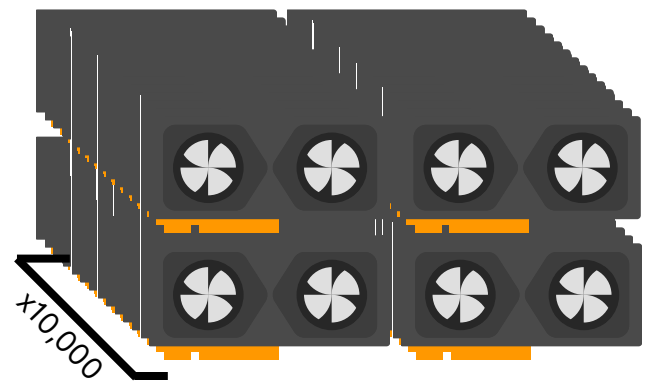
What are **Small Language Models** (SLMs)?

SLM
2B Params

LLM
700B Params

Cost: $10,000-500,000

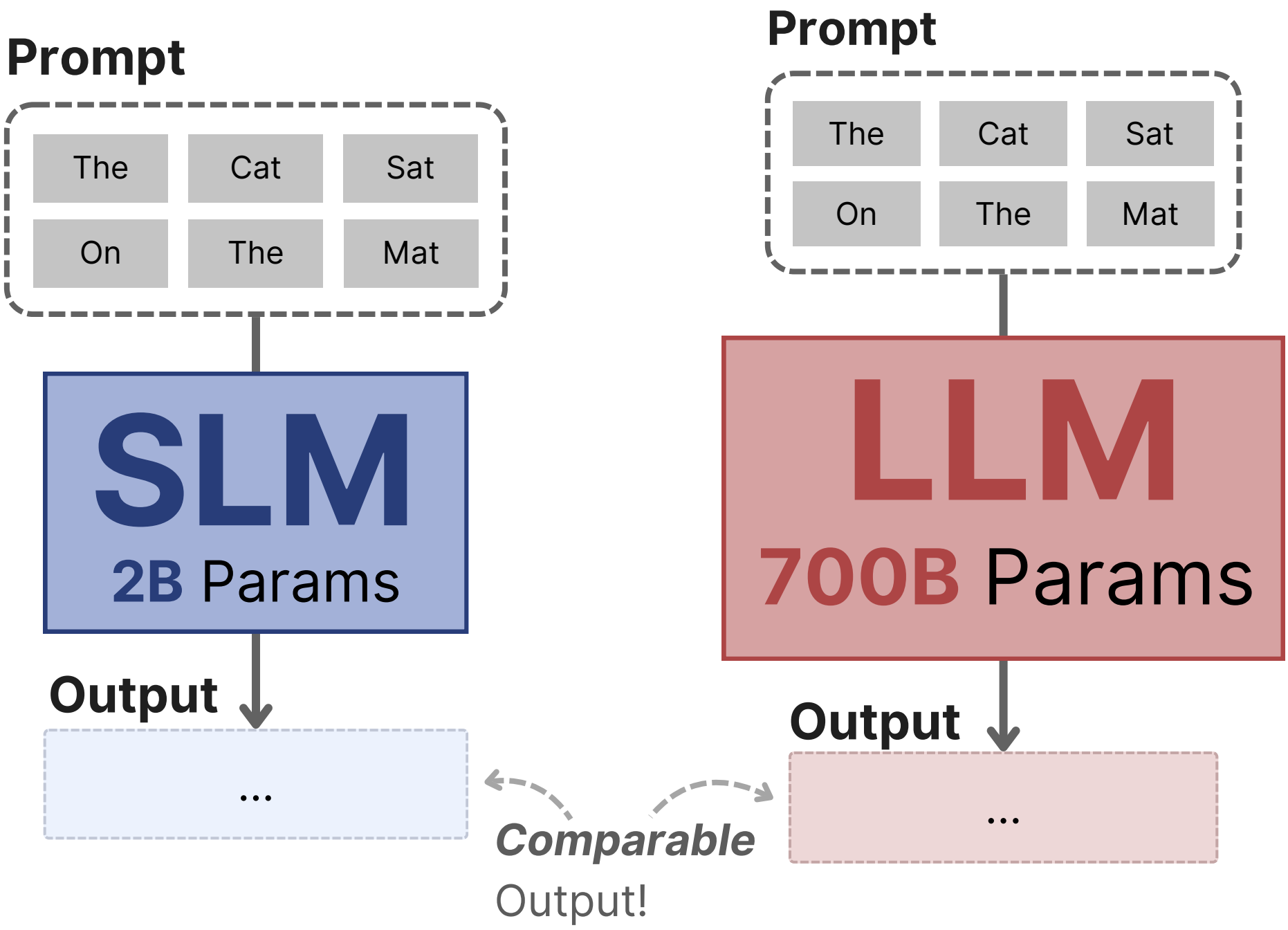
Cost: $10M - $500B



Swipe For More ⟶

SLMs are LLMs That are **Much Smaller** in Size

(< 10B parameters)

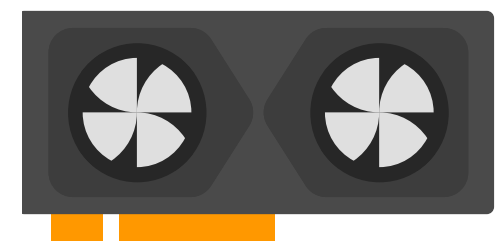


Although small, SLMs show *similar* performance on task-specific use cases

SLMs are **Particularly Useful**

SLM
2B Params

Locally Hostable



Cost: $10,000-500,000

Benefits:

- Locally Hostable
- Runs on Consumer-Grade Hardware

NVIDIA deemed SLMs were the **future of agentic AI**

But why not LLMs?

LLMs Are Good For **Broad Knowledge**

Why LLMs = **hundreds of billions** of parameters

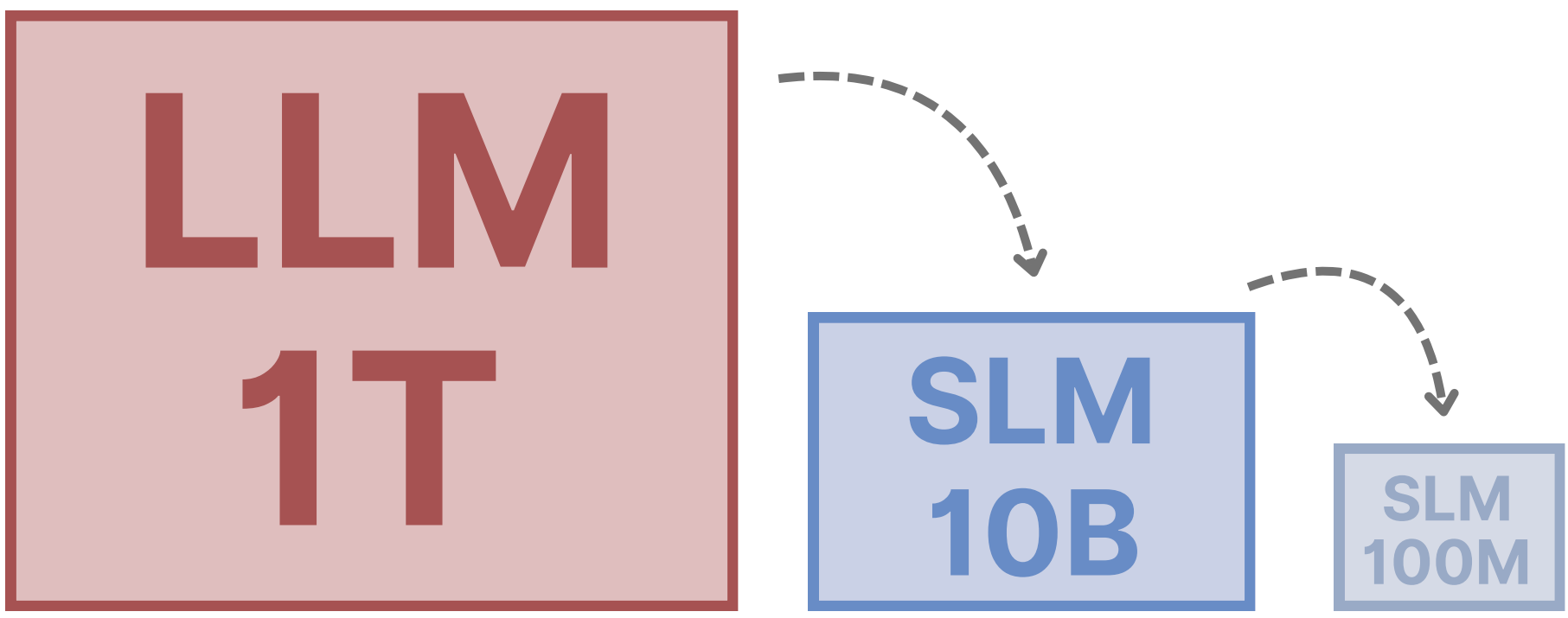
Domain ***Specific*** Tasks

Similar or **Better** Performance Than LLMs on Task

SLMs are good for **specializing** in specific tasks (e.g. *coding*)

How Did SLMs Get **So Small**?

They use **state-of-the-art** optimization techniques



Examples:

- Quantization
- LoRA Fine-Tuning
- FlashAttention

Quantization: **Reduce Precision** of Parameters

High-Precision (e.g., FP32)

Scalar

3.14

Vector

$$\begin{bmatrix} 3.14 \\ 2.72 \end{bmatrix}$$

Matrix

$$\begin{bmatrix} [3.14 & 2.72] \\ [1.62 & 1.41] \end{bmatrix}$$

Tensor

$$\begin{bmatrix} [3.14 & 2.72] \\ [1.62 & 1.41] \end{bmatrix} \begin{bmatrix} [4.5 & 6.2] \\ [8.8 & 7.1] \end{bmatrix}$$

Quantized (e.g., INT8)

Scalar

3

Vector

$$\begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

Matrix

$$\begin{bmatrix} [3 & 2] \\ [1 & 1] \end{bmatrix}$$

Tensor

$$\begin{bmatrix} [3 & 2] \\ [1 & 1] \end{bmatrix} \begin{bmatrix} [4 & 6] \\ [8 & 7] \end{bmatrix}$$

Stored in GPU as **parameters**

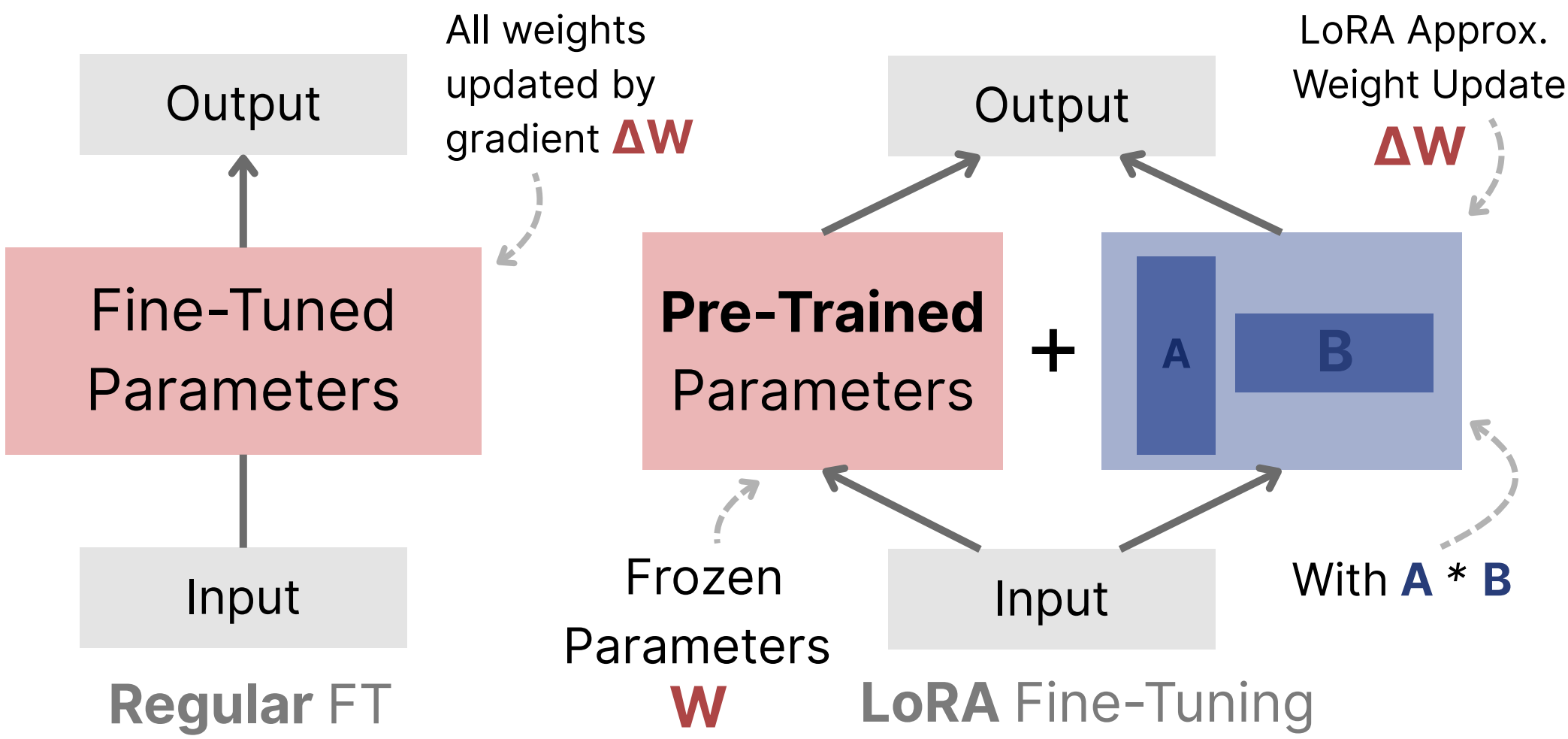
GPU

```
[[[0, 0.1, 0.2],[0.2, 0.4, 0.6]], [[-0.4, -0.3, 0.6], [0.7, 0.4, 0.3]]]
[[[0.5, 0.4, 0.2],[0.2, 0.9, 0.5]], [[-0.1, -0.3, 0.5], [0.6, -0.1, 0]]]
[[[0, 0.1, 0.2],[0.2, 0.4, 0.6]], [[-0.1, -0.3, 0.6], [0.7, 0.1, 0.1]]]
```

Quantization trades **model accuracy** for a smaller size

- Quantizing FP16 → INT4
  - → **3x** size reduction

Fine Tuning: **Low Rank Adaptation (LoRA)**

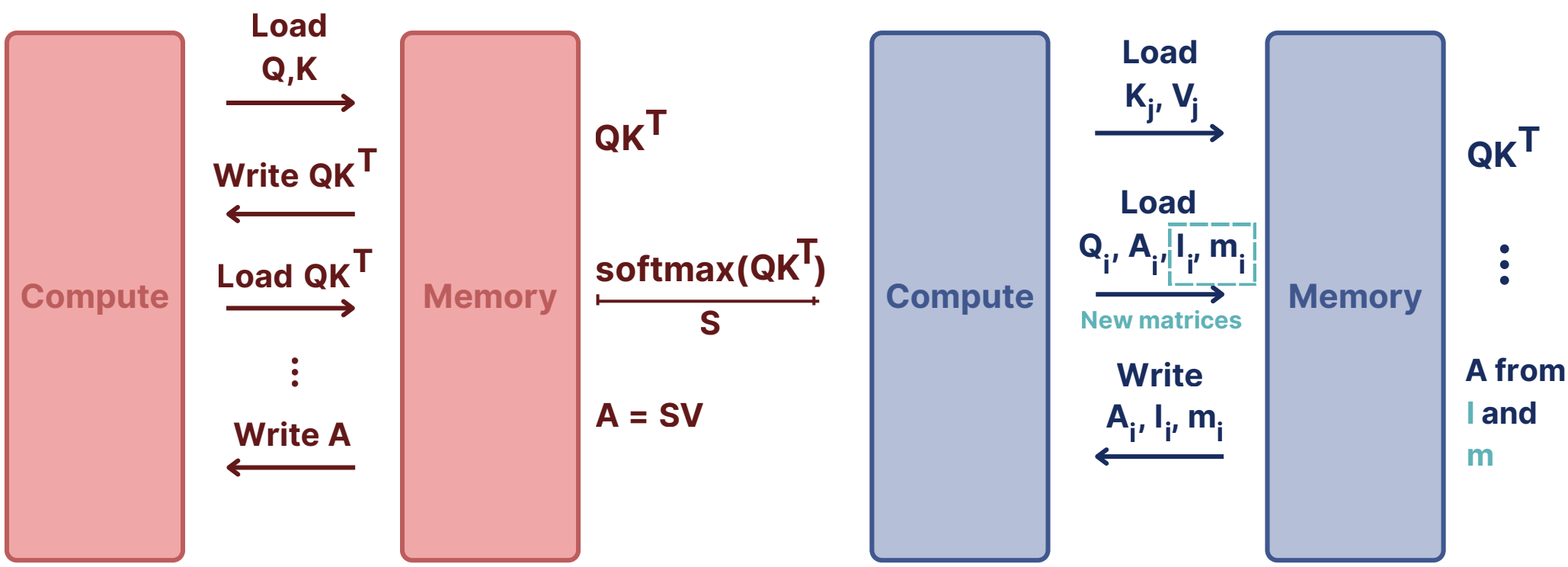


Idea: Add trainable **low-rank matrices** to weights

- Freeze most parameters

Fraction of parameters used for transfer learning

Optimizing Attention: **FlashAttention**



Fused GPU Kernel:

- Computes efficiently and in *parallel*

Speeds up attention and **reduces memory** overhead

What Was The **Result**?

<10B Parameter Models That are:

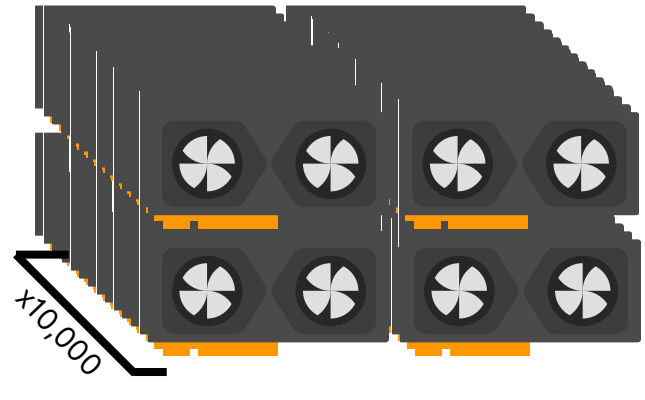
- Fast and ***Cheap***
- Memory-Efficient
- **Performant** on task-specific applications

SLM
2B Params

LLM
700B Params

Cost: $10,000-500,000

Cost: $10M - $500B



Like this **Post**?

Accelerate Your Learning With Our **Weekly AI Newsletter**

- Learn Industry Knowledge *Without* The Headache

Weekly AI Insights, **Visually Explained**

Link in Bio 🔗