24. Topics in Efficient ML

EEC454 Introduction to Machine Learning Systems

2023 Fall, Jaeho Lee

Motivation

Modern AI is big

Last generation of Google Bard required...

Dataset. Text corpus of 7.8×10^{11} tokens

Total dataset size = 780 billion tokens	
Data source	**Proportion of data**
Social media conversations (multilingual)	50%
Filtered webpages (multilingual)	27%
Books (English)	13%
GitHub (code)	5%
Wikipedia (multilingual)	4%
News (English)	1%

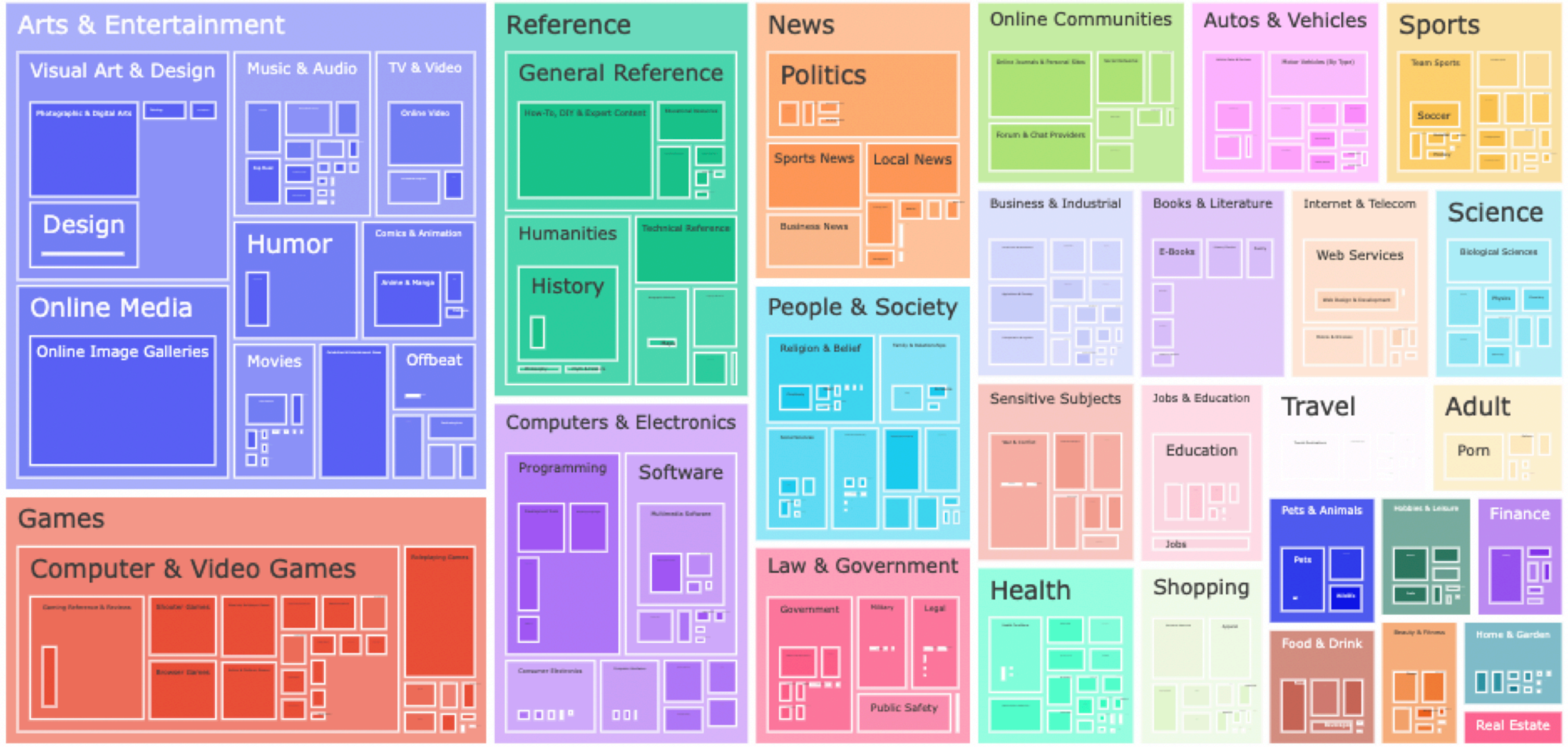


Figure 25: Hierarchical topics detected in the dataset.

Modern AI is big

Last generation of Google Bard required...

Parameters. Total 5.4×10^{11} parameters (in various precisions)

$\approx$ 1TB memory (in 16bits)

Computation. Total 2.56×10^{24} FLOPs for training

$\approx$ $27M, 1500 hours

Model	TFLOPs per token		Train FLOPs	PetaFLOP/s-days
	(non-attn+attn)	(non-attn+attn+remat)		
8B	0.0550	0.0561	4.29×10^{22}	497
62B	0.388	0.392	3.08×10^{23}	3570
540B	3.28	4.10	2.56×10^{24}	29600

Modern AI is big

Last generation of Google Bard required...

Hardware. Total $6{,}144$ TPUv4 chips

Modern AI is big

Last generation of Google Bard required...

Human. 67 Engineers

PaLM: Scaling Language Modeling with Pathways

Aakanksha Chowdhery* Sharan Narang* Jacob Devlin*
Maarten Bosma Gaurav Mishra Adam Roberts Paul Barham
Hyung Won Chung Charles Sutton Sebastian Gehrmann Parker Schuh Kensen Shi
Sasha Tsvyashchenko Joshua Maynez Abhishek Rao† Parker Barnes Yi Tay
Noam Shazeer‡ Vinodkumar Prabhakaran Emily Reif Nan Du Ben Hutchinson
Reiner Pope James Bradbury Jacob Austin Michael Isard Guy Gur-Ari
Pengcheng Yin Toju Duke Anselm Levskaya Sanjay Ghemawat Sunipa Dev
Henryk Michalewski Xavier Garcia Vedant Misra Kevin Robinson Liam Fedus
Denny Zhou Daphne Ippolito David Luan‡ Hyeontaek Lim Barret Zoph
Alexander Spiridonov Ryan Sepassi David Dohan Shivani Agrawal Mark Omernick
Andrew M. Dai Thanumalayan Sankaranarayana Pillai Marie Pellat Aitor Lewkowycz
Erica Moreira Rewon Child Oleksandr Polozov† Katherine Lee Zongwei Zhou
Xuezhi Wang Brennan Saeta Mark Diaz Orhan Firat Michele Catasta† Jason Wei
Kathy Meier-Hellstern Douglas Eck Jeff Dean Slav Petrov Noah Fiedel

Google Research

Preparation

Wrote the initial proposal: Sharan Narang, Alexander Spiridonov, Noah Fiedel, Noam Shazeer, David Luan

Model architecture and optimizer selection: Noam Shazeer, Yi Tay, Sharan Narang, Rewon Child, Aakanksha Chowdhery

Model scaling validation: Aakanksha Chowdhery, Noam Shazeer, Rewon Child

Low-precision finetuning and inference: Shivani Agrawal, Reiner Pope

Training strategy and efficiency: Noam Shazeer, Aakanksha Chowdhery, James Bradbury, Zongwei Zhou, Anselm Levskaya, Reiner Pope

Pod-level Data Parallelism Aakanksha Chowdhery, Paul Barham, Sasha Tsvyashchenko, Parker Schuh

T5X Model Parallelism and Flaxformer Adam Roberts, Hyung Won Chung, Anselm Levskaya, James Bradbury, Mark Omernick, Brennan Saeta

Deterministic data pipeline: Gaurav Mishra, Adam Roberts, Noam Shazeer, Maarten Bosma

Efficient Checkpointing: Sasha Tsvyashchenko, Paul Barham, Hyeontaek Lim

Pathways system: Aakanksha Chowdhery, Paul Barham, Hyeontaek Lim, Thanunalayan Sankaranayana Pillai, Michael Isard, Ryan Sepassi, Sanjay Ghemawat, Jeff Dean

Dataset and Vocabulary development: Maarten Bosma, Rewon Child, Andrew Dai, Sharan Narang, Noah Fiedel

Model Training

Large-scale Training: Aakanksha Chowdhery, Jacob Devlin, Sharan Narang
Large-scale Training includes in-flight debugging of training instability issues, architecture and optimizer improvements, training strategy improvements, and resolving infrastructure bottlenecks.

Infrastructure improvements: Paul Barham, Hyeontaek Lim, Adam Roberts, Hyung Won Chung, Maarten Bosma, Gaurav Mishra, James Bradbury

Model performance validation on downstream tasks: Sharan Narang, Gaurav Mishra

Post-Training

Coordination of results and model analyses: Sharan Narang

Few-shot evaluation infrastructure: Maarten Bosma, Sharan Narang, Adam Roberts

English NLP tasks (few-shot evaluation): Sharan Narang, Nan Du

Finetuning on SuperGlue: Sharan Narang, Yi Tay, Liam Fedus

BIG-bench tasks (few-shot evaluation): Gaurav Mishra, Noah Fiedel, Guy Gur-Ari, Jacob Devlin, Aakanksha Chowdhery, Sharan Narang

Reasoning tasks (few-shot evaluation): Jason Wei, Xuezhi Wang, Denny Zhou

Code tasks (few-shot evaluation and finetuning): Jacob Austin, Henryk Michalewski, Charles Sutton, Aitor Lewkowycz, Kensen Shi, Pengcheng Yin, Oleksandr Polozov, Vedant Misra, Michele Catasta, Abhishek Rao, David Dohan, Aakanksha Chowdhery

Translation tasks (few-shot evaluation): Xavier Garcia, Orhan Firat

Multilingual Natural Language Generation (few-shot evaluation and finetuning): Joshua Maynez, Sebastian Gehrmann

Multilingual Question Answering (few-shot evaluation and finetuning): Sharan Narang, Yi Tay

Analysis of noise in few-shot performance: Barret Zoph

Representational Bias Analysis (few-shot evaluation and dataset analysis): Marie Pellat, Kevin Robinson, Sharan Narang, Jacob Devlin, Emily Reif, Parker Barnes

Dataset contamination: Jacob Devlin, Sharan Narang

Memorization: Katherine Lee, Daphne Ippolito, Jacob Devlin

Exploring Explanations: Jacob Devlin

Ethical Considerations: Marie Pellat, Kevin Robinson, Mark Díaz, Sunipa Dev, Parker Barnes, Toju Duke, Ben Hutchinson, Vinodkumar Prabhakaran, Kathy Meier-Hellstern

Compute Usage and Environmental Impact: Aakanksha Chowdhery, James Bradbury, Zongwei Zhou

Model serving (API, use cases and efficiency): Sharan Narang, Jacob Devlin, Jacob Austin, James Bradbury, Aakanksha Chowdhery, Zongwei Zhou, Reiner Pope, Noah Fiedel

Model card and datasheet: Alexander Spiridonov, Andrew Dai, Maarten Bosma, Jacob Devlin

Product Management: Alexander Spiridonov

Paper Writing and Reviewing: All authors contributed to writing and reviewing the paper

Full Project Lifecycle

Overall project leadership: Sharan Narang, Aakanksha Chowdhery, Noah Fiedel

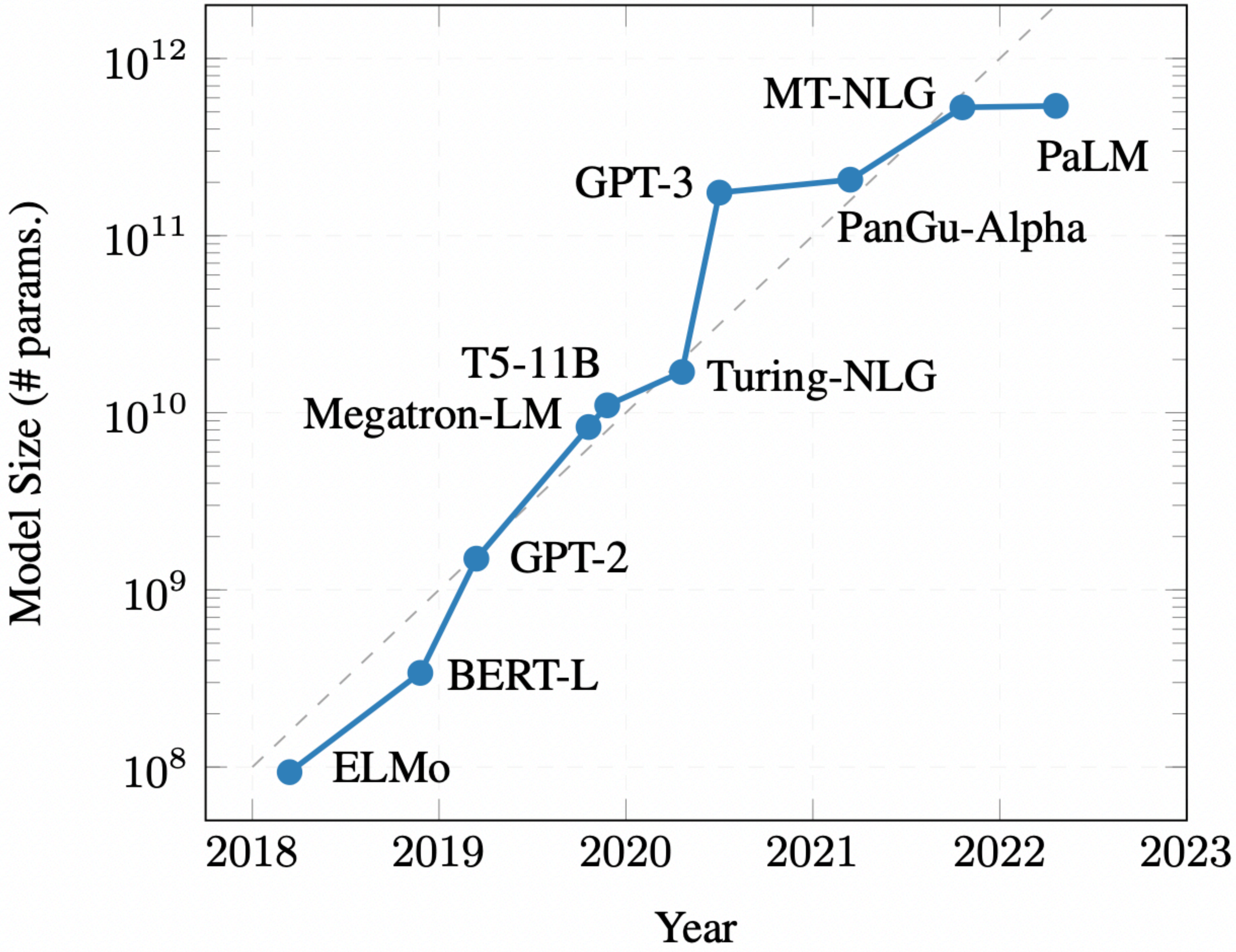
Responsible AI and Safety leadership: Kathy Meier-Hellstern

Resource management: Erica Moreira

Advisors: Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, Noah Fiedel

Modern AI is big

Even worse, models are going fast!



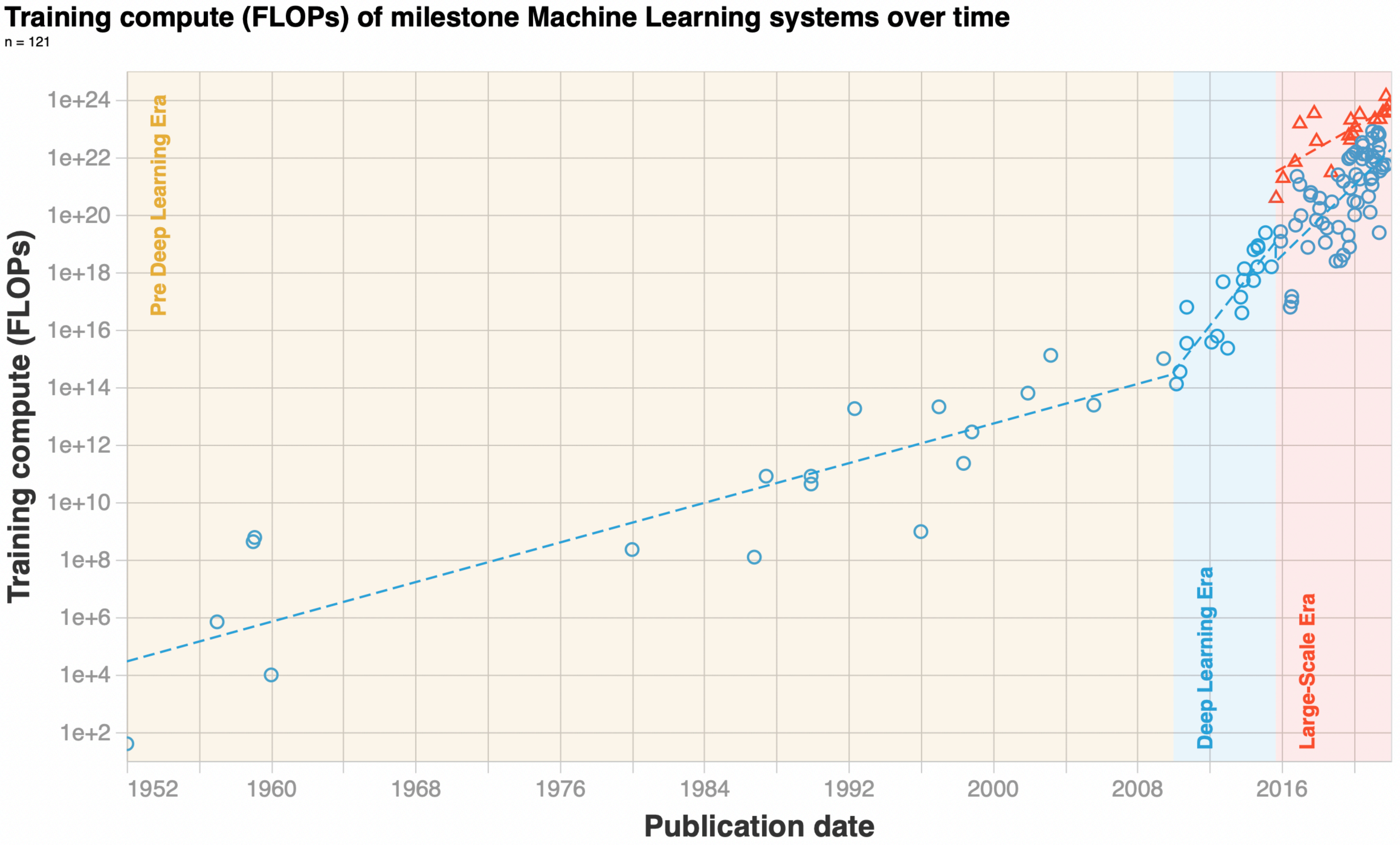


Figure 1: Trends in $n = 121$ milestone ML models between 1952 and 2022. We distinguish three eras. Notice the change of slope circa 2010, matching the advent of Deep Learning; and the emergence of a new large-scale trend in late 2015.

https://twitter.com/kchonyc/status/1732467029808951732

제미나이에 쓰인 데이타의 quantity랑 모델 싸이즈를 tell me, please

제미나이에 쓰인 데이터의 quantity는 텍스트와 이미지, 비디오를 포함해 총 1.56조 개의 파라미터를 갖고 있습니다. 파라미터는 모델의 크기와 복잡성을 나타내는 지표로, 파라미터 수가 많을수록 더 복잡한 모델을 만들 수 있습니다.

제미나이의 모델 크기는 1.56조 개의 파라미터로, 이는 GPT-4의 1.37조 개의 파라미터보다 약 13% 더 큽니다. 모델 크기가 클수록 더 많은 정보를 학습할 수 있고 더 복잡한 작업을 수행할 수 있습니다.

구체적으로, 제미나이에 사용된 데이터는 다음과 같습니다.

- 텍스트: 구글 검색, 유튜브, 구글 맵스 등 구글의 다양한 서비스에서 수집한 텍스트 데이터
- 이미지: 구글 이미지 검색, 구글 포토 등 구글의 다양한 서비스에서 수집한 이미지 데이터
- 비디오: 구글 유튜브, 구글 맵스 등 구글의 다양한 서비스에서 수집한 비디오 데이터

이러한 다양한 데이터를 통해 제미나이는 텍스트, 이미지, 비디오를 포함한 다양한 정보를 학습할 수 있었습니다.

According to the jailbreak, Google Gemini (2023) requires...

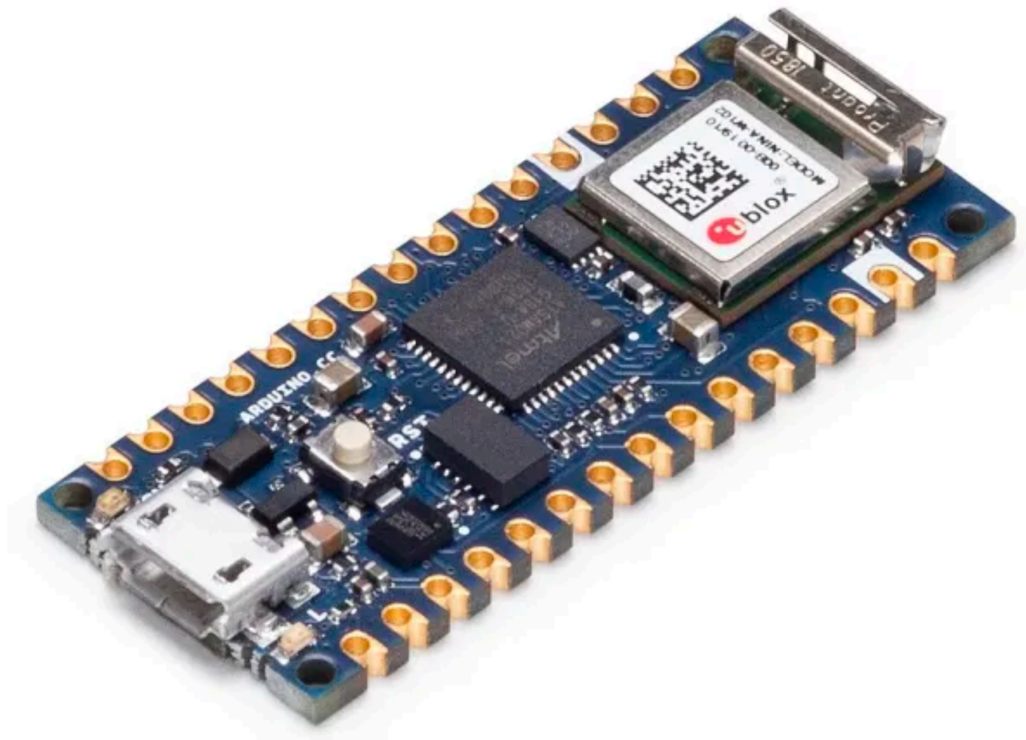
- 1.56 trillion parameters
- 6.24 trillion bytes of data
- $1~2B for training.

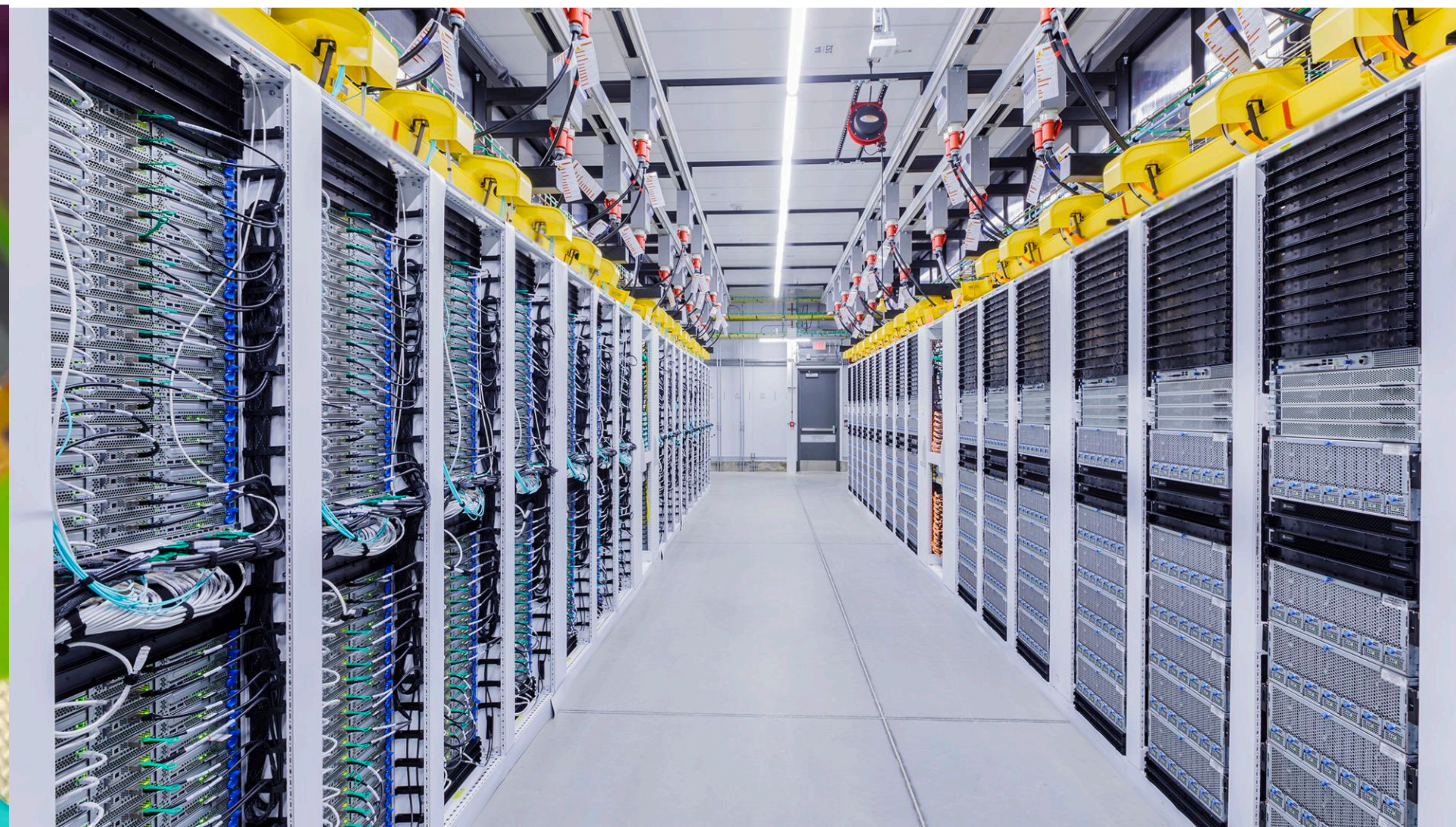
Goals

Goals

Efficient ML is a collection of techniques to reduces various costs of ML,

- **Scale.** Microcontrollers (a ConvNet)
  Mobile phones (Google Gemini Nano)
  Laptop (small LLMs)
  GPU clusters (Giant LLMs)

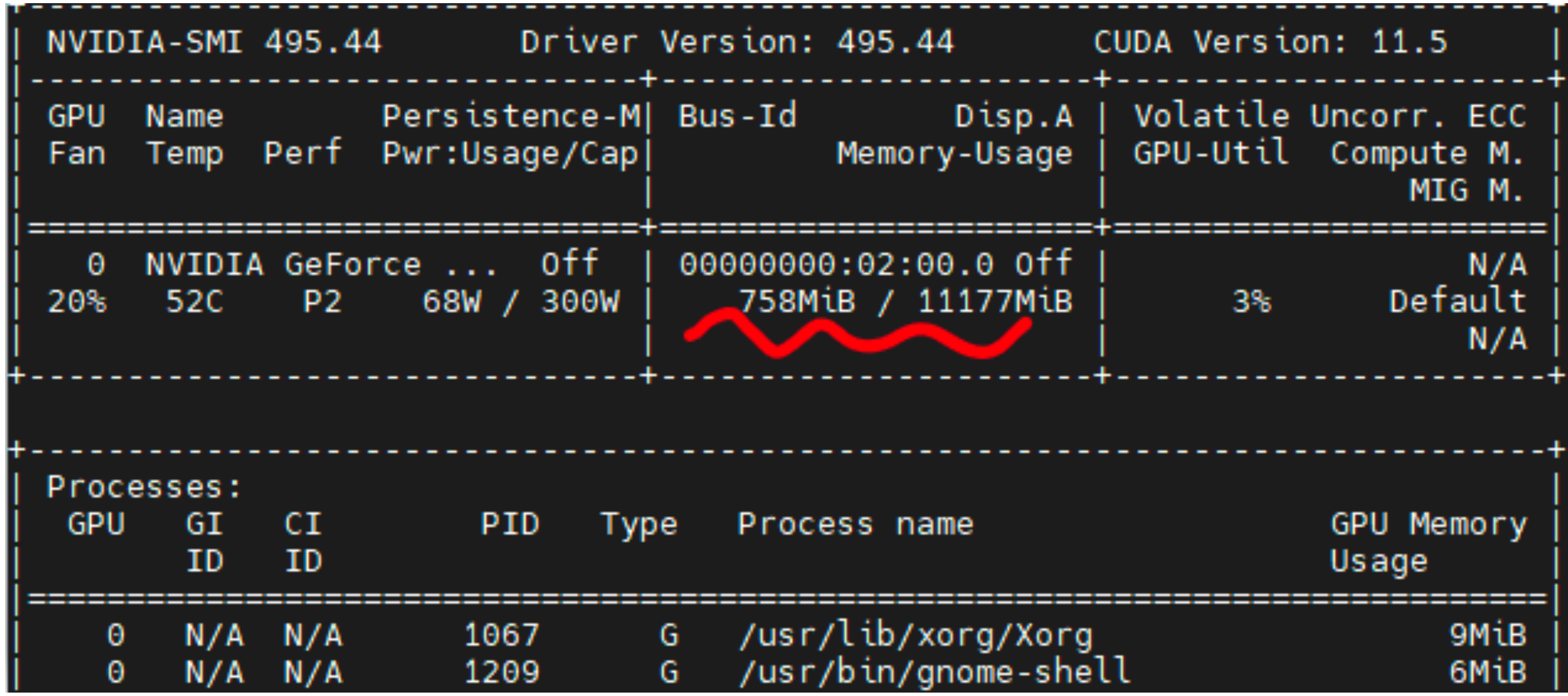




Goals

Efficient ML is a collection of techniques to reduces various costs of ML,

- **Scale.** From microcontrollers to LLMs
- **Focus.** Inference latency
  Inference peak memory
  Training memory
  Training cost
  …



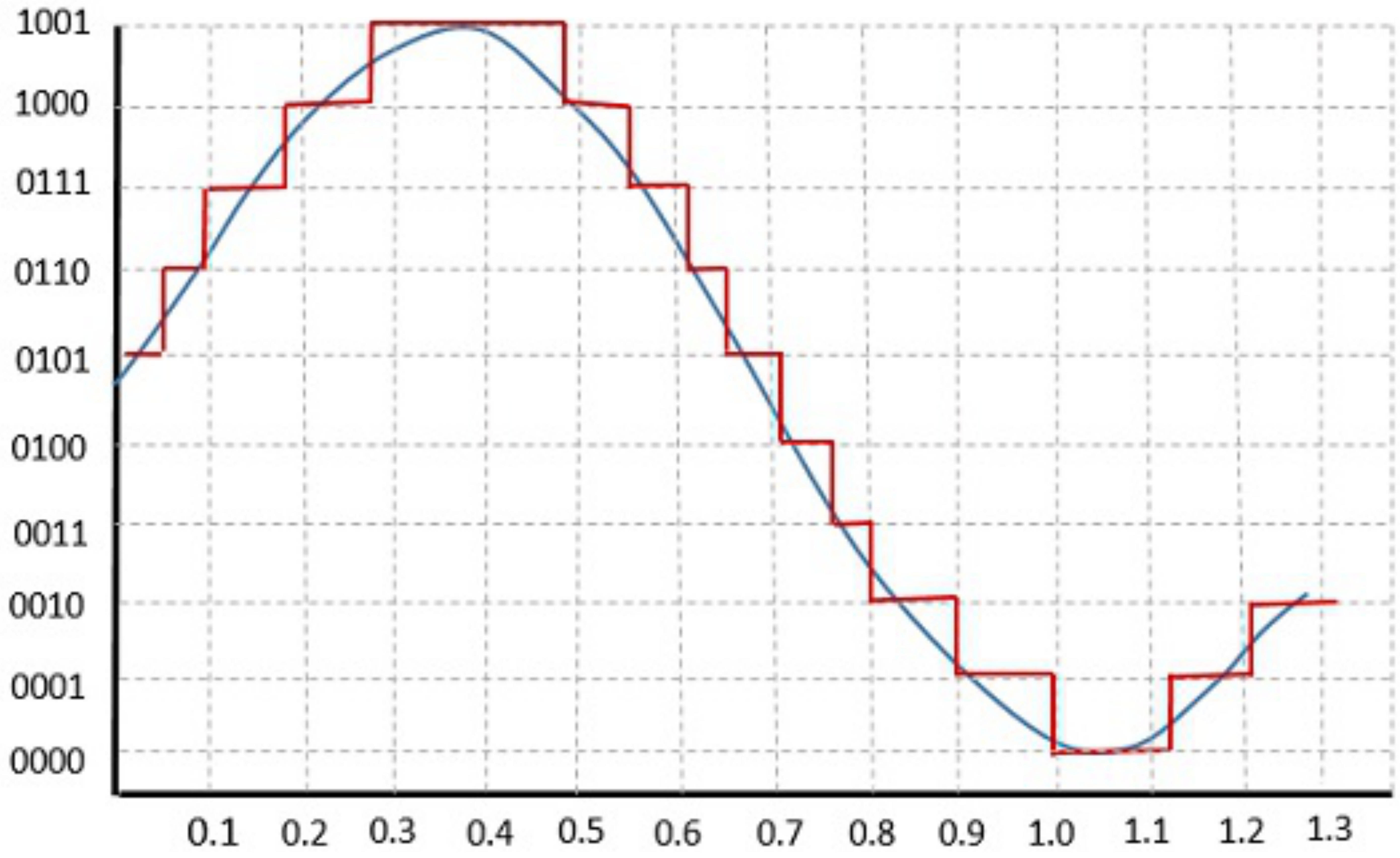

2. On-device training
Train done
Prediction:
class 1
Ground True:
class 1
fps:1.798
STM32F746G-DISCO
MB1191B
www.st.com/stm32f7-discovery
CAMERA
Prediction:
green: correct
red: incorrect

Techniques

1. Quantization

Quantization

- **Idea.** Reducing the precision level of parameters in deep learning.
  - Weight only / Weight & Activation



Quantization

- **Benefit.** A lot!
  - Energy
  - Memory bandwidth
  - Storage space on RAM/SSD
  - Chip area

Add energy (pJ)	
INT8	FP32
0.03	0.9
30X energy reduction	

Mult energy (pJ)	
INT8	FP32
0.2	3.7
18.5X energy reduction	

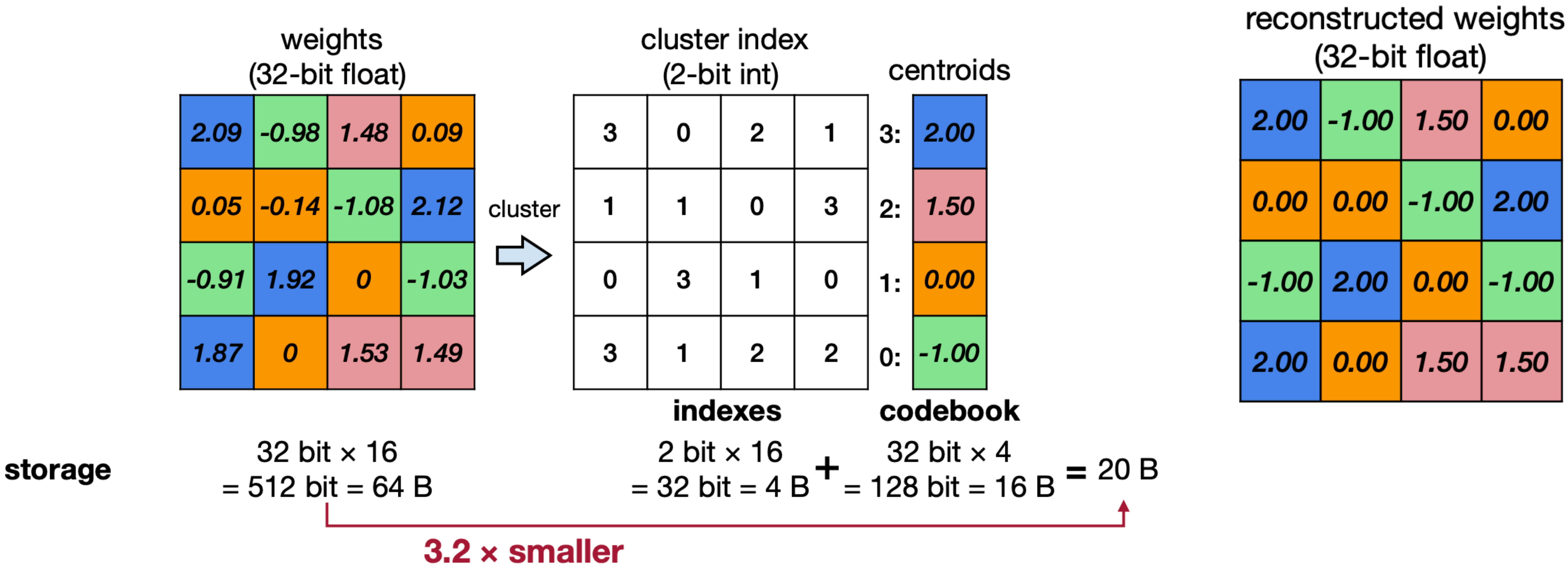
Mem access energy (pJ)	
Cache (64-bit)	
8KB	10
32KB	20
1MB	100
DRAM	1300-2600
Up to 4X energy reduction	

Add area (μm^2)	
INT8	FP32
36	4184
116X area reduction	

Mult area (μm^2)	
INT8	FP32
282	7700
27X area reduction	

Quantization

- **Key question.** Find the right quantization level.
  - Similar to K-means, but in 1-dimension.



Quantization

- **Popular.** The *linear quantization*
  - Optimized for inference; allows full computation in quantized form.

weights (32-bit float) | quantized weights (2-bit signed int) | zero point (2-bit signed int) | scale (32-bit float)

$$\begin{bmatrix}2.09&-0.98&1.48&0.09\\0.05&-0.14&-1.08&2.12\\-0.91&1.92&0&-1.03\\1.87&0&1.53&1.49\end{bmatrix} \Rightarrow \left(\begin{bmatrix}1&-2&0&-1\\-1&-1&-2&1\\-2&1&-1&-2\\1&-1&0&0\end{bmatrix} - \; -1\right) \times 1.07$$

$$r = (q - Z) \times S$$

reconstructed weights (32-bit float)

$$\begin{bmatrix}2.14&-1.07&1.07&0\\0&0&-1.07&2.14\\-1.07&2.14&0&-1.07\\2.14&0&1.07&1.07\end{bmatrix}$$

Floating-point (r)

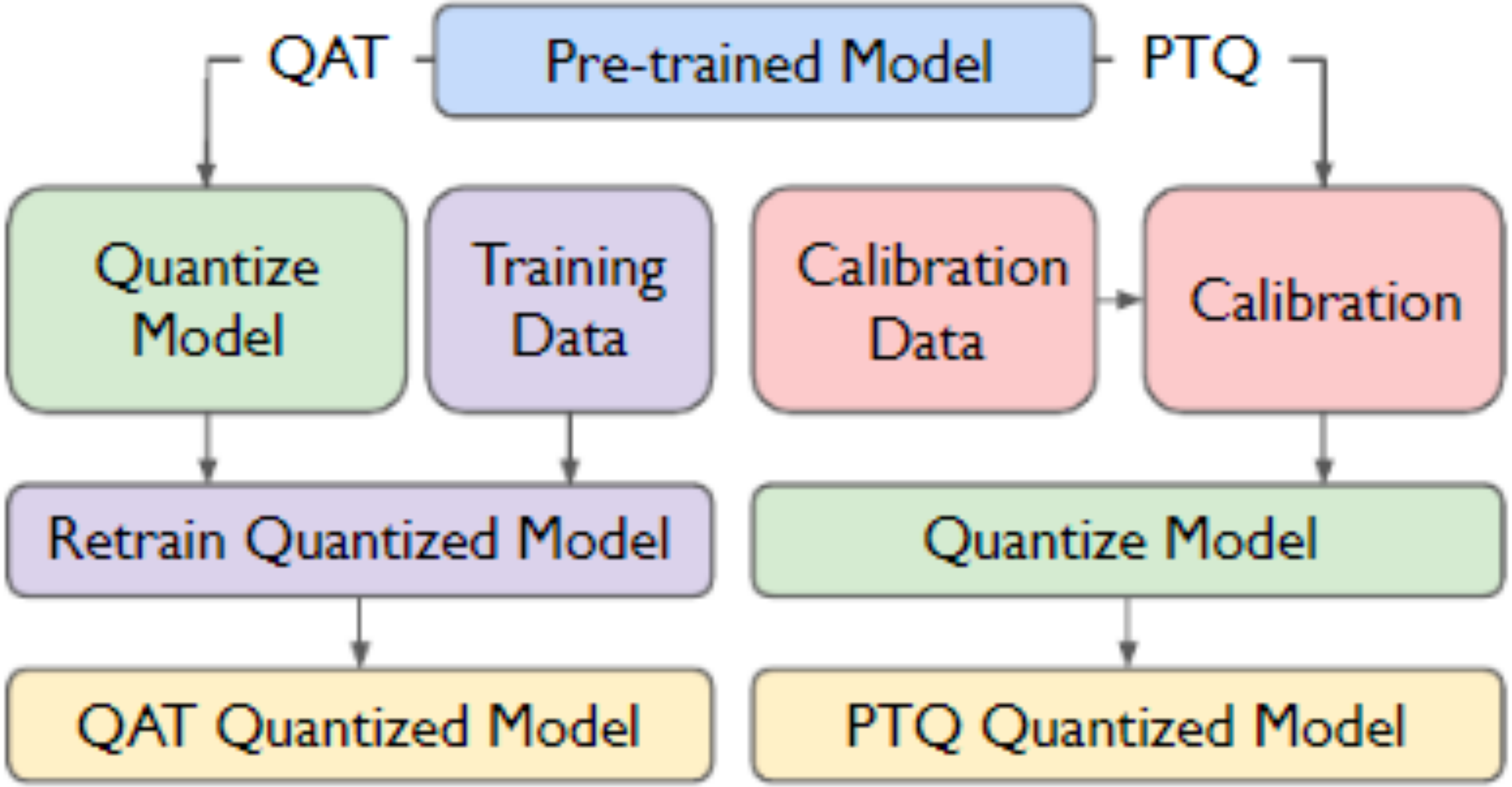
Integer (q)

Integer (Z)
- quantization parameter
- allow real number *r=0* be exactly representable by a quantized integer Z

Floating-point (S)
- quantization parameter

Quantization

- **Advanced.** PTQ vs QAT, Quantized training, Tree-based quantization

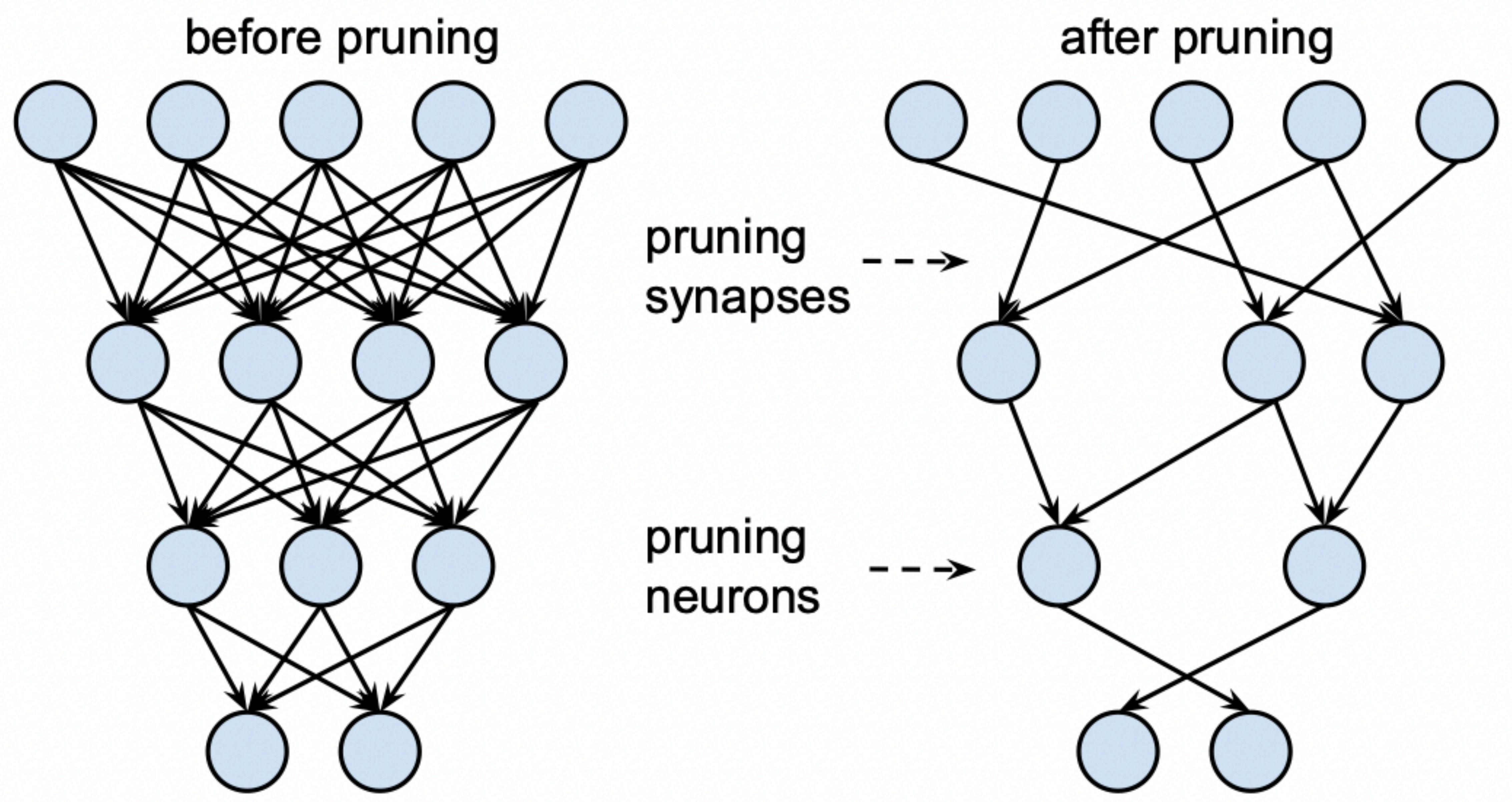


2. Pruning

Pruning

- **Idea.** Making ***some weights equal to zero***.

$$\begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ a_5 & a_6 & a_7 & a_8 \\ a_9 & a_{10} & a_{11} & a_{12} \\ a_{13} & a_{14} & a_{15} & a_{16} \end{bmatrix} \longrightarrow \begin{bmatrix} 0 & 0 & \tilde{a}_3 & \tilde{a}_4 \\ \tilde{a}_5 & 0 & \tilde{a}_7 & 0 \\ \tilde{a}_9 & 0 & 0 & \tilde{a}_{12} \\ \tilde{a}_{13} & 0 & \tilde{a}_{15} & \tilde{a}_{16} \end{bmatrix}$$



Pruning

- **Benefit.** Reduce both memory and computation associated with zeros

$$\begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}$$

32bits x 4 = 128bits

$$\begin{bmatrix} a_1 & 0 \\ 0 & a_4 \end{bmatrix}$$

32bits x 2 + α = 64bits + α

$$\begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \begin{bmatrix} b_1 & b_2 \\ b_3 & b_4 \end{bmatrix} = \begin{bmatrix} a_1b_1 + a_2b_3 & a_1b_2 + a_2b_4 \\ a_3b_1 + a_4b_3 & a_3b_1 + a_4b_4 \end{bmatrix}$$

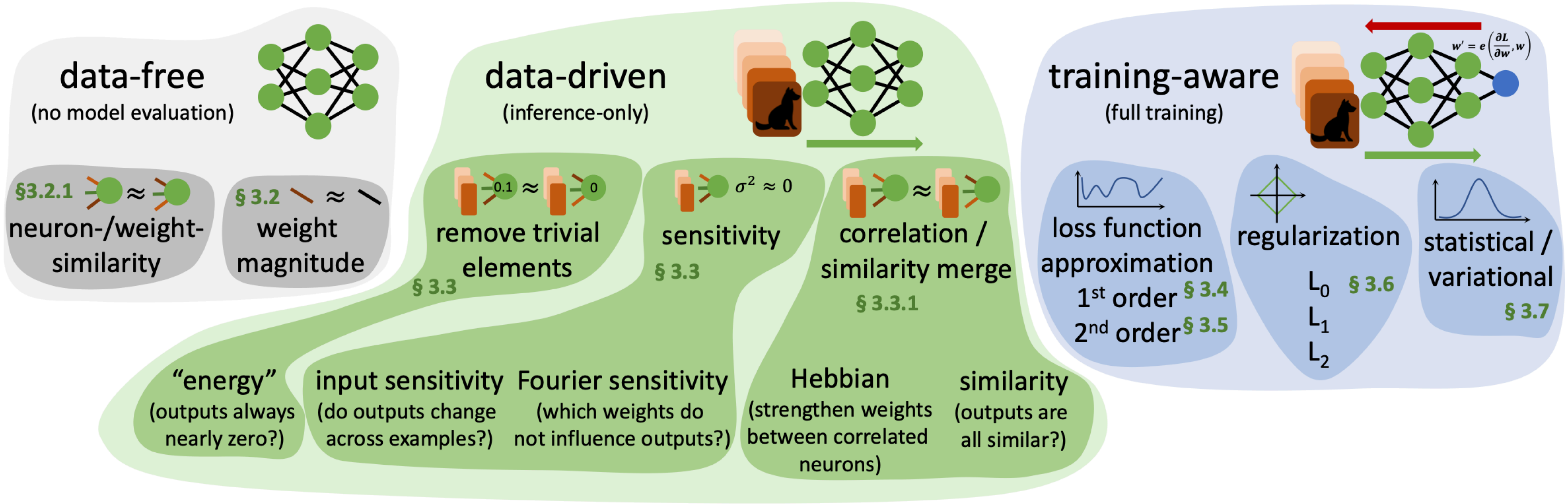
8 Multiplications, 4 Additions

$$\begin{bmatrix} a_1 & 0 \\ 0 & a_4 \end{bmatrix} \begin{bmatrix} b_1 & b_2 \\ b_3 & b_4 \end{bmatrix} = \begin{bmatrix} a_1b_1+0 & a_1b_2+0 \\ 0+a_4b_3 & 0+a_4b_4 \end{bmatrix}$$

4 Multiplications, 0 Additions

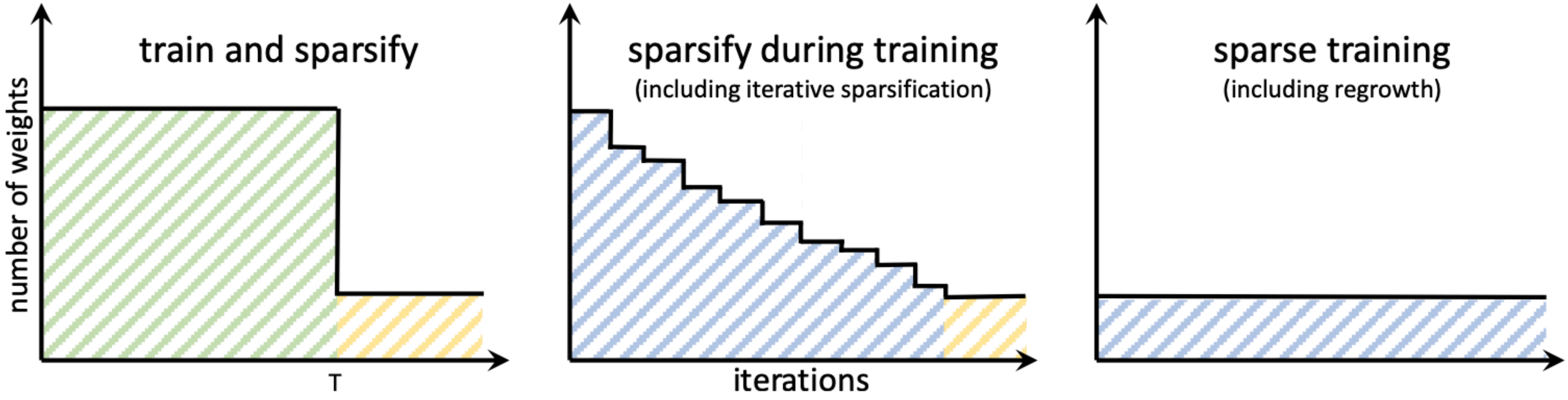
Pruning

- **Key question.** Selecting the weights to remove
  - Which weights? When to prune? How much?



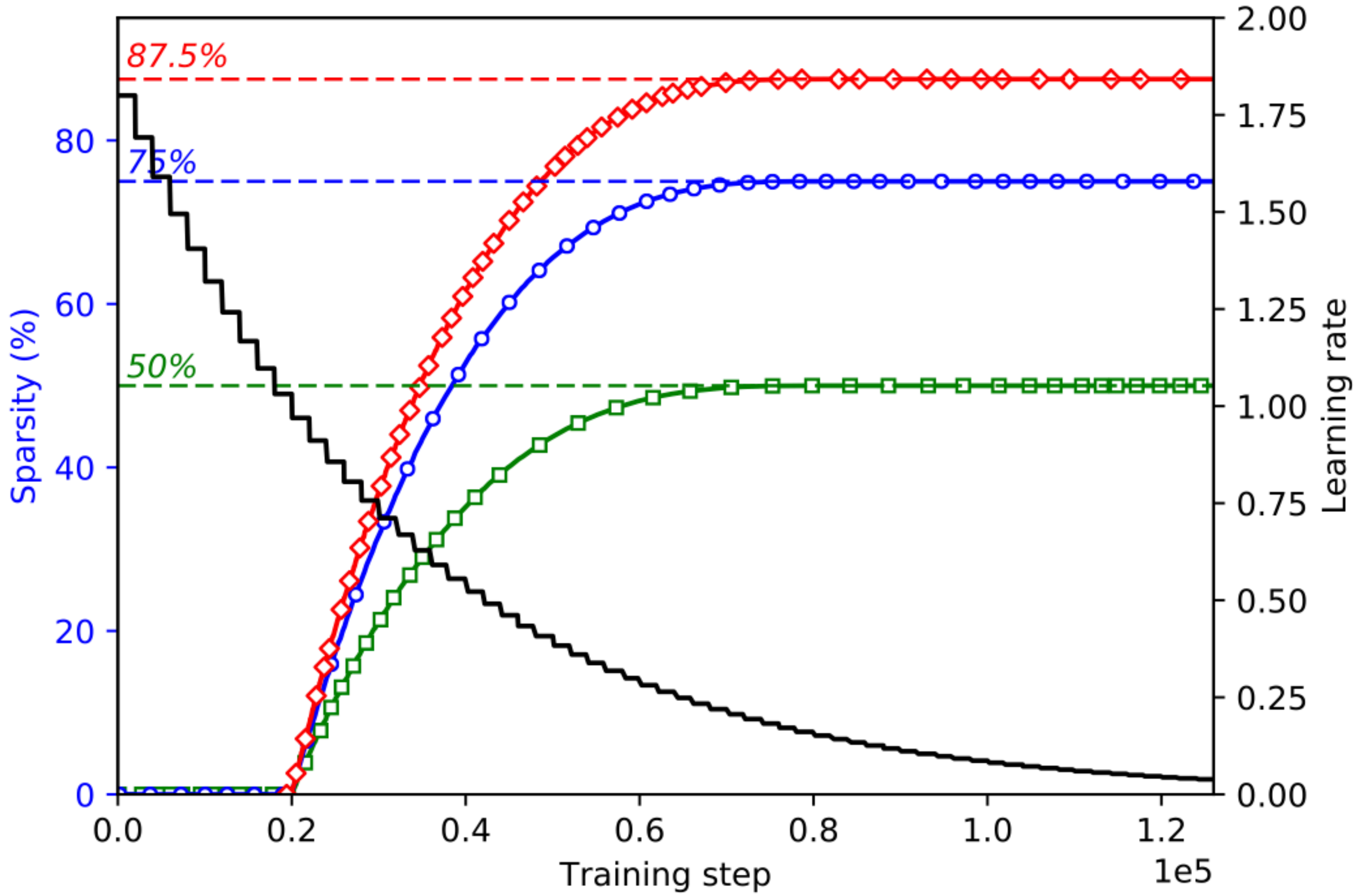
Pruning

- **Key question.** Selecting the weights to remove
  - Which weights? When to prune? How much?



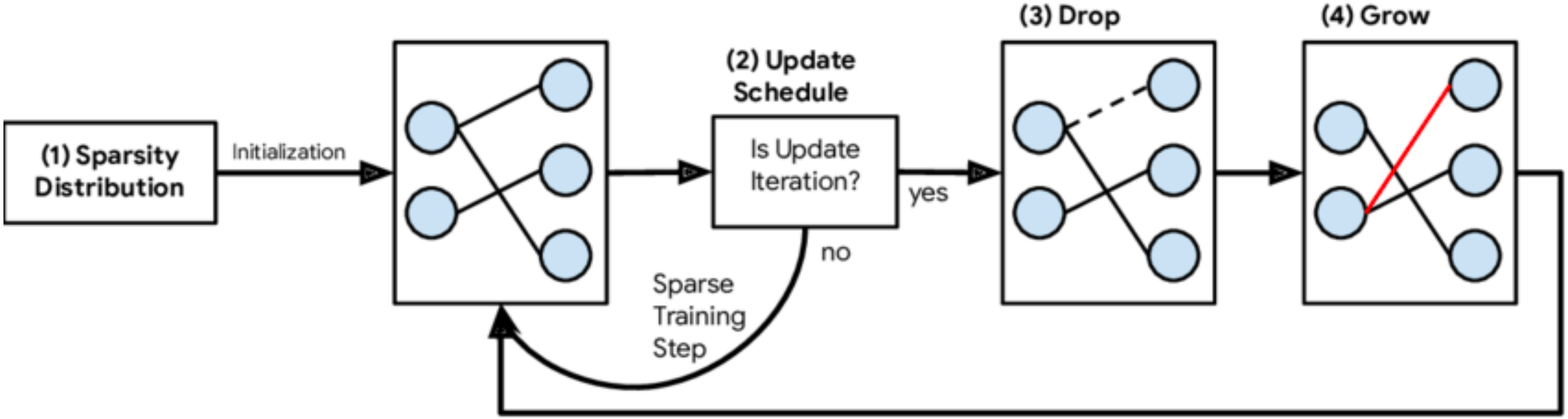
Pruning

- **Popular.** Gradual, magnitude-based pruning (for inference compute)
  - Remove small-magnitude weights from each layer.



Pruning

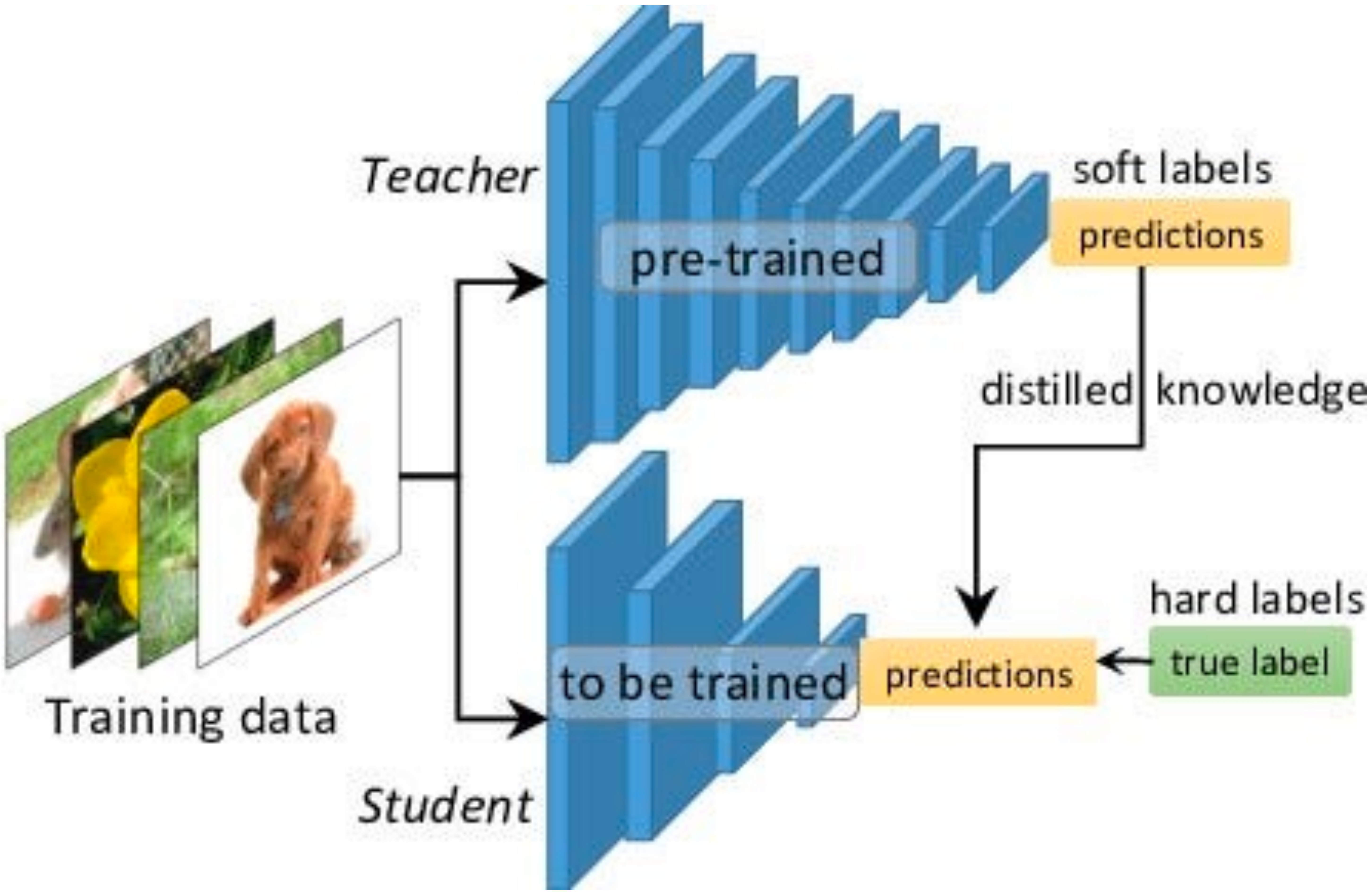
- **Advanced.** Sparse training, 2:4 Sparsity, Post-training sparsity



3. Knowledge Distillation

Distillation

- **Idea.** Use a large model to better train a small model



Distillation

- **Benefits.** Better accuracy of the student model
  - Sometimes can utilize the knowledge of ***teacher dataset***

System	Test Frame Accuracy	WER
Baseline	58.9%	10.9%
10xEnsemble	61.1%	10.7%
Distilled Single model	60.8%	10.7%

Table 1: Frame classification accuracy and WER showing that the distilled single model performs about as well as the averaged predictions of 10 models that were used to create the soft targets.

Distillation

- **Key question.** What should we distill?
  - Prediction, features, relations, attention, …

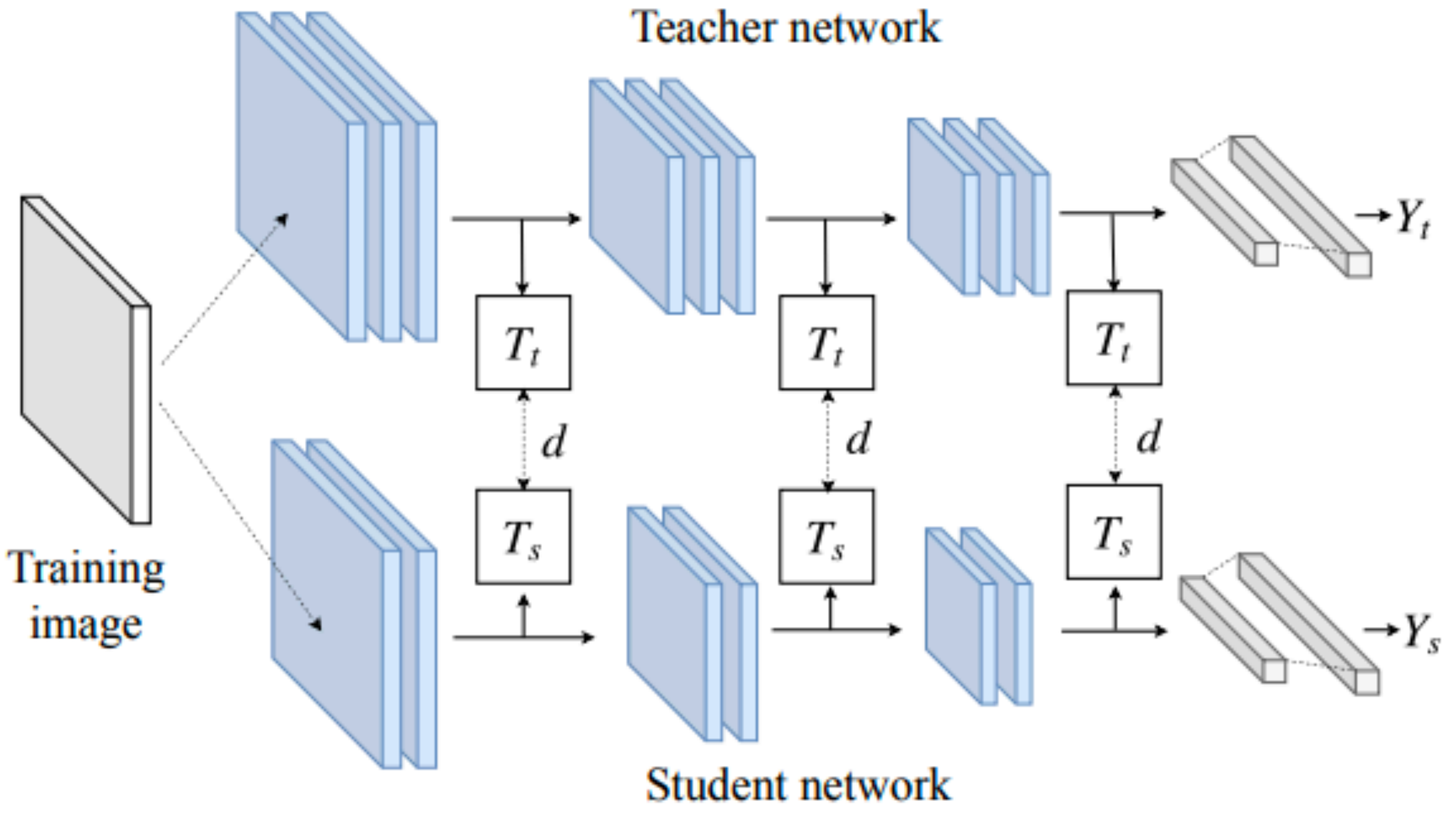
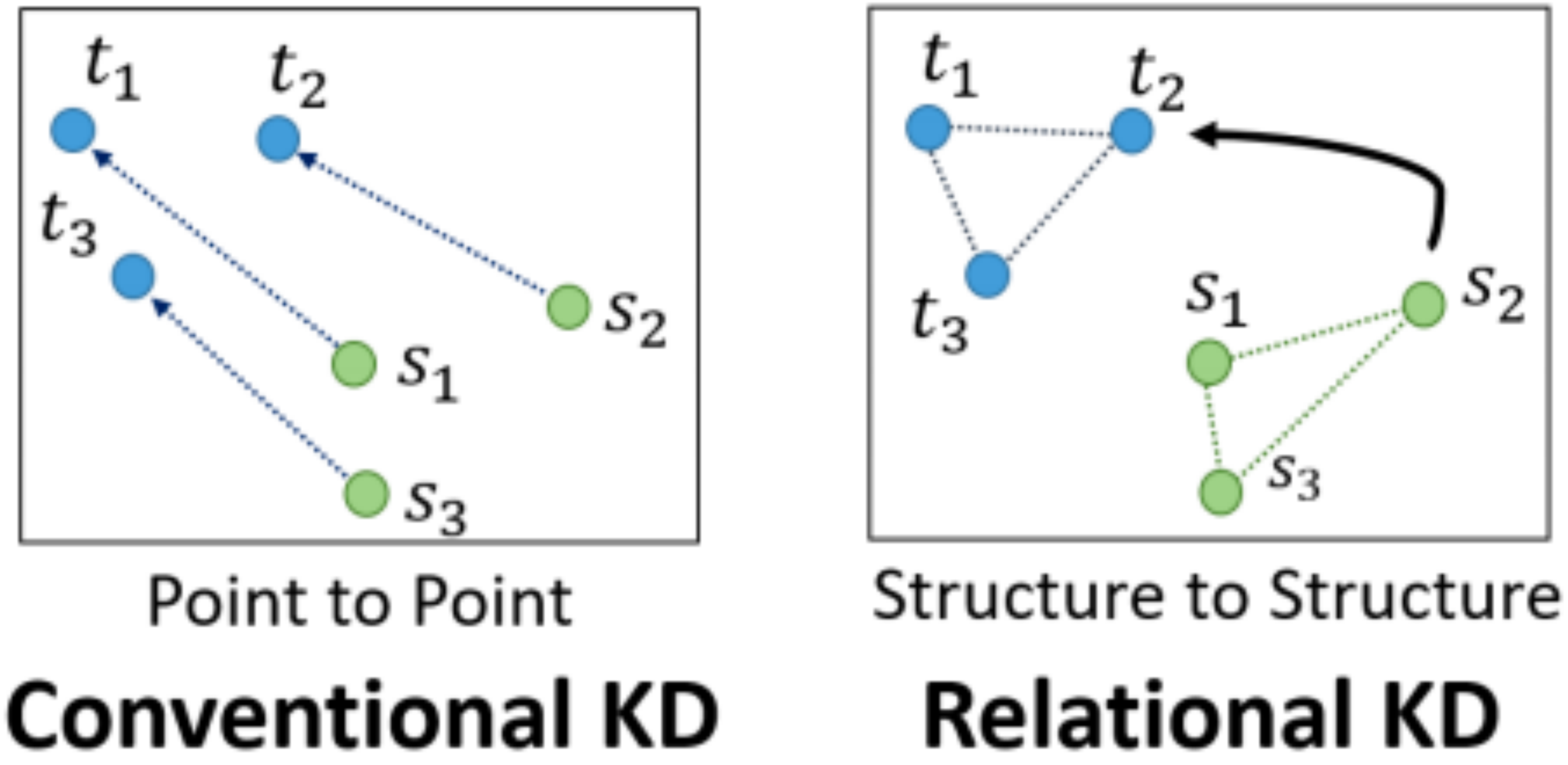
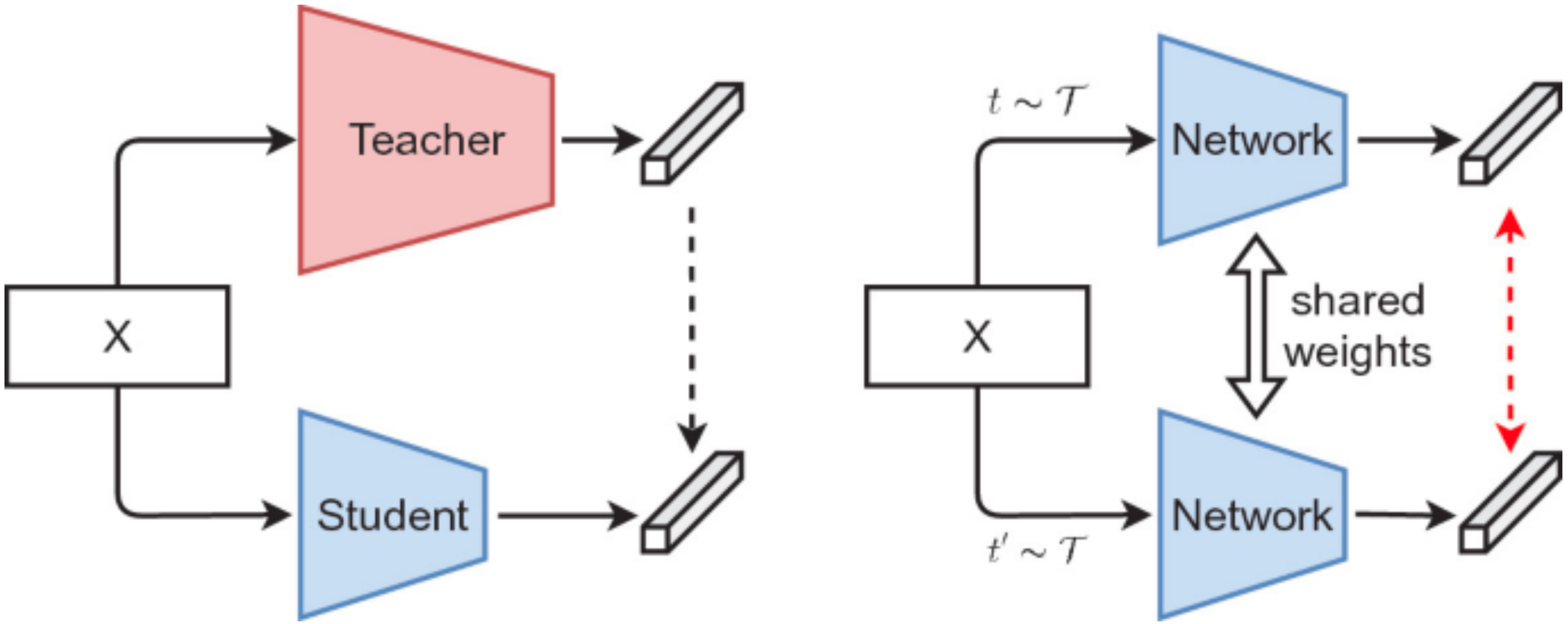


Figure 2. The general training scheme of feature distillation. The form of teacher transform T_t, student transform T_s and distance d differ from method to method.



Distillation

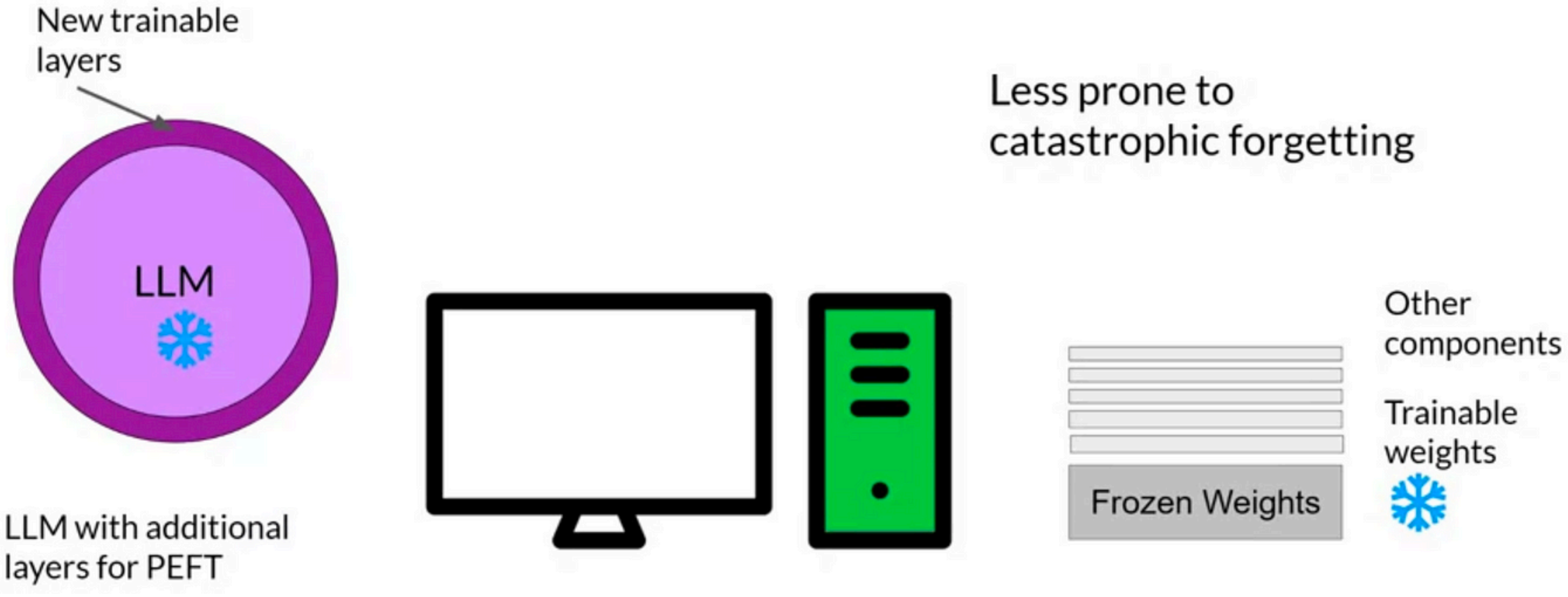
- **Advanced.** Data-free distillation, Self-distillation, Self-training



4. Parameter-efficient fine-tuning

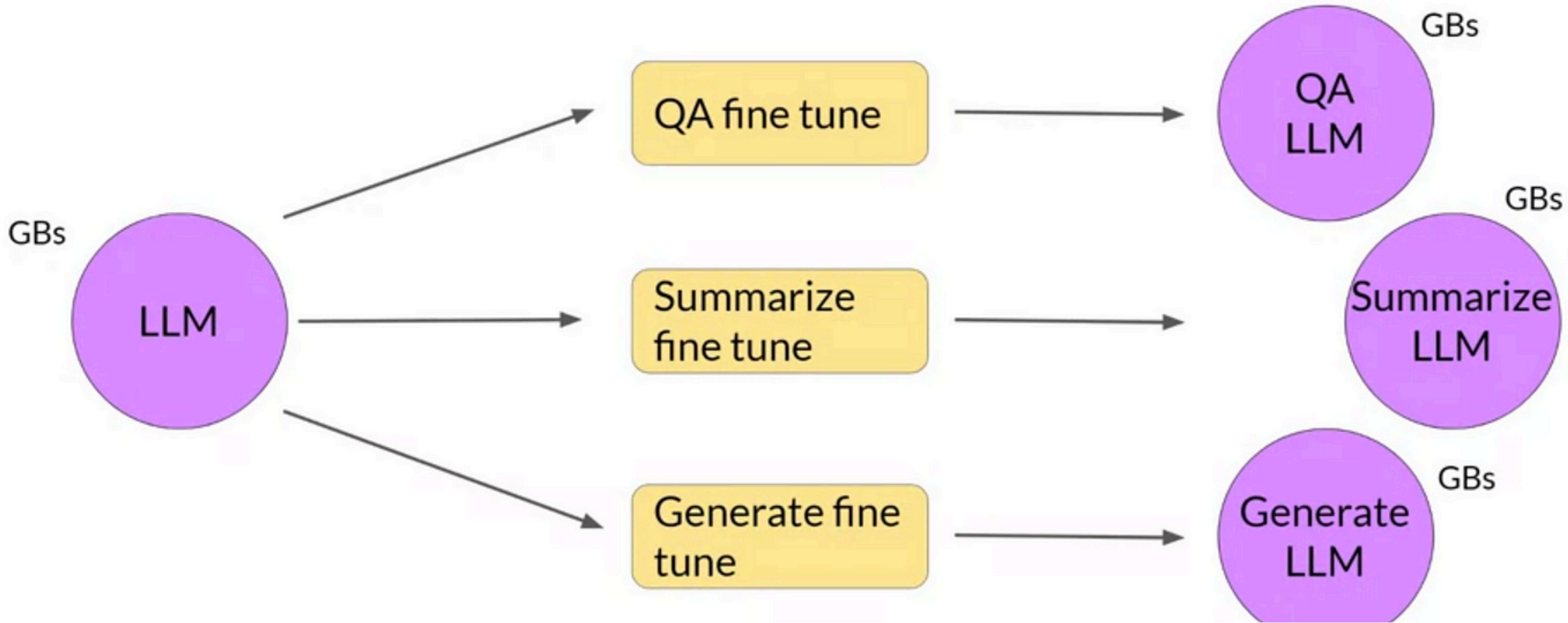
PEFT

- **Idea.** Use only a small number of additional weight for fine-tuning.



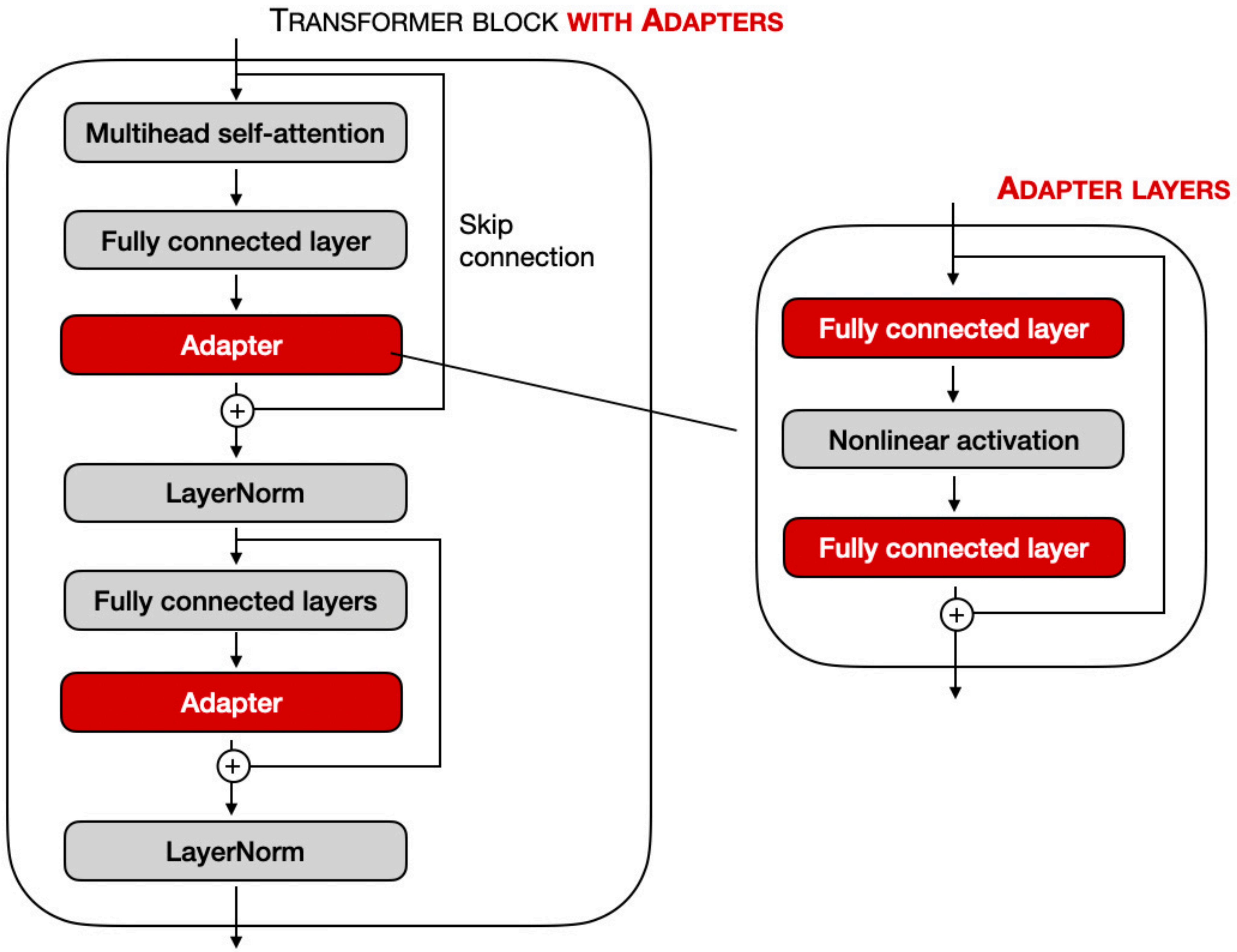
PEFT

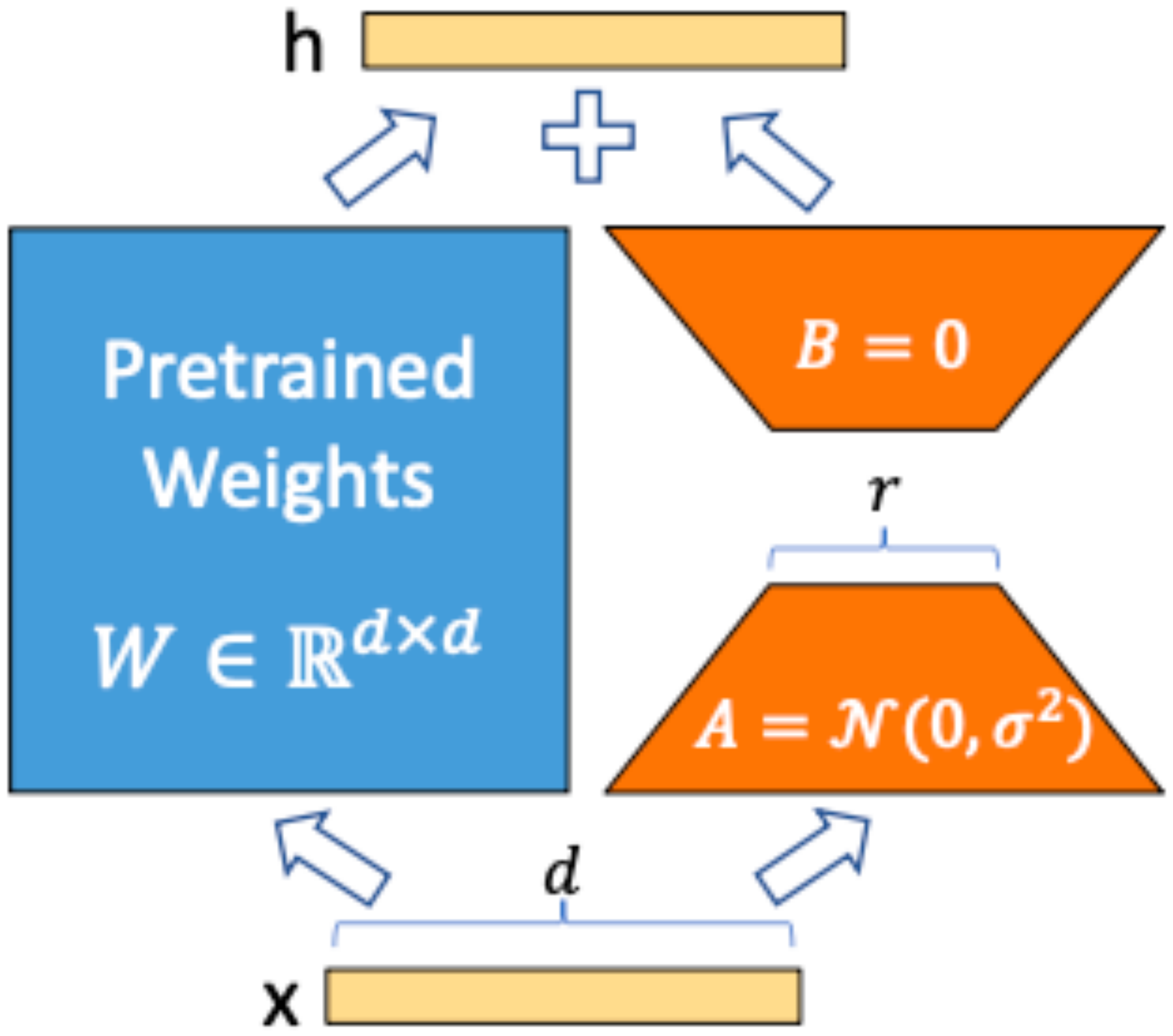
- **Benefit.** Low training cost, small per-task storage
    - Easy personalization / specialization.



PEFT

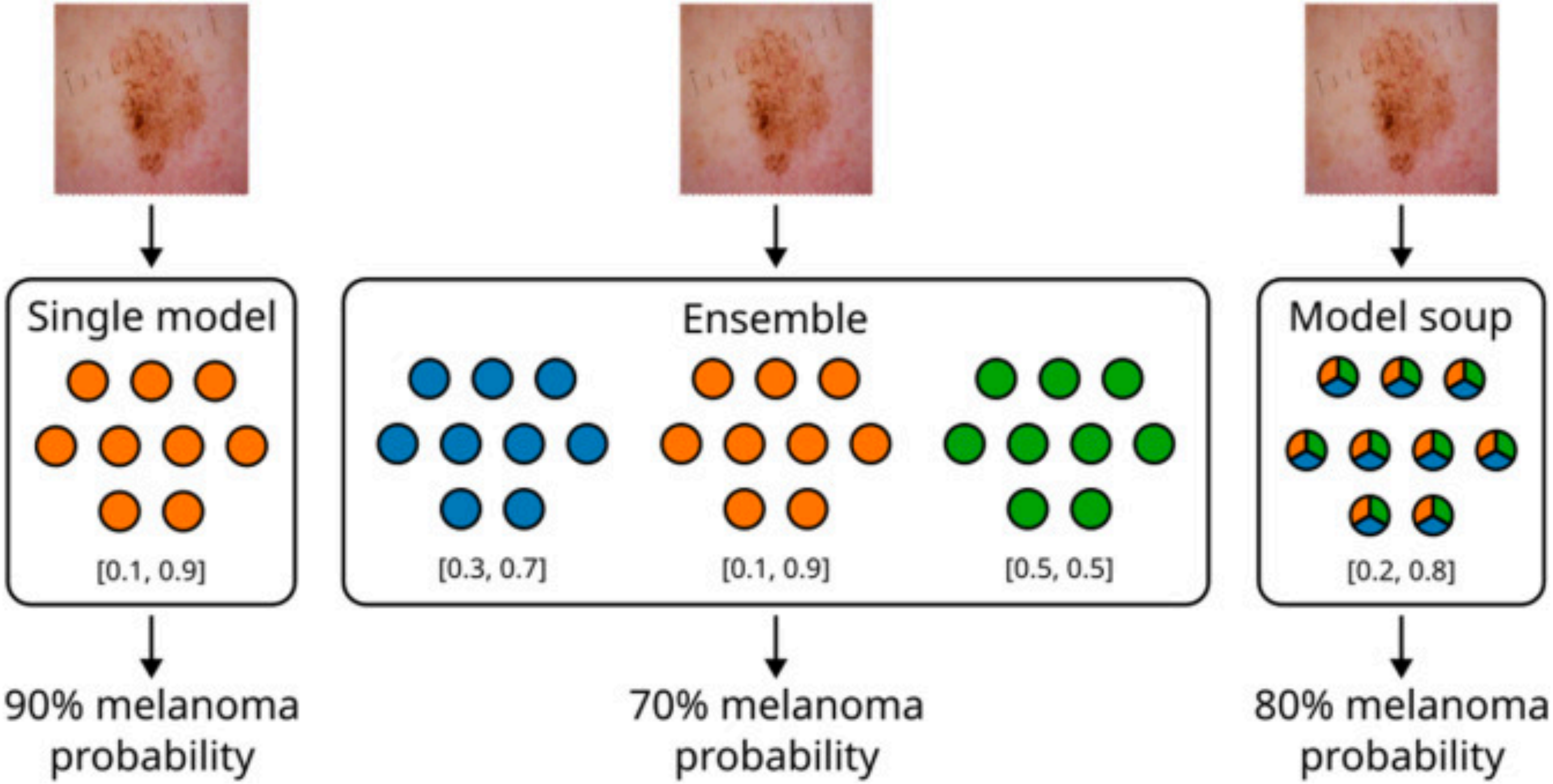
- **Key question.** How to augment the original model?
  - Add layers (adapter), Additive low-rank matrices (LoRA)





PEFT

- **Advanced.** Model Soup, QLoRA …



Remarks

Concluding Remarks

- Making model efficient ***requires***...
  - Understanding what is going on
  - Identifying the essence of ML practices
  - In-depth math & system knowledges
- As a result, we get...
  - Saving $$$
  - Cleaner environment
  - Democratization / Decentralization in ML

Cheers