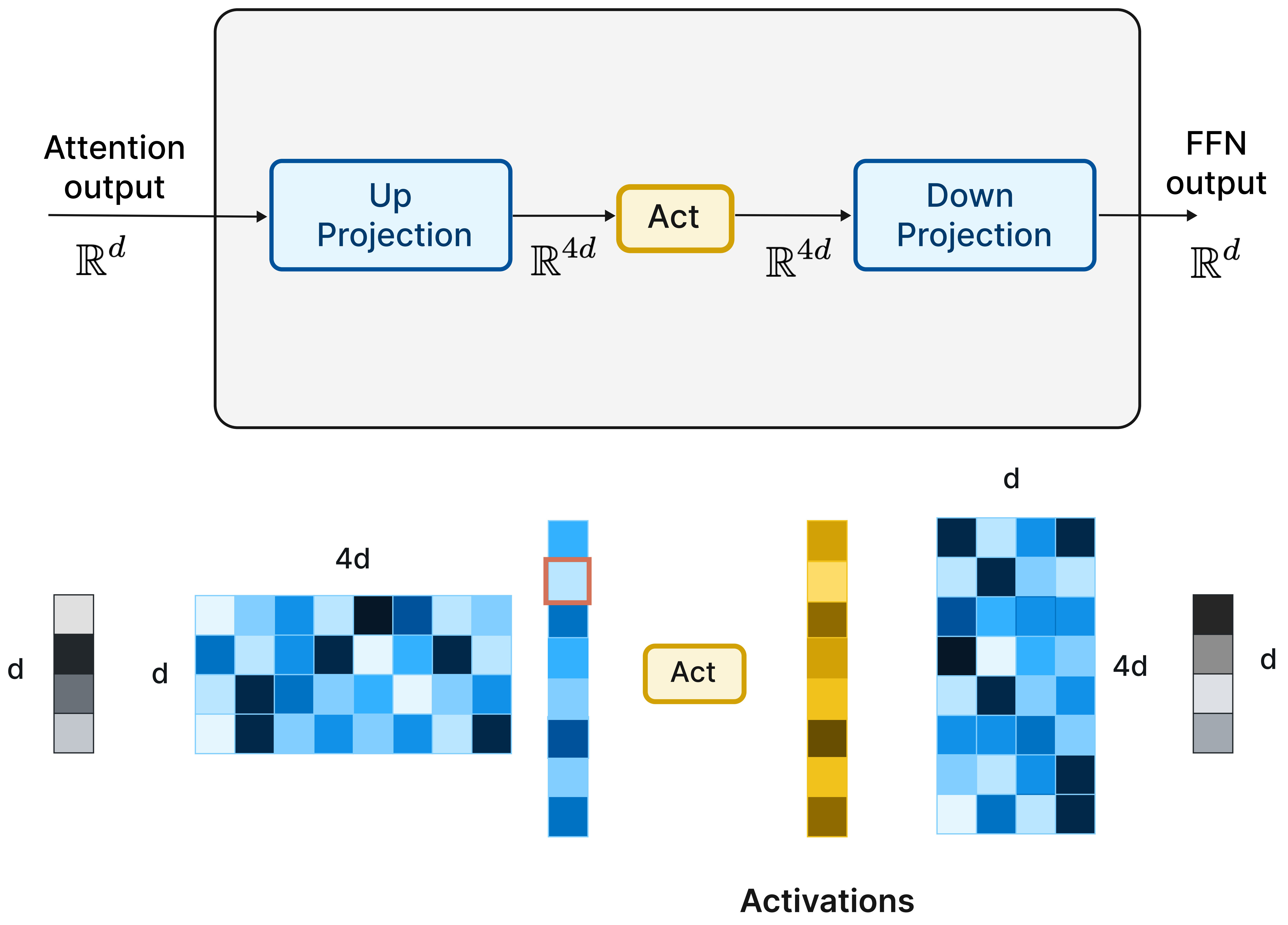
Motivation: Skipping FFN computation



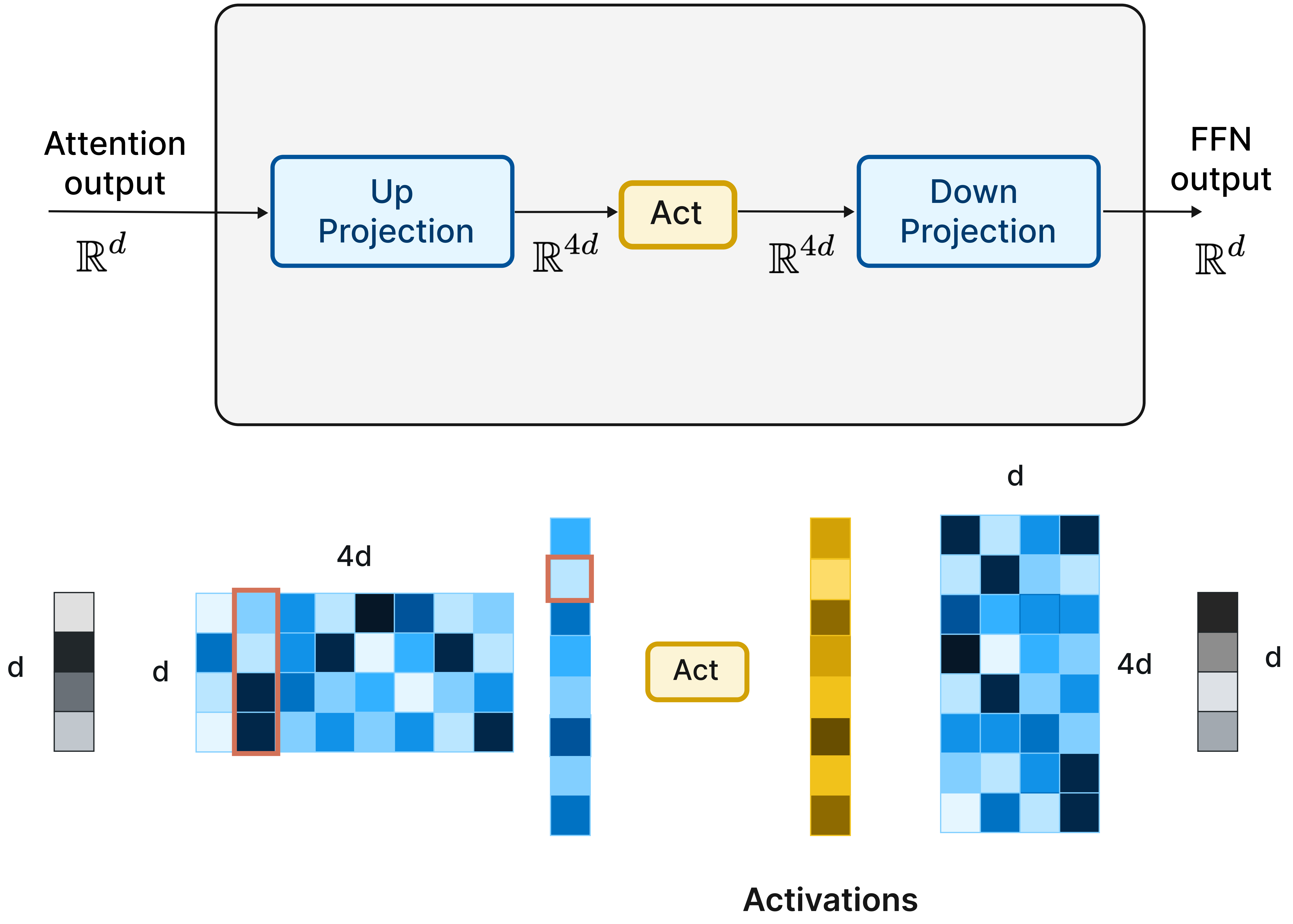
Motivation: Skipping FFN computation



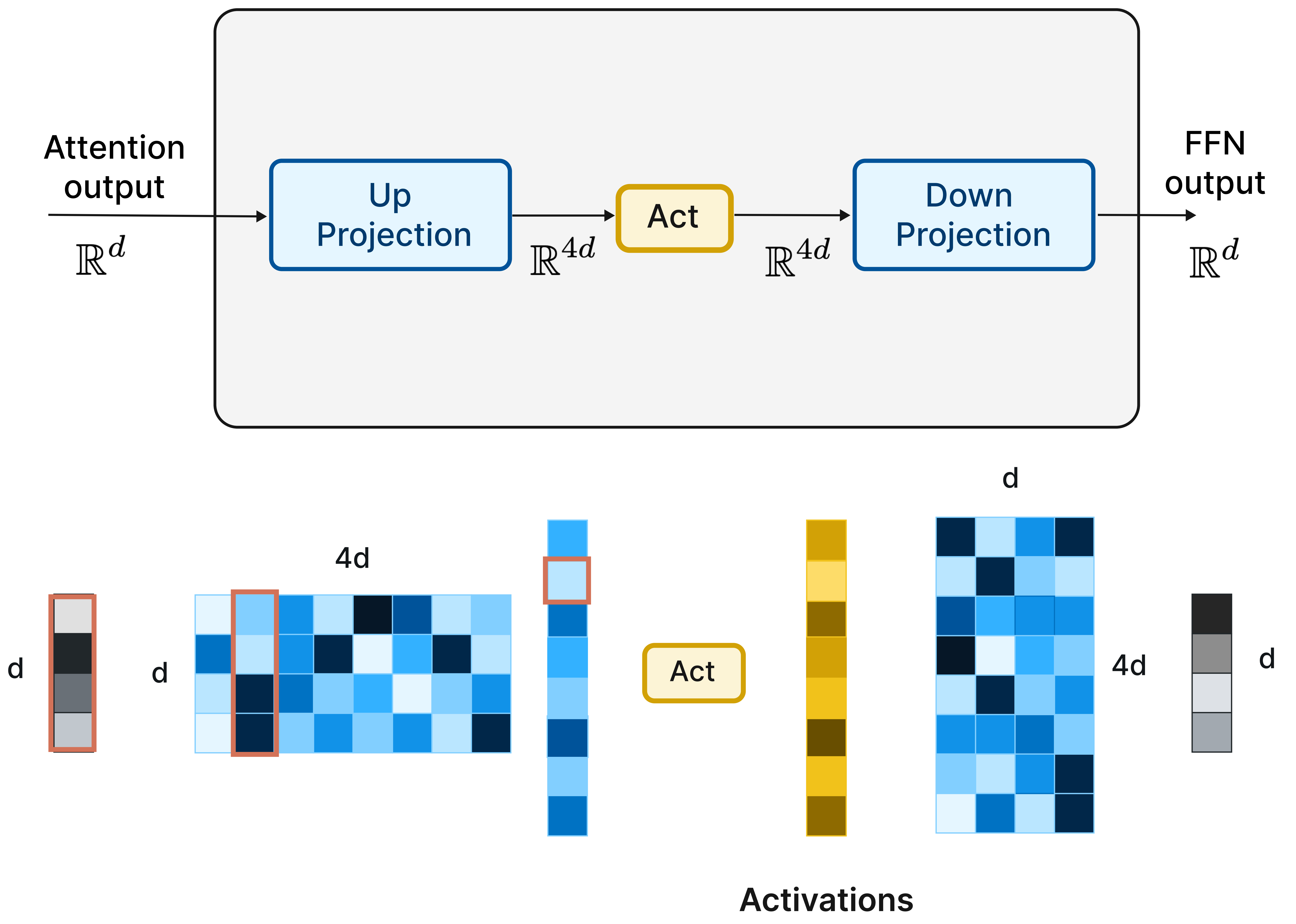
Motivation: Skipping FFN computation



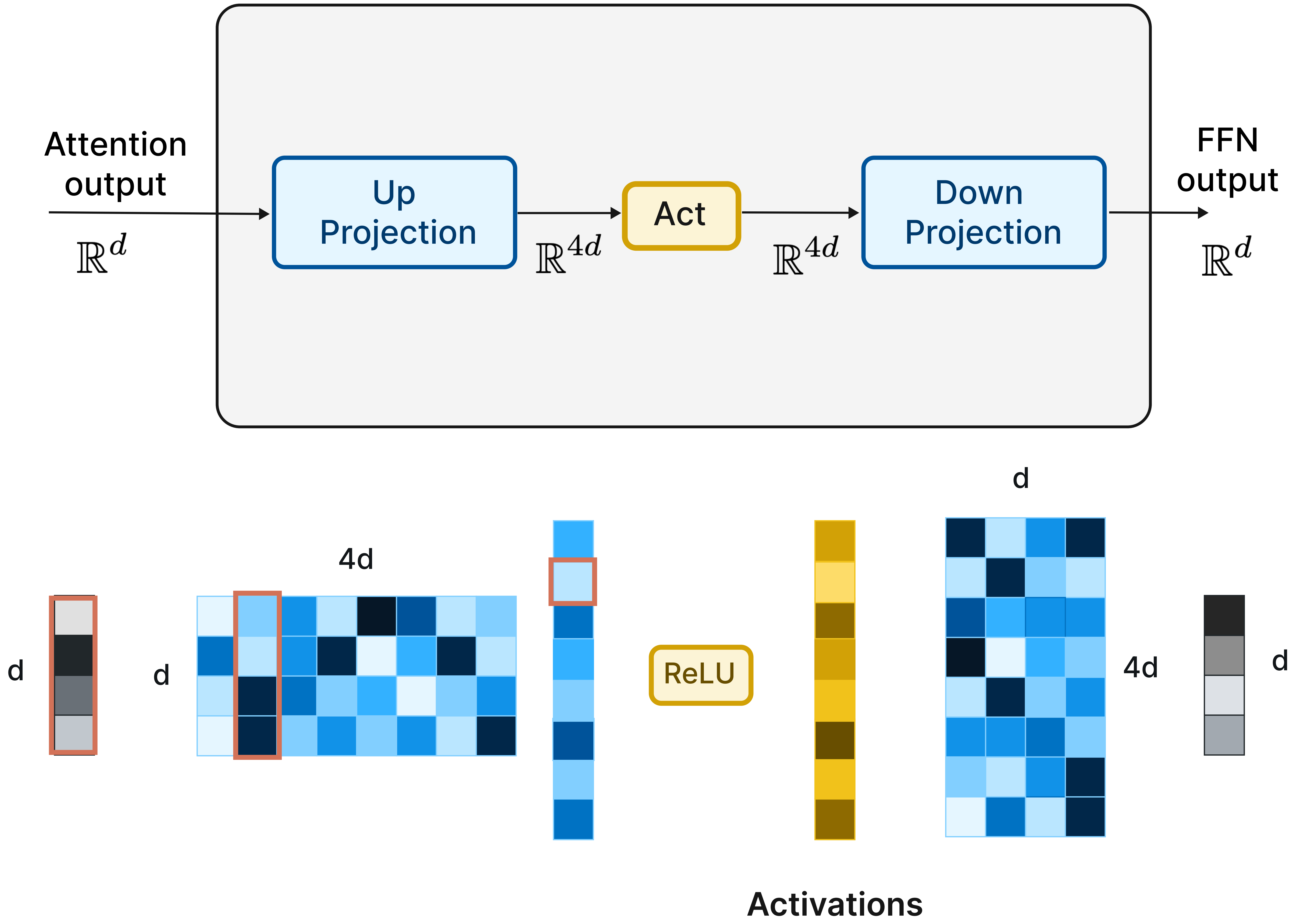
Motivation: Skipping FFN computation



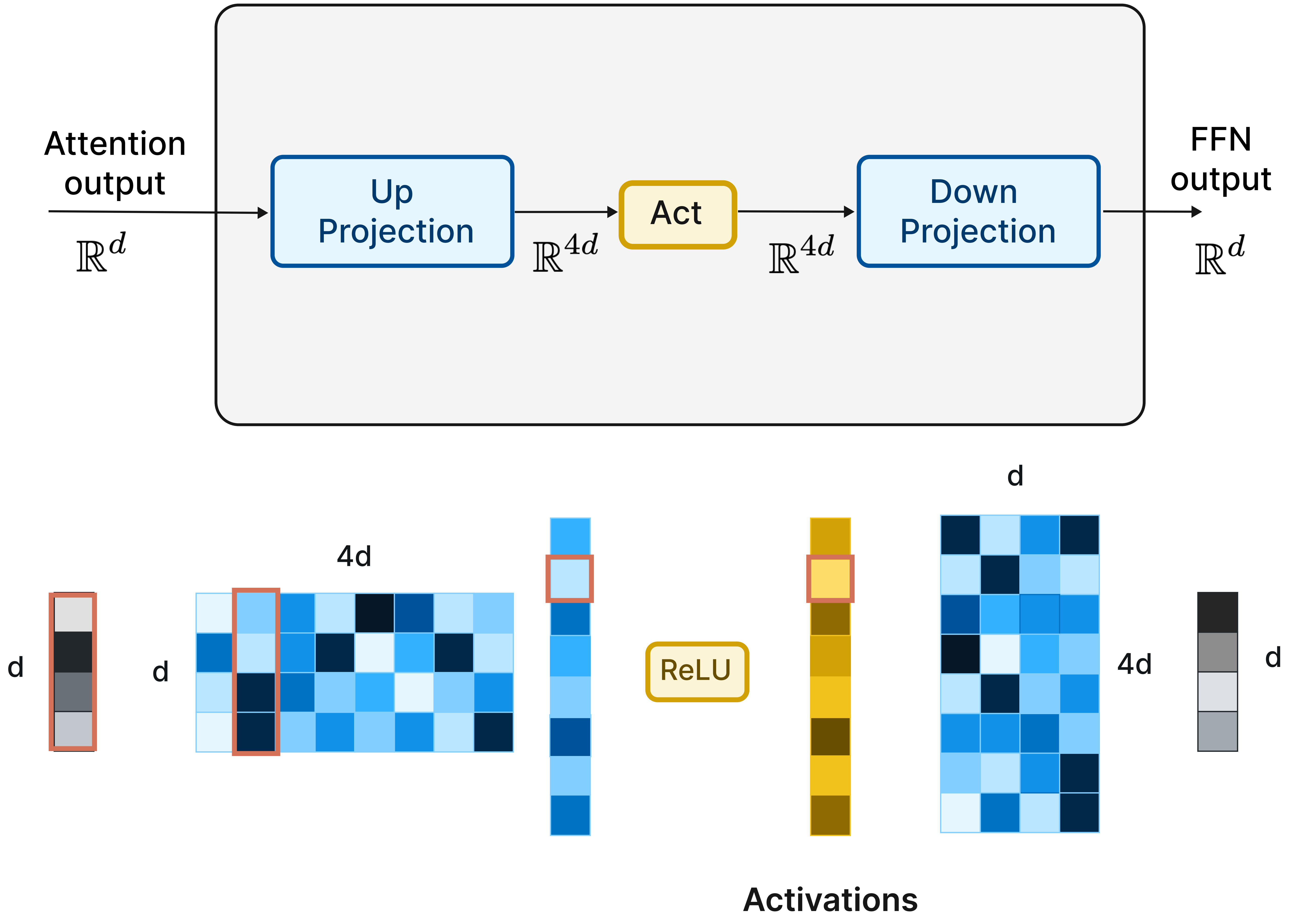
Motivation: Skipping FFN computation



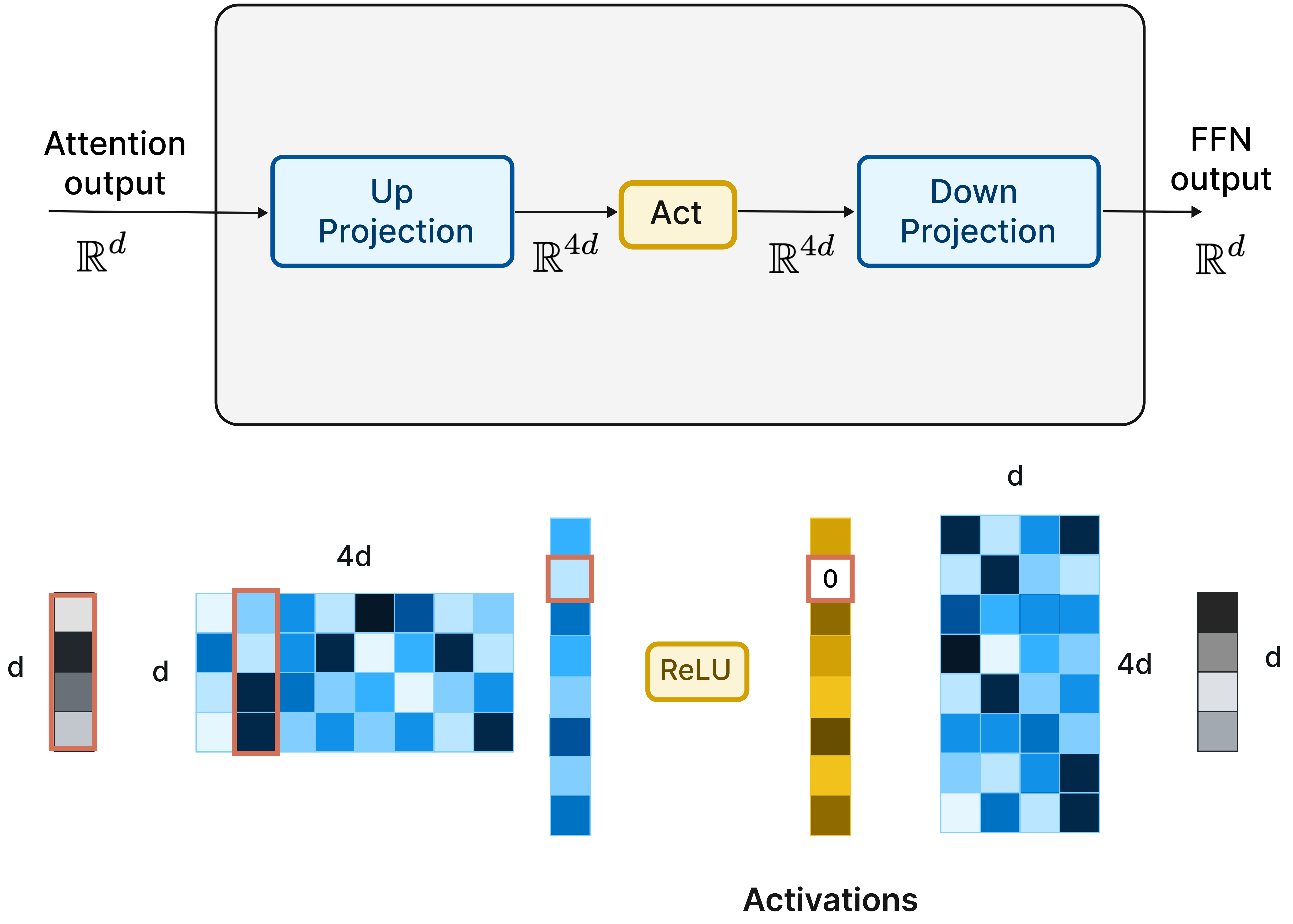
Motivation: Skipping FFN computation



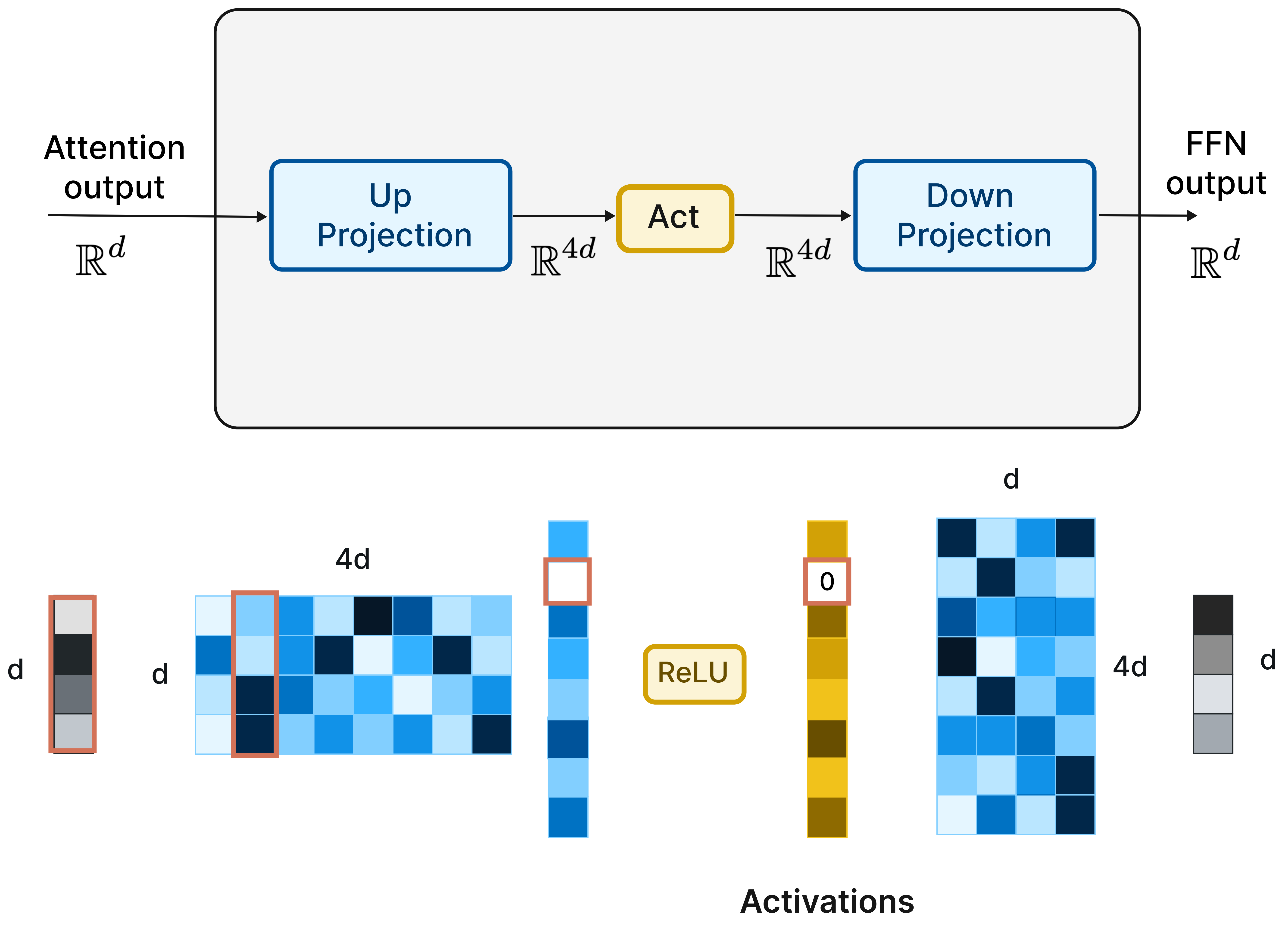
Motivation: Skipping FFN computation



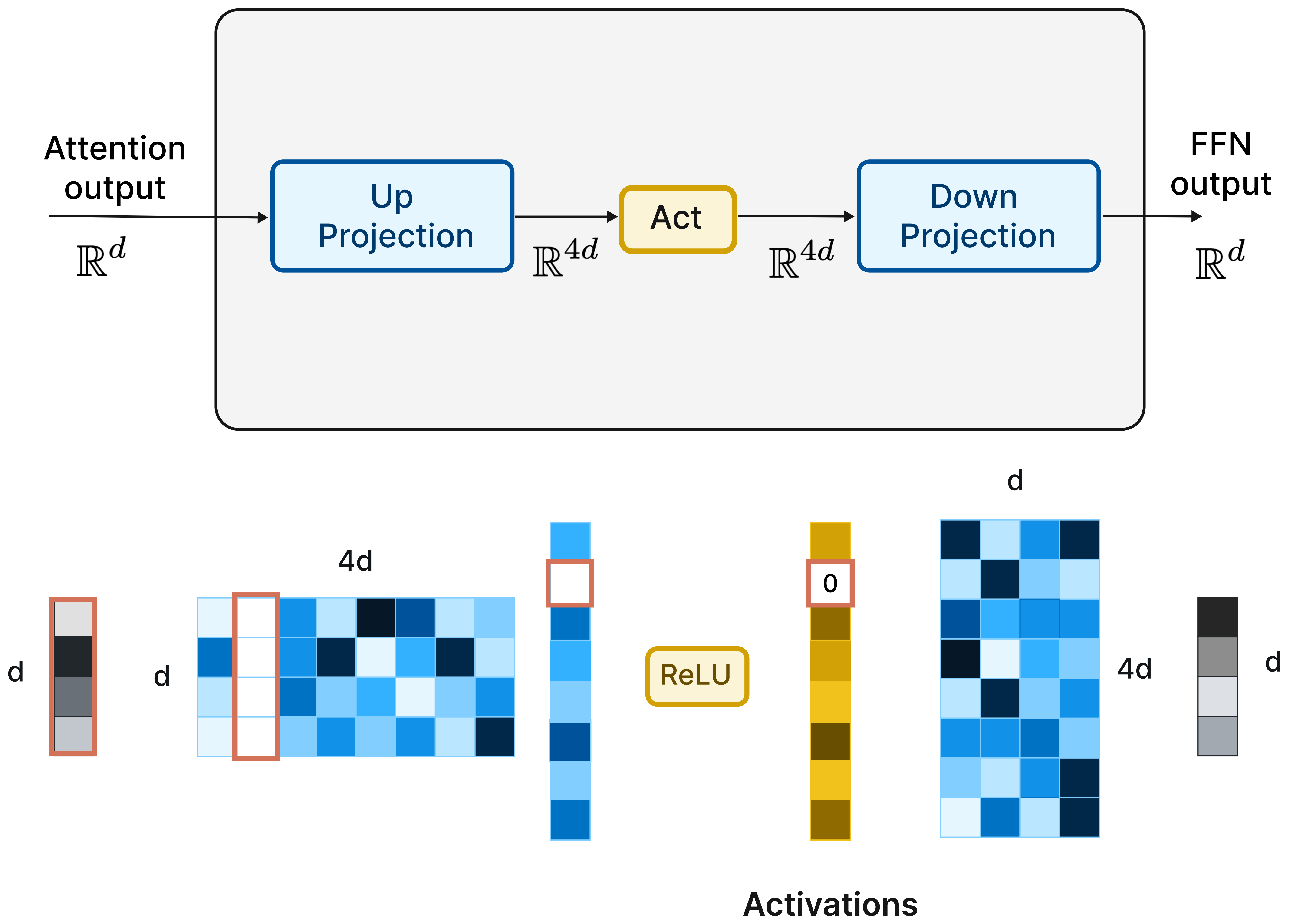
Motivation: Skipping FFN computation



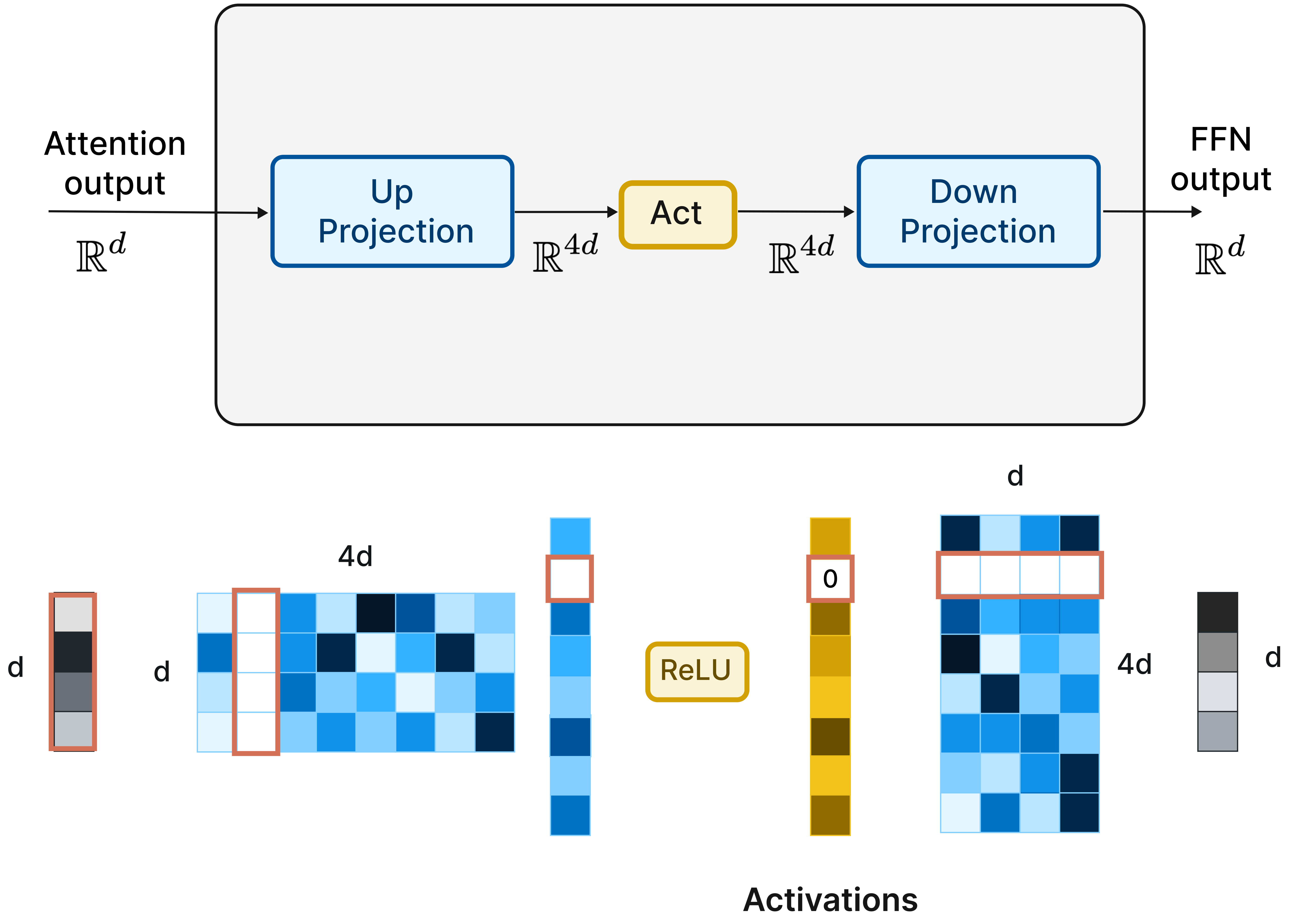
Motivation: Skipping FFN computation



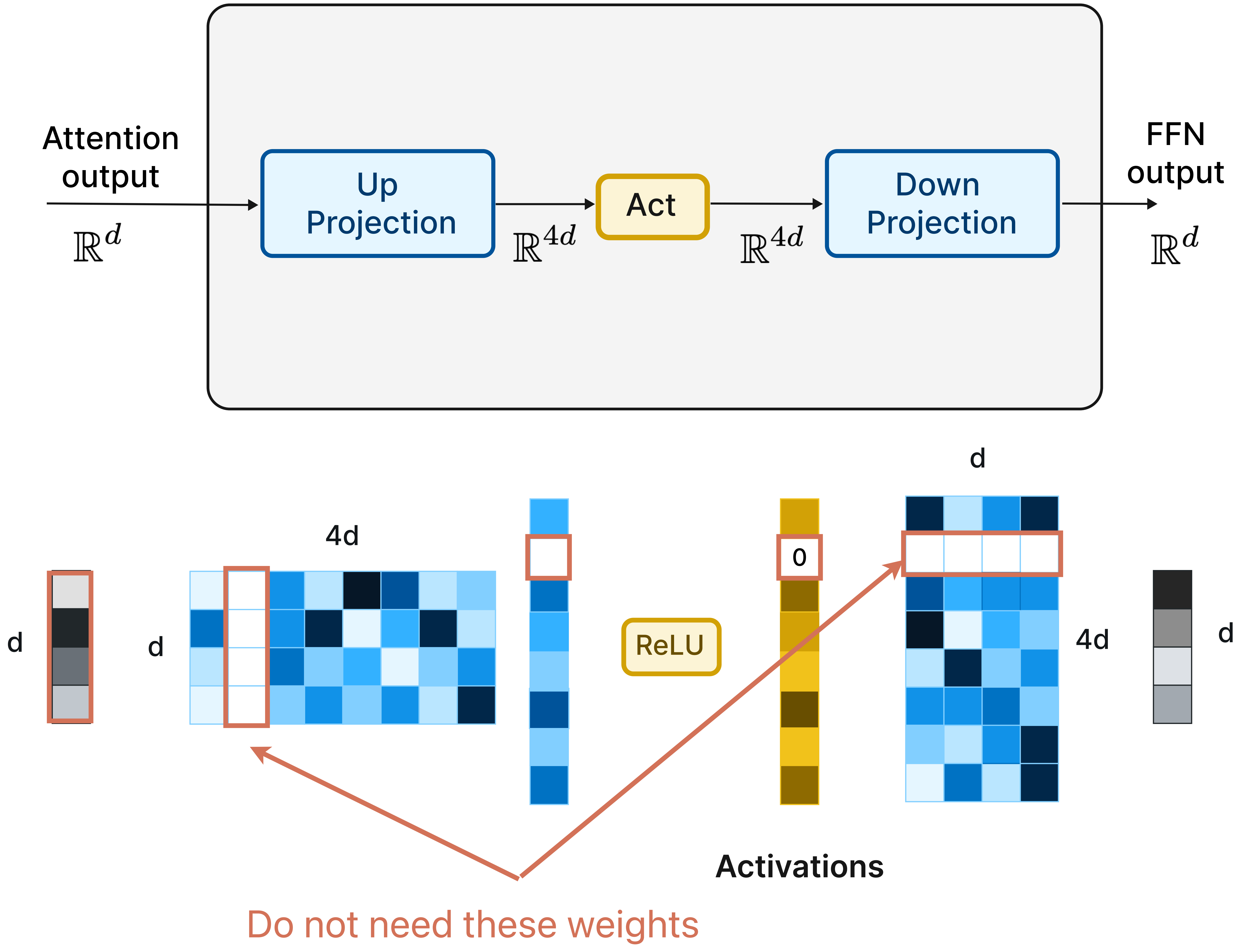
Motivation: Skipping FFN computation



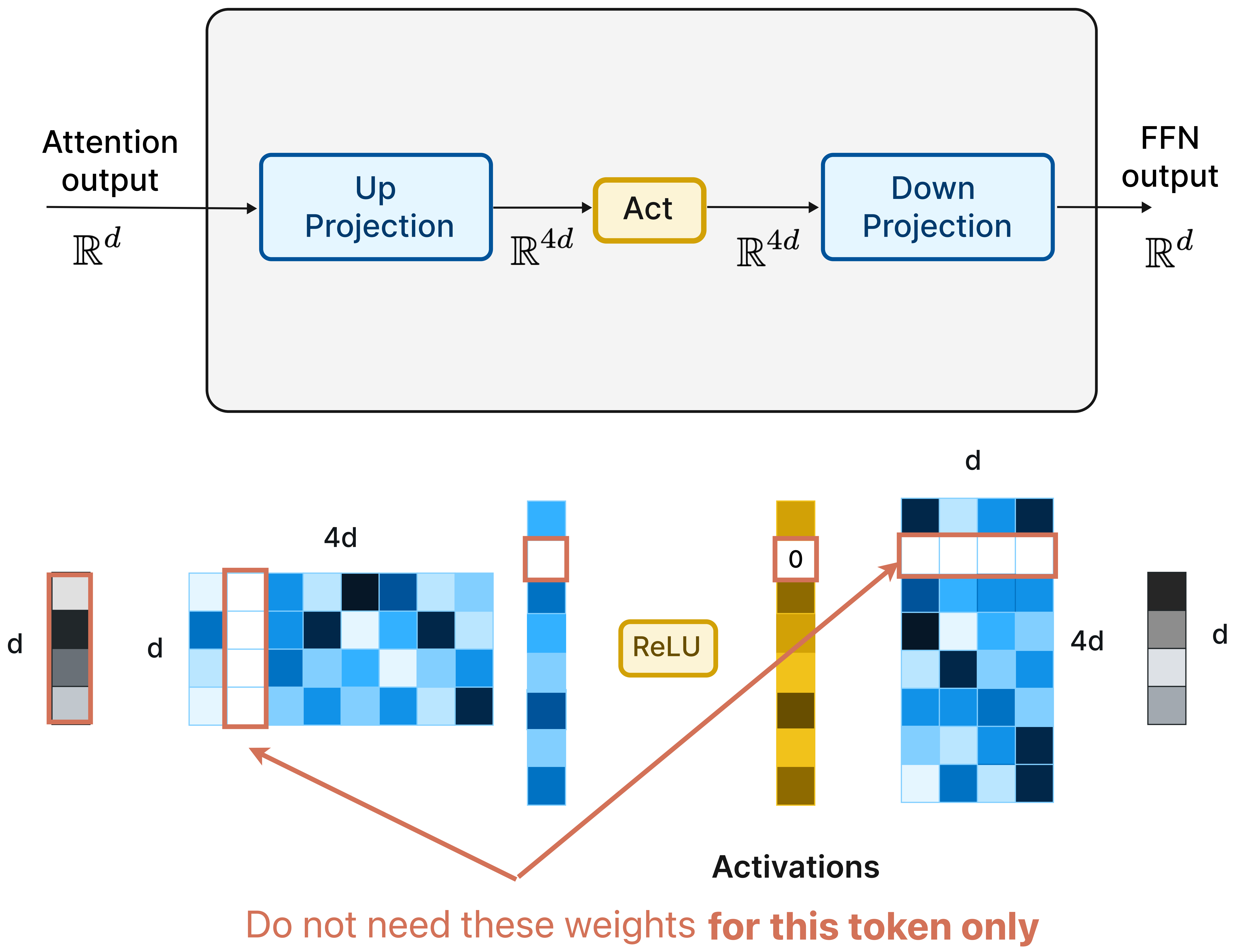
Motivation: Skipping FFN computation



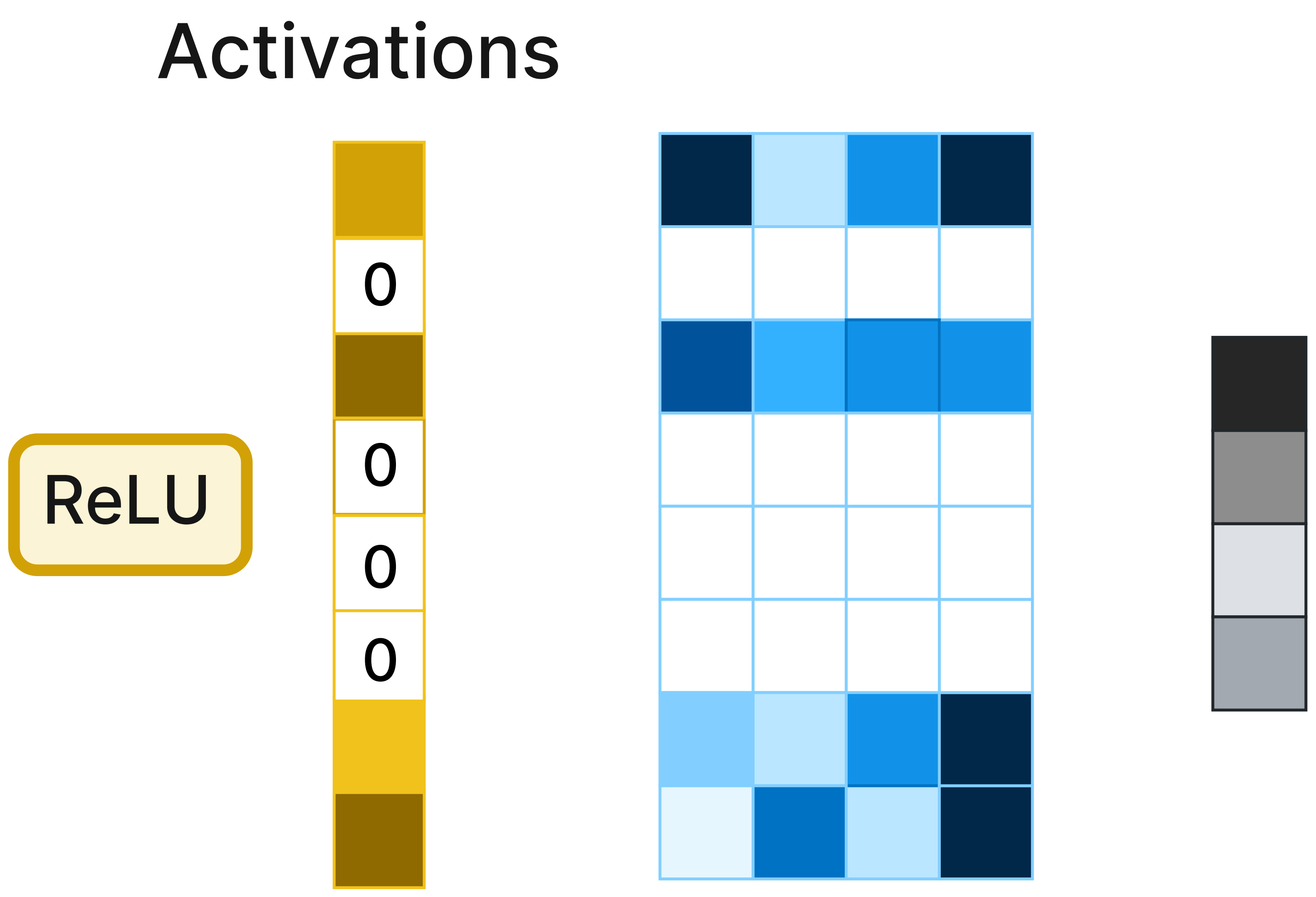
Motivation: Skipping FFN computation



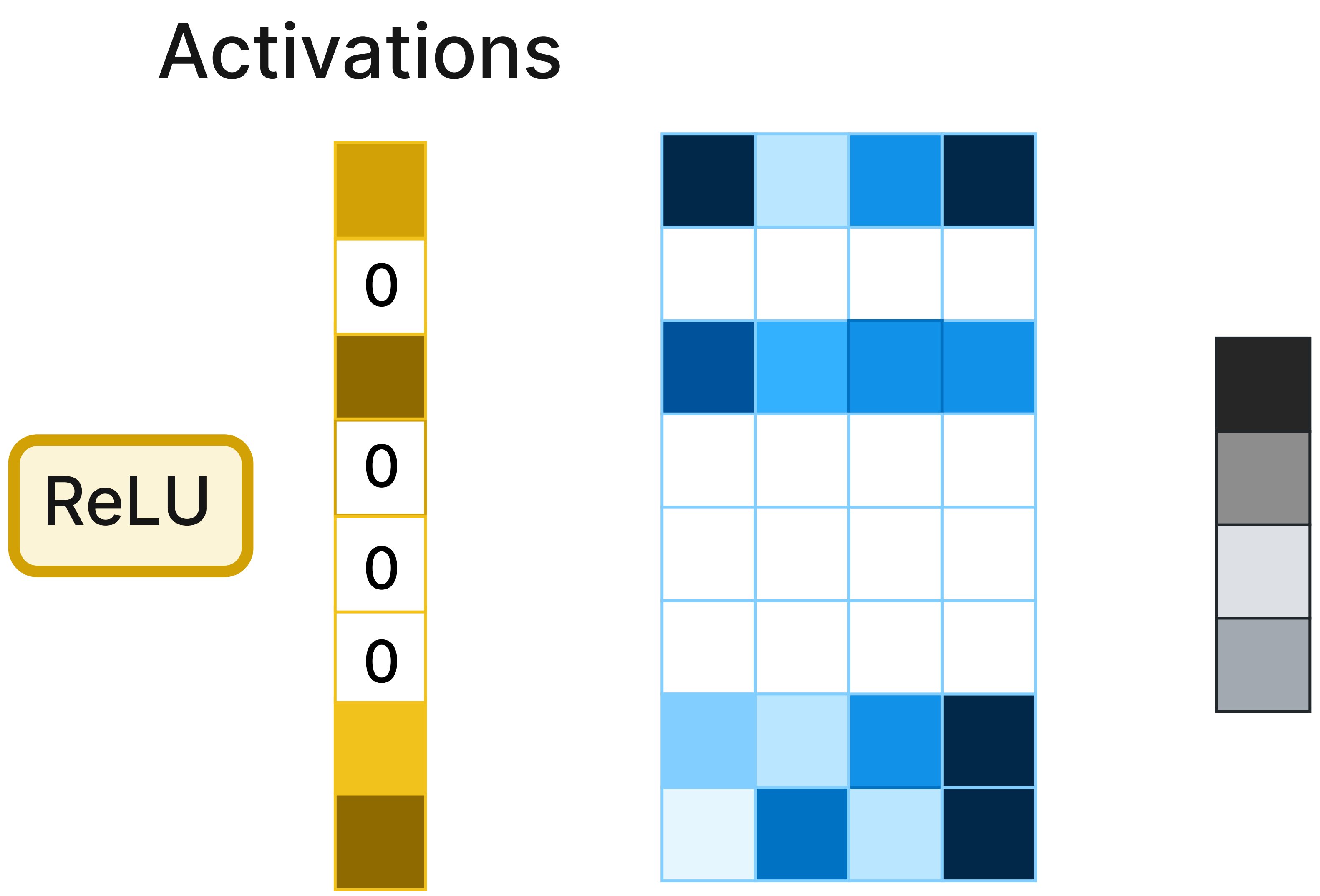
Motivation: Skipping FFN computation

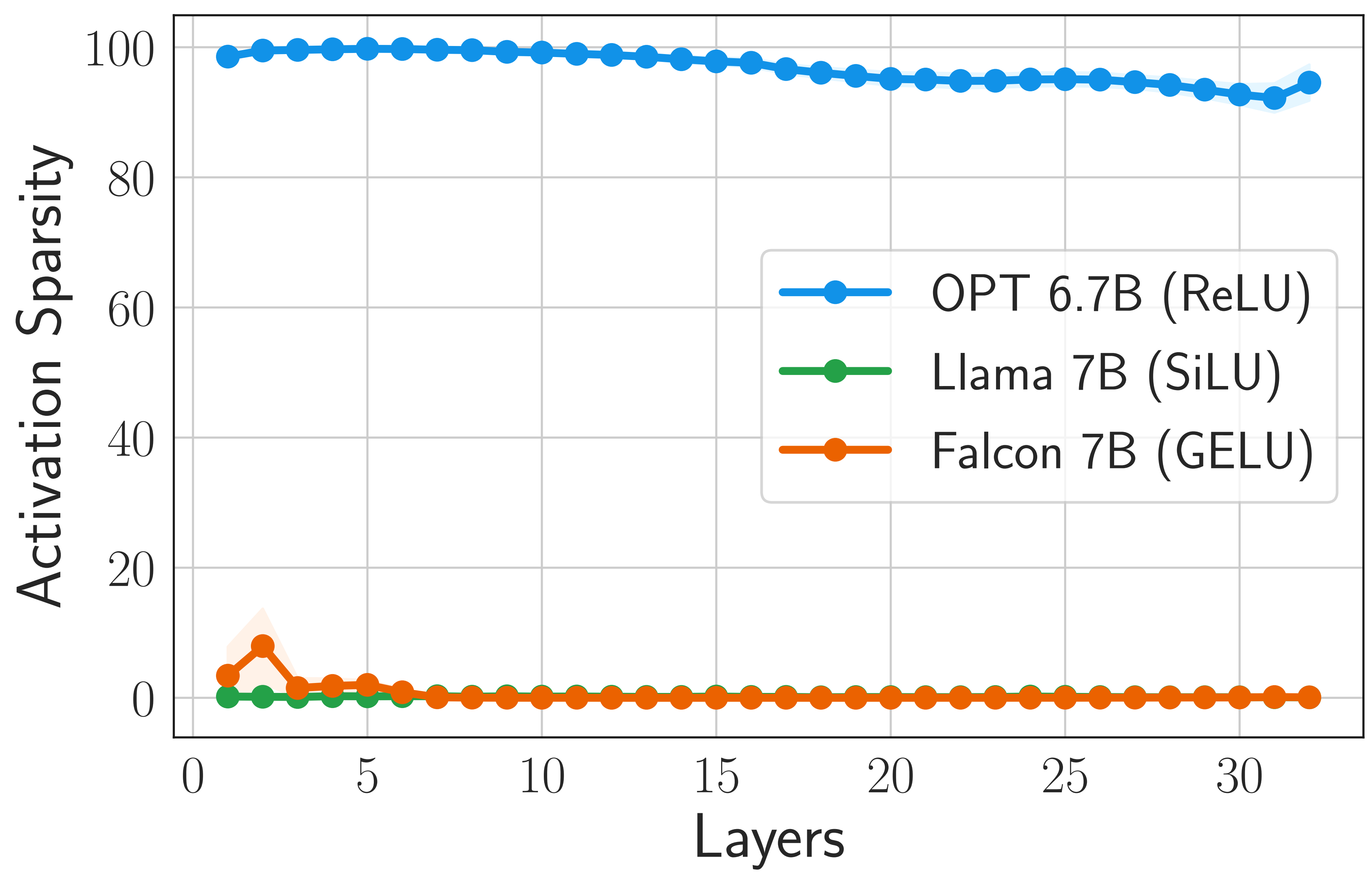


Activation Sparsity: The Benefits of ReLU

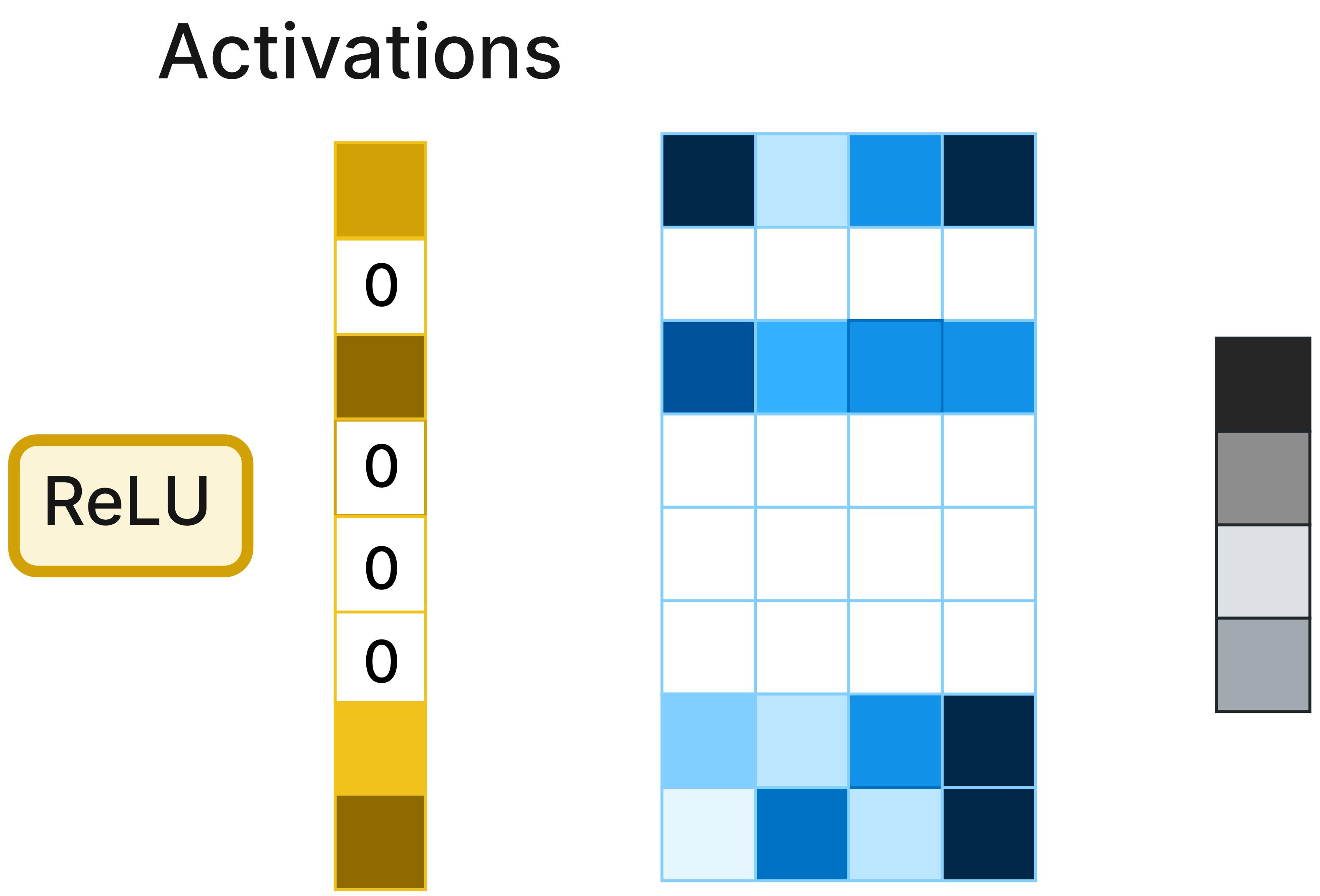


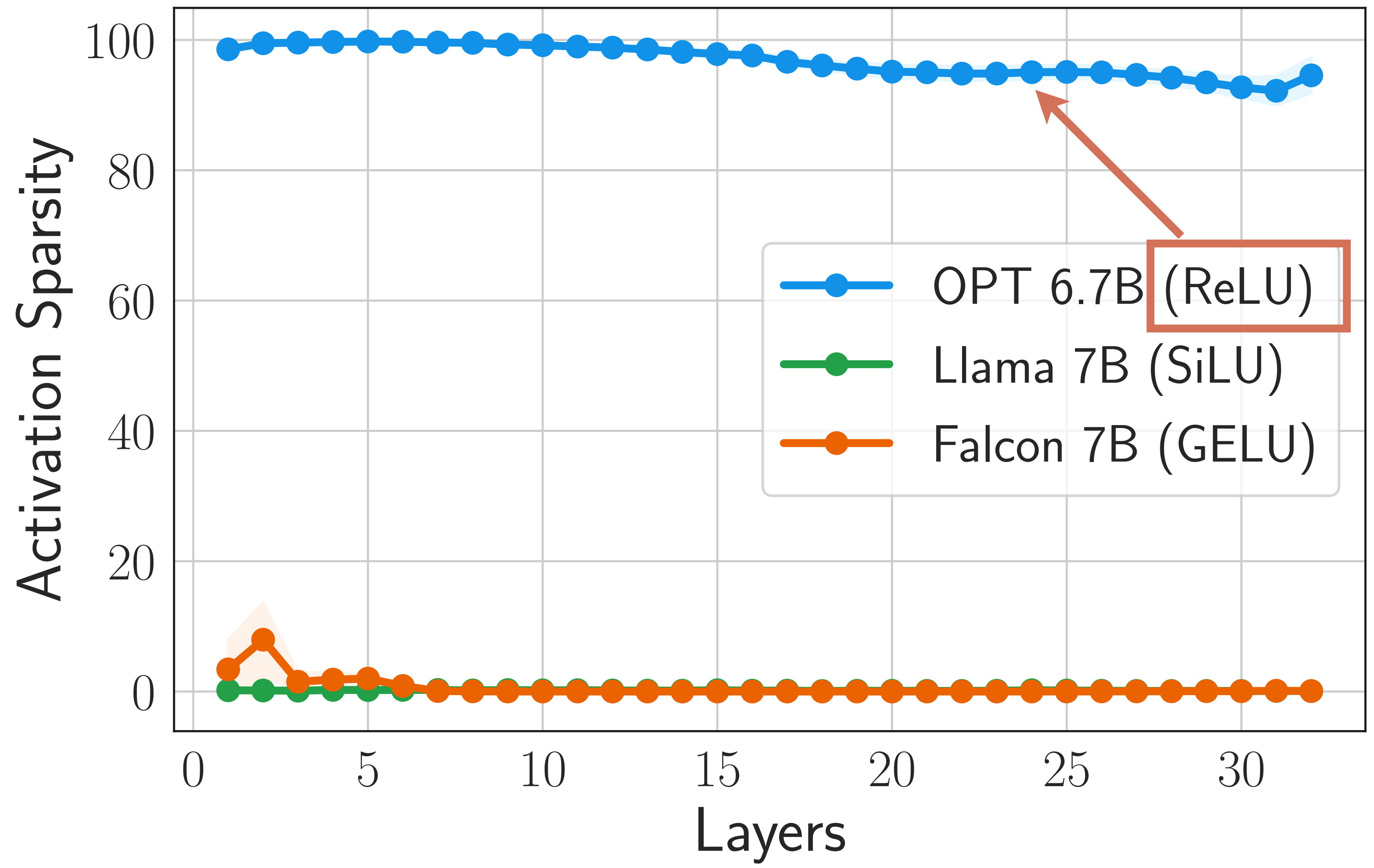
Activation Sparsity: The Benefits of ReLU



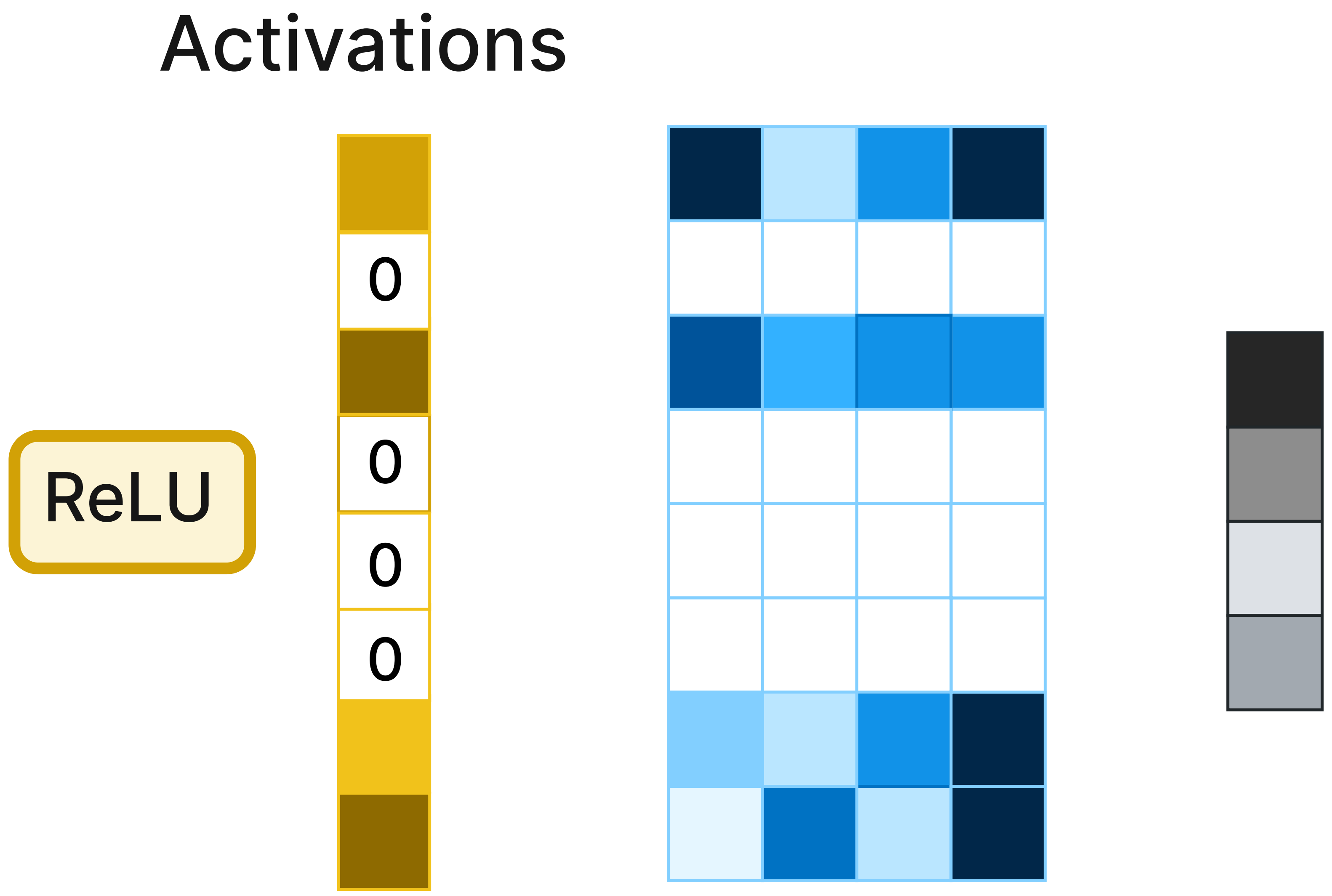


Activation Sparsity: The Benefits of ReLU

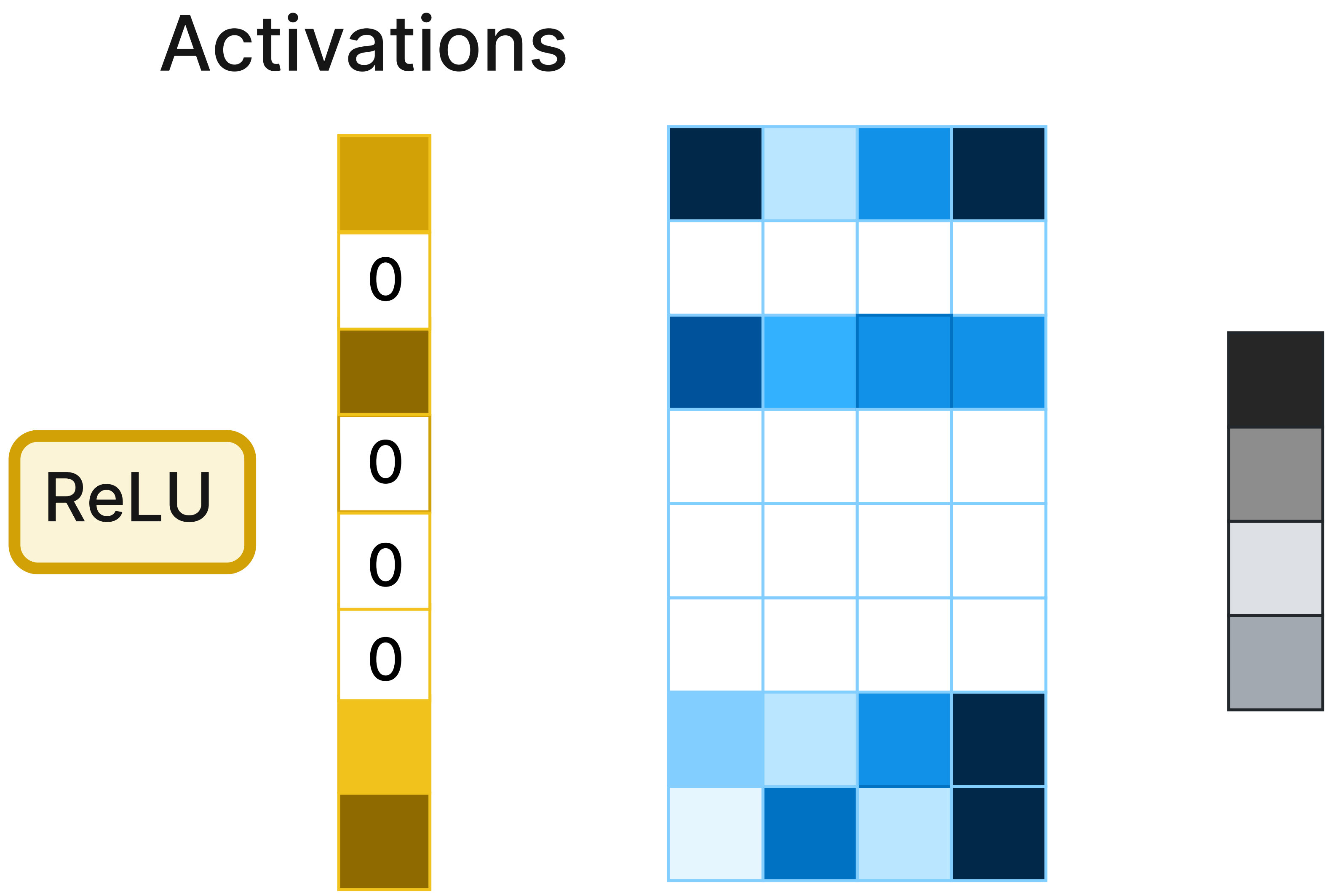


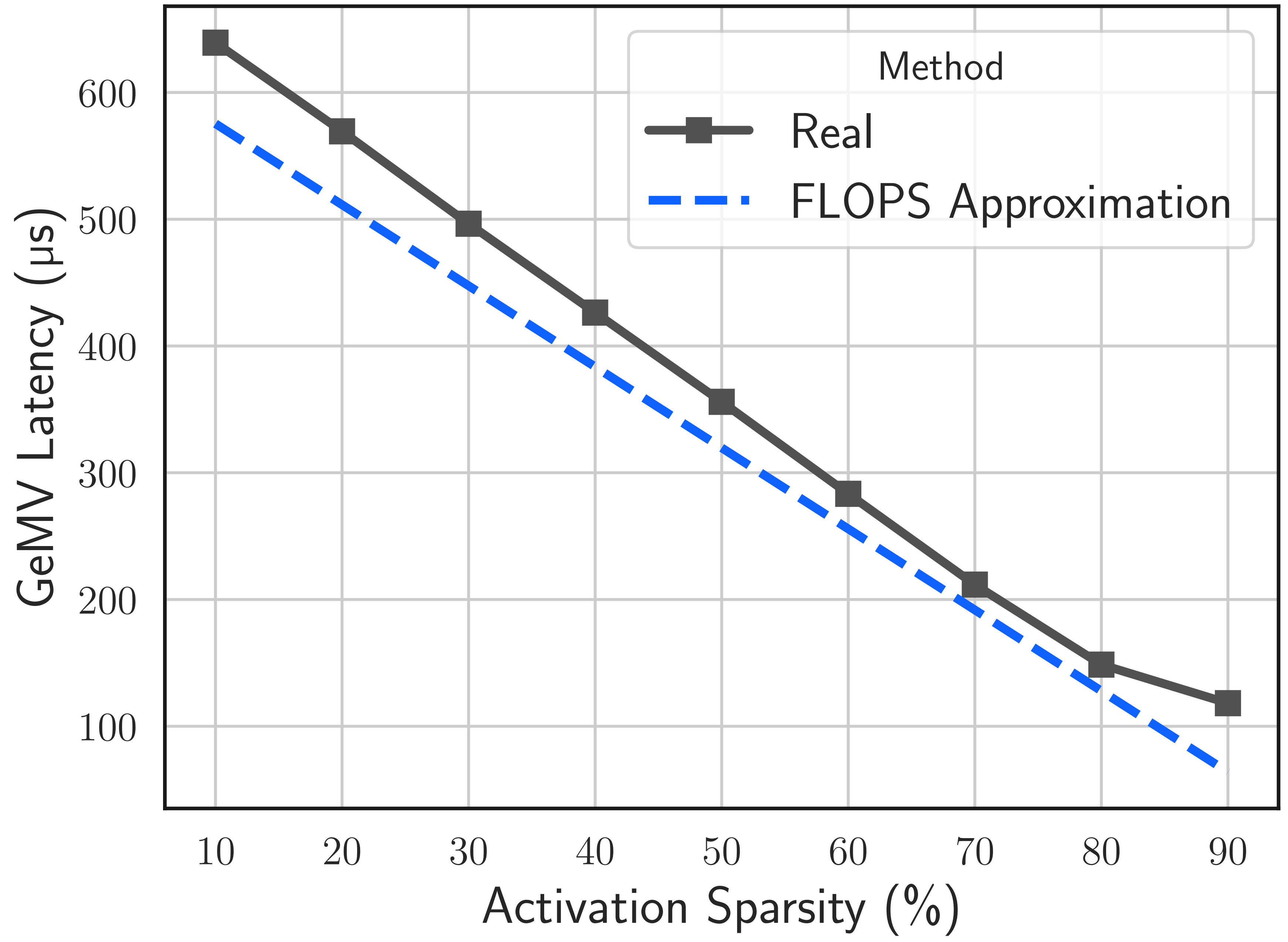


Activation Sparsity: Efficient Implementation



Activation Sparsity: Efficient Implementation



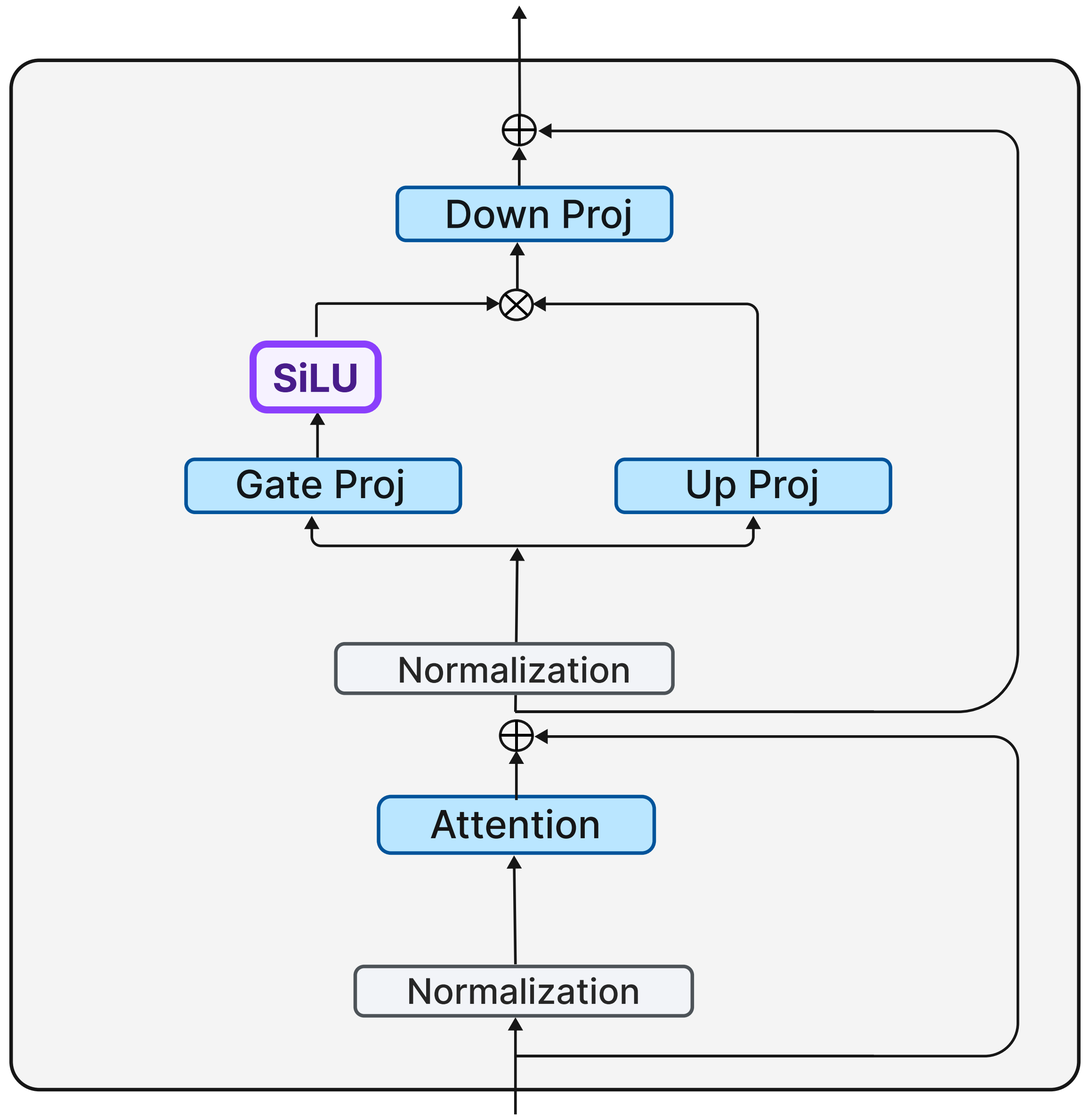


Efficient Matrix-Vector product Metal Kernel on M2-Macbook Pro

But most of the LLMs are already pre-trained without ReLU

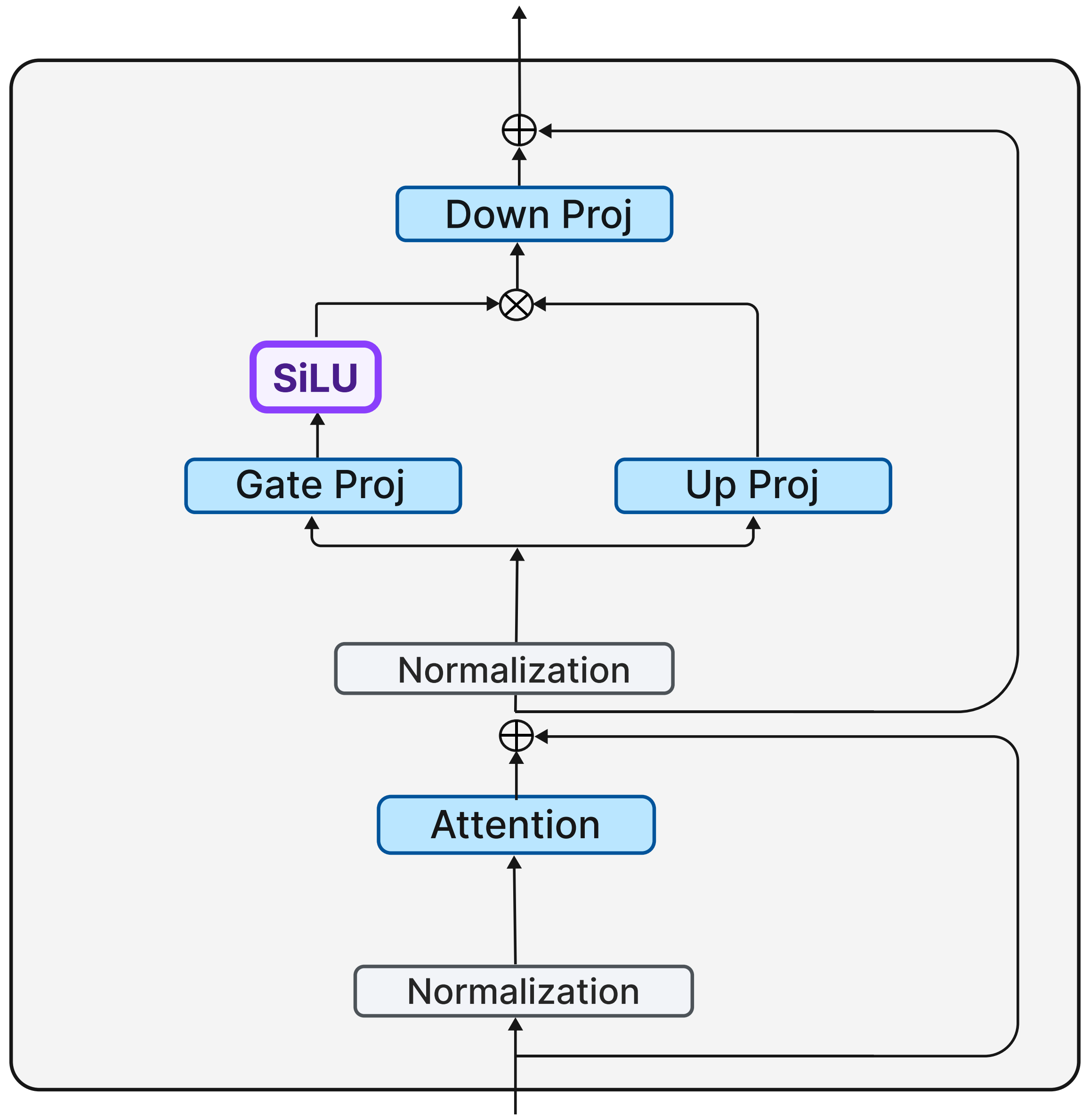
ReLUfication: Bringing ReLU Back to non-ReLU Pre-Trained Models

ReLUfication: Bringing ReLU Back to non-ReLU Pre-Trained Models

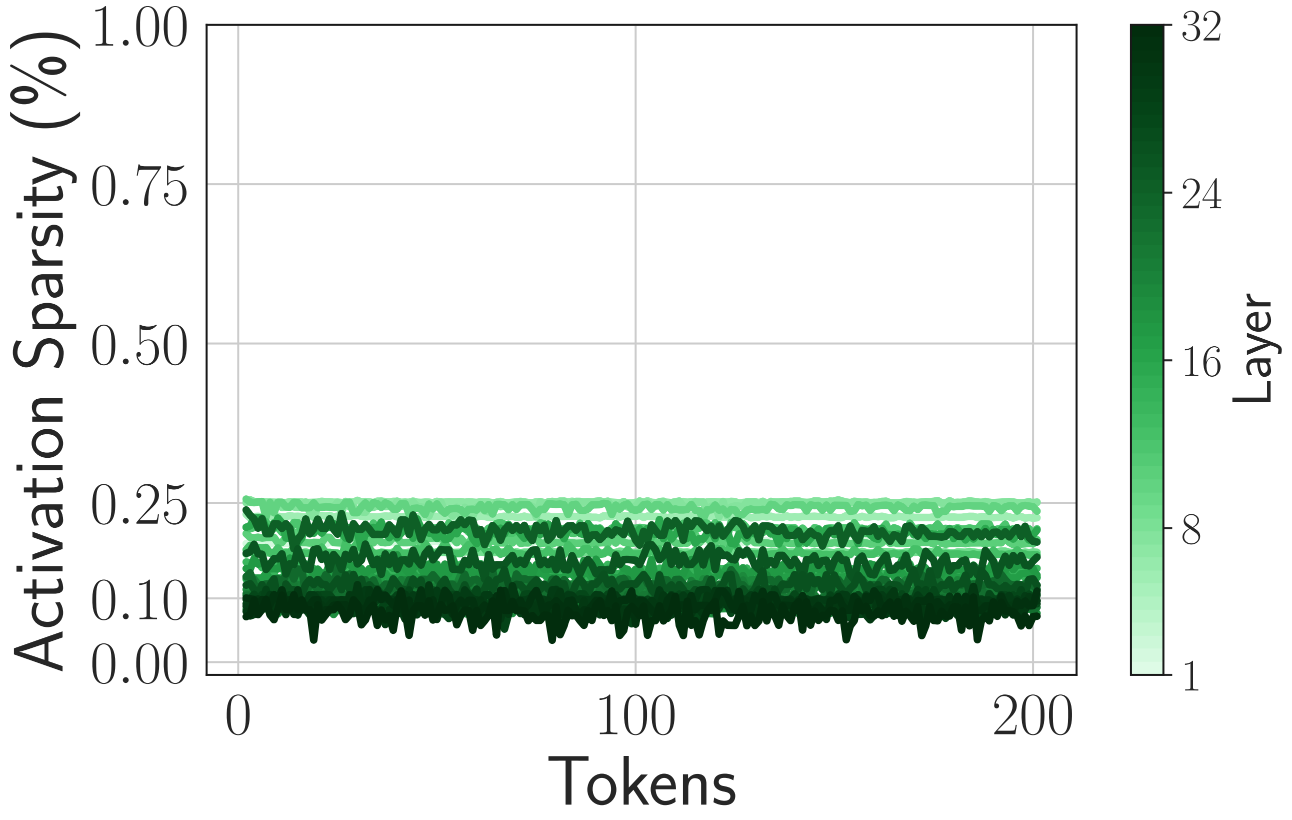


LLaMA Layer

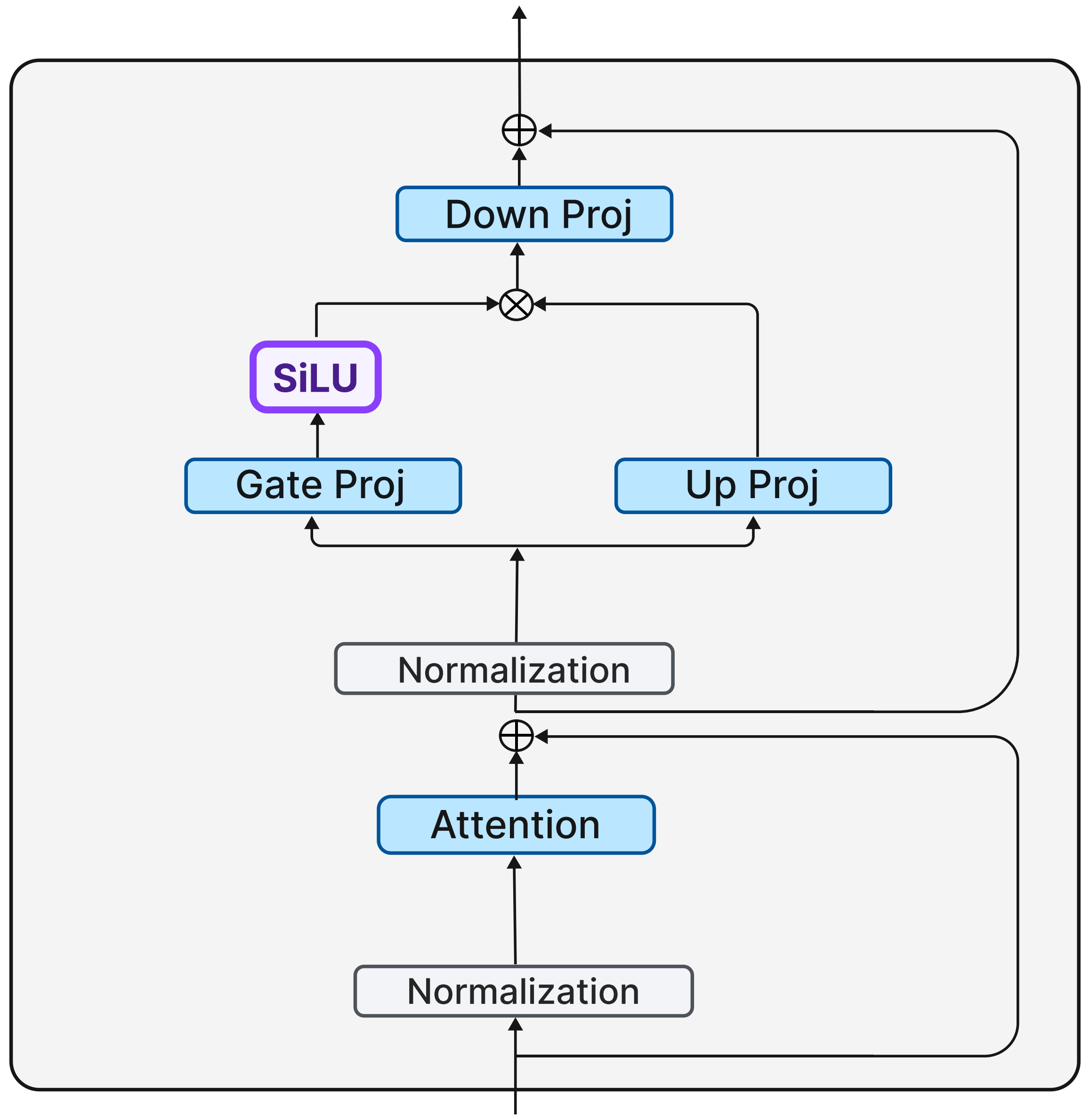
ReLUfication: Bringing ReLU Back to non-ReLU Pre-Trained Models



LLaMA Layer

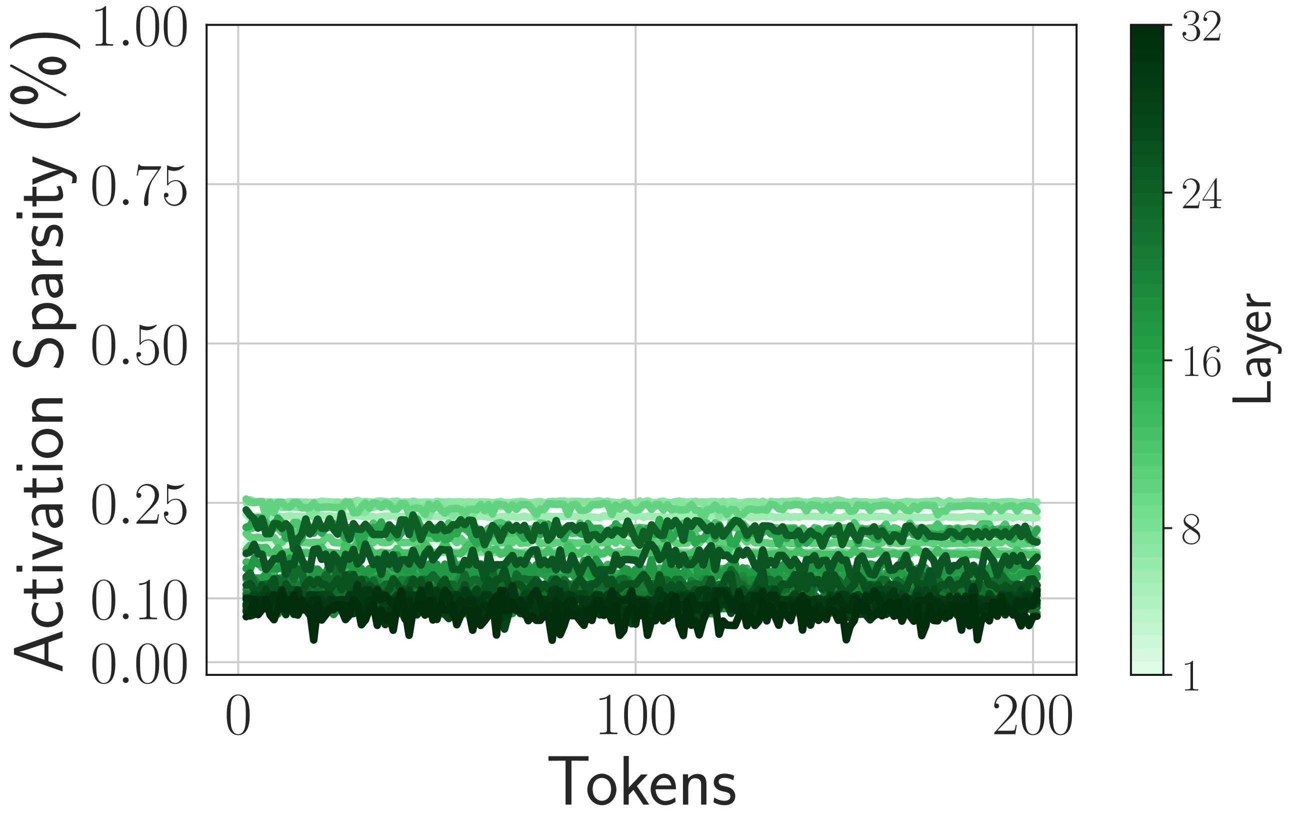


ReLUfication: Bringing ReLU Back to non-ReLU Pre-Trained Models

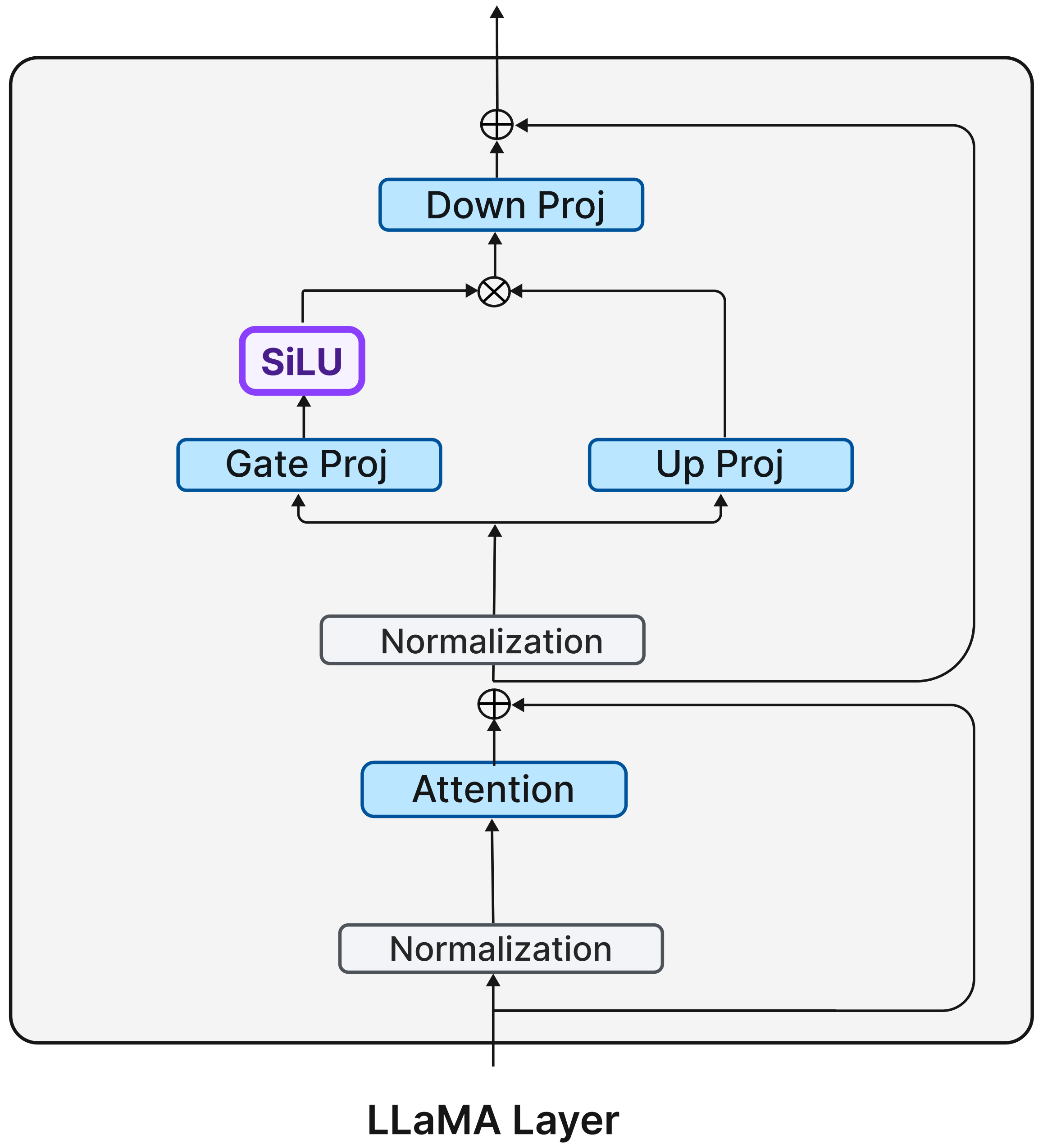


LLaMA Layer

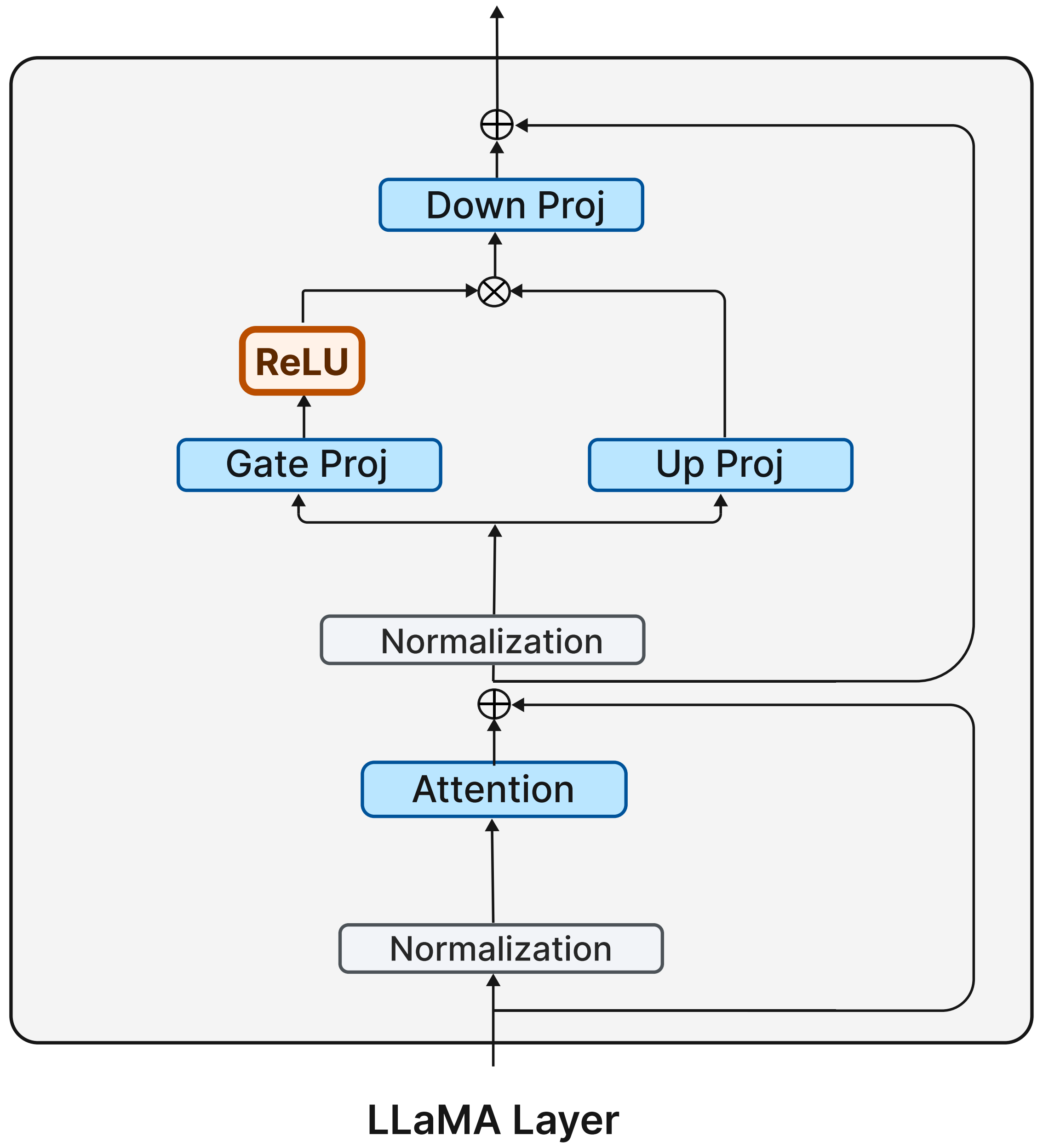
relufication

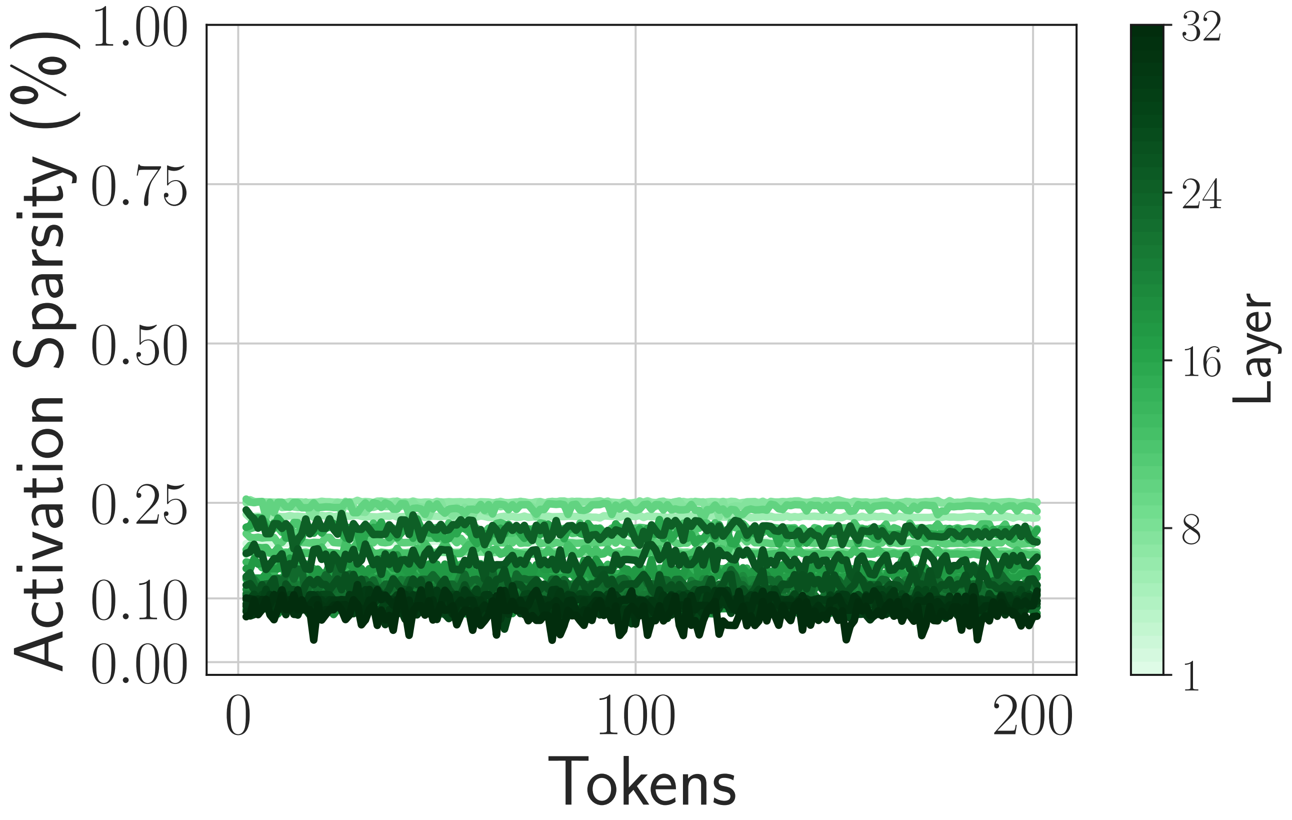


ReLUfication: Bringing ReLU Back to non-ReLU Pre-Trained Models

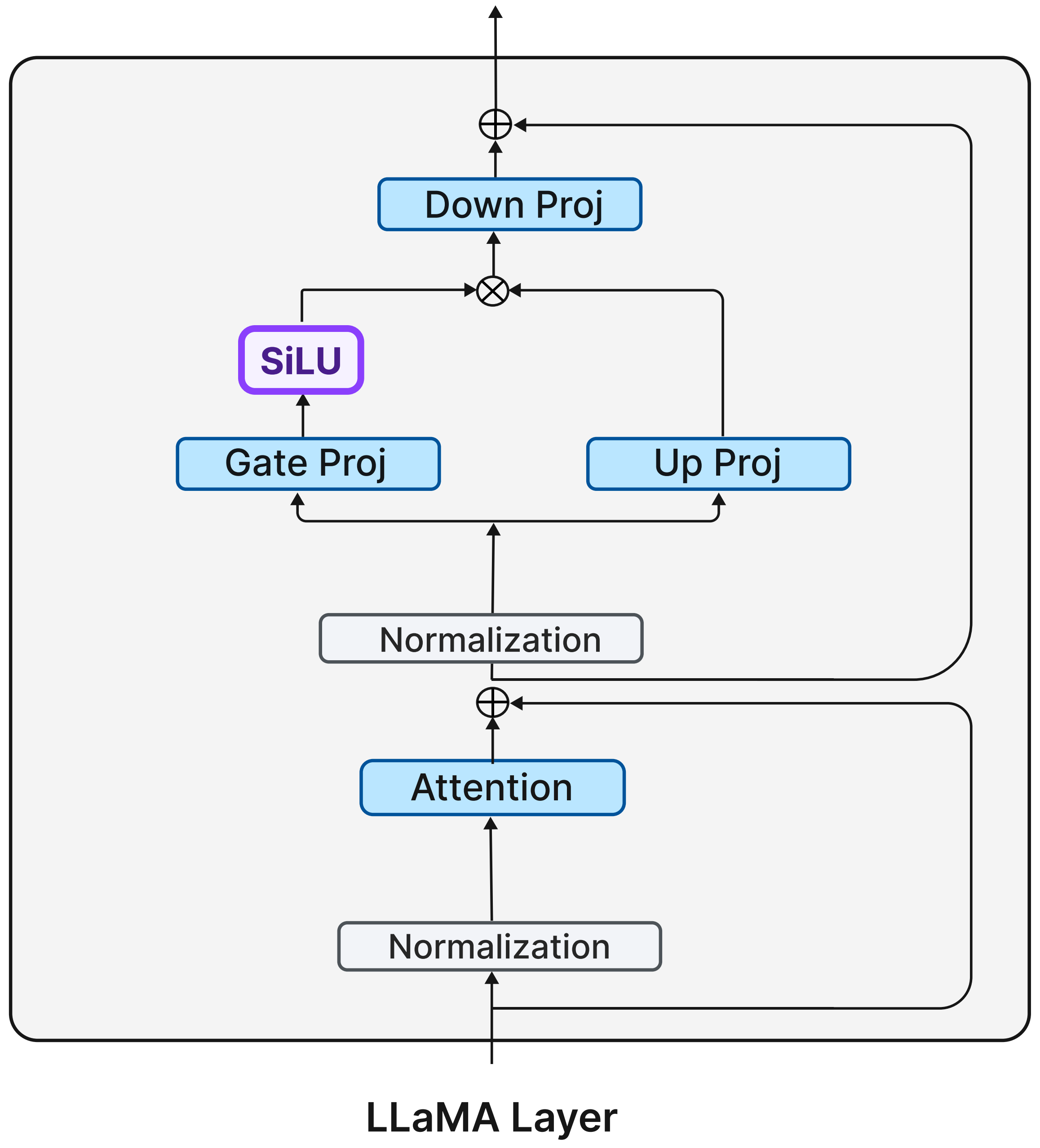


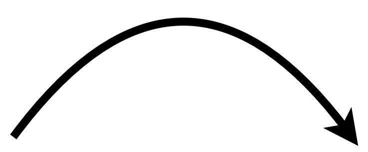
relufication

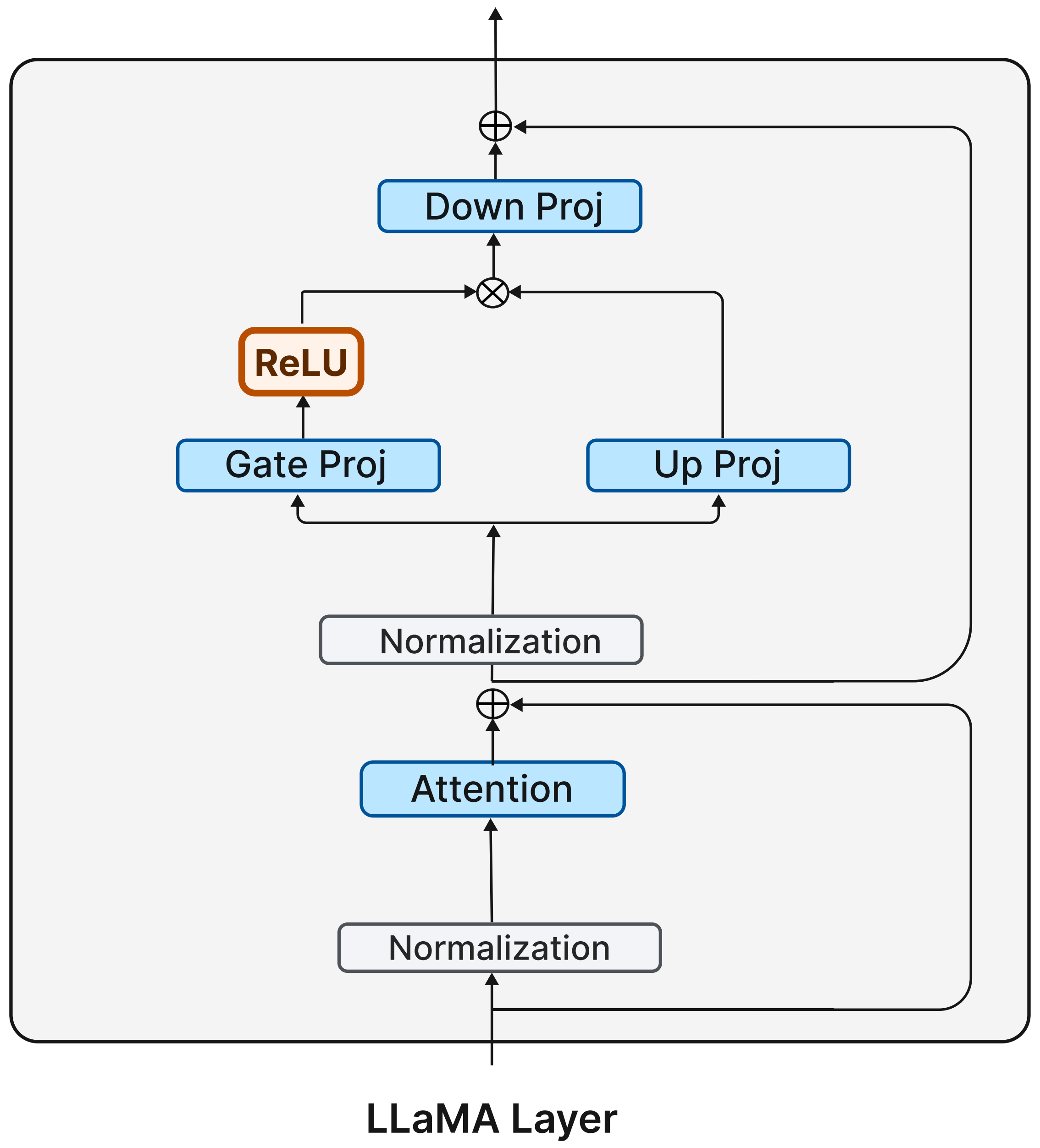


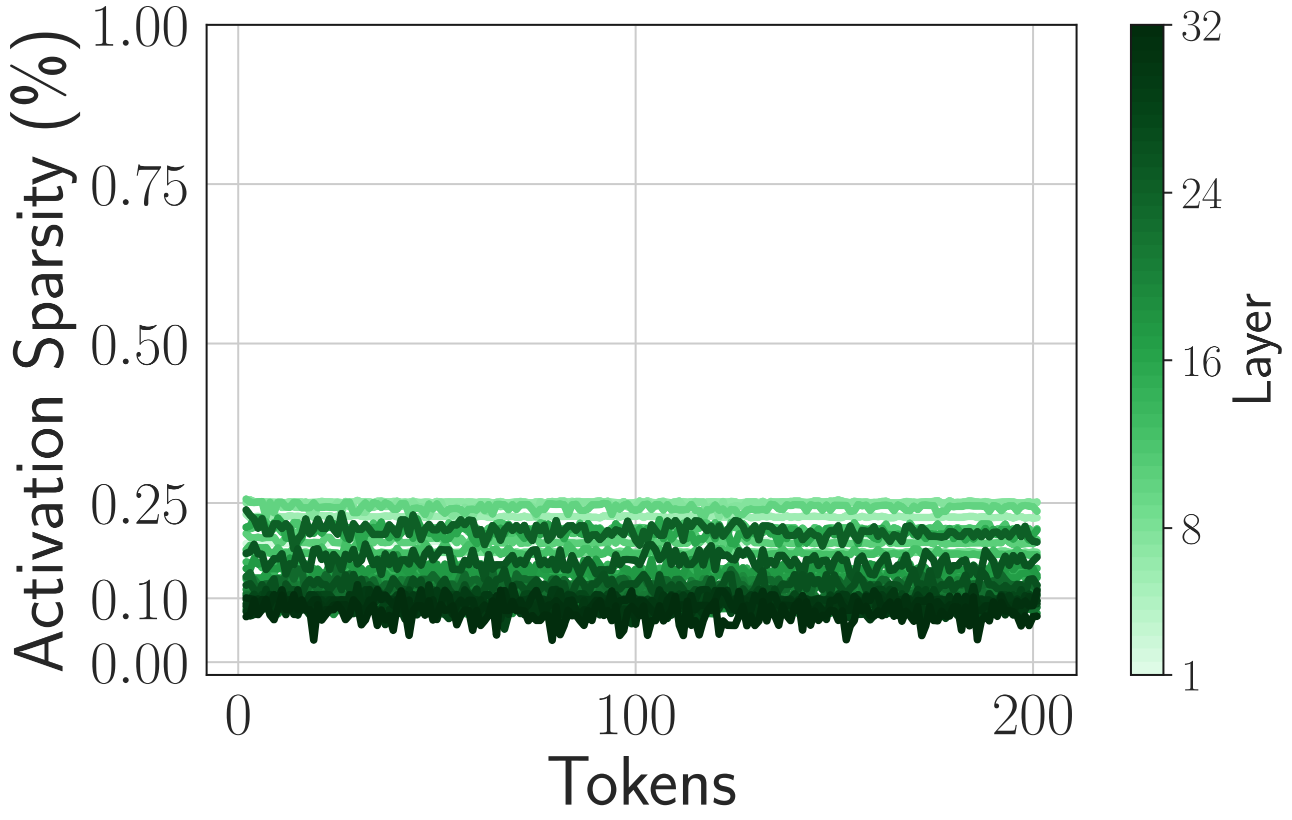


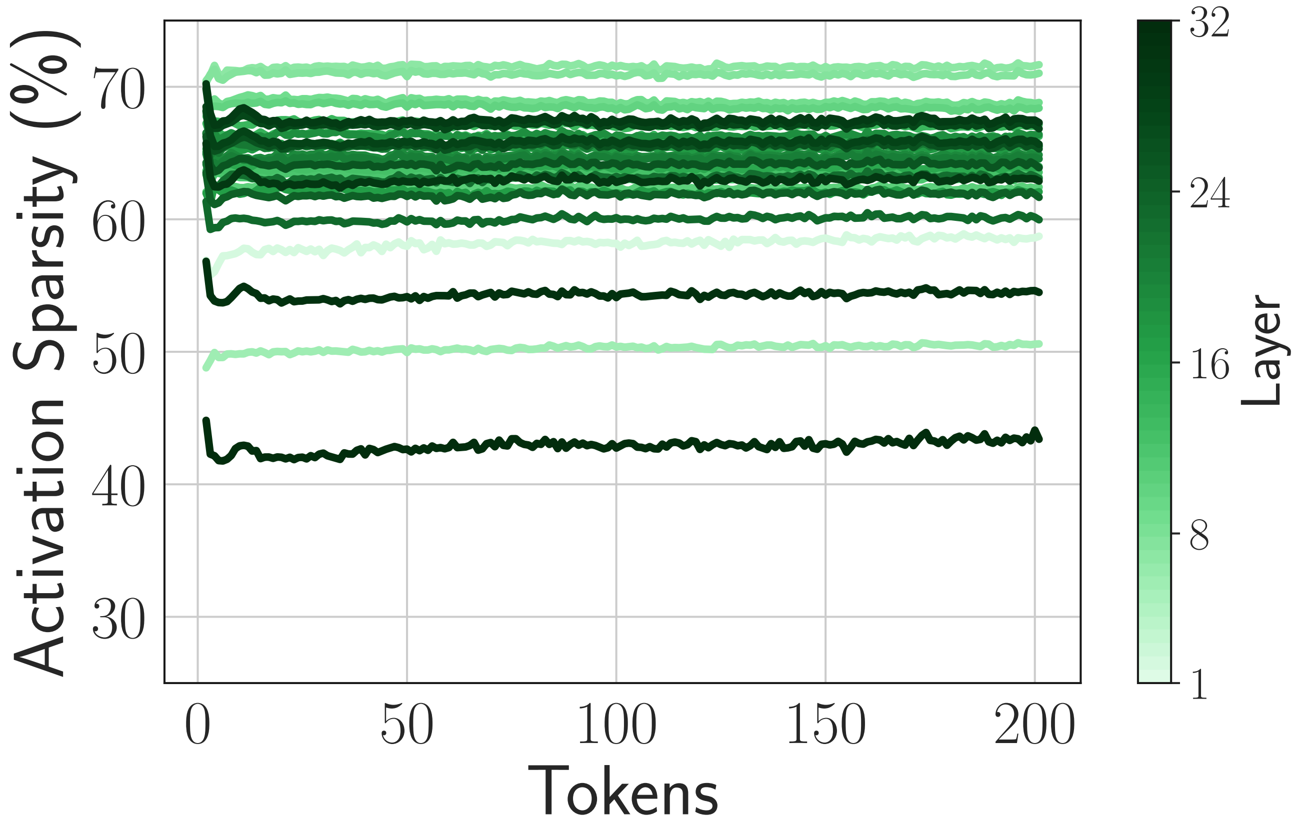
ReLUfication: Bringing ReLU Back to non-ReLU Pre-Trained Models



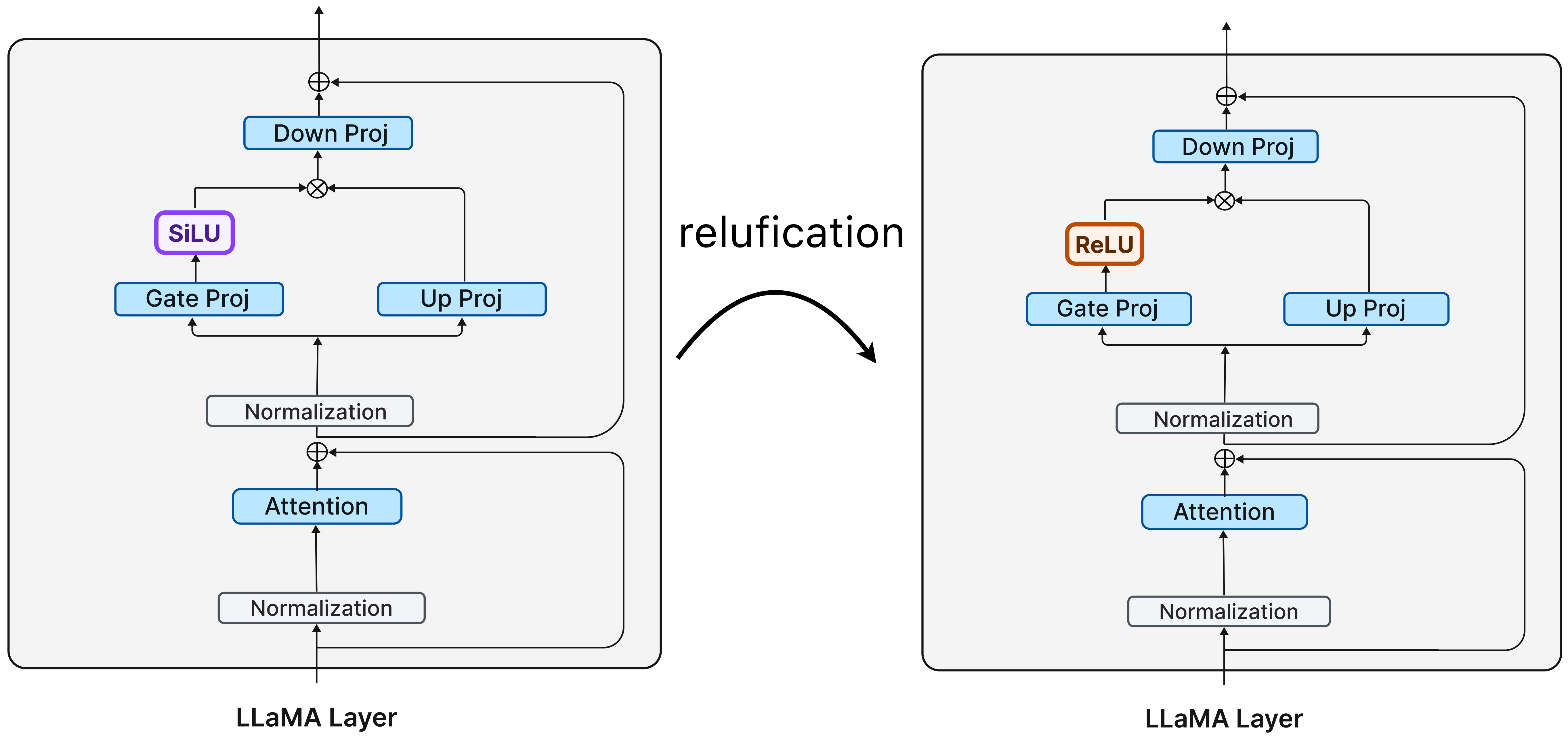


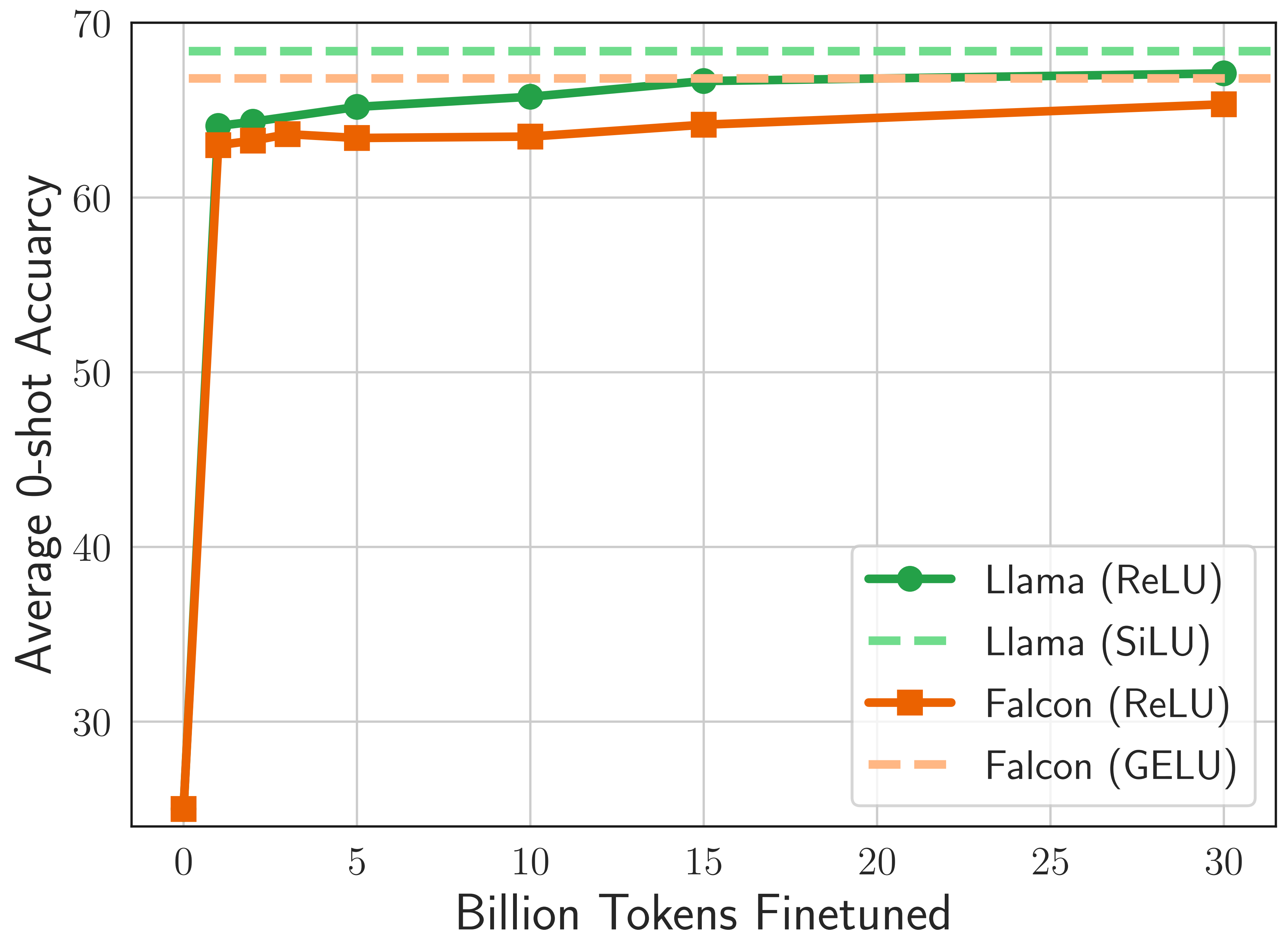


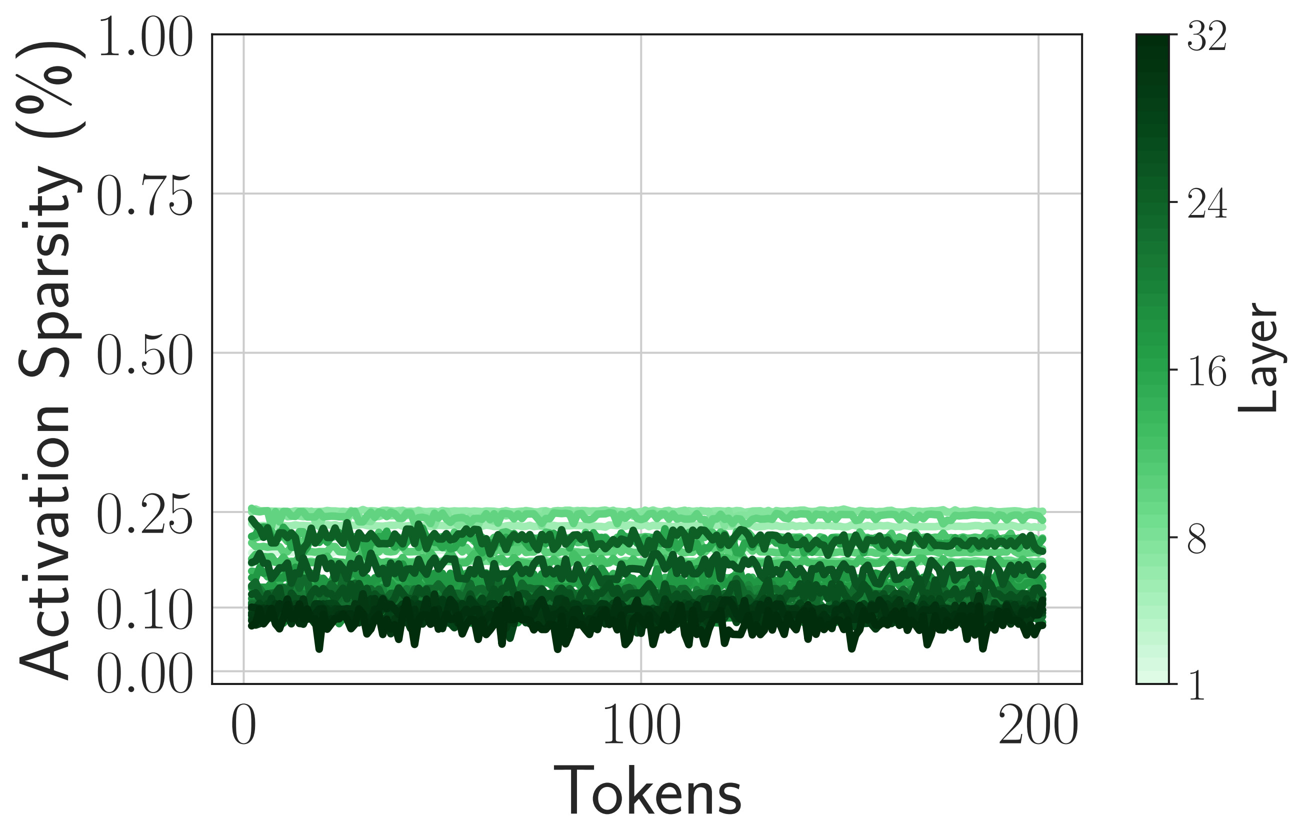


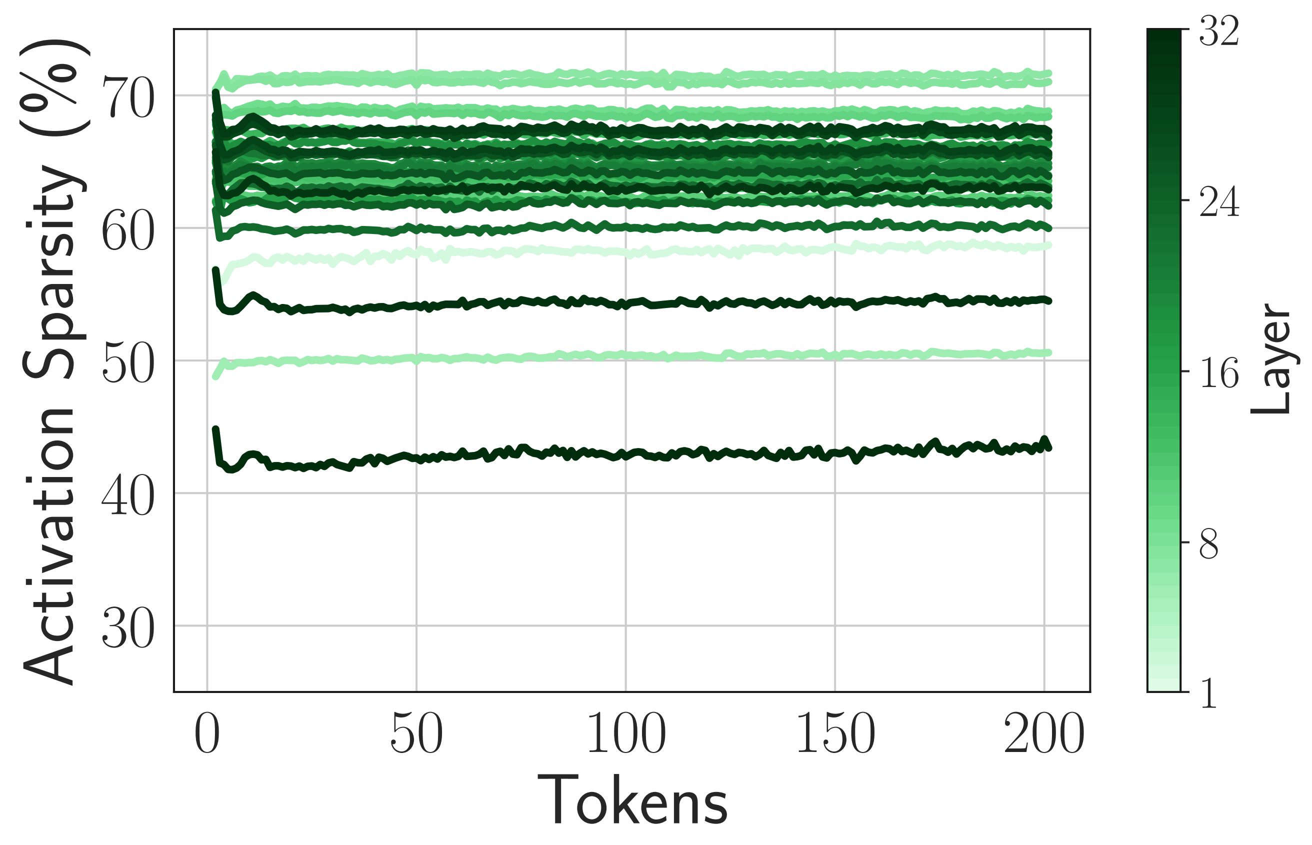


ReLUfication: Bringing ReLU Back to non-ReLU Pre-Trained Models

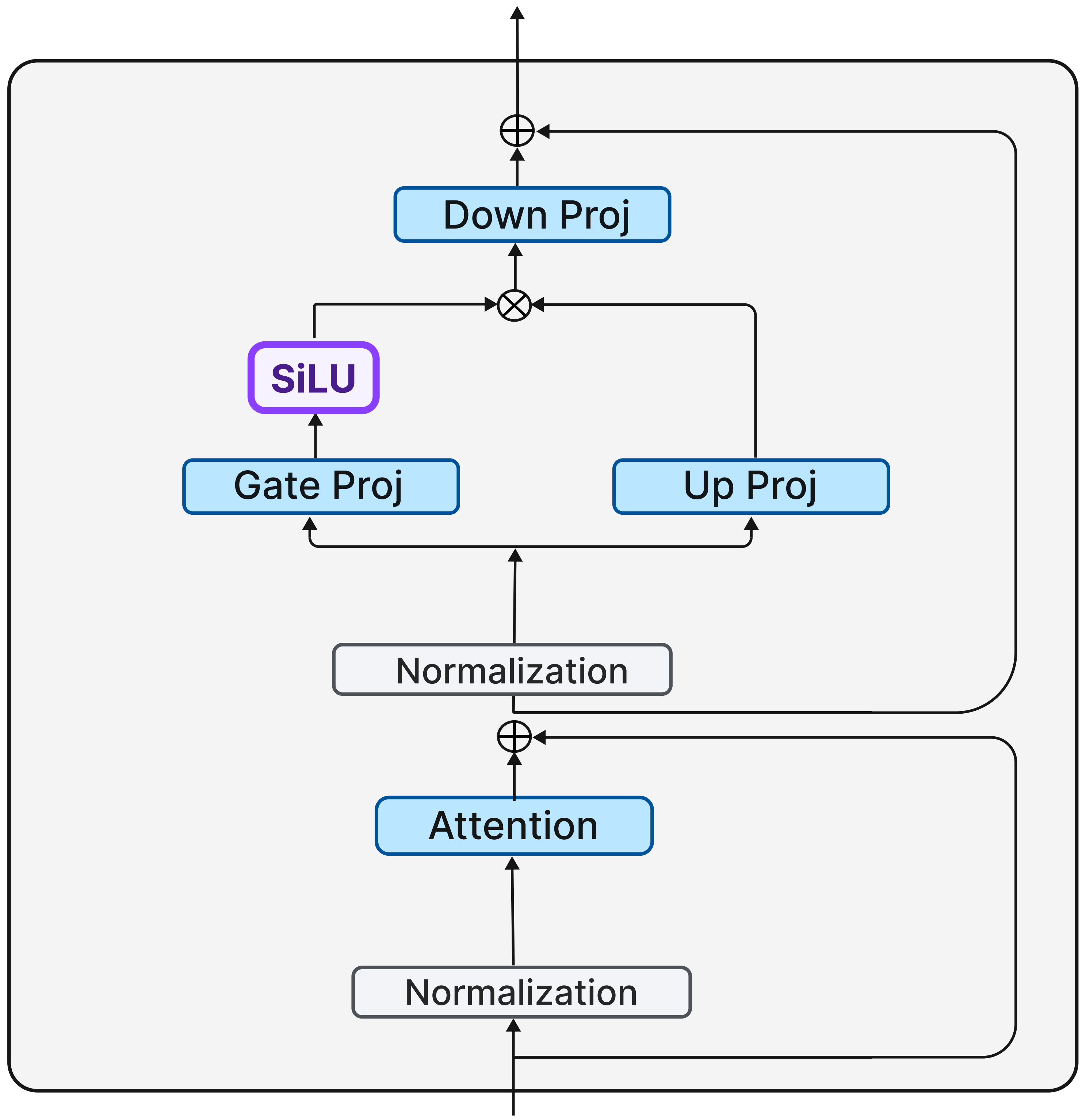




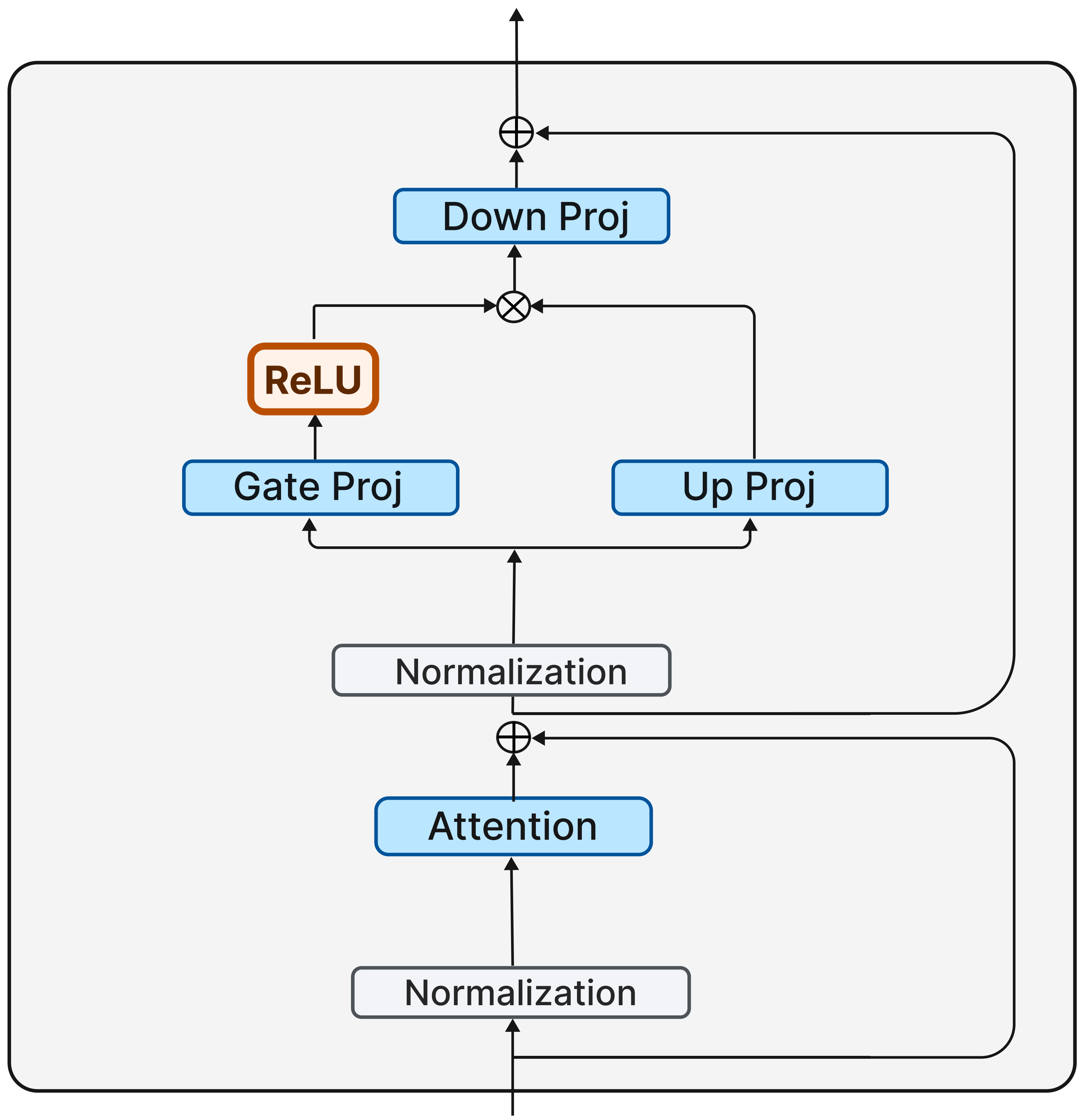




ReLUfication Stages

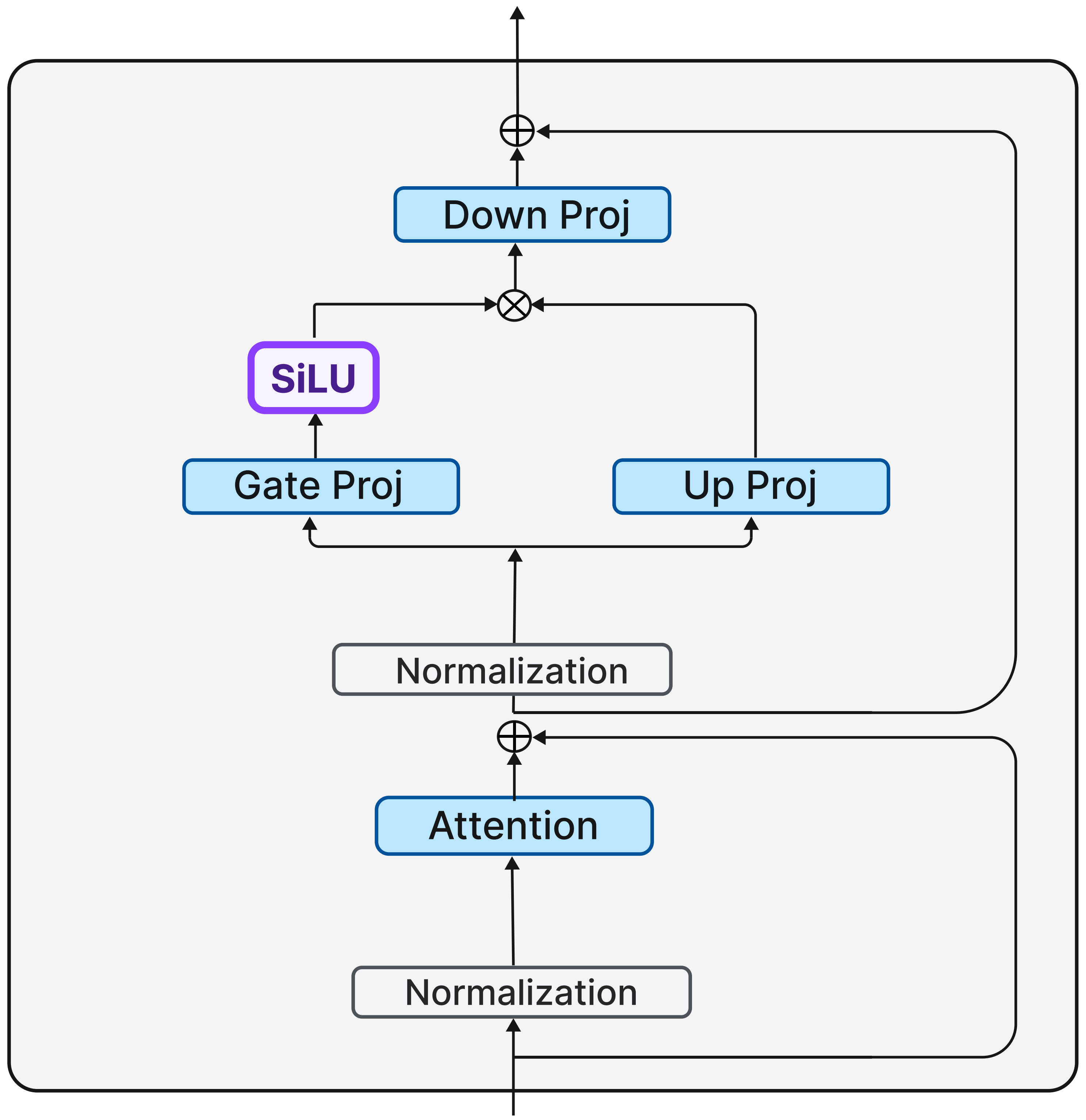


no relufication

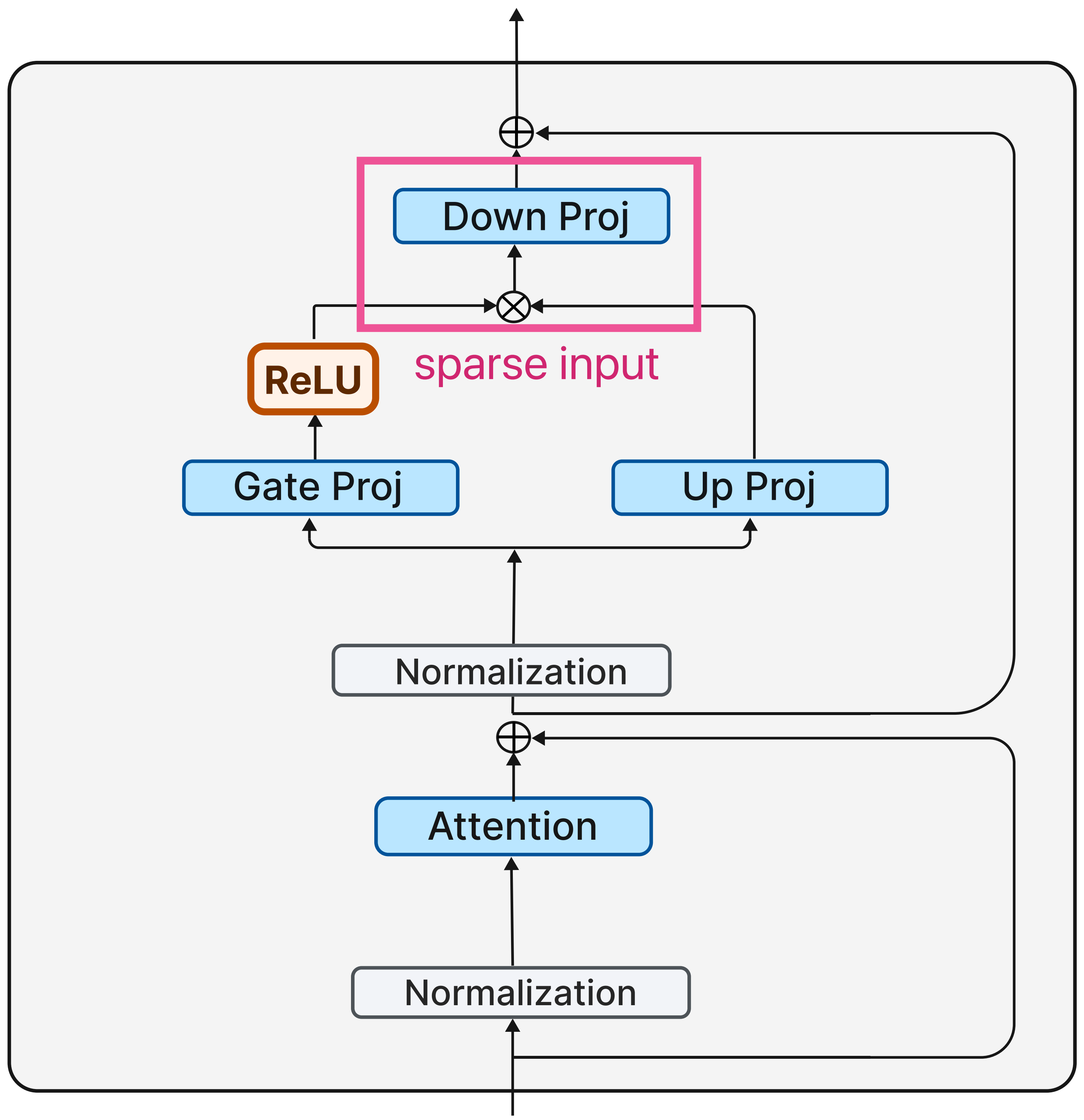


relufication - stage 1

ReLUfication Stages

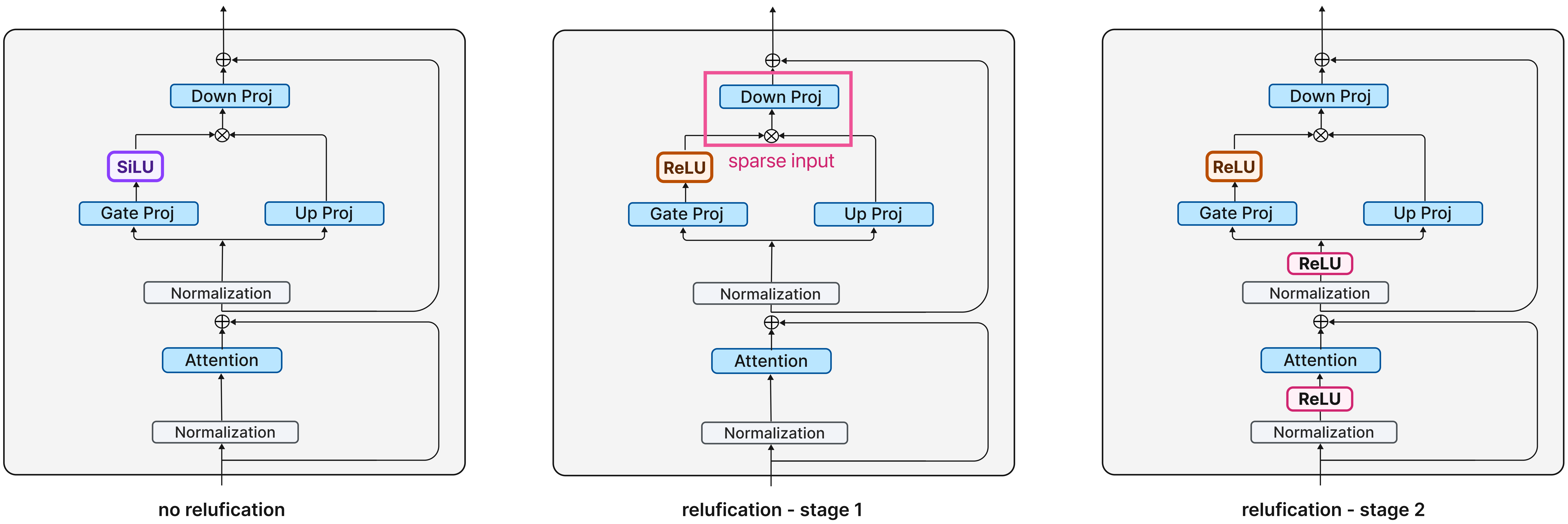


no relufication

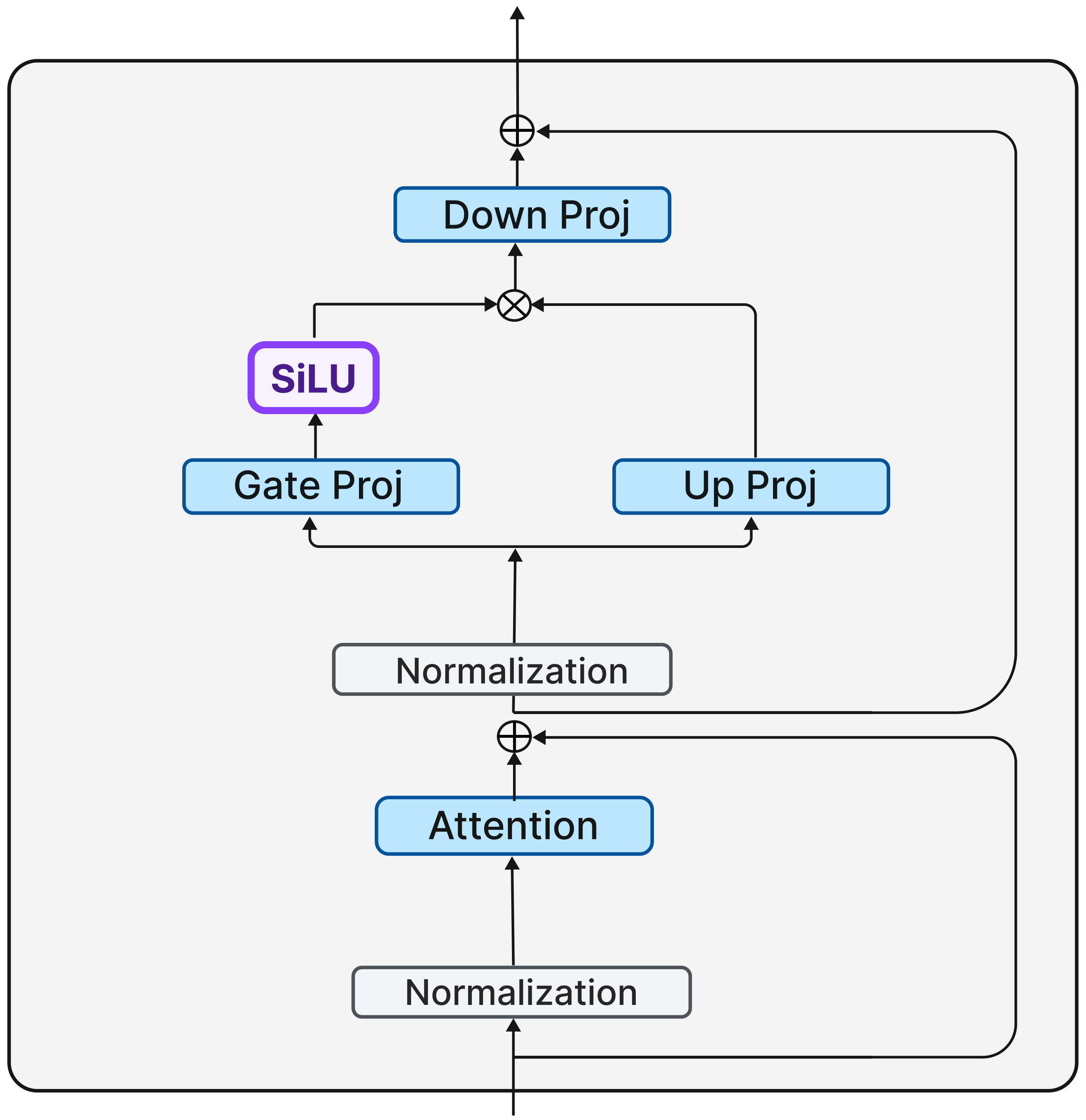


relufication - stage 1

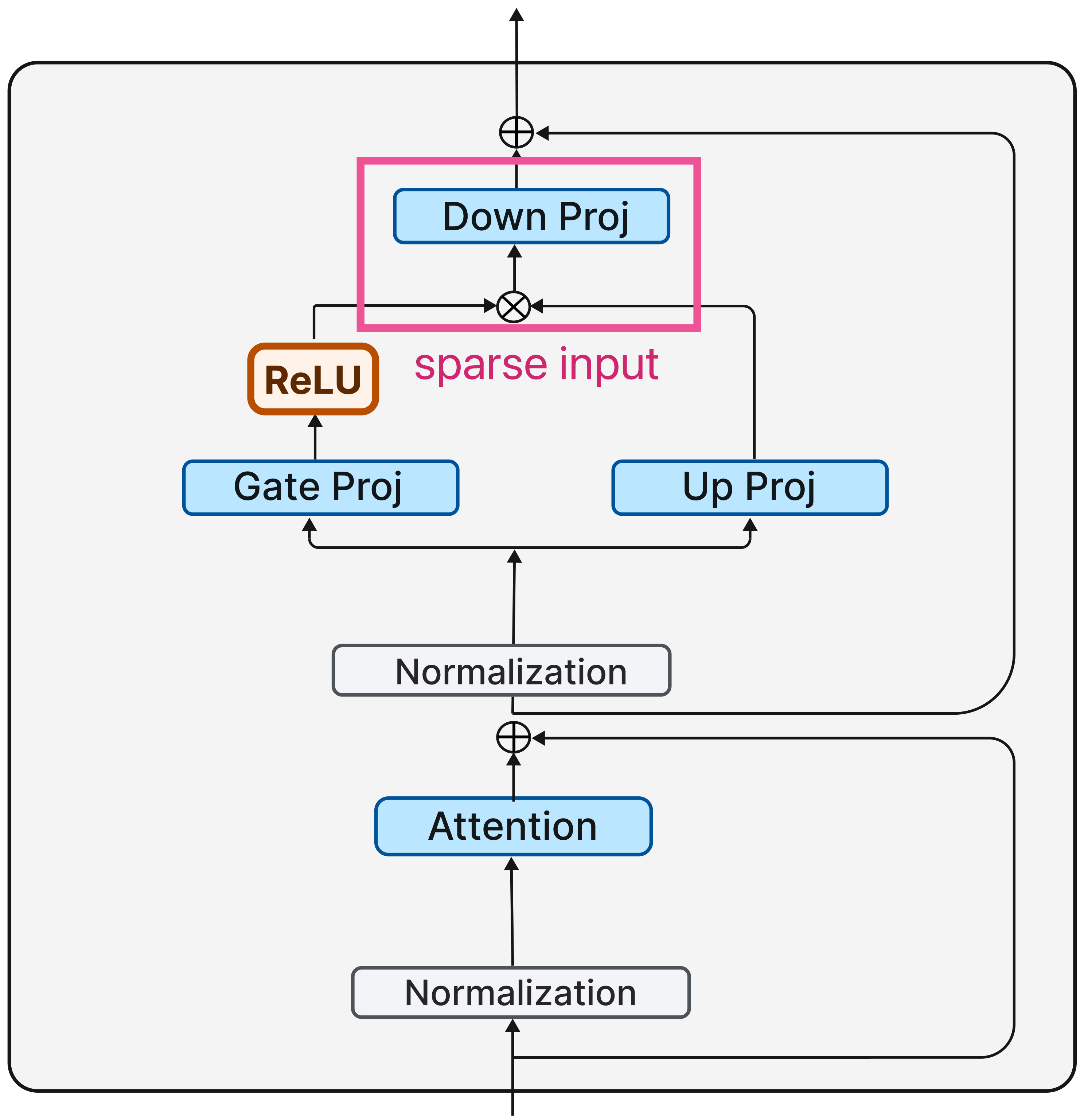
ReLUfication Stages



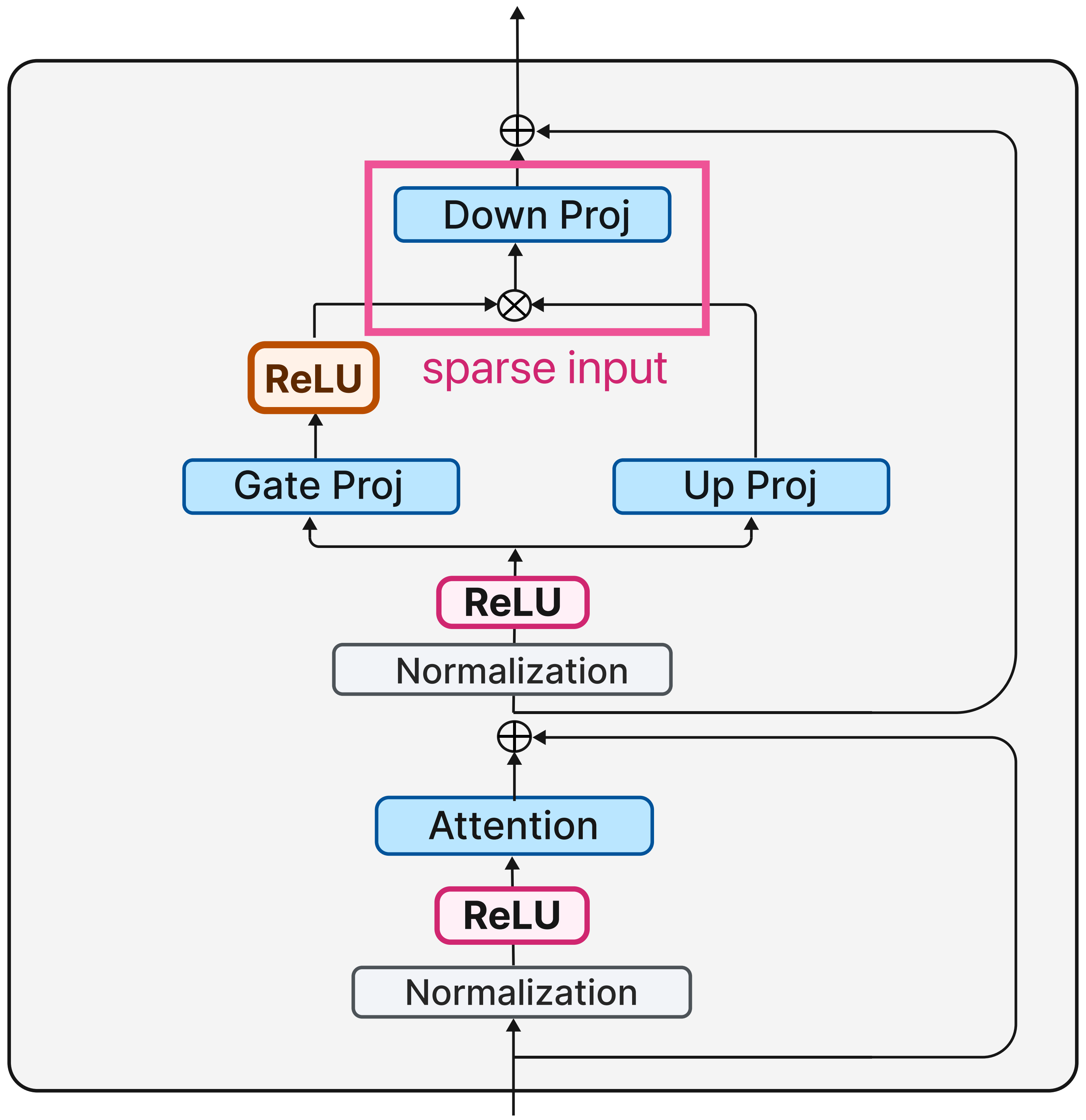
ReLUfication Stages



no relufication

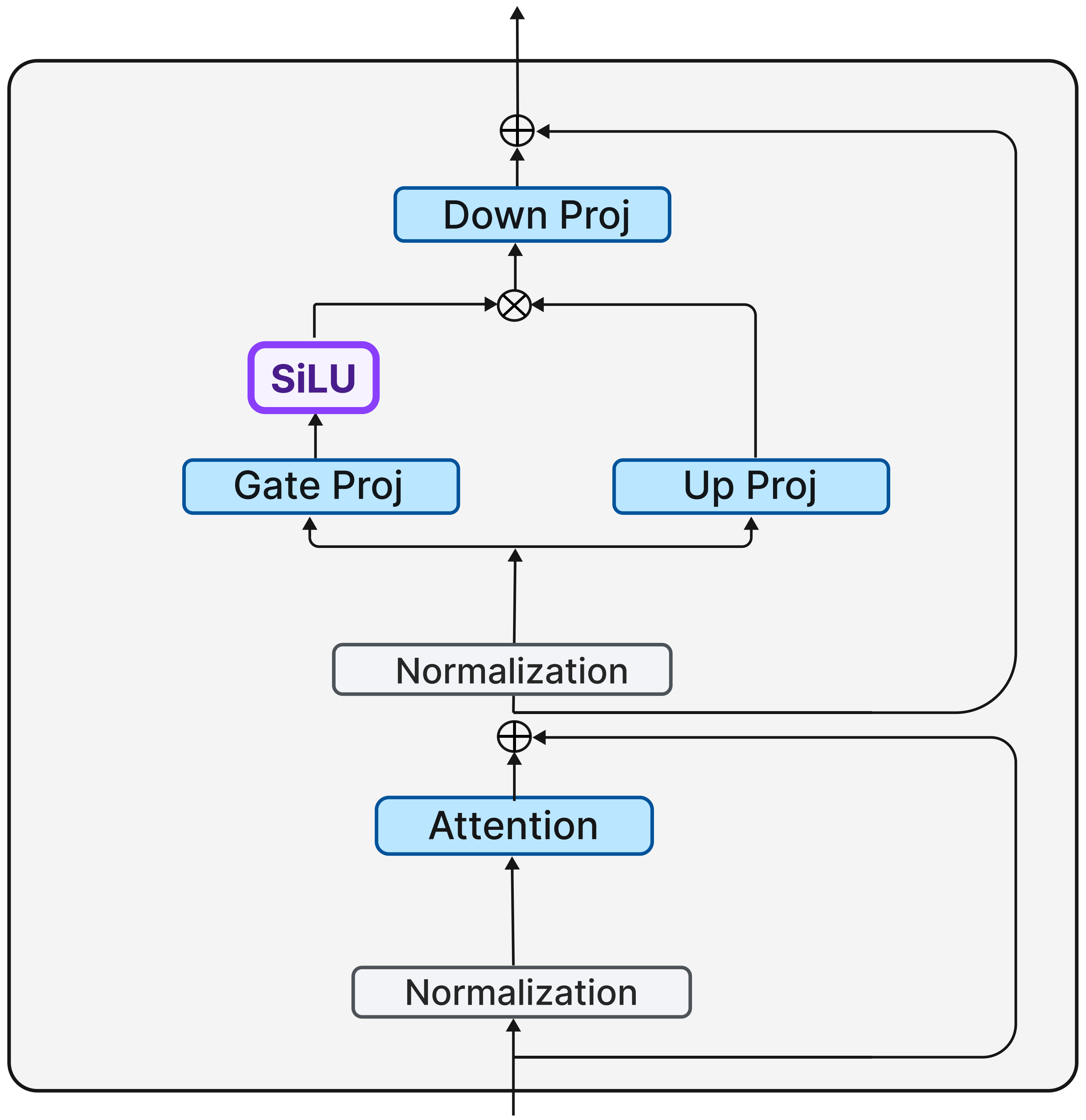


relufication - stage 1

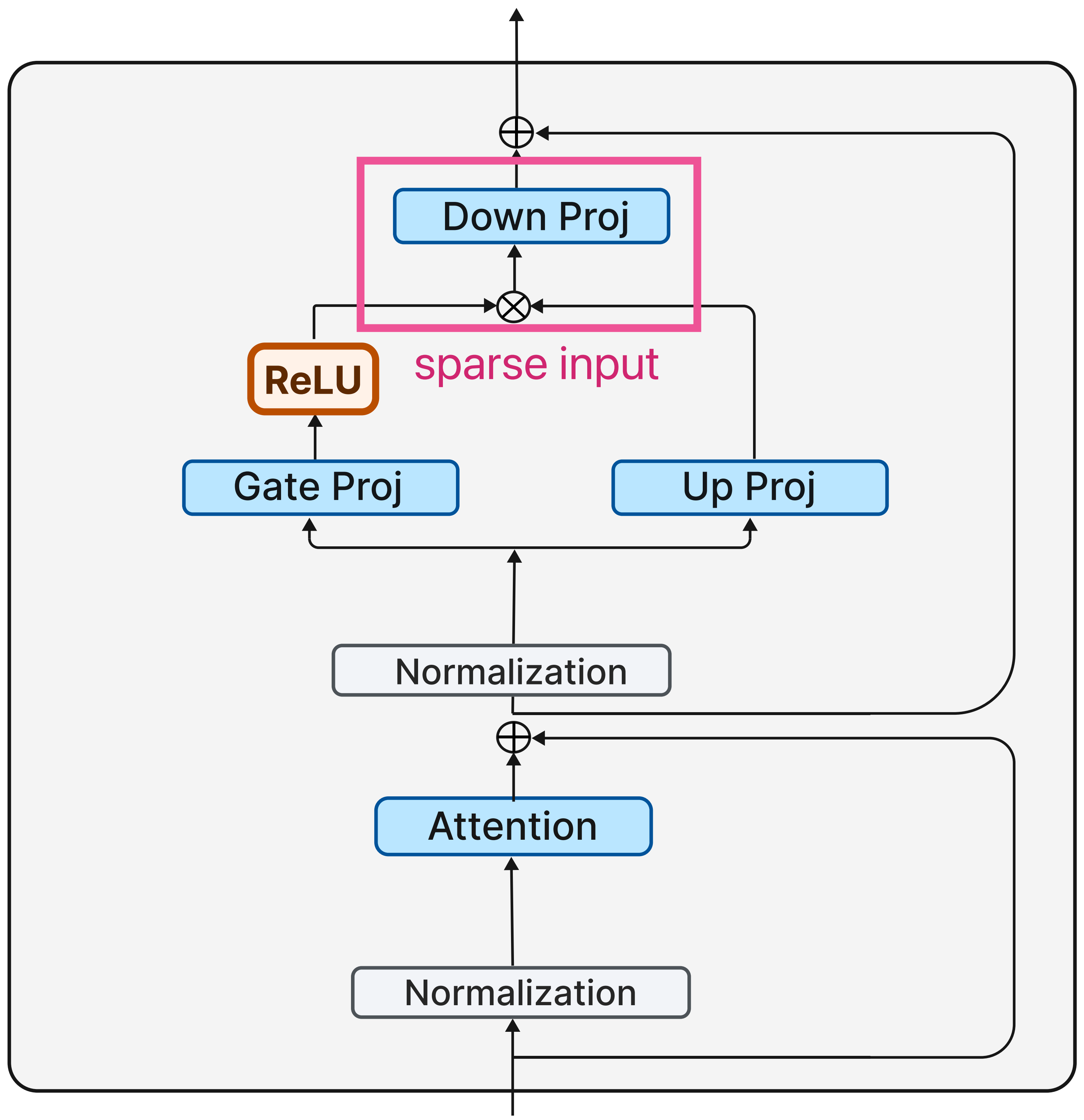


relufication - stage 2

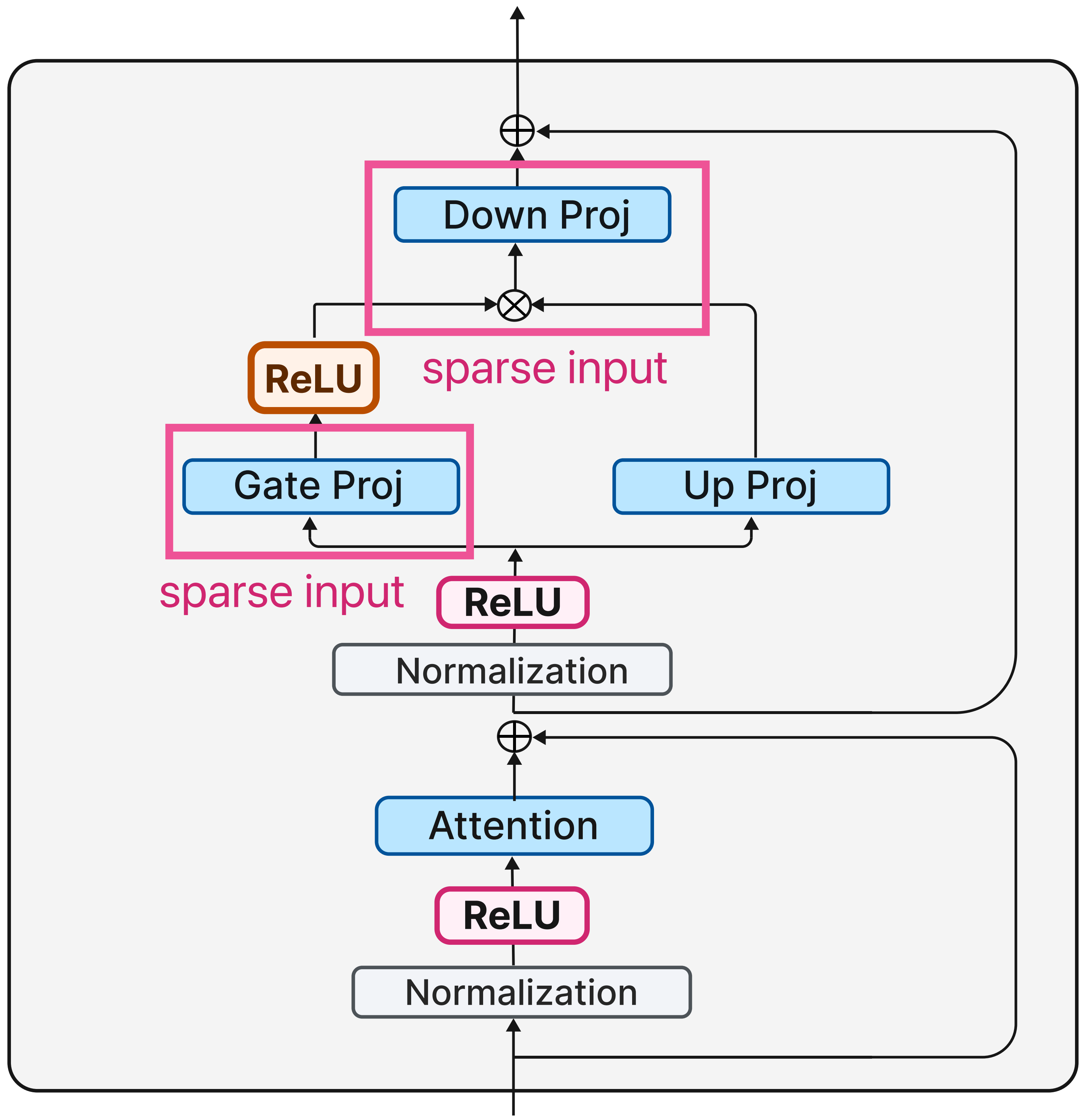
ReLUfication Stages



no relufication

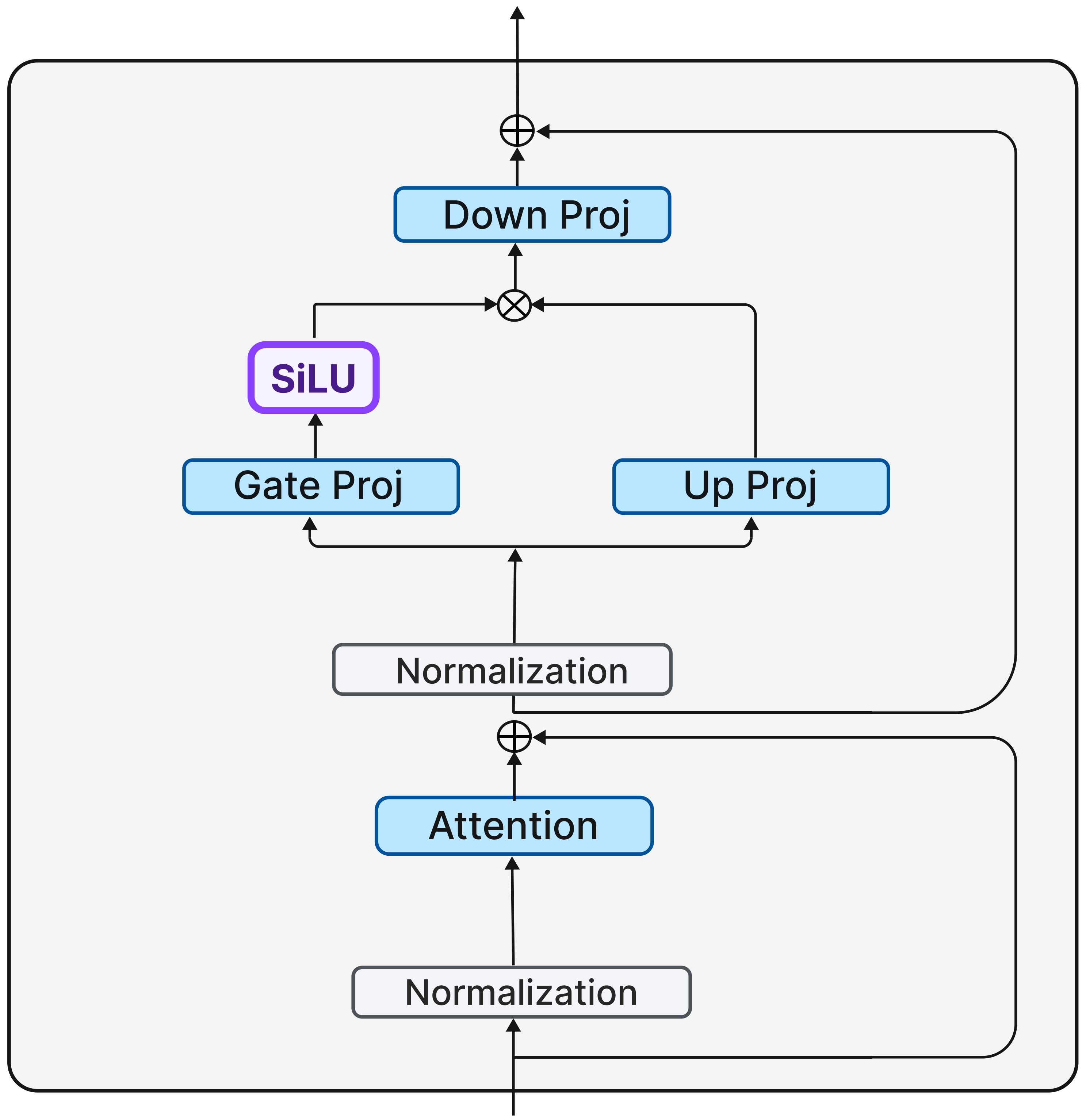


relufication - stage 1

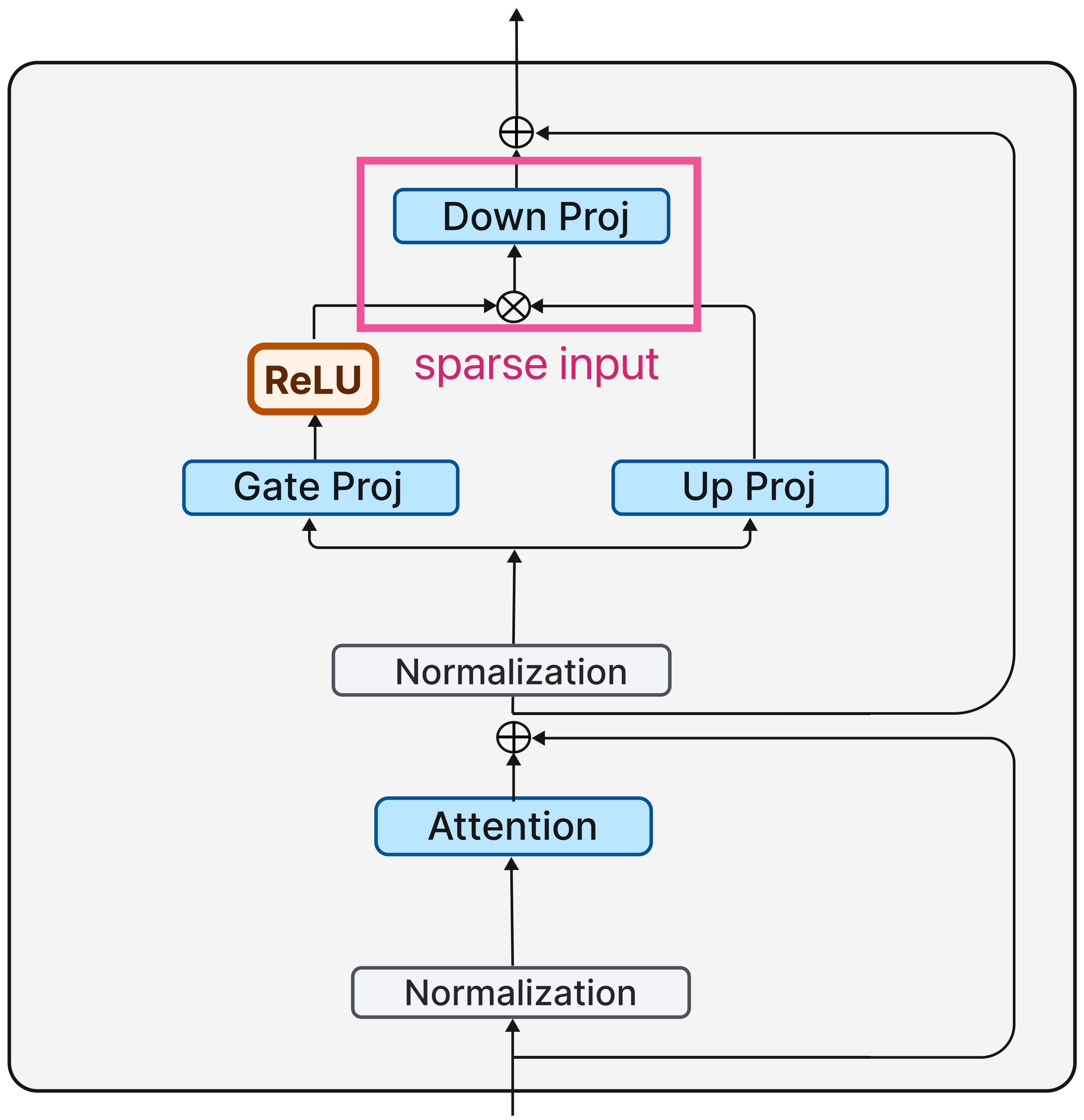


relufication - stage 2

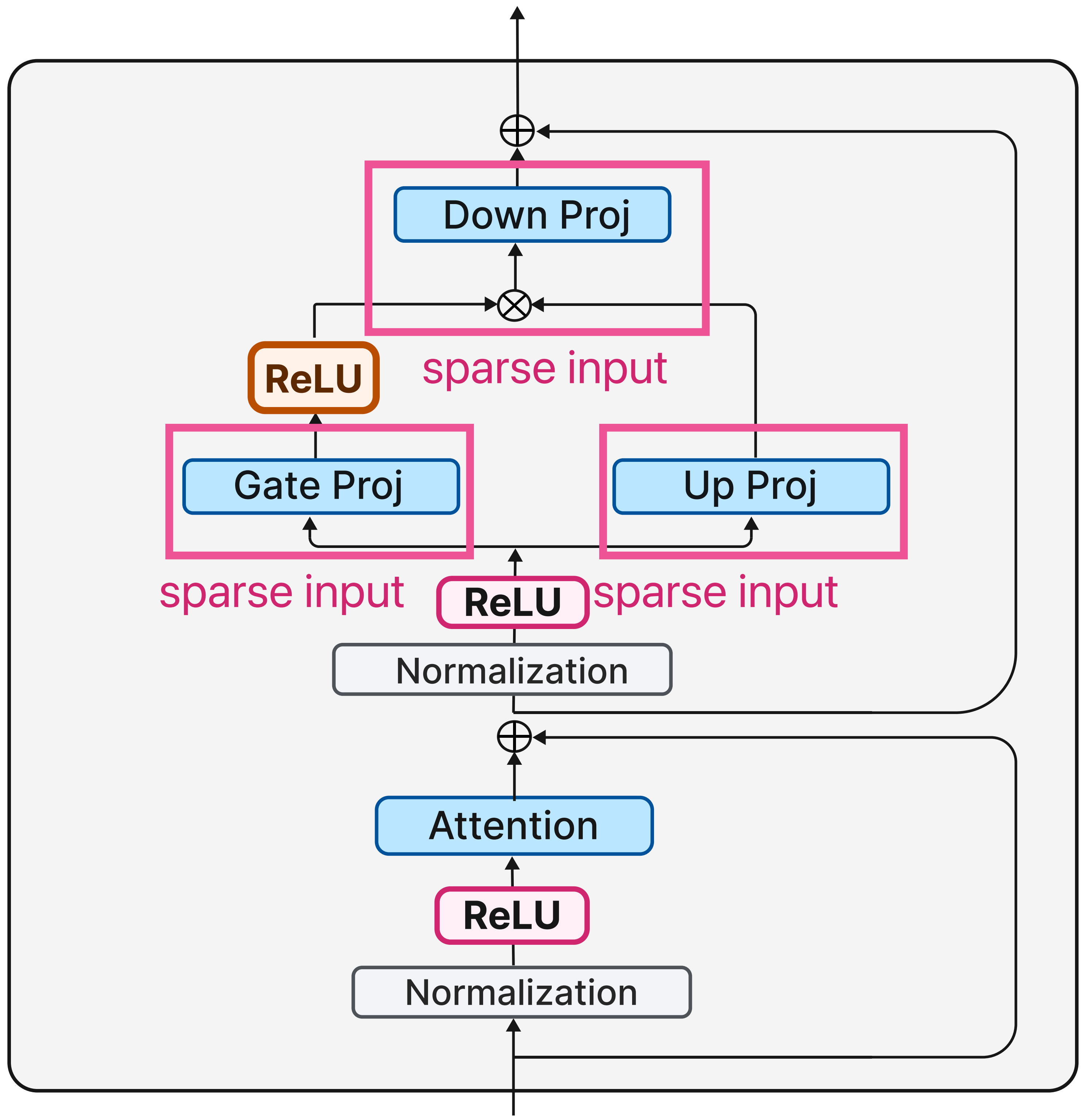
ReLUfication Stages



no relufication

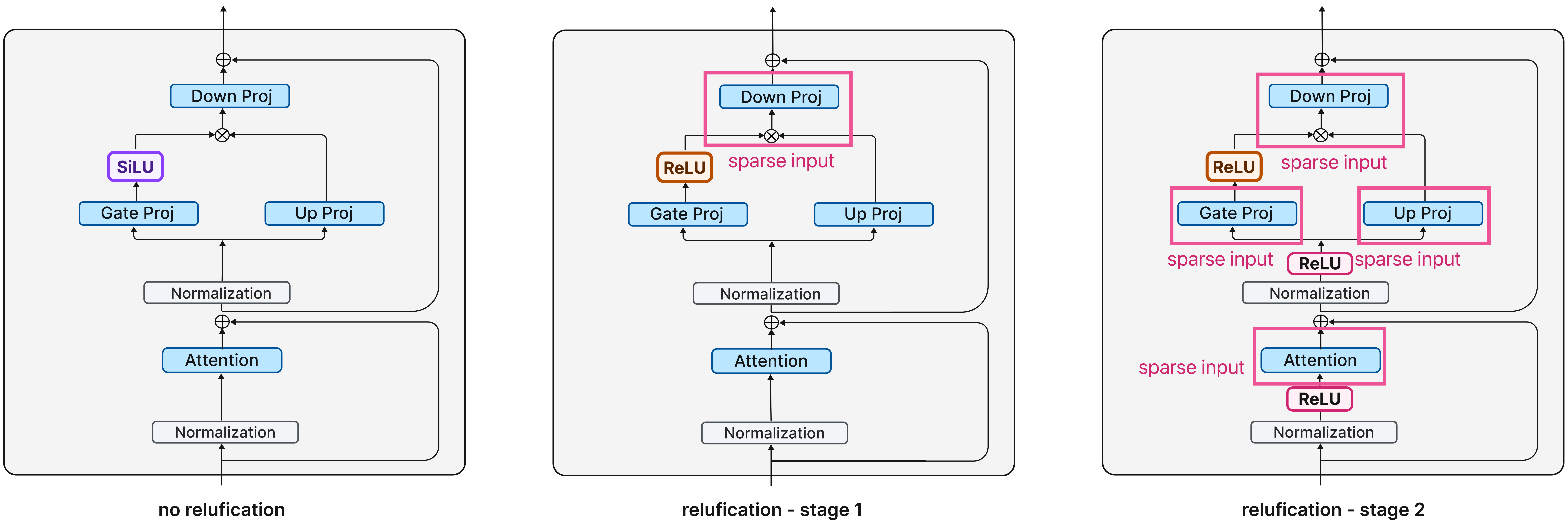


relufication - stage 1



relufication - stage 2

ReLUfication Stages

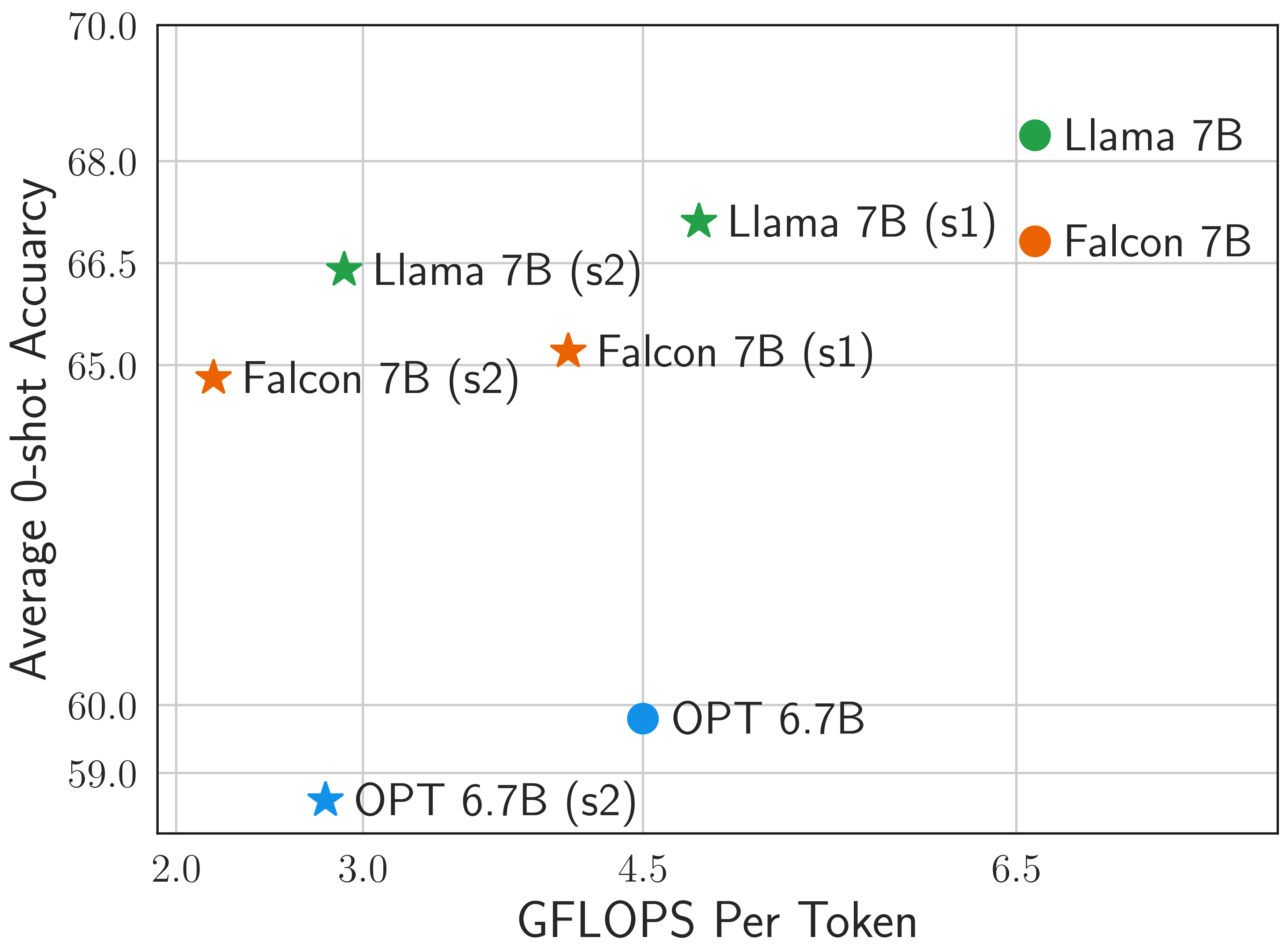


ReLUfication: Results

Model	Input Sparsity (%)			FLOPS (G)	Avg 0-shot Acc %
	QKV	UpProj	DownProj		
Llama 7B	0	0	0	6.6	68.4
Llama 7B (relufied-s1)	0	0	62	4.8	67.1
Llama 7B (relufied-s2)	51	67	65	2.9	66.4
Falcon 7B	0	1	0	6.6	66.8
Falcon 7B (relufied-s1)	0	0	94	4.1	65.2
Falcon 7B (relufied-s2)	56	56	95	2.2	64.8
OPT 6.7B	0	0	97	4.5	59.8
OPT 6.7B (relufied-s2)	50	40	97	2.8	58.6

ReLUfication: Results

Model	Input Sparsity (%)			FLOPS (G)	Avg 0-shot Acc %
	QKV	UpProj	DownProj		
Llama 7B	0	0	0	6.6	68.4
Llama 7B (relufied-s1)	0	0	62	4.8	67.1
Llama 7B (relufied-s2)	51	67	65	2.9	66.4
Falcon 7B	0	1	0	6.6	66.8
Falcon 7B (relufied-s1)	0	0	94	4.1	65.2
Falcon 7B (relufied-s2)	56	56	95	2.2	64.8
OPT 6.7B	0	0	97	4.5	59.8
OPT 6.7B (relufied-s2)	50	40	97	2.8	58.6



Why aren’t we training our LLMs with ReLU?

The Impact of Activation Function on Performance

When trained from scratch...

The Impact of Activation Function on Performance

When trained from scratch…

$$\mathrm{act}(x) = x\sigma(\beta x)$$

The Impact of Activation Function on Performance

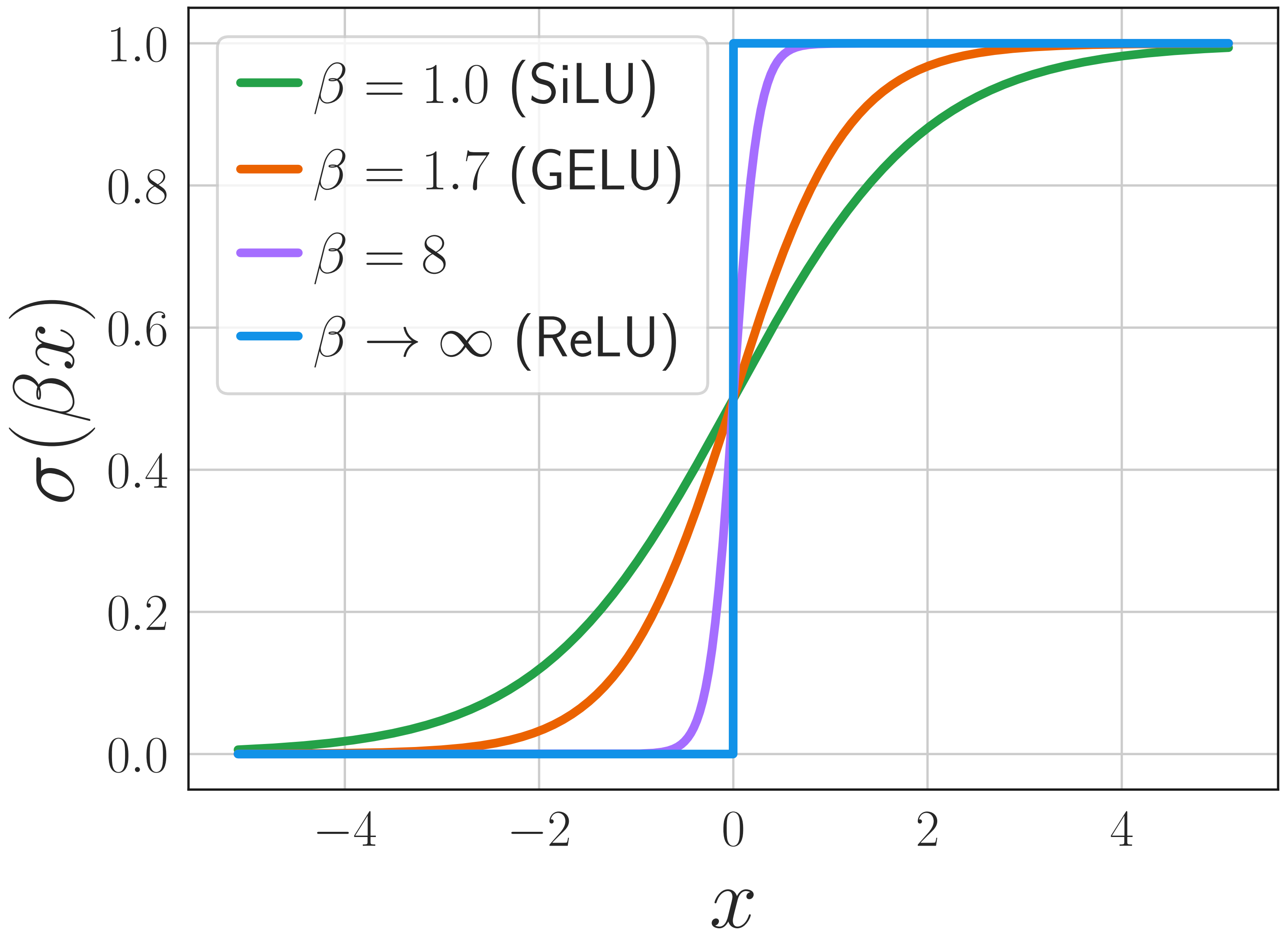
When trained from scratch...

$$\text{act}(x) = x\sigma(\beta x)$$
$$\text{SiLU}(x) = x\sigma(x)$$
$$\text{GELU}(x) \approx x\sigma(1.7x)$$

The Impact of Activation Function on Performance

When trained from scratch...

Activation Function

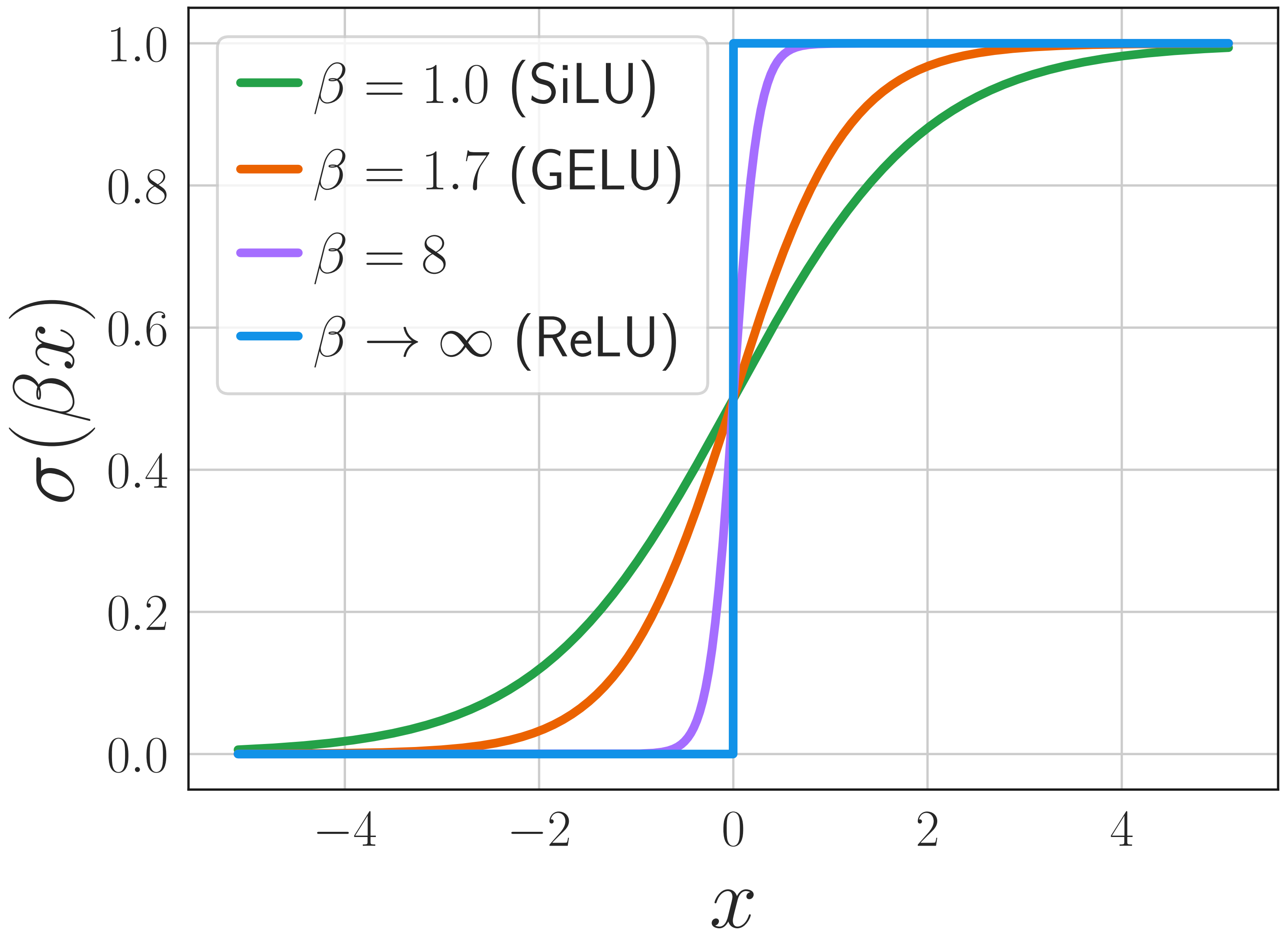


$$\mathrm{act}(x) = x\sigma(\beta x)$$
$$\mathrm{SiLU}(x) = x\sigma(x)$$
$$\mathrm{GELU}(x) \approx x\sigma(1.7x)$$

The Impact of Activation Function on Performance

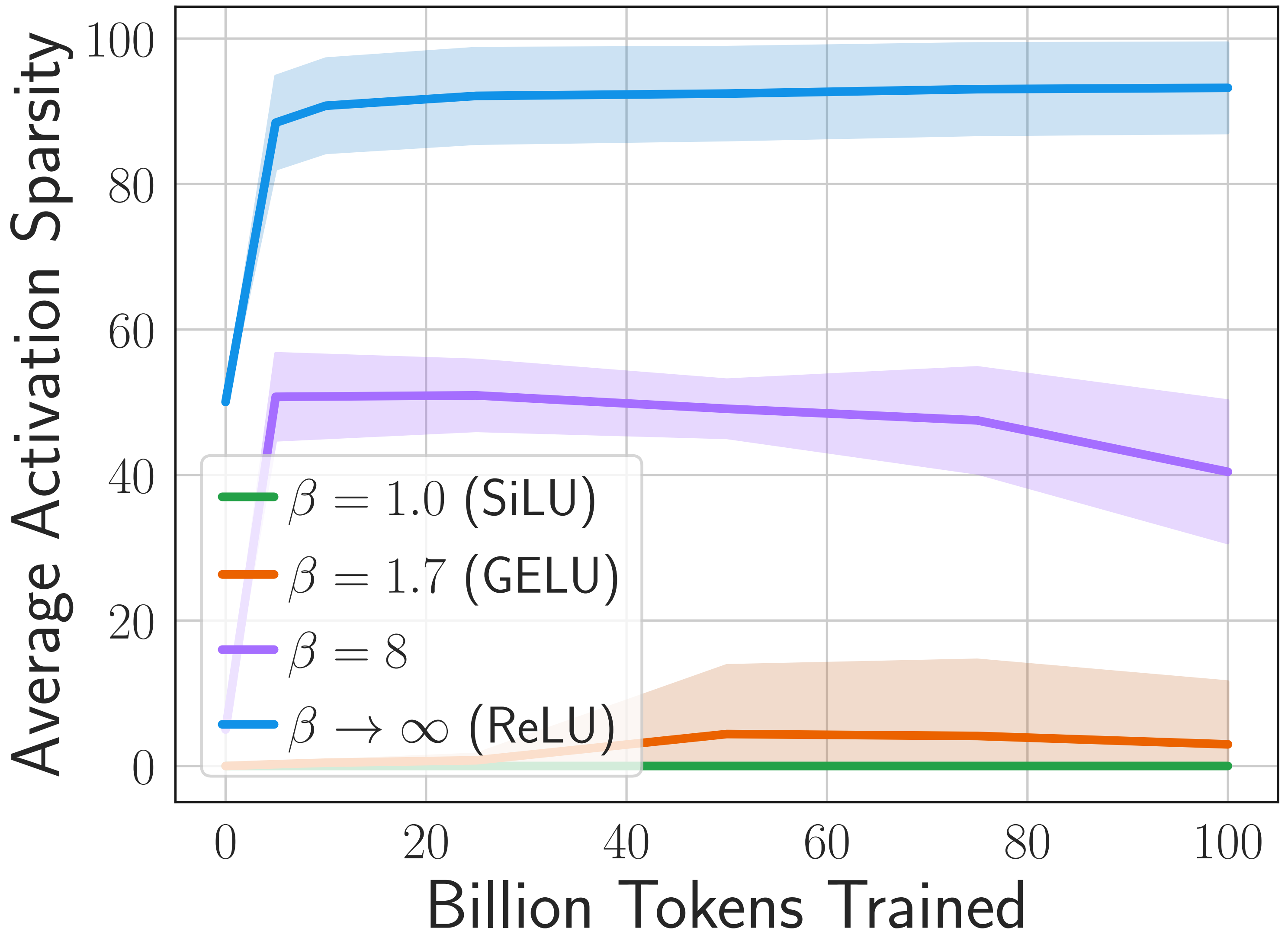
When trained from scratch...

Activation Function



$$\mathrm{act}(x) = x\sigma(\beta x)$$
$$\mathrm{SiLU}(x) = x\sigma(x)$$
$$\mathrm{GELU}(x) \approx x\sigma(1.7x)$$

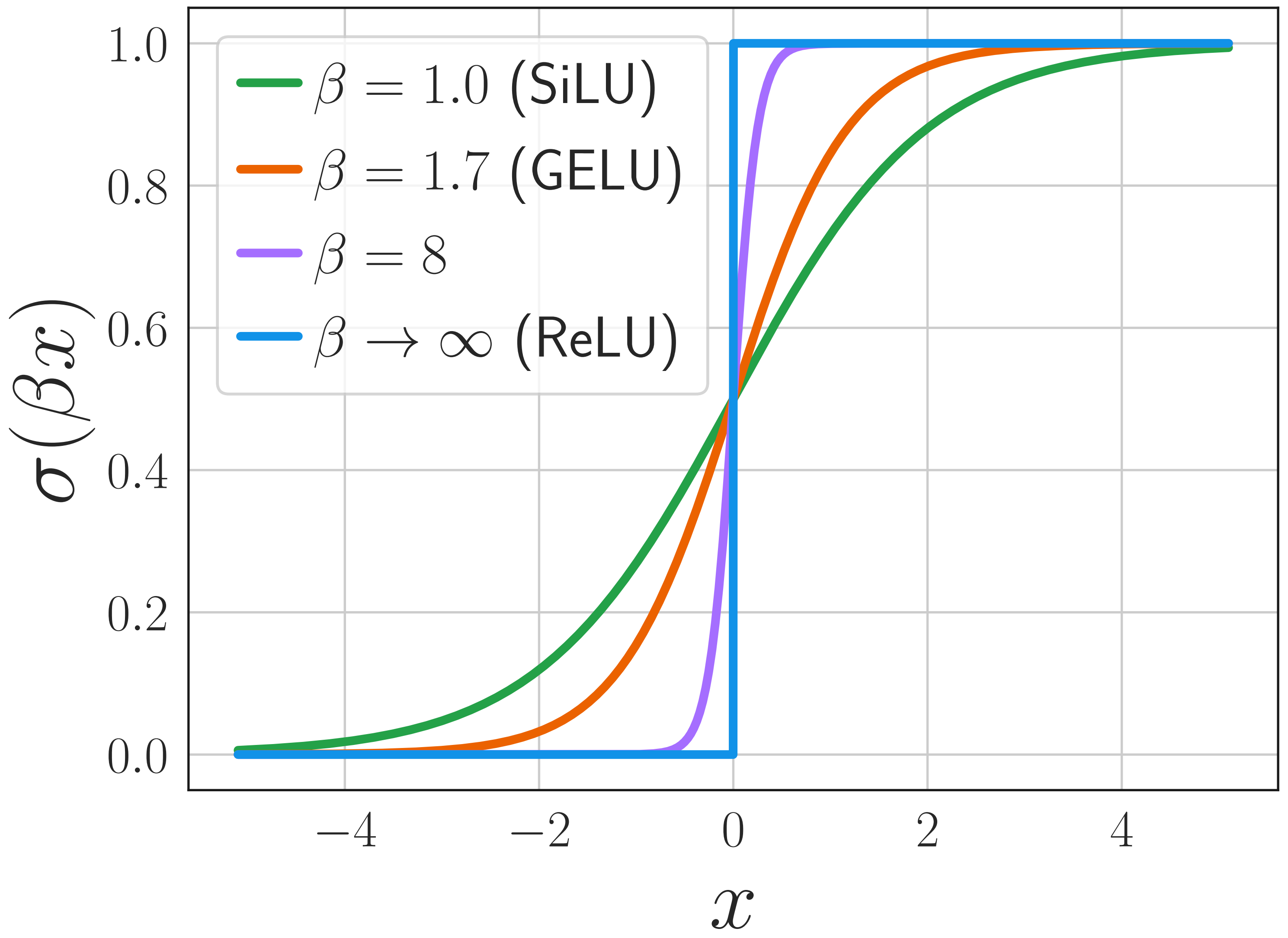
Activation Sparsity



The Impact of Activation Function on Performance

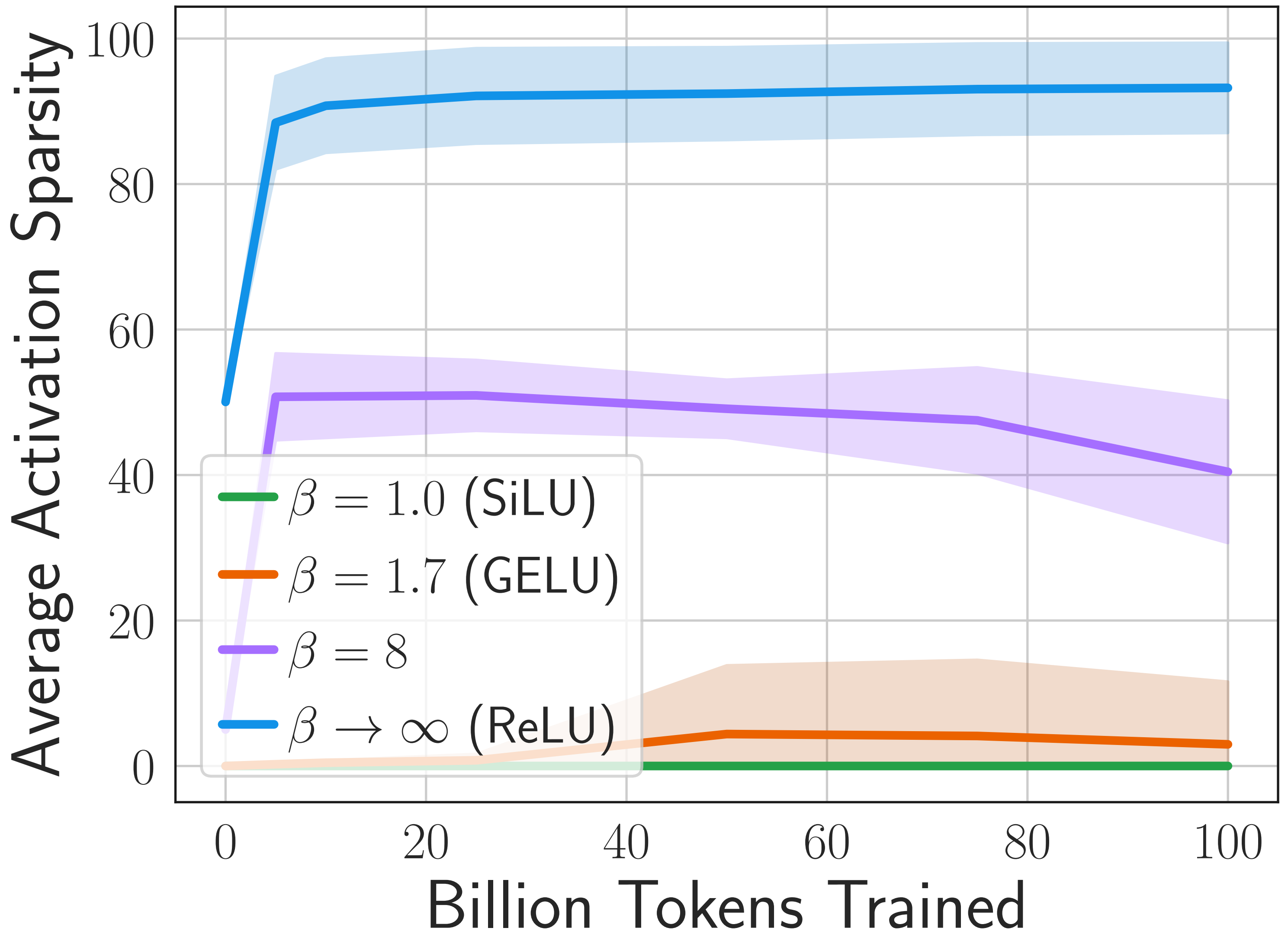
When trained from scratch...

Activation Function

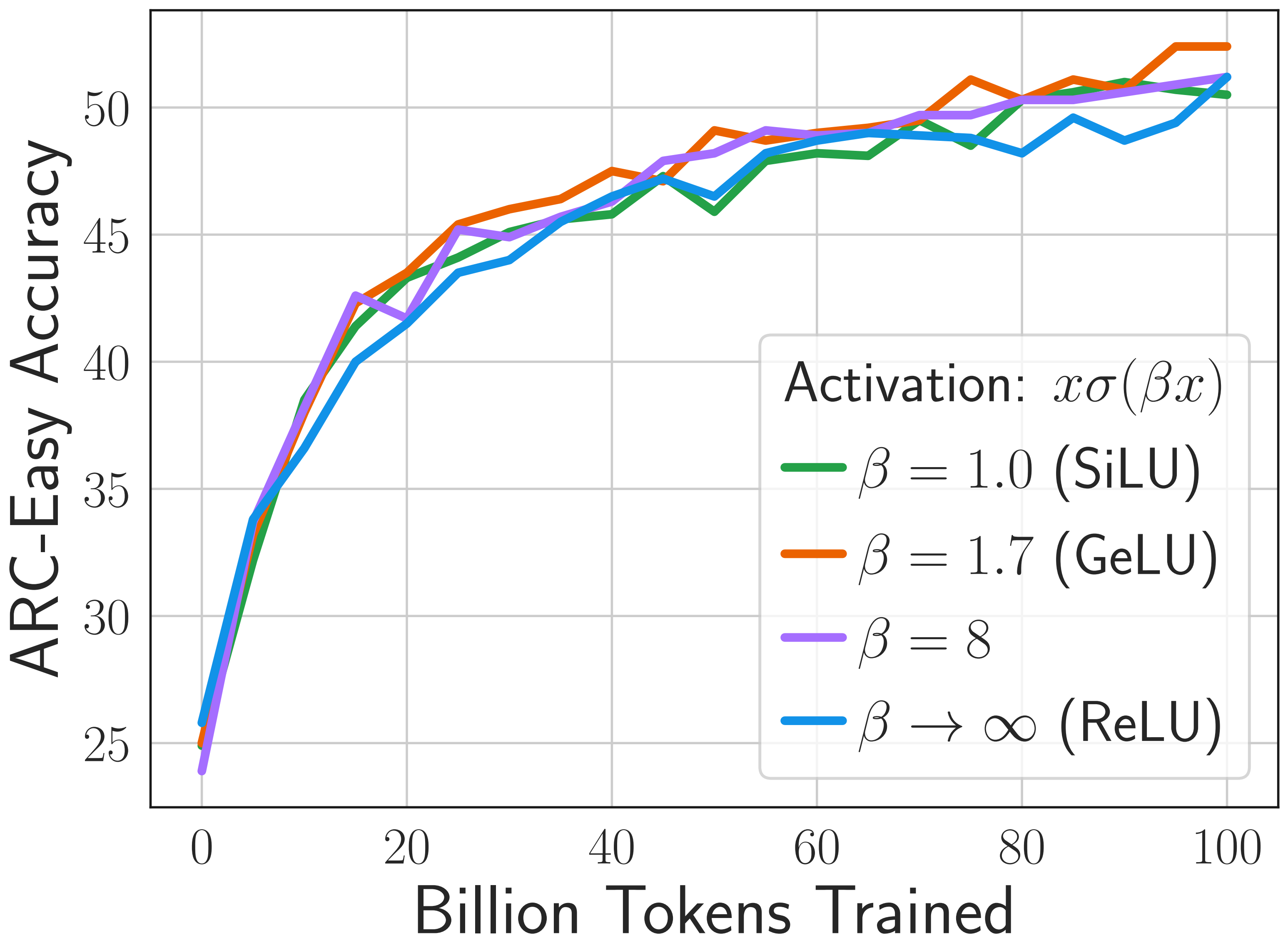


$$\mathrm{act}(x) = x\sigma(\beta x)$$
$$\mathrm{SiLU}(x) = x\sigma(x)$$
$$\mathrm{GELU}(x) \approx x\sigma(1.7x)$$

Activation Sparsity



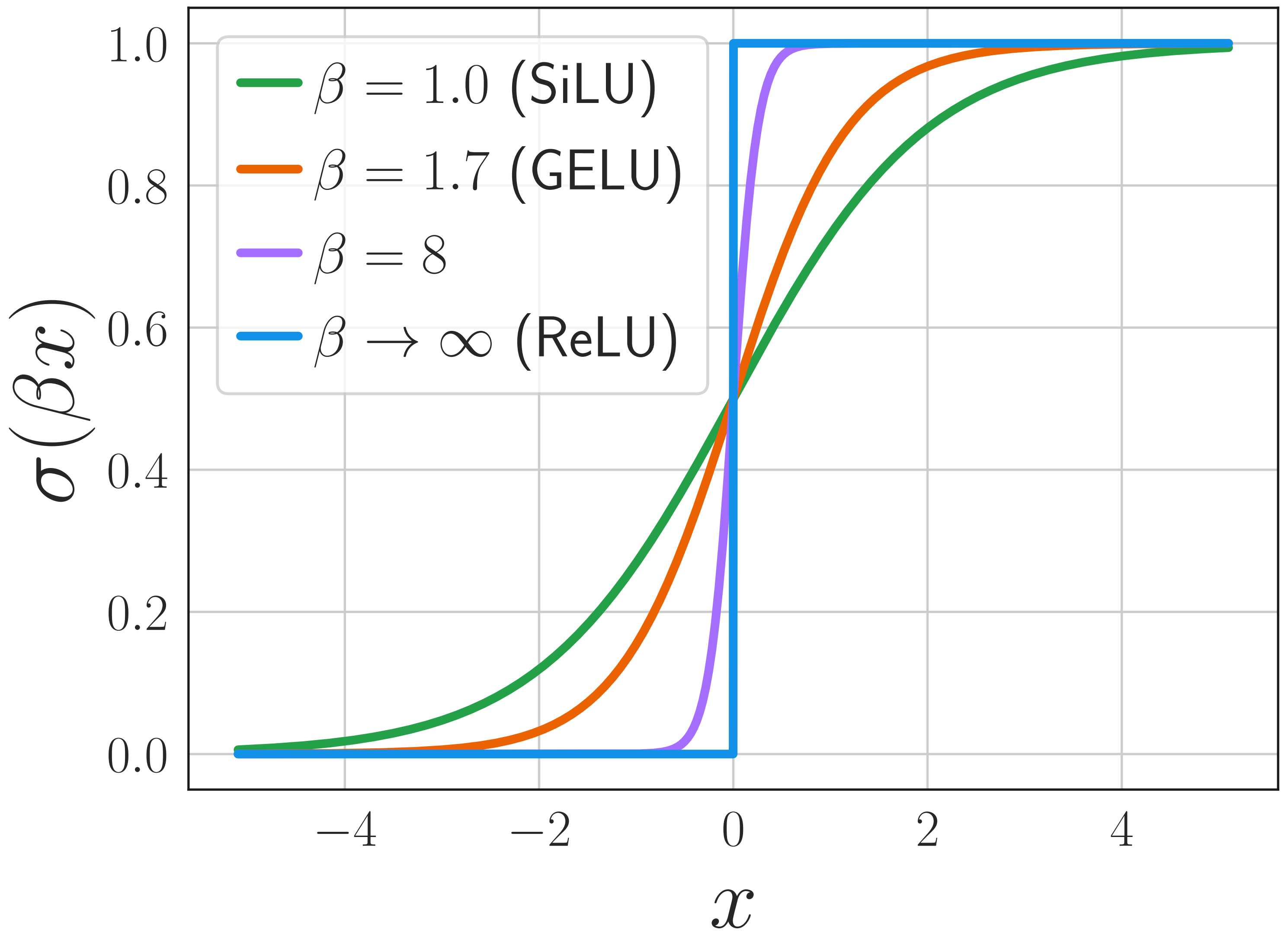
Performance



The Impact of Activation Function on Performance

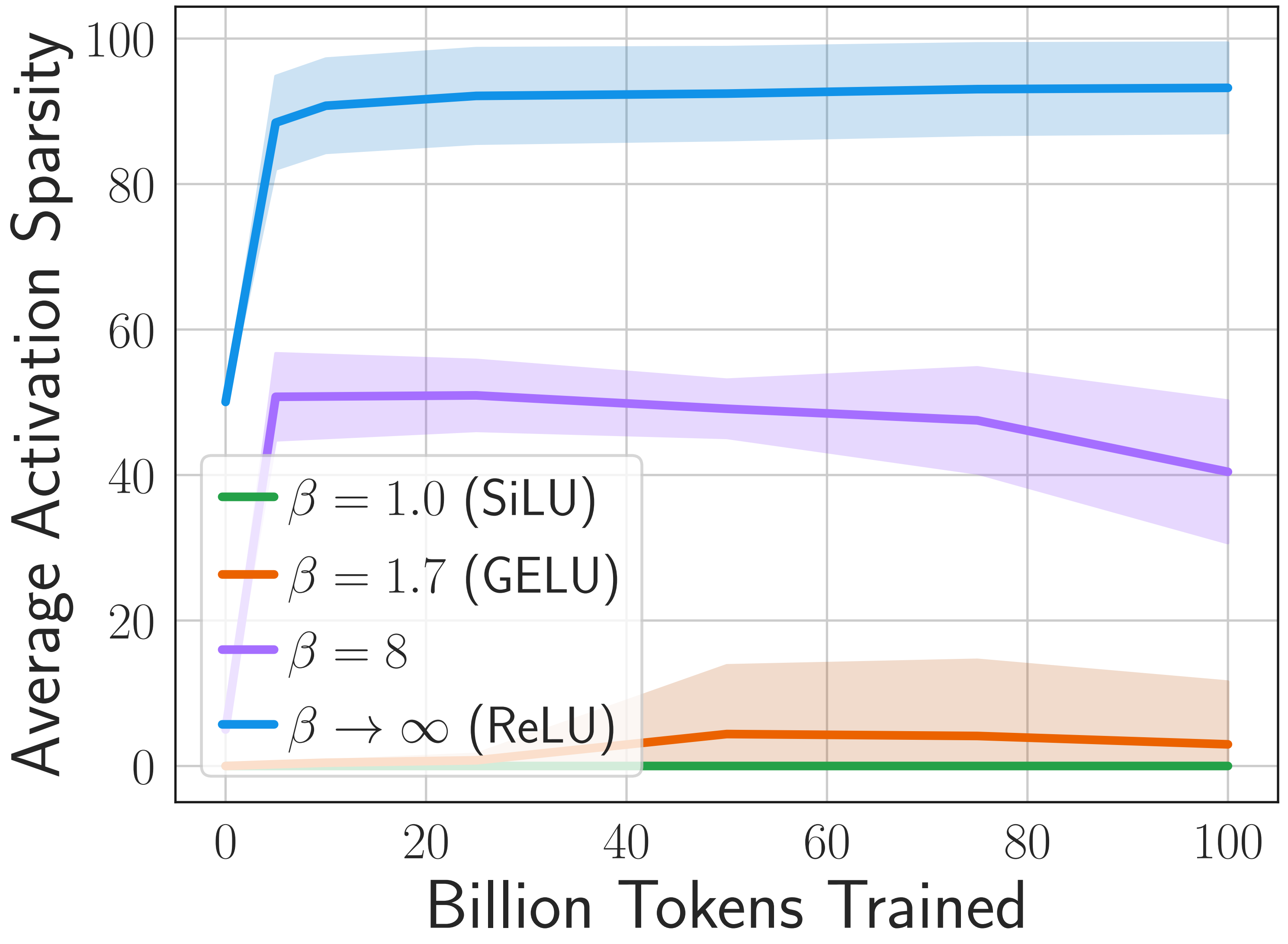
When trained from scratch...

Activation Function

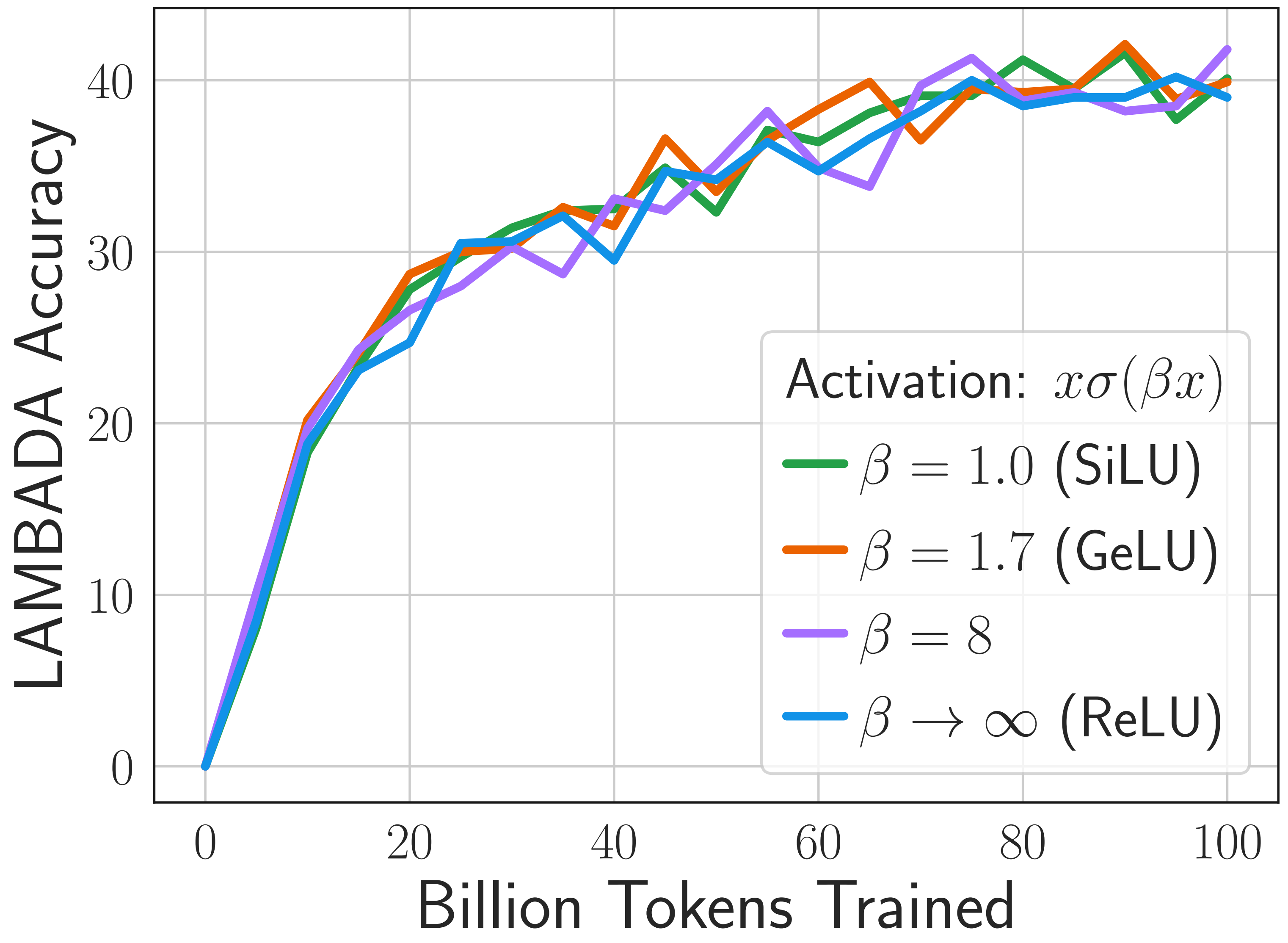


$$\mathrm{act}(x) = x\sigma(\beta x)$$
$$\mathrm{SiLU}(x) = x\sigma(x)$$
$$\mathrm{GELU}(x) \approx x\sigma(1.7x)$$

Activation Sparsity



Performance



Concluding Remarks

Recap

- ReLU activations are sparse. We can use this property for faster inference.
- The majority of LLMs are trained without ReLU.
- We can change their activation function to ReLU, and fine-tune them for a short time (ReLUfication).
- But do these alternative activation functions (e.g., GELU, SwiGLU) really improve performance?
  Maybe we should reconsider the choice of activation function.

Future Directions: Multiple Tokens

Future Directions: Multiple Tokens

Limitation of our work:
- The assumption that bsz = 1,
- We are generating one token at a time.

Future Directions: Multiple Tokens

Limitation of our work:
- The assumption that bsz = 1,
- We are generating one token at a time.

It is difficult to exploit activation sparsity when we have beam search, speculative decoding, batched inference

Future Directions: Multiple Tokens

Limitation of our work:
- The assumption that bsz = 1,
- We are generating one token at a time.

It is difficult to exploit activation sparsity when we have beam search, speculative decoding, batched inference

Aggregated Sparsity
How much of total available neurons have **not been used** for the first **t** tokens?

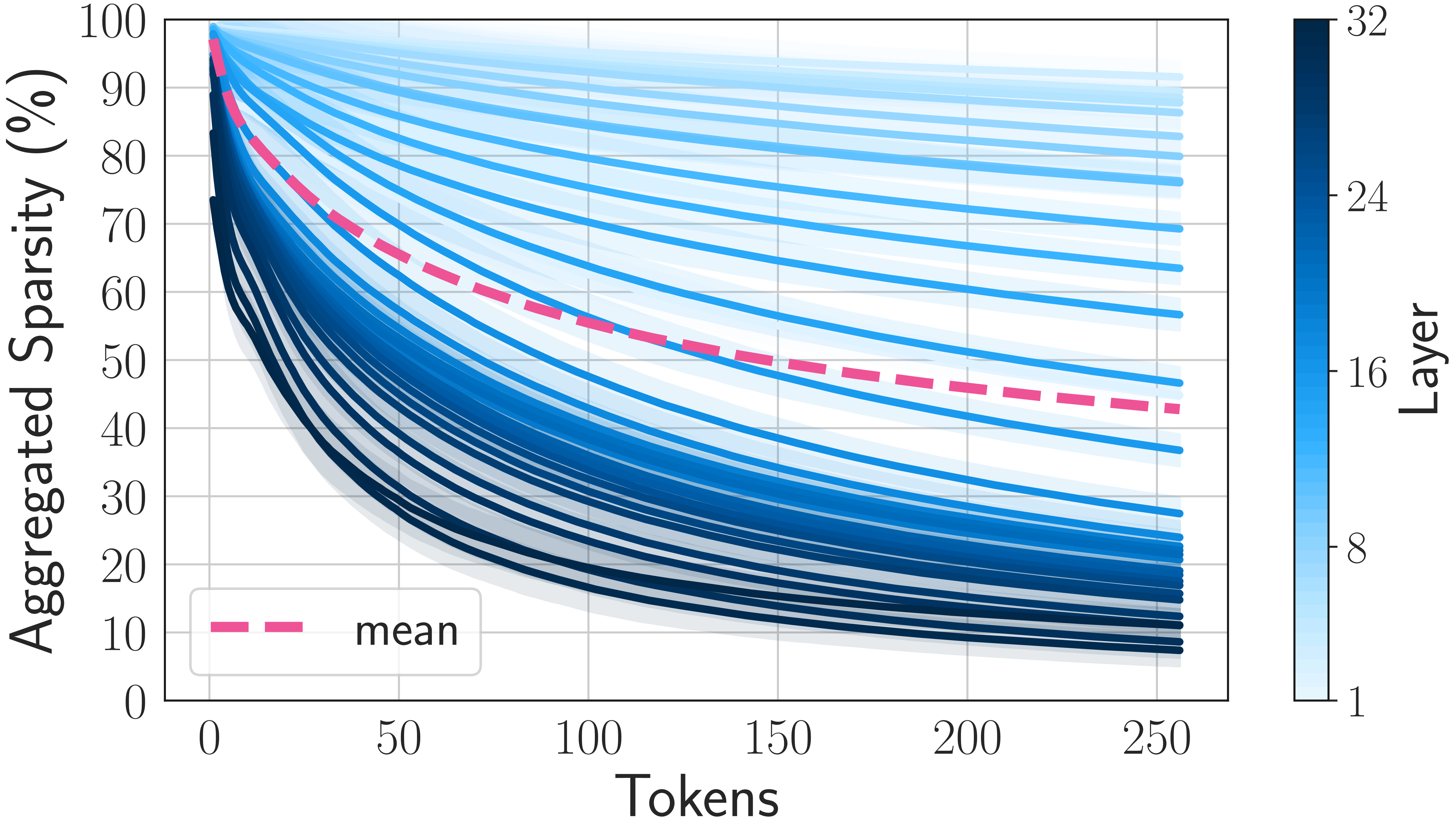
Future Directions: Multiple Tokens

Limitation of our work:
- The assumption that bsz = 1,
- We are generating one token at a time.

It is difficult to exploit activation sparsity when we have beam search, speculative decoding, batched inference

Aggregated Sparsity

How much of total available neurons have **not been used** for the first **t** tokens?



Future Directions: Improved Sparsity

For the GLU-based activation functions (e.g., SwiGLU), we notice lower activation sparsity (~60%) even after ReLUfication.

Tackling this issue:

[1] Song, Chenyang, et al. "ProSparse: Introducing and Enhancing Intrinsic Activation Sparsity within Large Language Models." arXiv preprint arXiv:2402.13516.

[2] Lee, Je-Yong, et al. "CATS: Contextually-Aware Thresholding for Sparsity in Large Language Models." arXiv:2404.08763.

Poster Session: Halle B #89