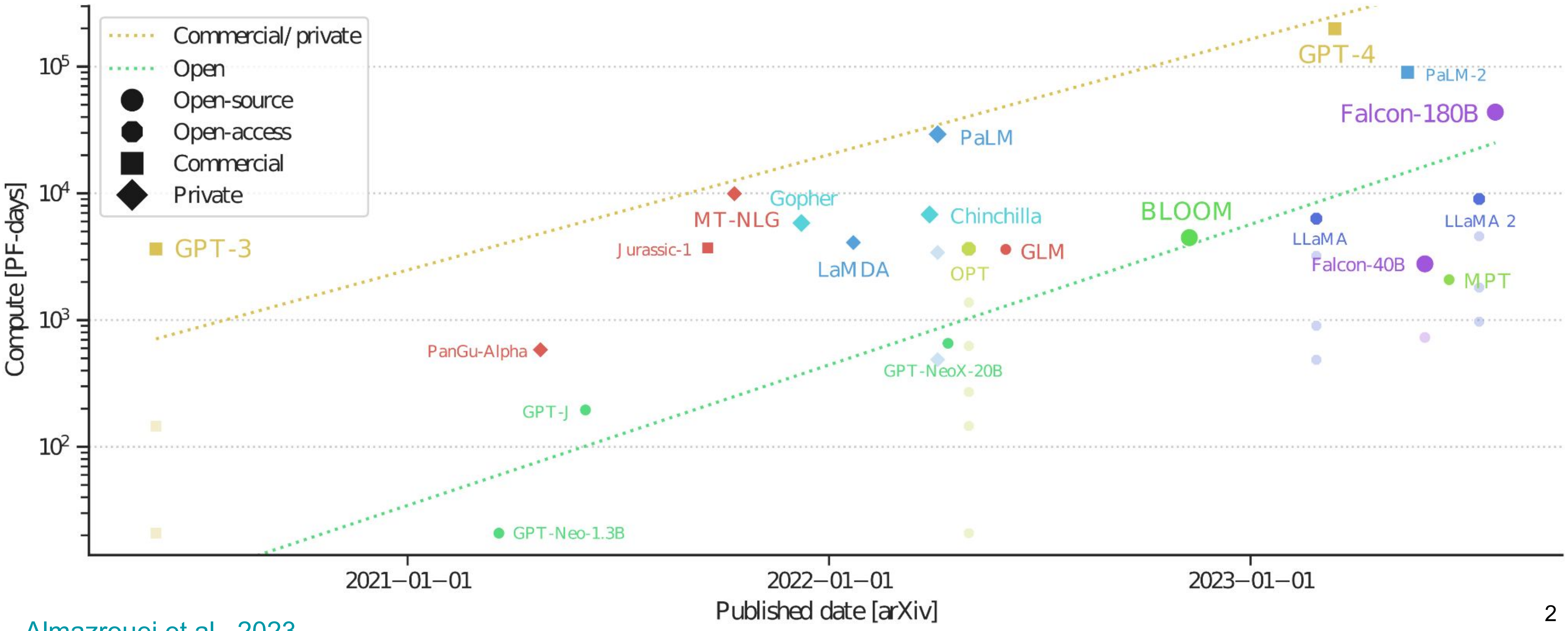
Efficient Foundation Models via Quantization

Tim Dettmers

Language models grew 100x in compute requirements in a few years



[Almazrouei et al., 2023]

This lecture …

This lecture …

Using foundation models

This lecture …

Using foundation models

Finetuning foundation models

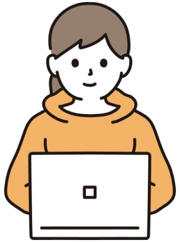
This lecture …

Using foundation models

Finetuning foundation models

Quantization: companies vs users

Typical user groups for each case

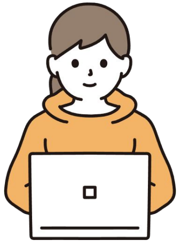


Using foundation models

Finetuning foundation models

Quantization: companies vs users

Typical user groups for each case

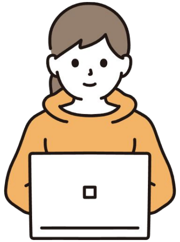


Using foundation models

Finetuning foundation models

Quantization: companies vs users

Typical user groups for each case



Using foundation models

Finetuning foundation models

Quantization: companies vs users

Accessibility challenges of foundation models

Using foundation models

Finetuning foundation models

Quantization: companies vs users

Challenges of accessible use of foundation models

Reduced memory footprint

Challenges of accessible use of foundation models

Reduced memory footprint

Maintain prediction/generation quality

Challenges of accessible use of foundation models

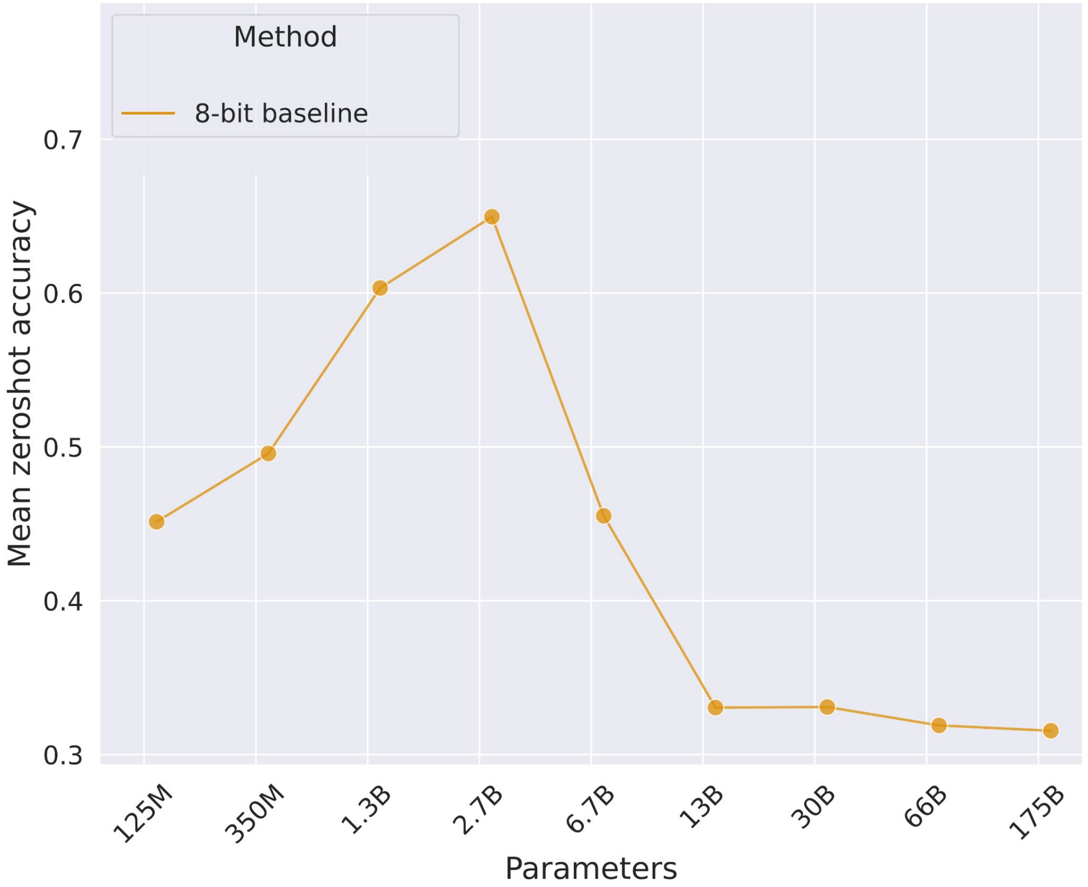
Reduced memory footprint

Maintain prediction/generation quality

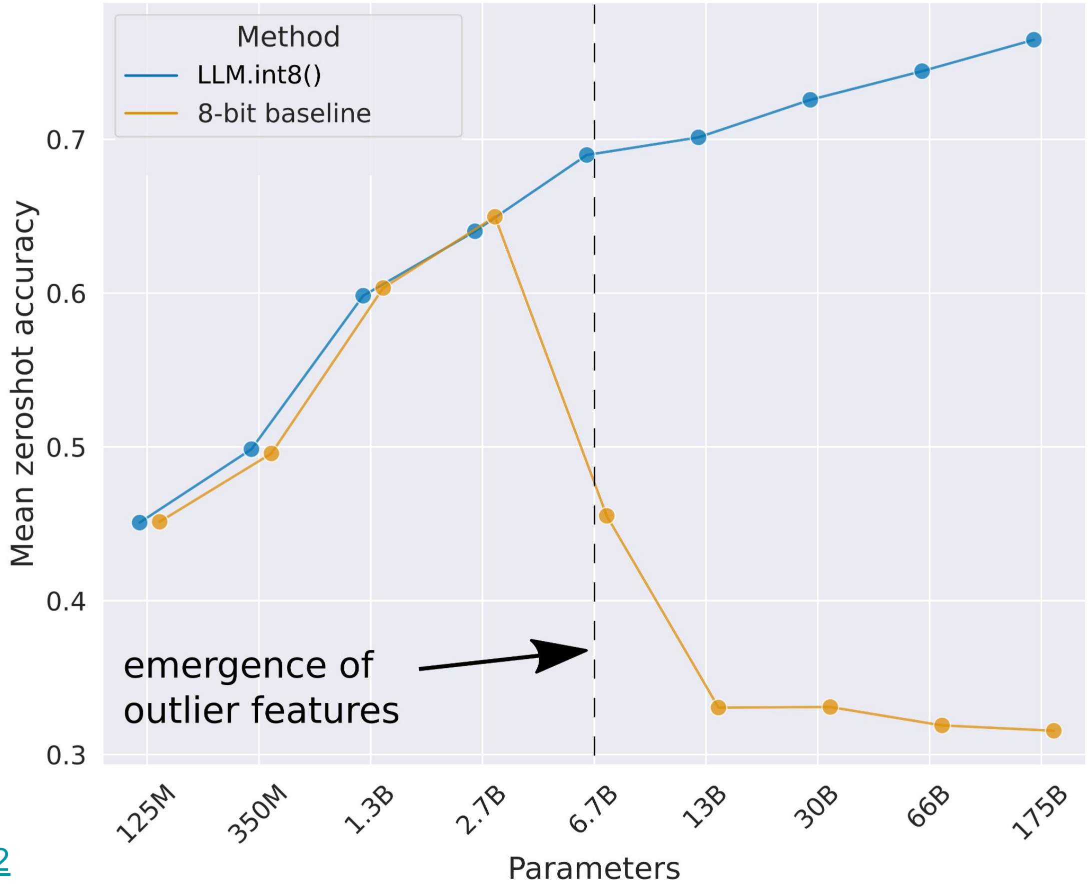
Generation / prediction speed

Compress 16-bit foundation models to 8 bit and 4 bit

8-bit Foundation Models Fail at Scale

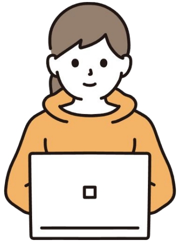


Our LLM.int8() method is the first method that works at scale



Dettmers et al., 2022

Accessibility challenges of foundation models



Using foundation models

Finetuning foundation models

Quantization: companies vs users

Evolution of scale of protein models

AlphaFold

21 Million Parameters

July 2021

ESM-1

650 Million Parameters

November 2021

Evolution of scale of protein models

AlphaFold

21 Million Parameters

July 2021

ESM-1

650 Million Parameters

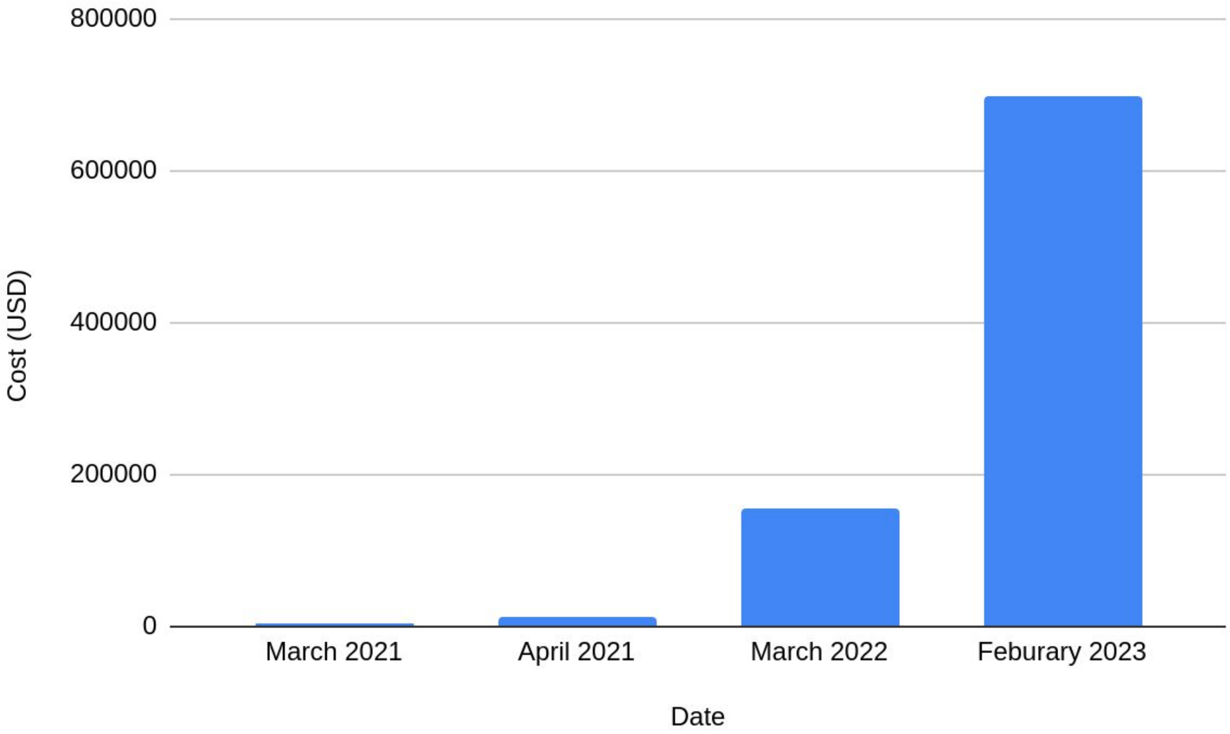
November 2021

ESM-2

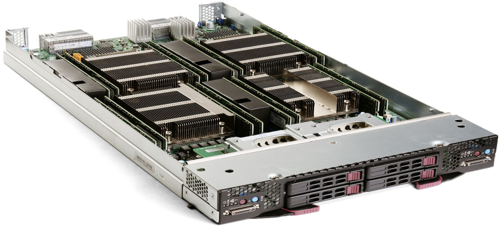
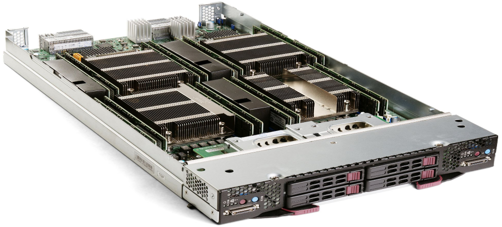
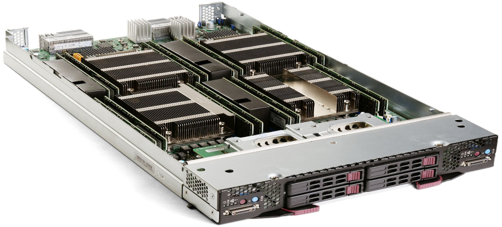
15 billion Parameters

August 2022

Finetuning is expensive due to GPU memory requirements



QLoRA: Finetuning large models on a single GPU.

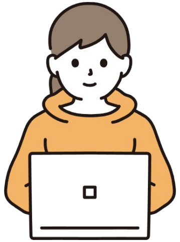
↓

QLoRA

(4-bit finetuning)

[Dettmers et al., 2023]

Accessibility challenges of foundation models



Using foundation models

Finetuning foundation models

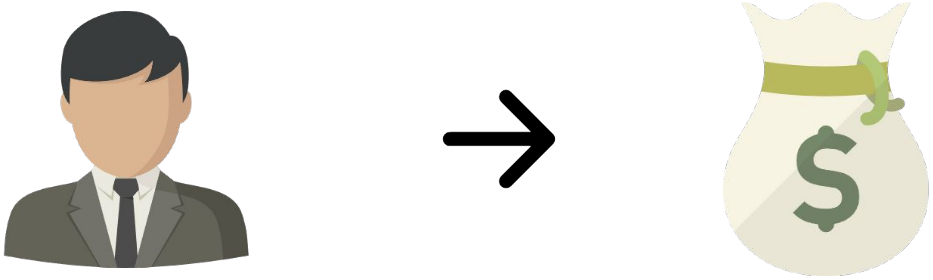
Quantization: companies vs users

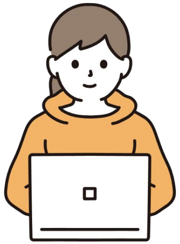
Quantization: companies vs users

Quantization: companies vs users

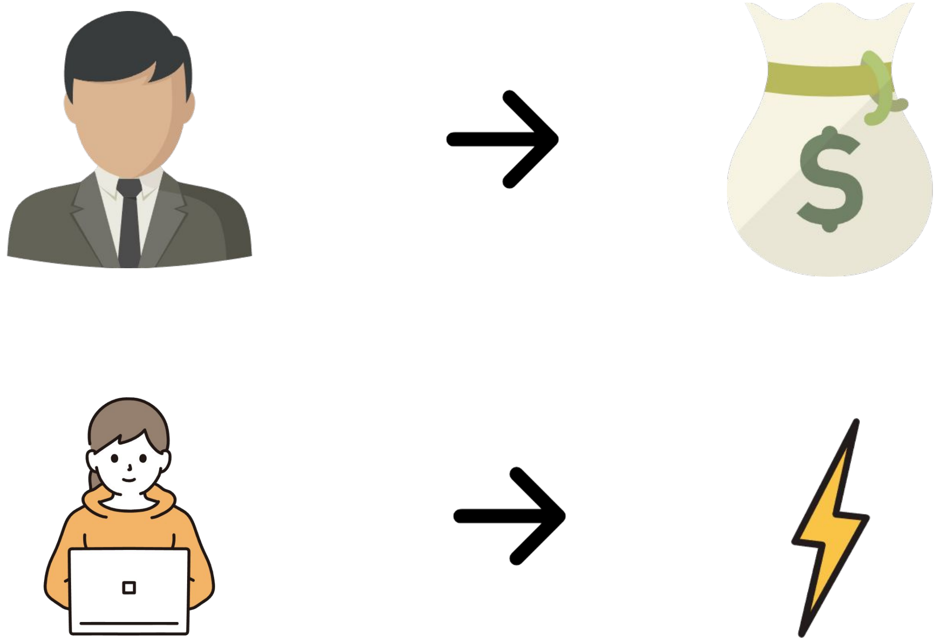
→

Quantization: companies vs users

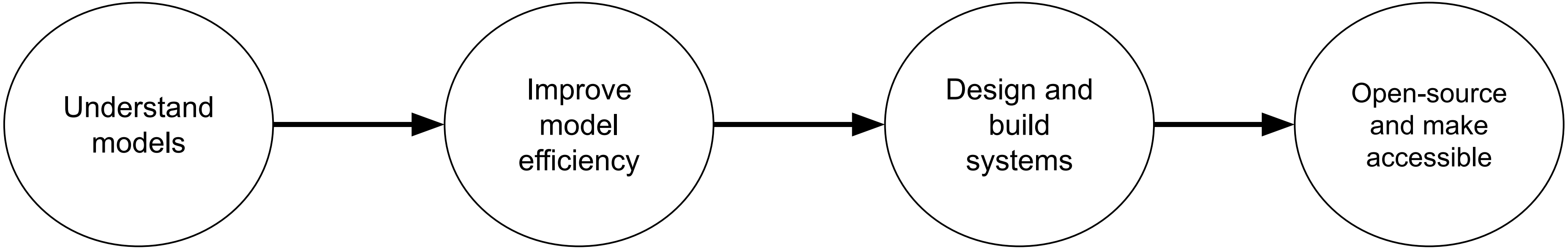




Quantization: companies vs users



Four steps to making foundation models accessible



The bitsandbytes library implements all my research algorithms.

One of the most popular machine learning libraries, growing at 1.7M installations per month.

Widely used in industry.

Usage of bitsandbytes outside of computer science after 2 years

Clinical research: Veen et al., 2023; Nerella et al., 2023; Shoham & Rappoport 2023; Liu et al., 2023; Gosh et al., 2024; Fan et al., 2024; Han et al., 2023; Yang et al., 2023; Schlegel et al., 2023; An et al., 2023

Biomedical: Ateia et al., 2023; Wang et al., 2023; Li et al., 2023; Wang et al., 2023; Delmas et al., 2023; Robinson et al., 2023; Ateia et al., 2023; Hong et al., 2023; Amara et al., 2023; Fries et al., 2022; He et al., 2023;

Humanities: Fok et al., 2023; Kuzman et al., 2023; Han et al., 2023, Deng et al., 2024;

Education: Zeilikman et al., 2023; Sonkar et al., 2023;

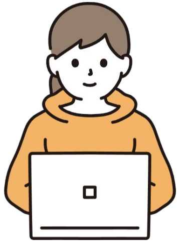
Political science: Linegar et al., 2023; He et al., 2023; Bornheim et al., 2023; Gesnouin et al., 2024; Allaham et al., 2024

Social science: Attanasio et al., 2023; Hu et al., 2023; Weld et al., 2024

Manufacturing: Freire et al., 2024; Zhang et al., 2024; Momodu 2023;

Other fields: Kraus et al., 2023; Hadi et al., 2023; Zelikman et al., 2023; Jiang 2023; Freudenberg 2023; Wang et al., 2023; Chu et al., 2024; Buehler et al., 2023; Saben & Chandrasekar, 2024;

Accessibility challenges of foundation models



Using foundation models

Finetuning foundation models

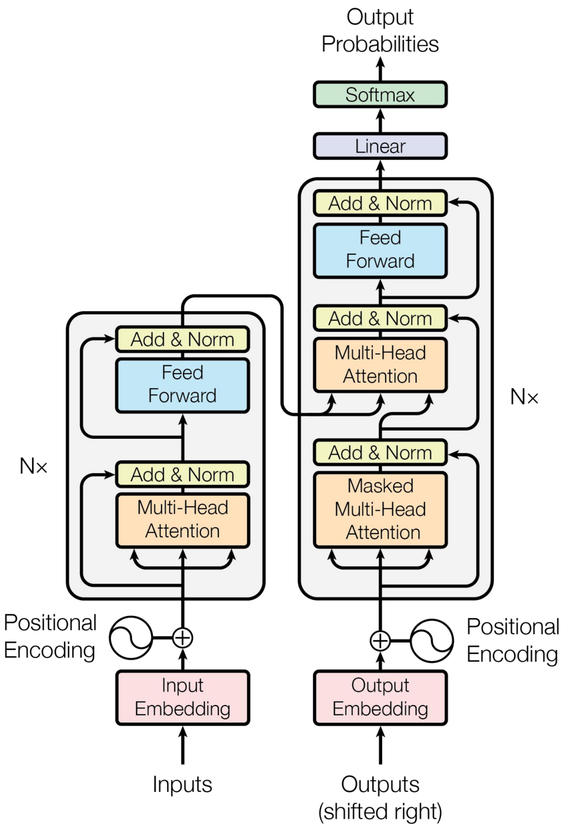
Quantization: companies vs users

Background

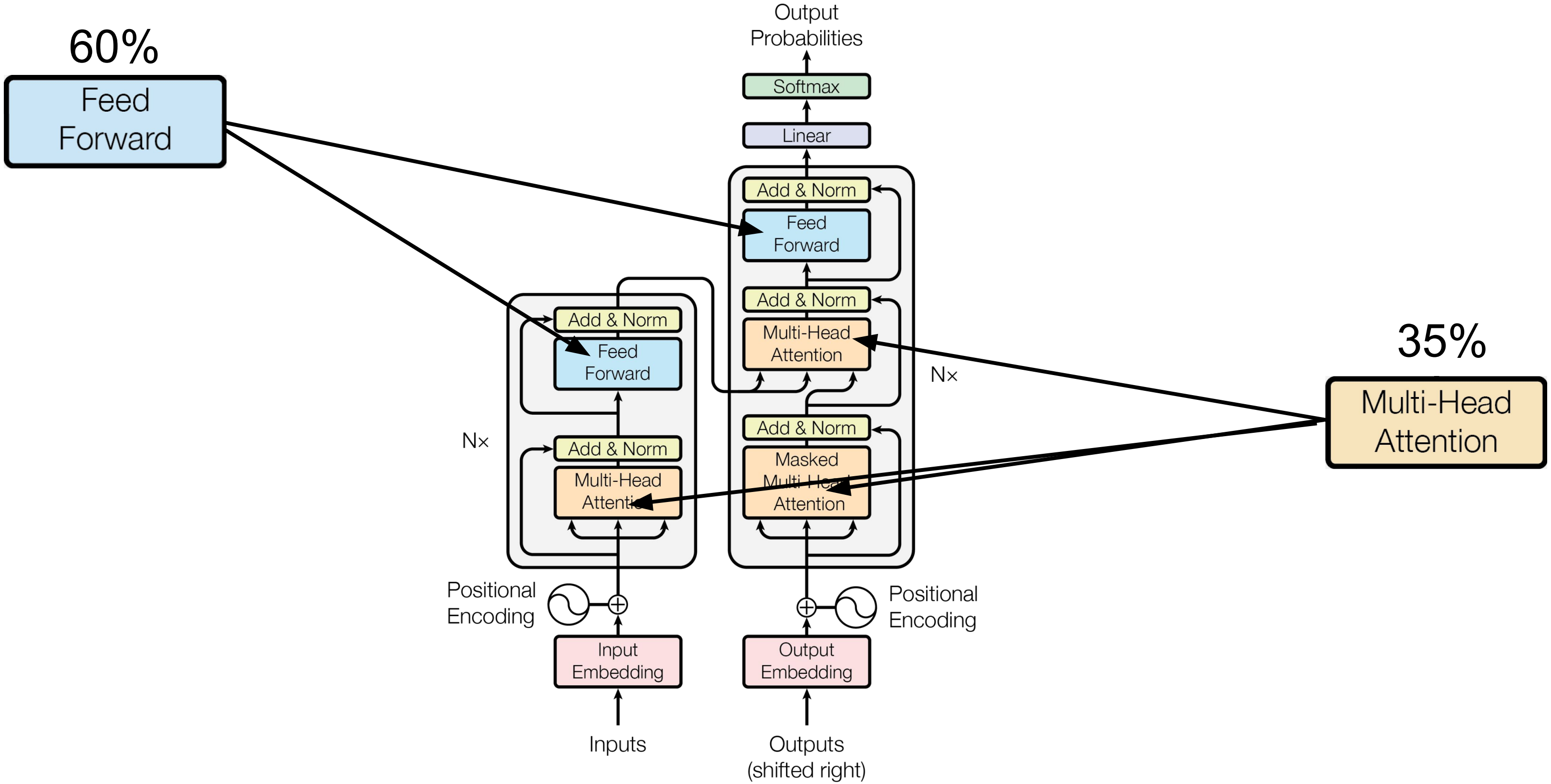
1. Resource use in neural networks
2. Neural networks
3. Quantization

1. Resource use in neural networks

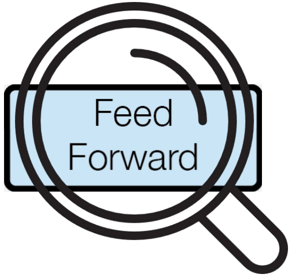
Transformers: The backbone of foundation models



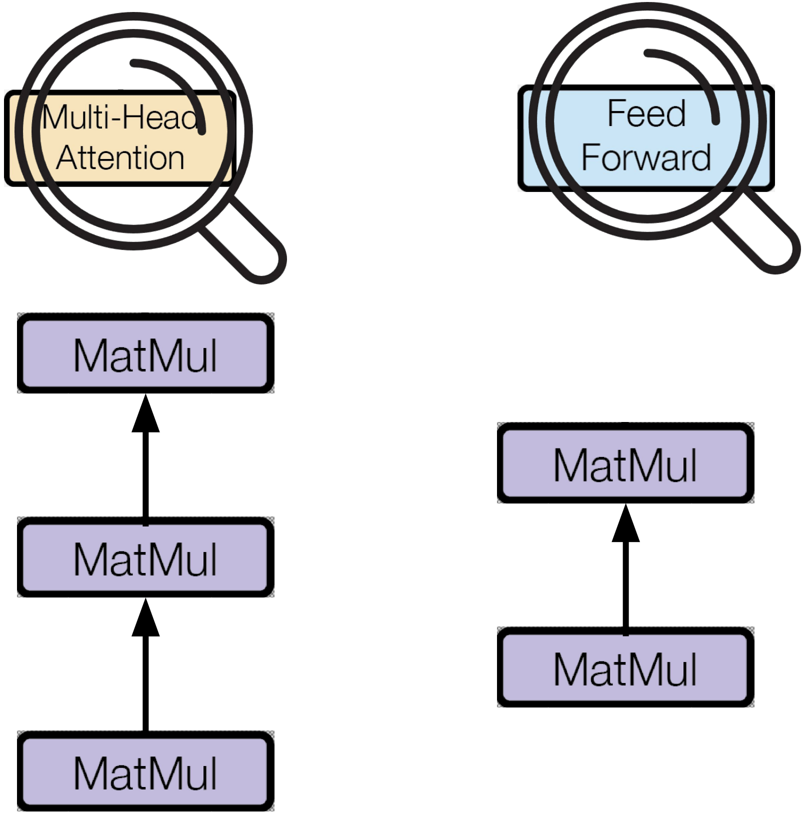
Transformers: The backbone of foundation models



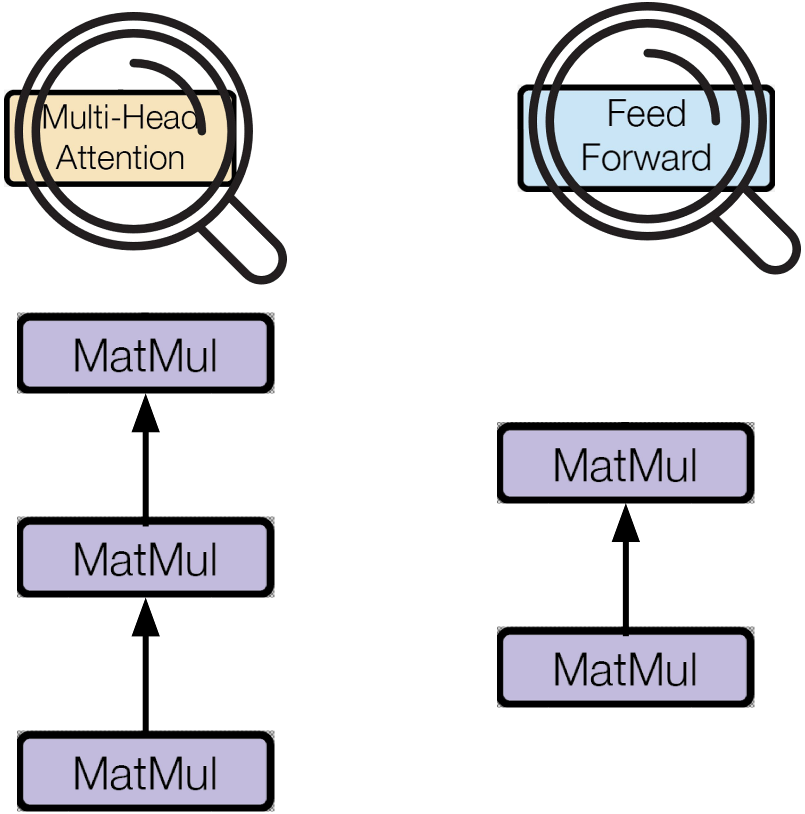
Transformers are mostly matrix multiplication



Transformers are mostly matrix multiplication



Transformers are mostly matrix multiplication



Matrix multiplication consumes:

- *95%* Memory
- *95%* Computation

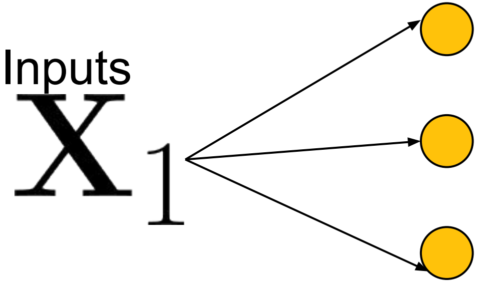
2. Neural networks

Background: neural networks. A sequence of layers.

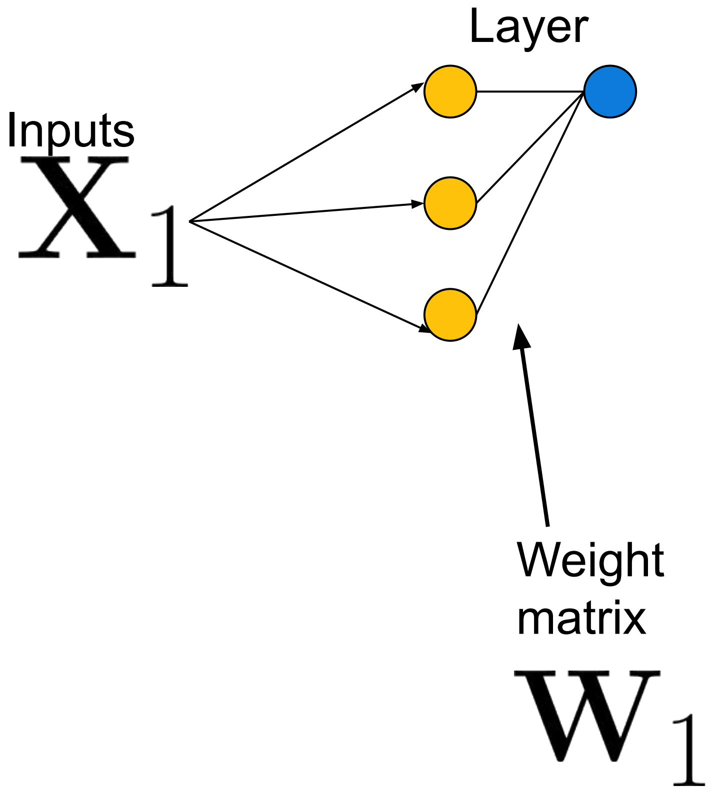
Inputs

$$\mathbf{x}_1$$

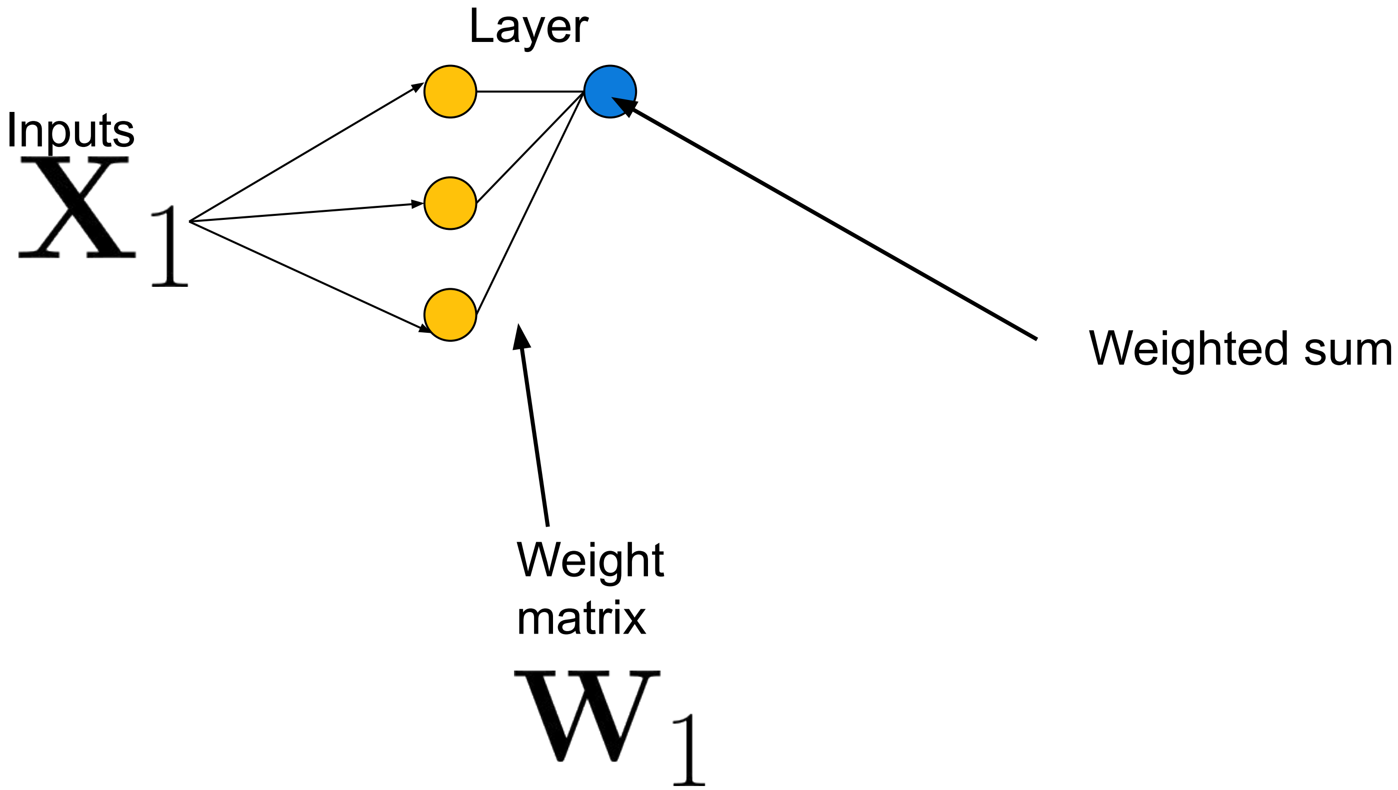
Background: neural networks. A sequence of layers.



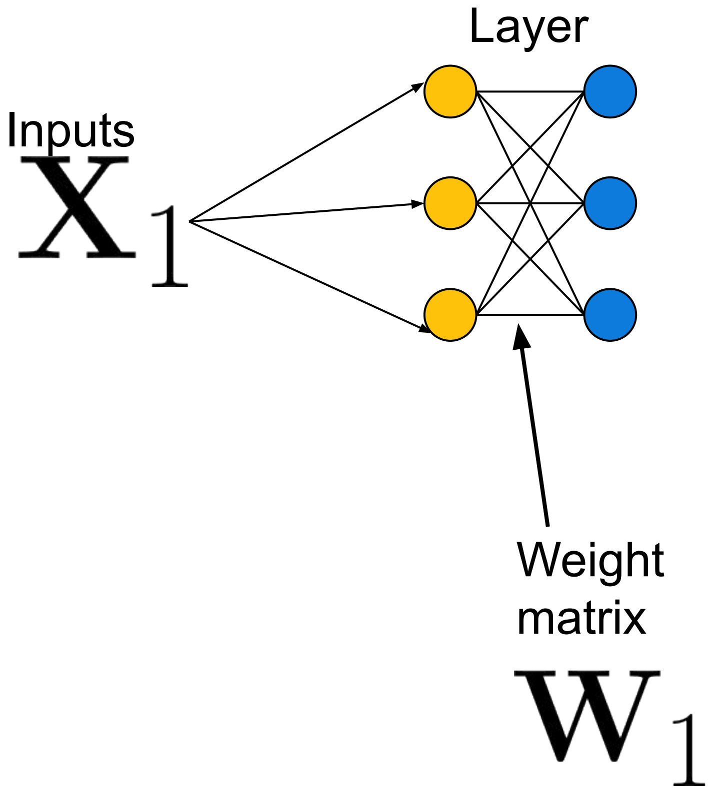
Background: neural networks. A sequence of layers.



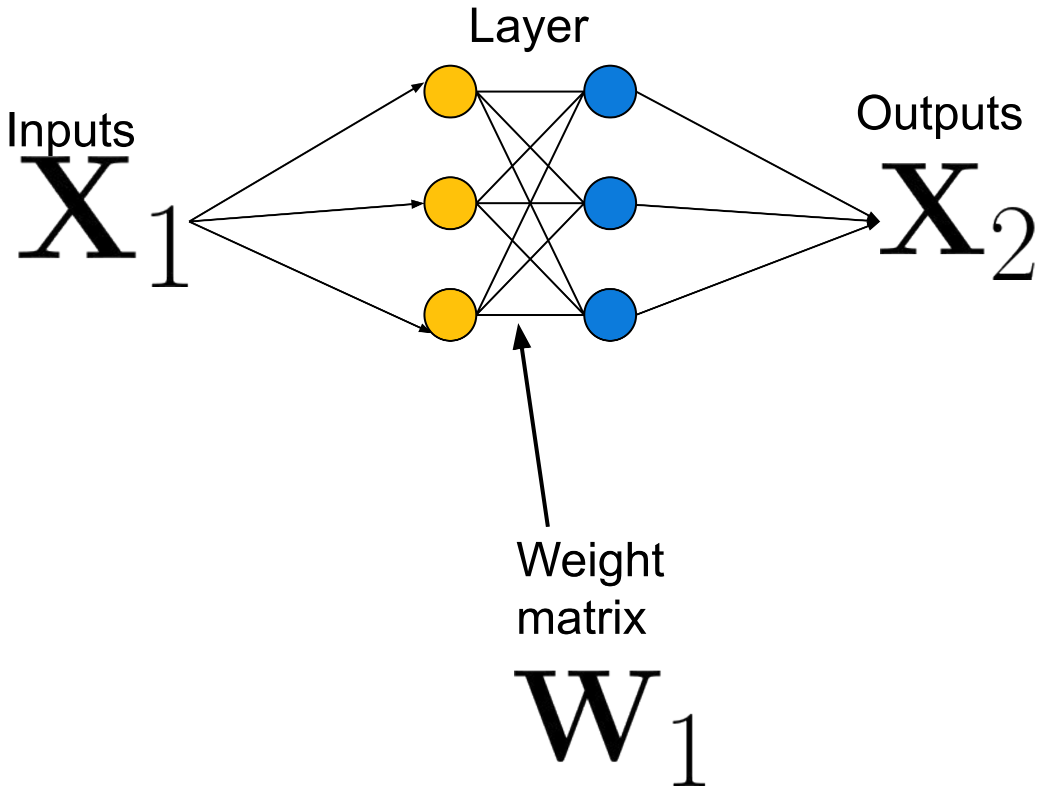
Background: neural networks. A sequence of layers.



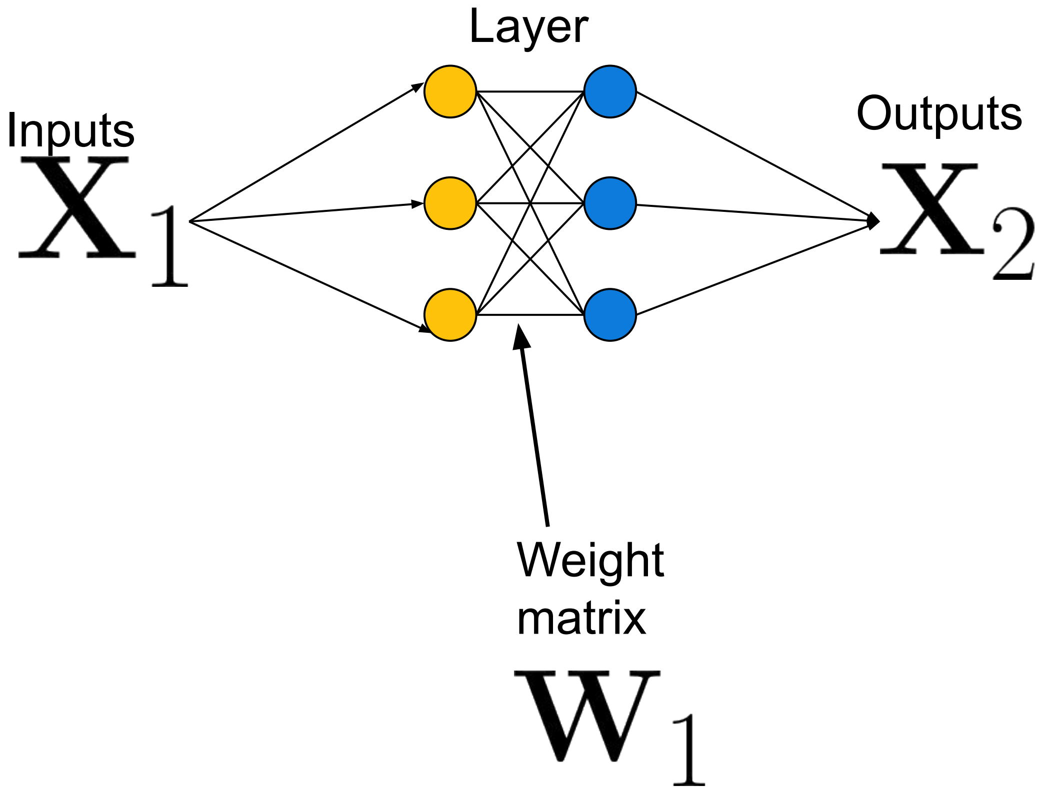
Background: neural networks. A sequence of layers.



Where are resources used in neural networks?

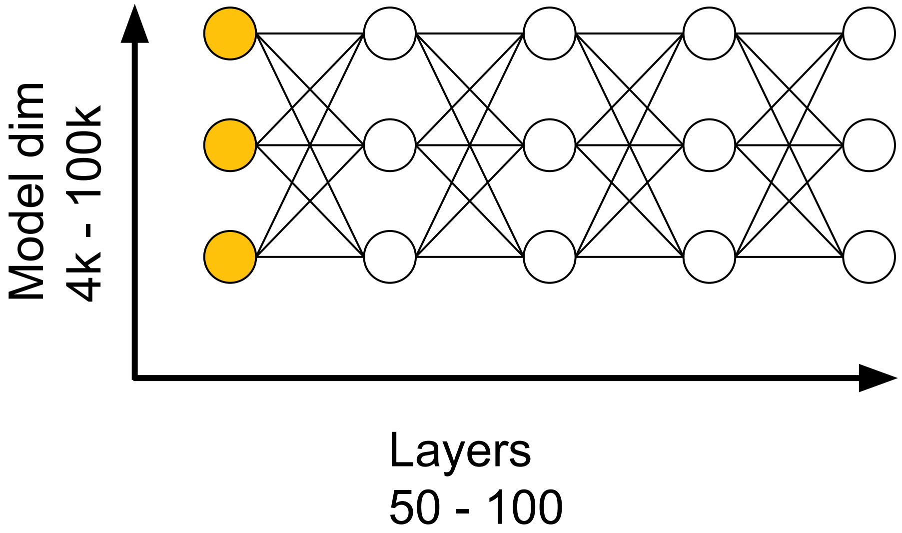


Background: neural networks. A sequence of layers.

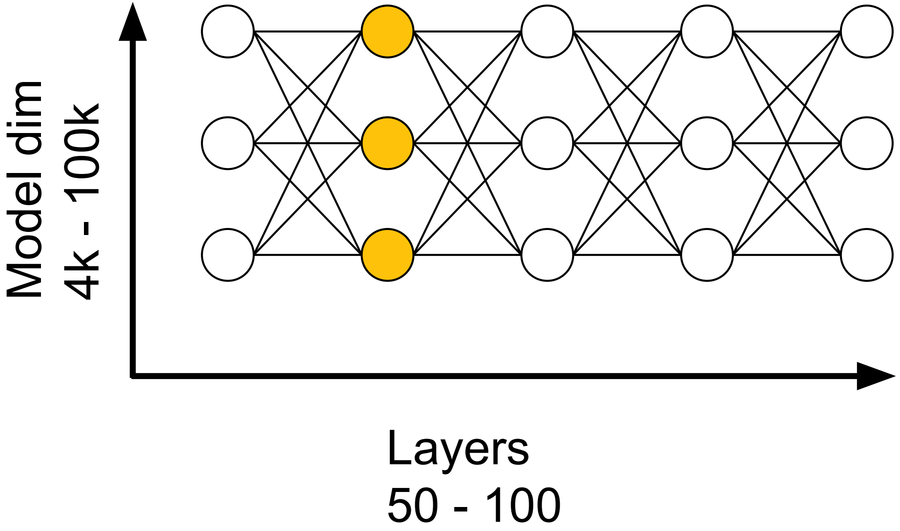


$$\mathbf{X}_1\mathbf{W}_1 = \mathbf{X}_2$$

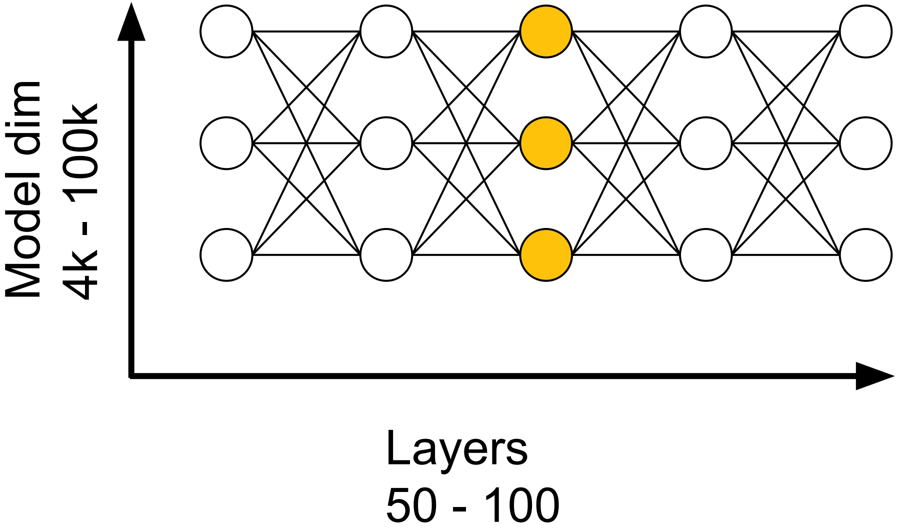
Background: neural networks. A sequence of layers.



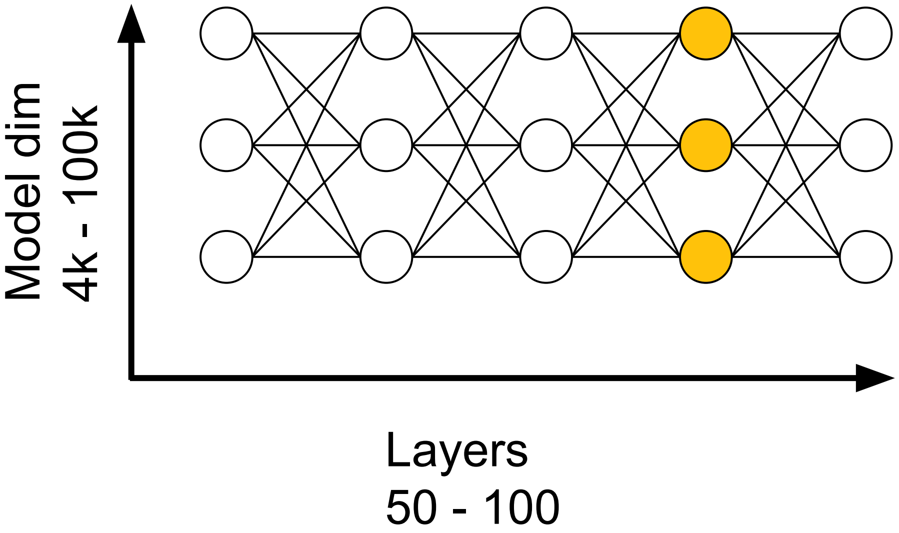
Background: neural networks. A sequence of layers.



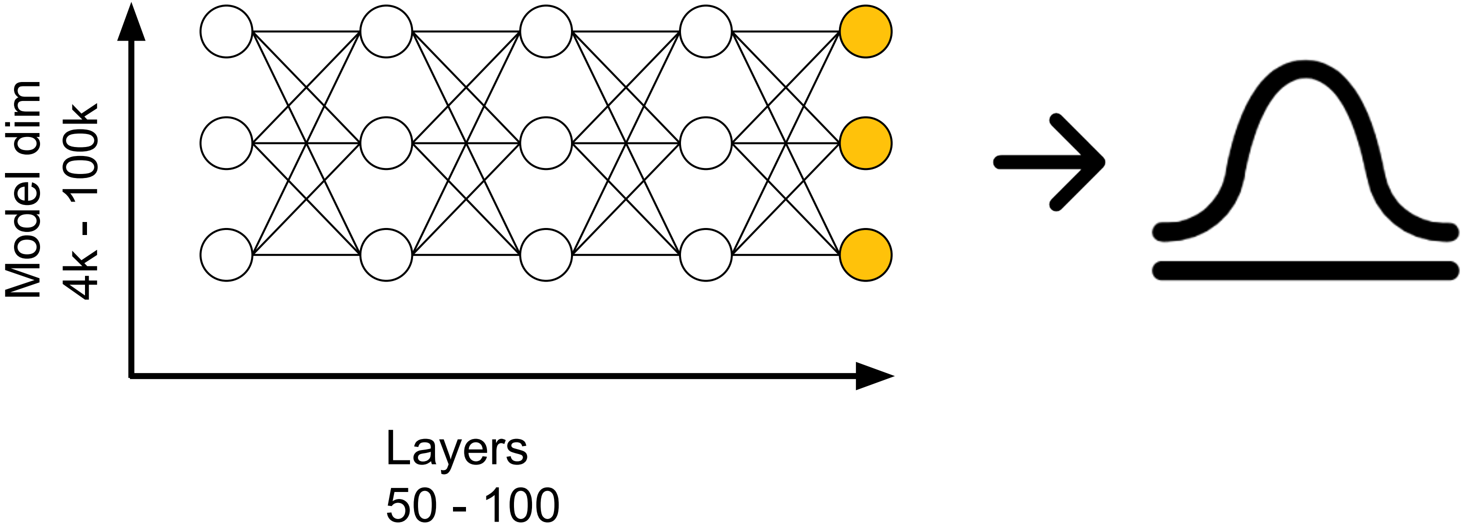
Background: neural networks. A sequence of layers.



Background: neural networks. A sequence of layers.



Background: neural networks. A sequence of layers.



Matrix multiplication is quite optimal in terms of hardware and software

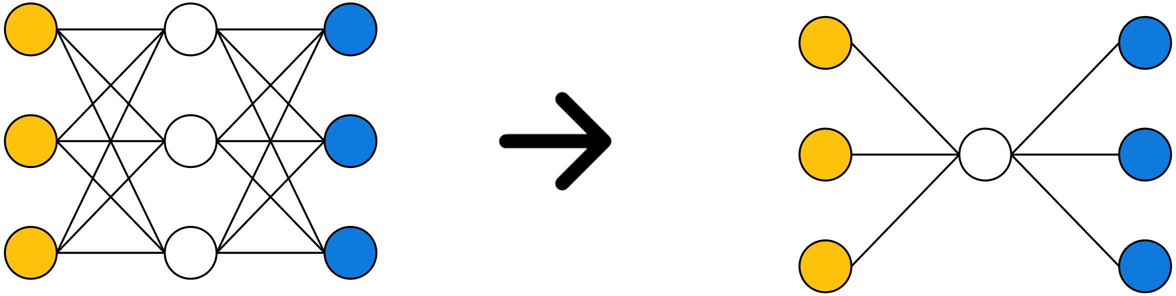
Matrix multiplication is quite optimal in terms of hardware and software

…

As such we need to find good approximations to gain efficiency

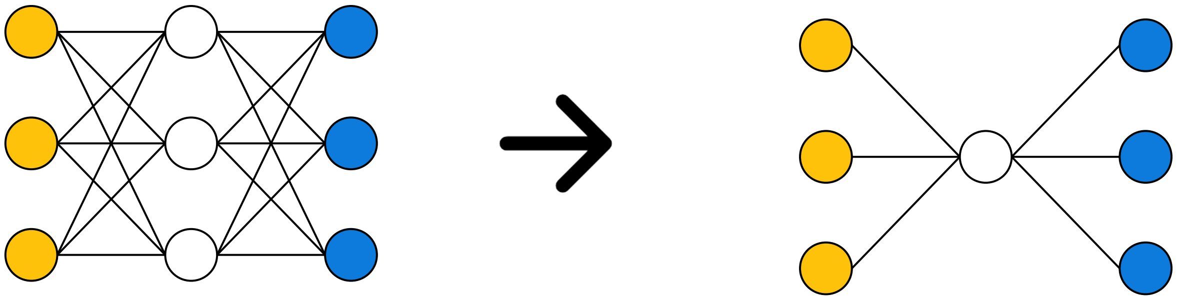
Approximations need to be faithful

Approximation
through
low-rank projection



Approximations need to be faithful

Approximation
through
low-rank projection

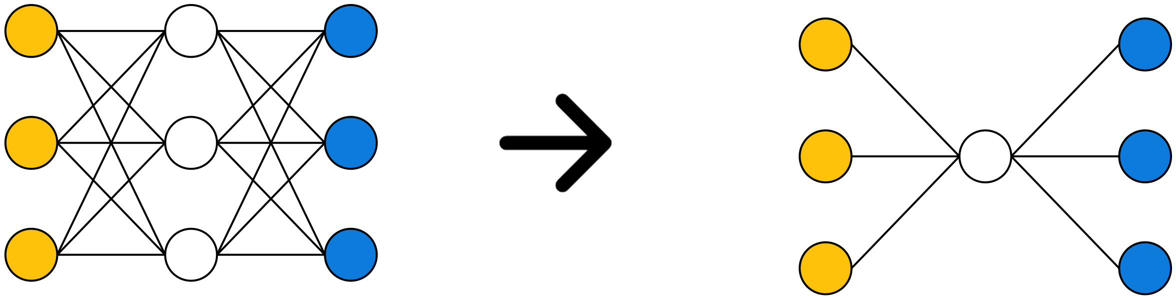


Speed ✅

Memory ✅

Approximations need to be faithful

Approximation
through
low-rank projection



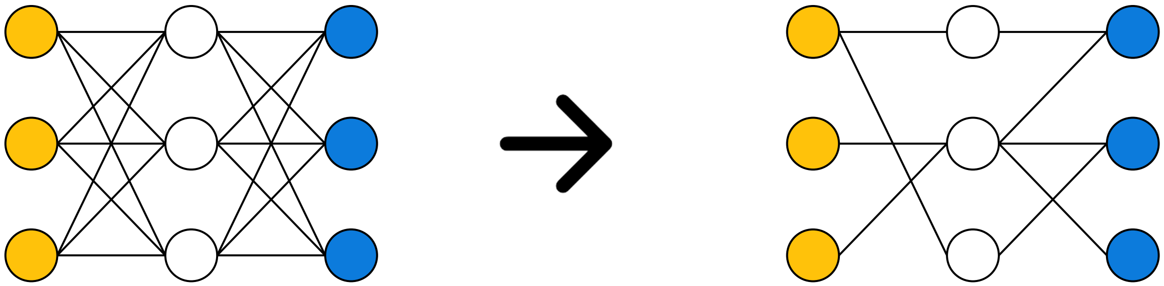
Speed ✅

Memory ✅

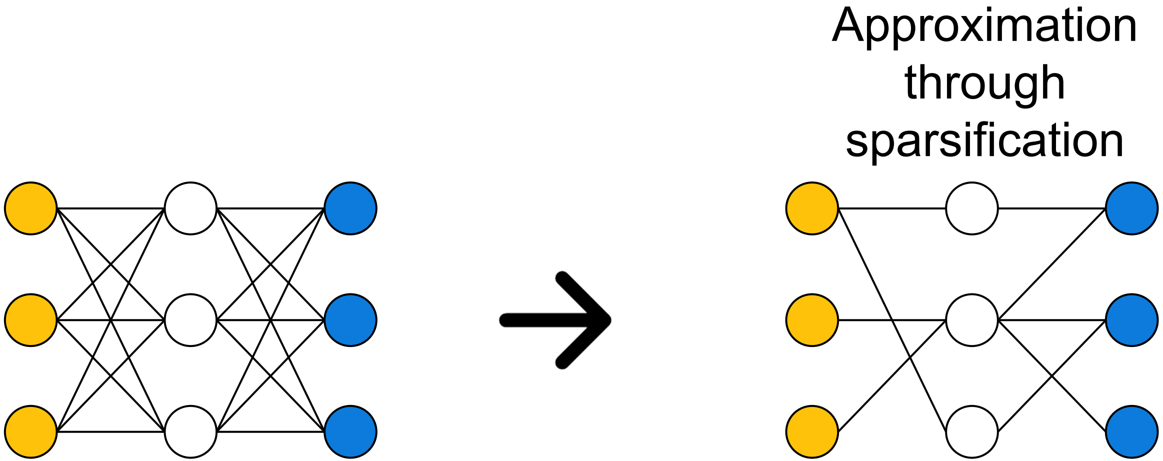
Quality ❌

Approximations need to be useful in practice

Approximation
through
sparsification



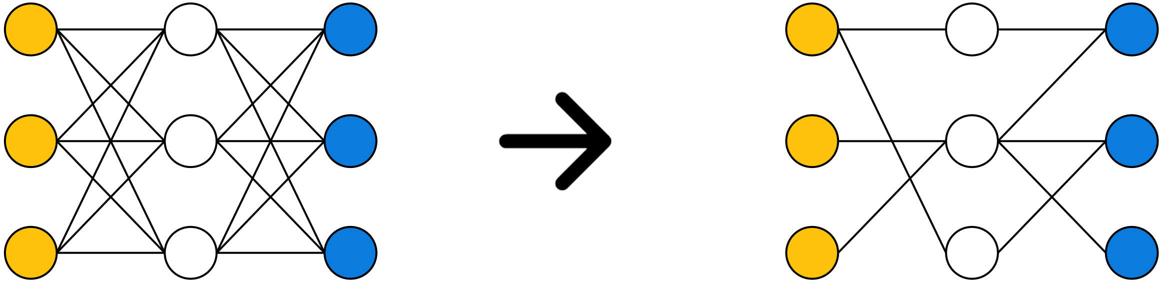
Approximations need to be useful in practice



Quality ✅

Approximations need to be useful in practice

Approximation through sparsification



Speed ❌

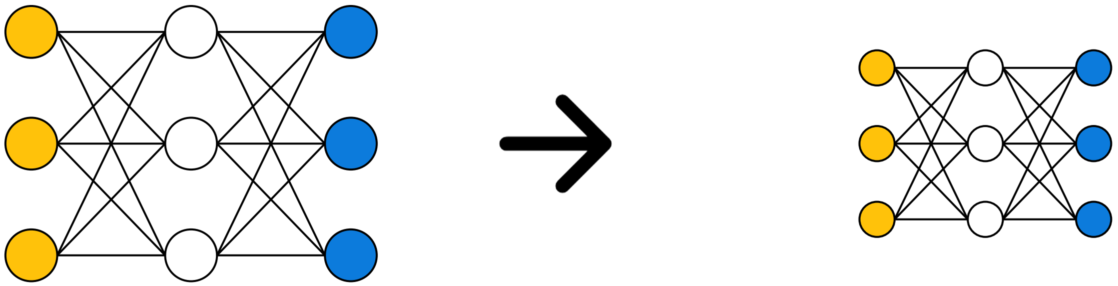
Memory ❌

Quality ✅

Quantization has challenges …

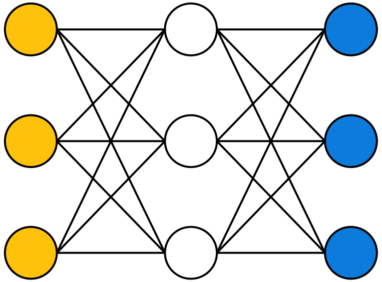
16-bit

Approximation
through
8-bit computation



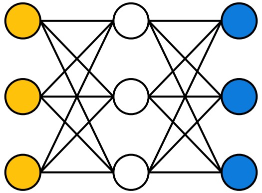
Quantization has challenges …

16-bit



→

Approximation
through
8-bit computation



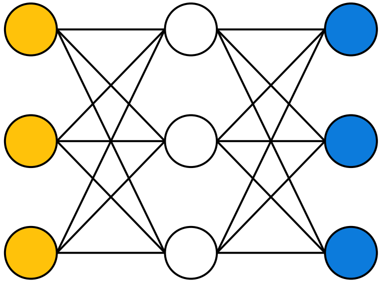
Speed ✅

Memory ✅

Quality ❓

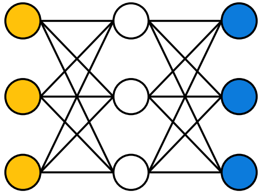
Quantization has challenges, but we can overcome them!

16-bit



→

Approximation
through
8-bit computation



Speed ✅

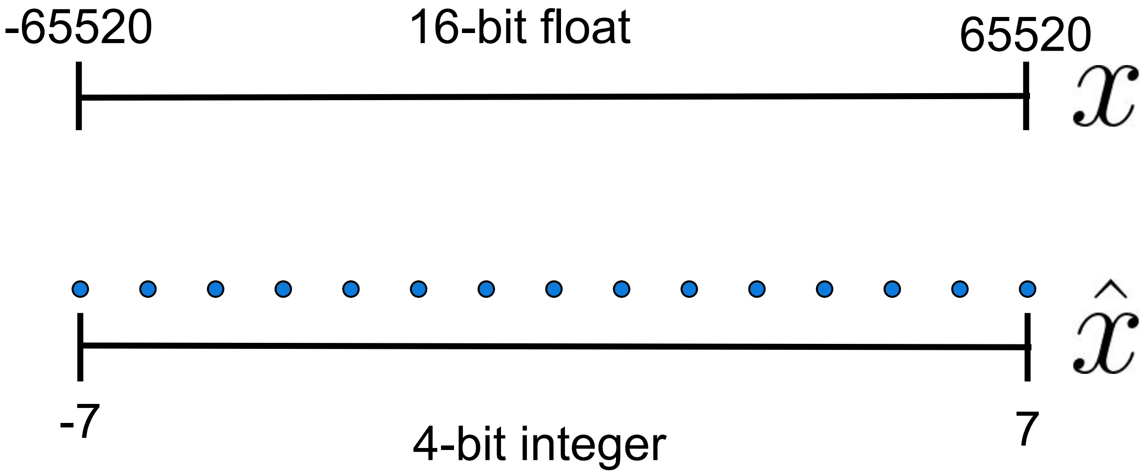
Memory ✅

Quality ✅

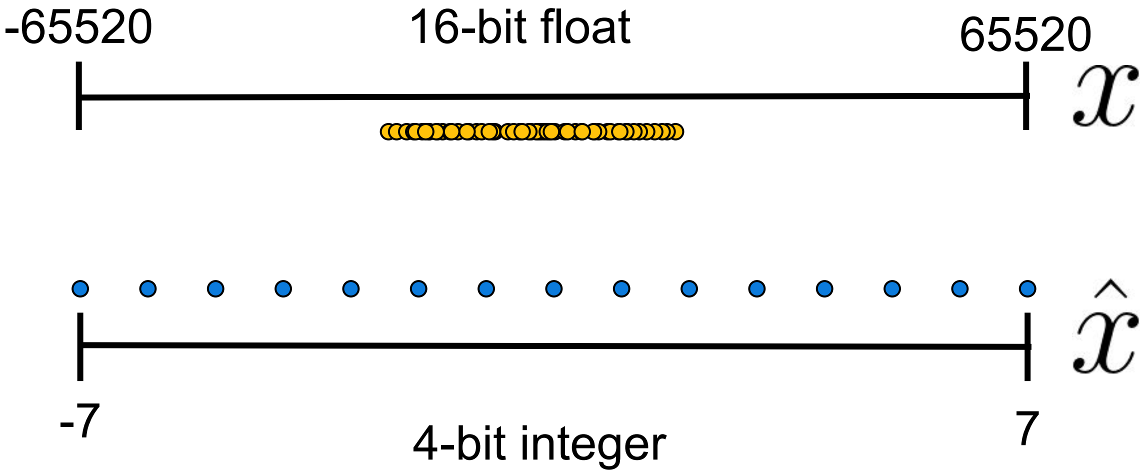
3. Quantization

Quantizing 16-bit normal distributed data to 4-bit integers

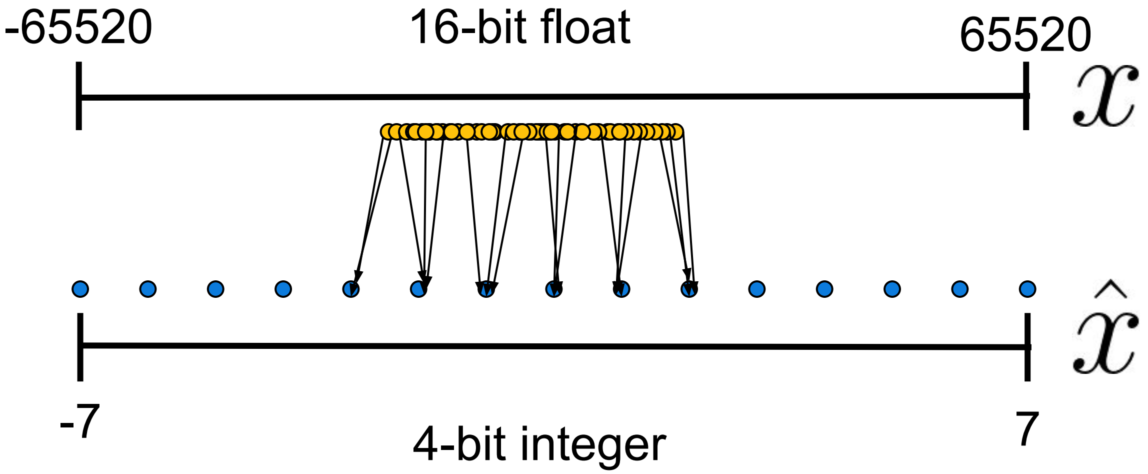
Quantizing 16-bit normal distributed data to 4-bit integers



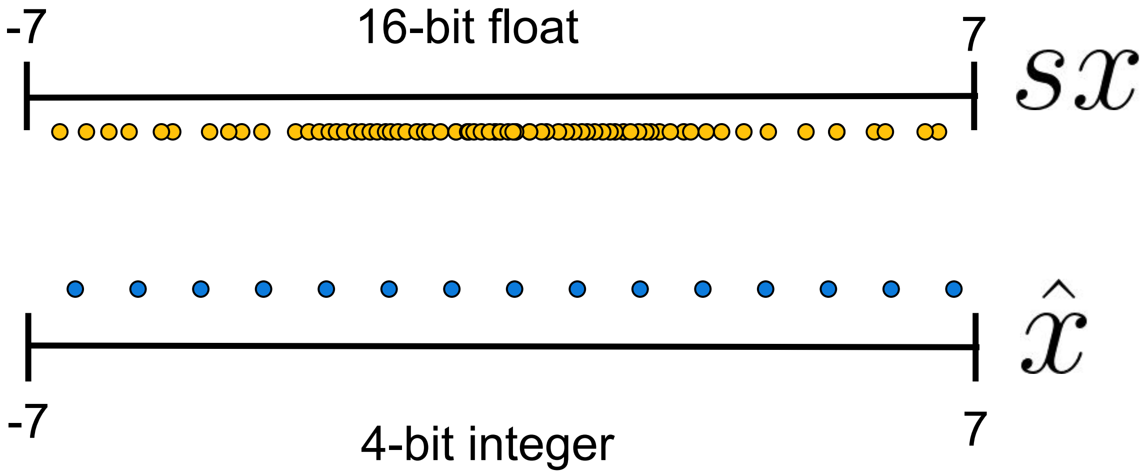
Quantizing 16-bit normal distributed data to 4-bit integers



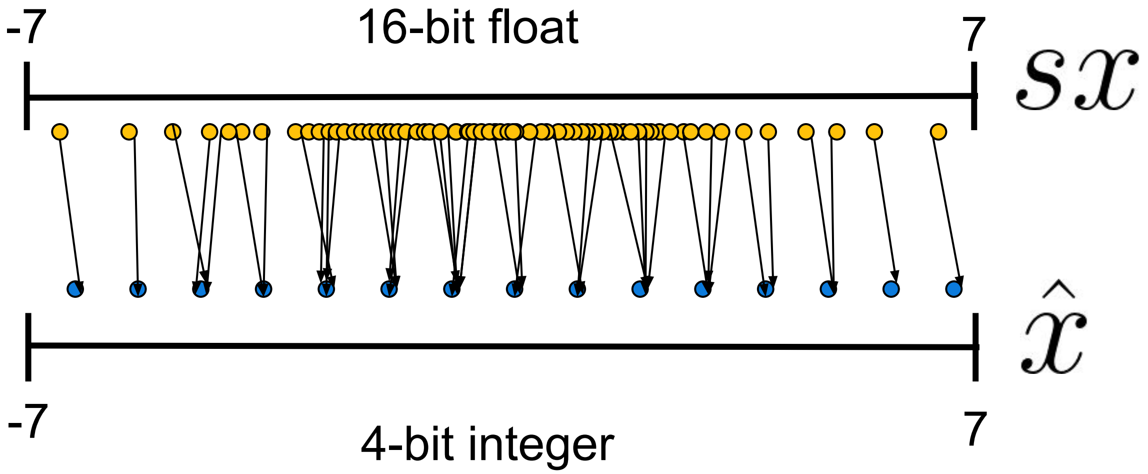
Quantizing 16-bit normal distributed data to 4-bit integers



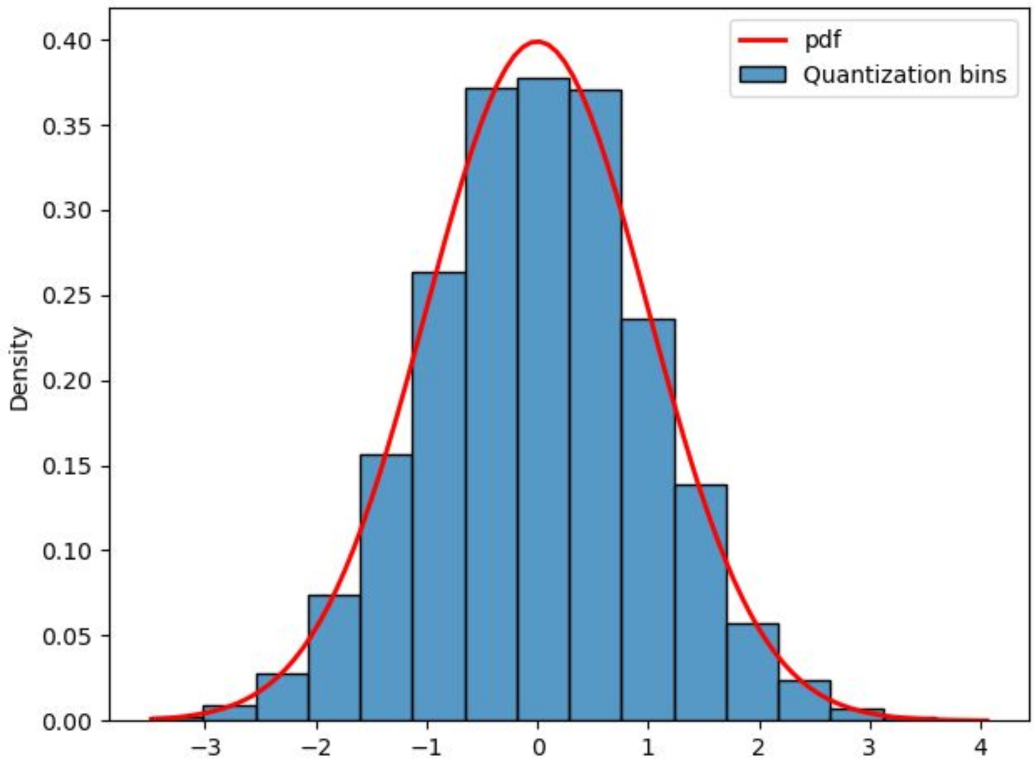
Quantizing 16-bit normal distributed data to 4-bit integers



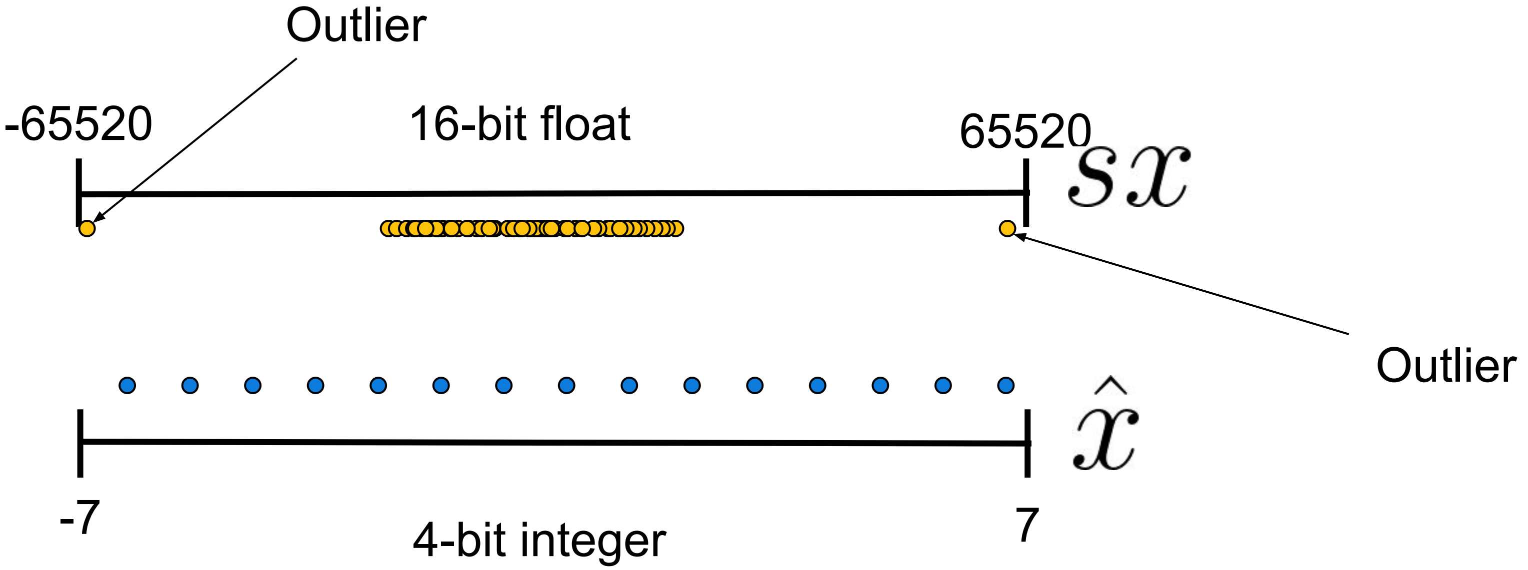
Quantizing 16-bit normal distributed data to 4-bit integers



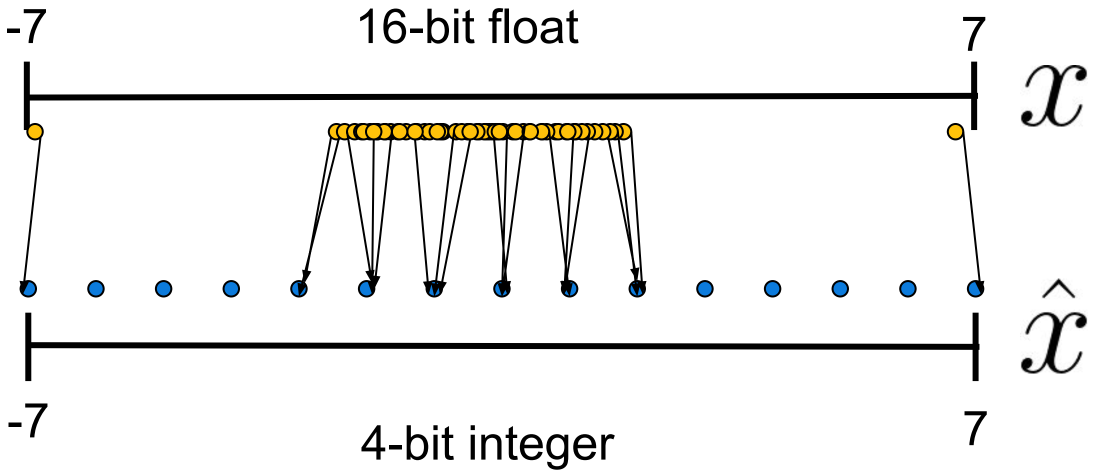
Integer quantization is similar to histogram binning



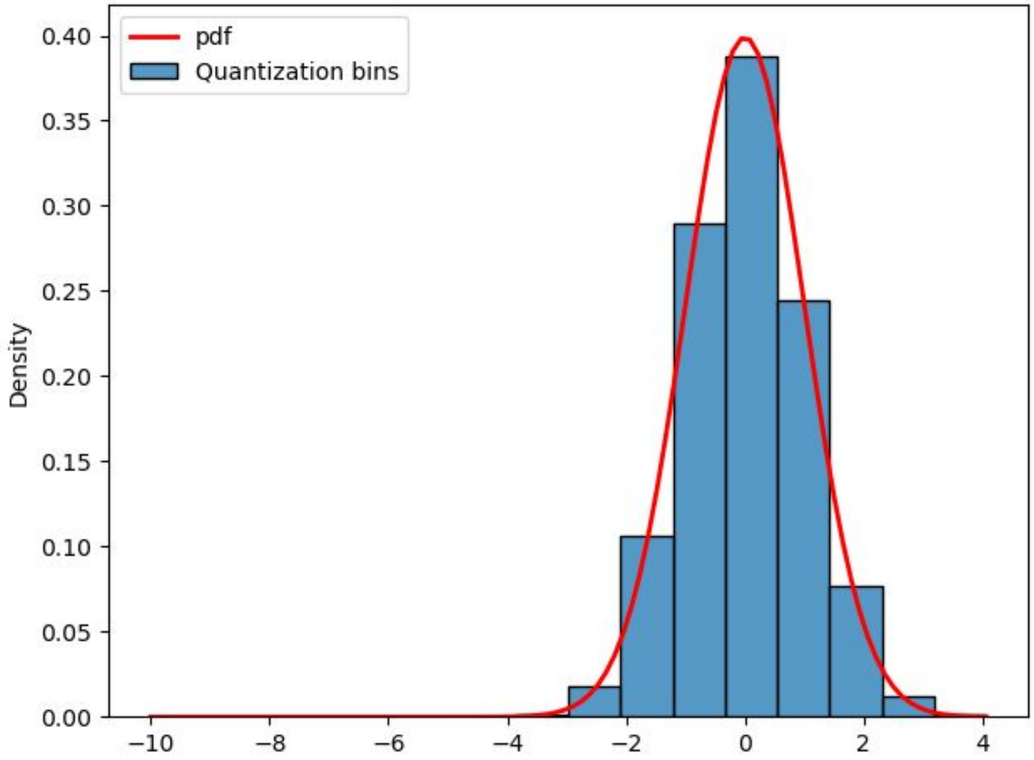
Quantizing 16-bit normal distributed data to 4-bit integers



Quantizing 16-bit normal distributed data to 4-bit integers



What do outliers in quantization look like?



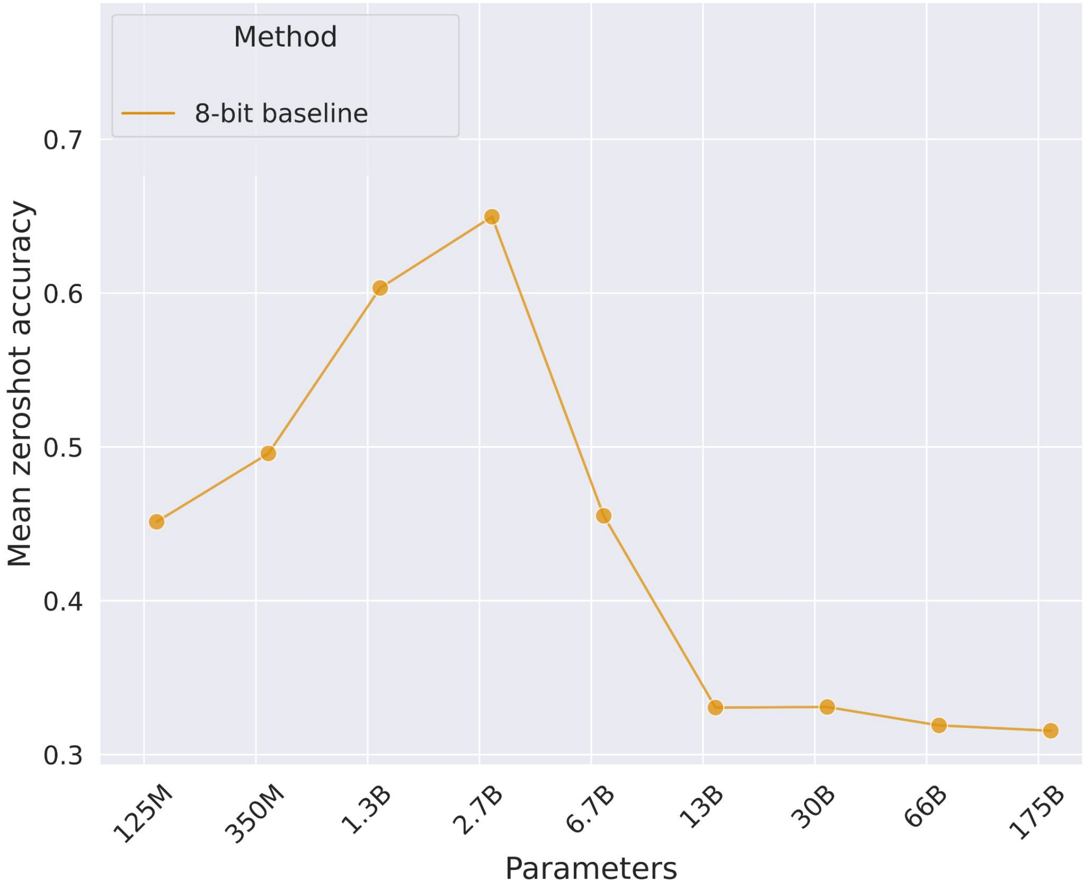
Accessibility challenges of foundation models

Using foundation models

Finetuning foundation models

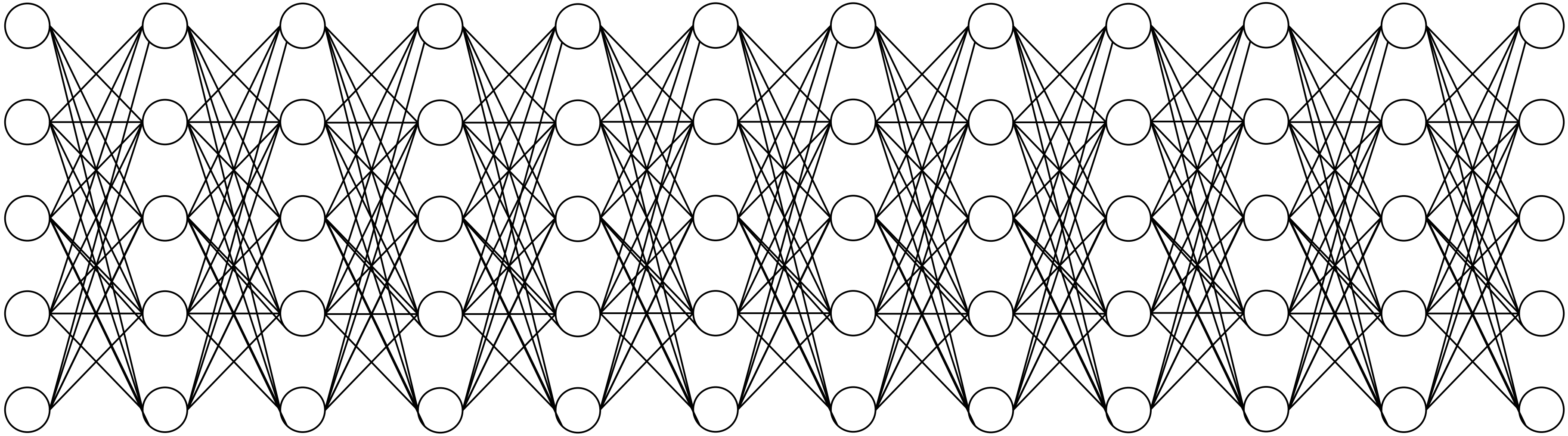
Quantization: companies vs users

8-bit Foundation Models Fail at Scale



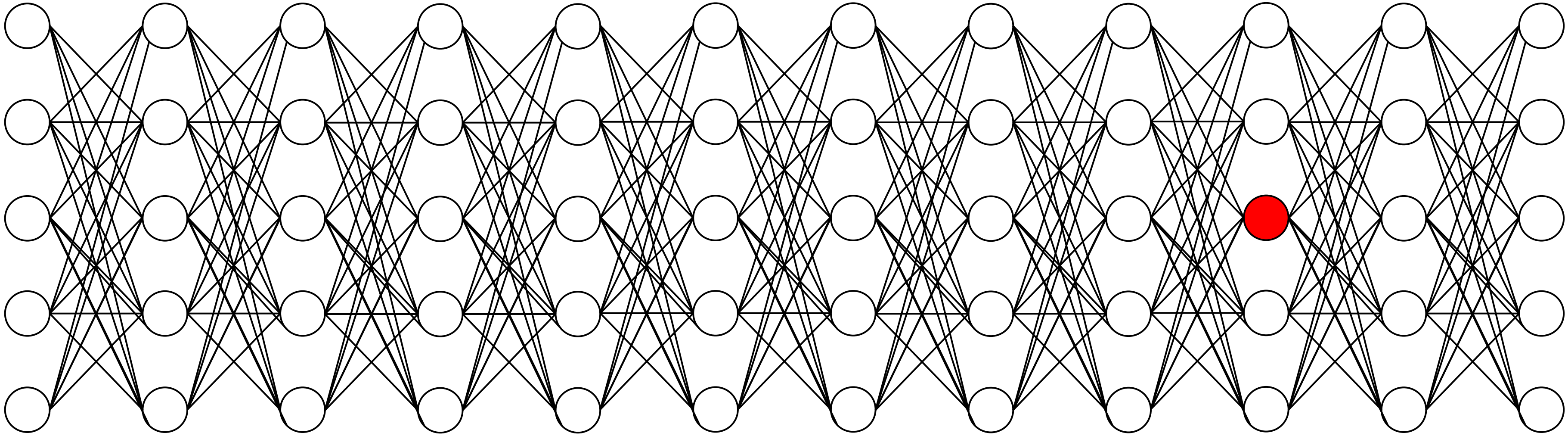
Outlier patterns in small neural networks (350M parameters)

Outlier at 6 sigma

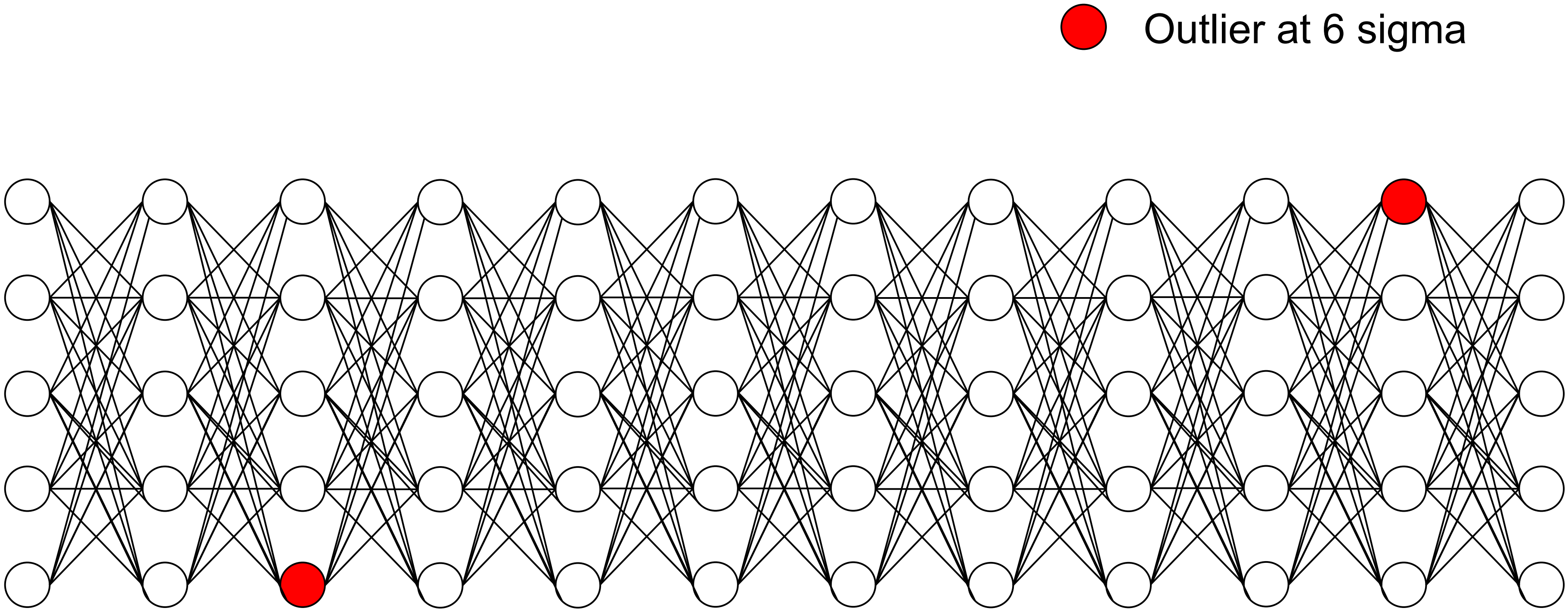


Outlier patterns in small neural networks (350M parameters)

Outlier at 6 sigma

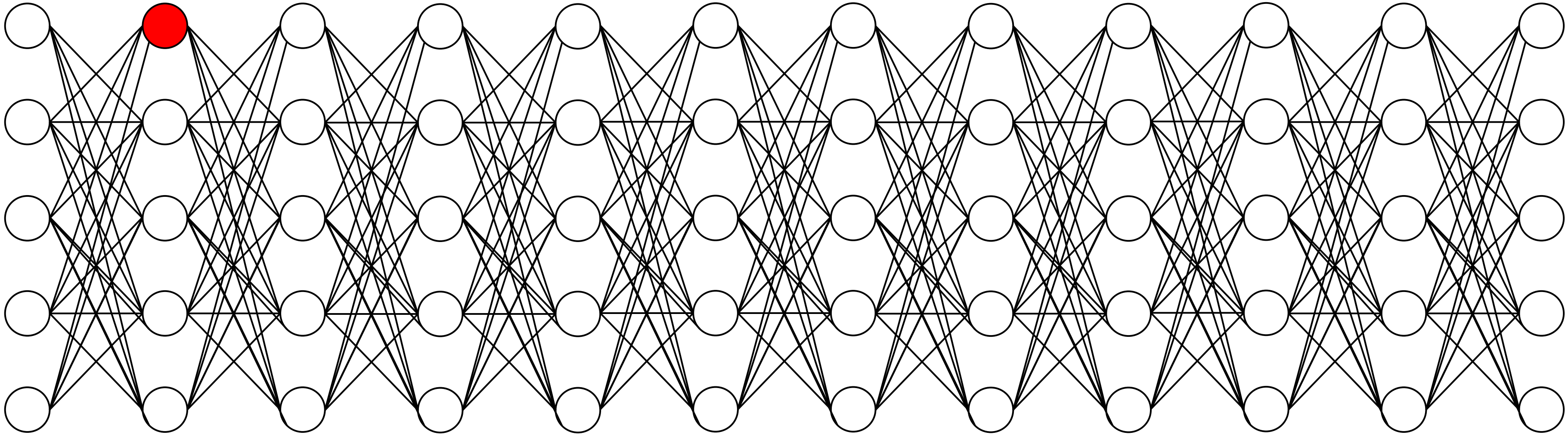


Outlier patterns in small neural networks (350M parameters)

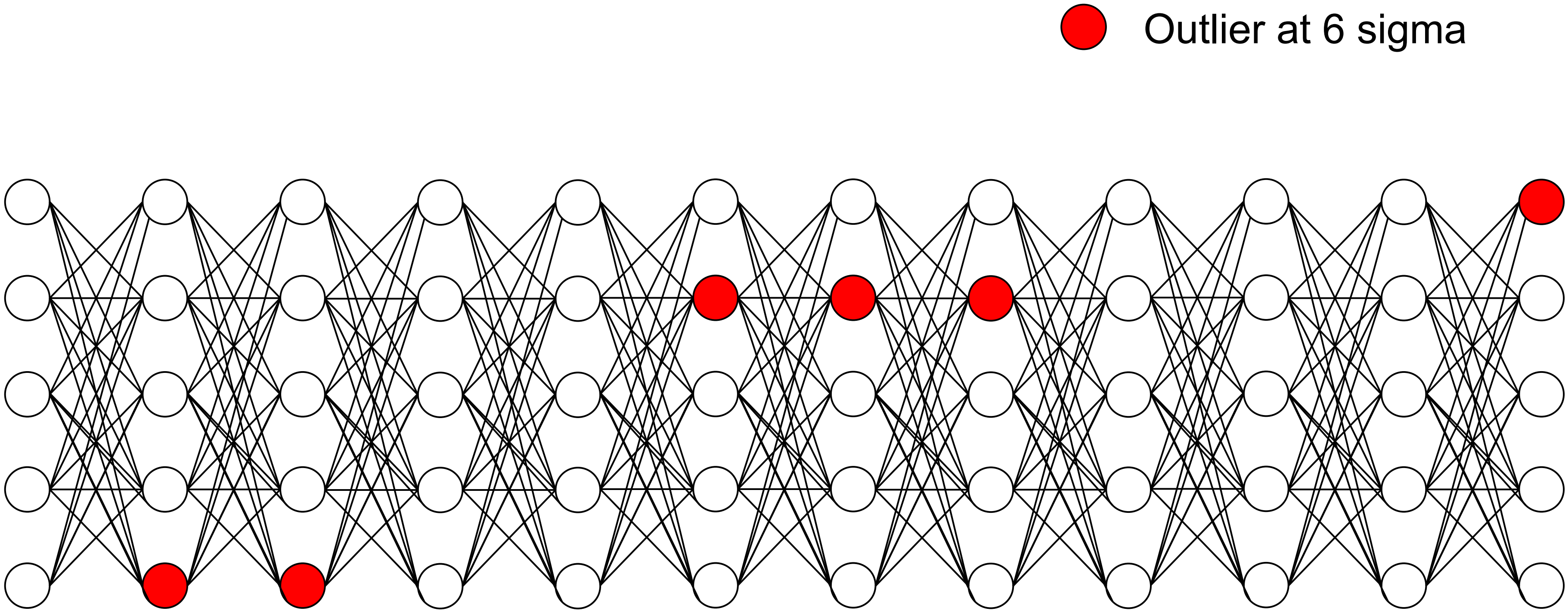


Outlier patterns in small neural networks (350M parameters)

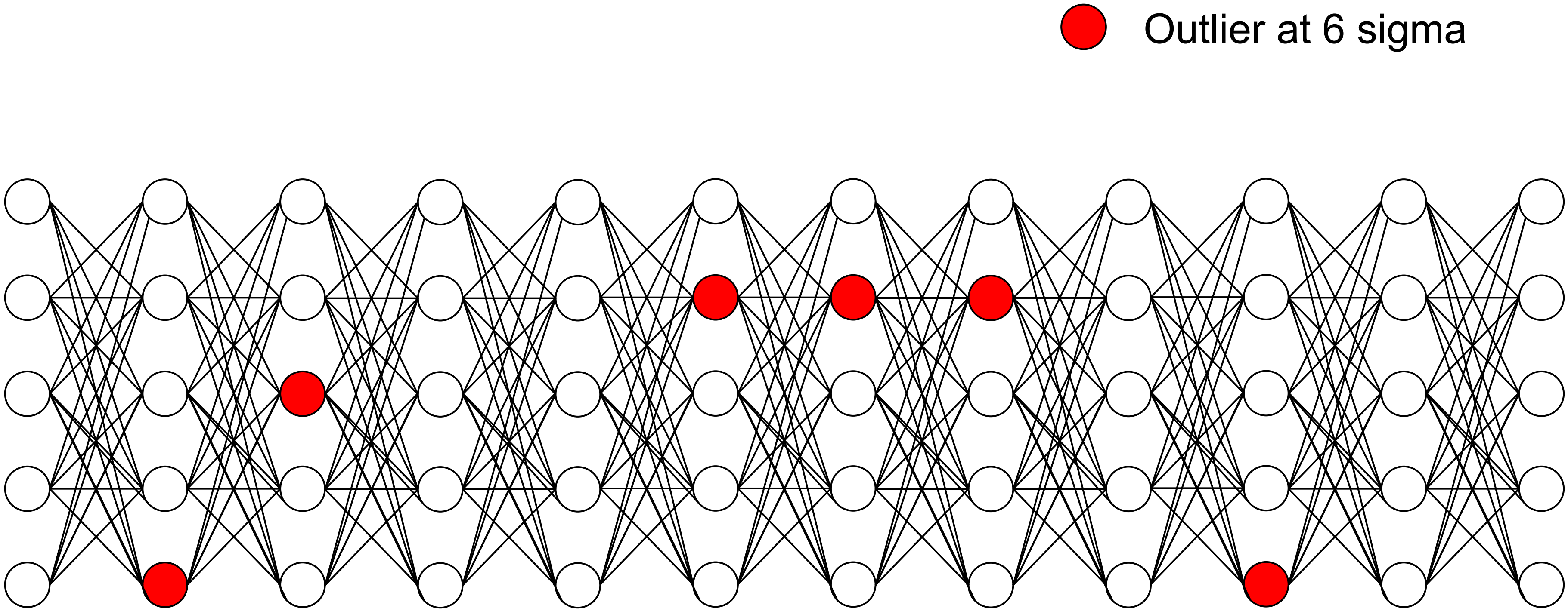
Outlier at 6 sigma



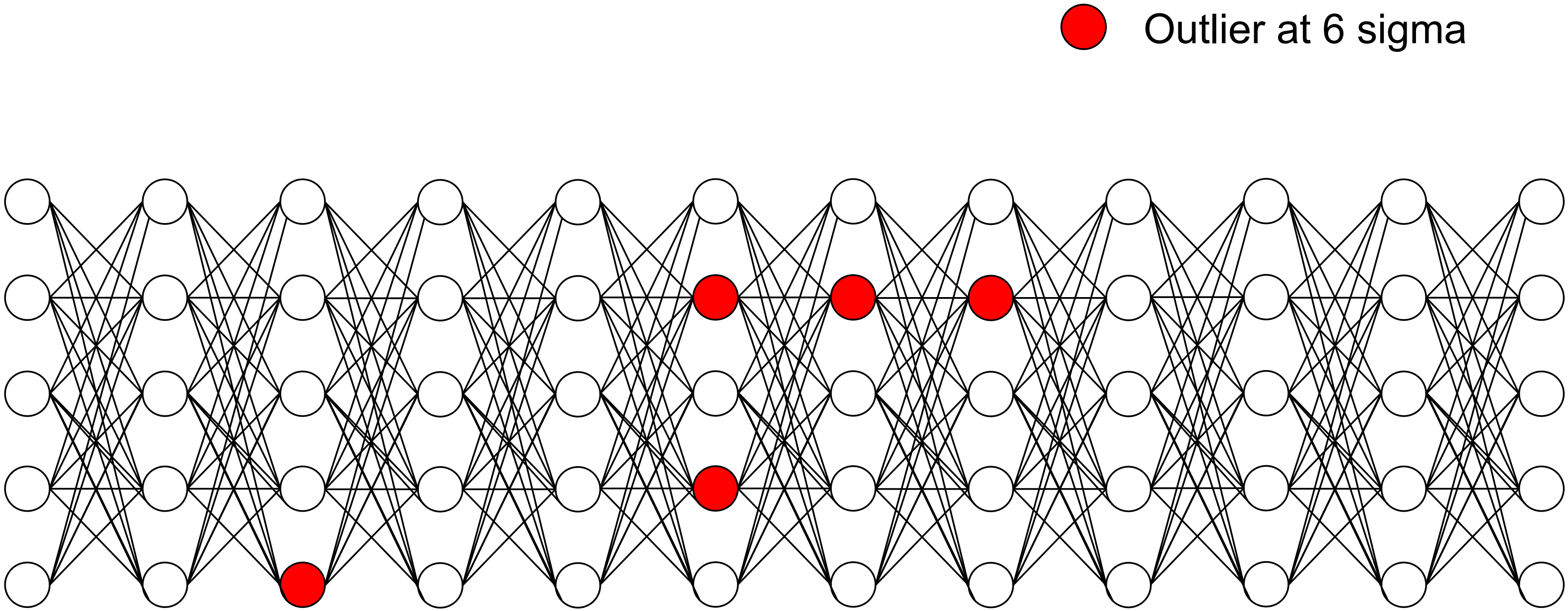
Outlier patterns in small neural networks (1.3B parameters)



Outlier patterns in small neural networks (1.3B parameters)

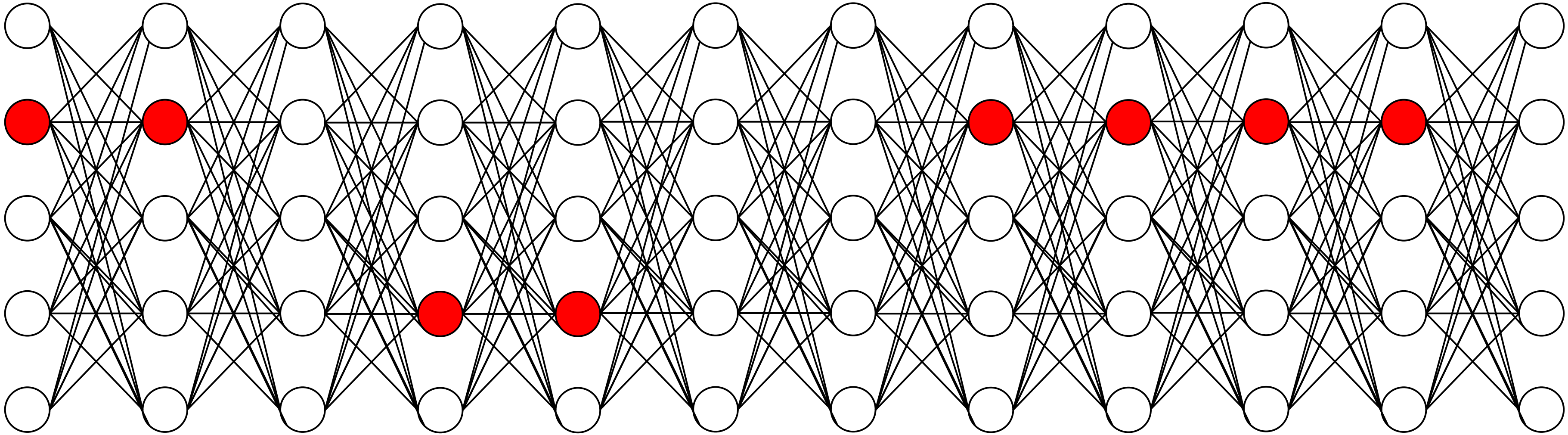


Outlier patterns in small neural networks (1.3B parameters)

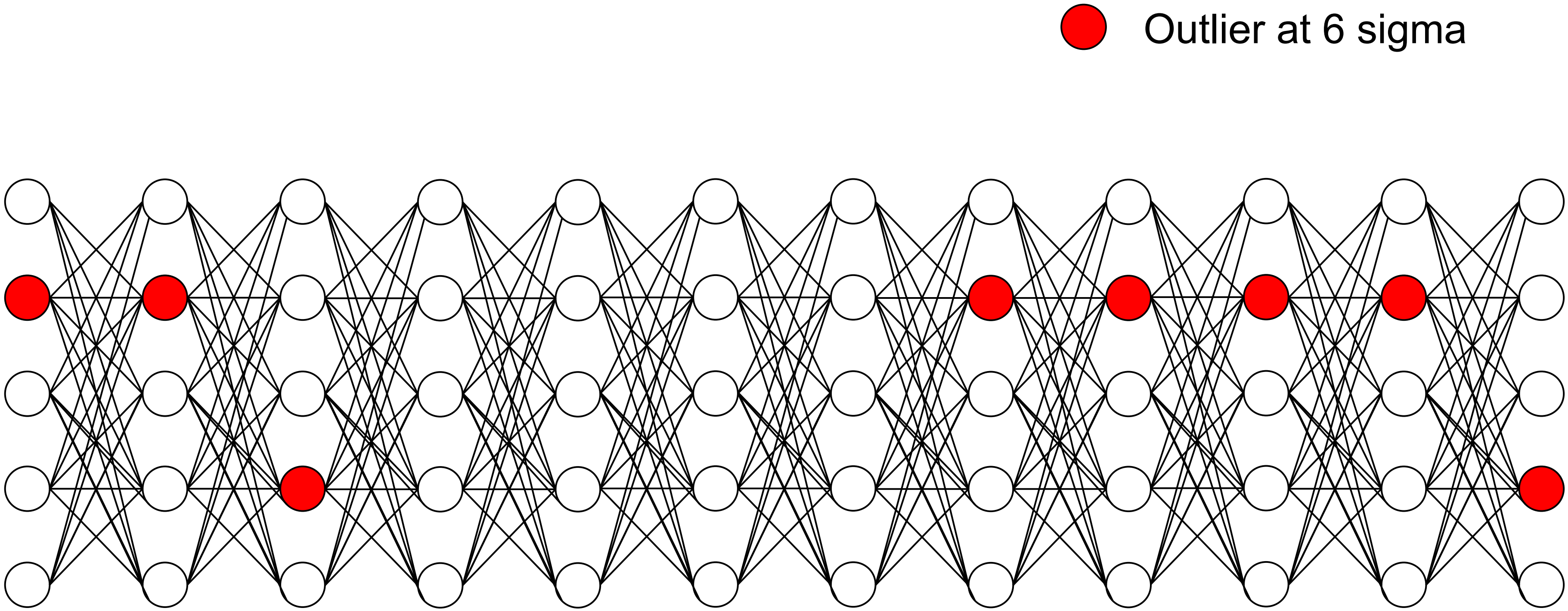


Outlier patterns in small neural networks (2.7B parameters)

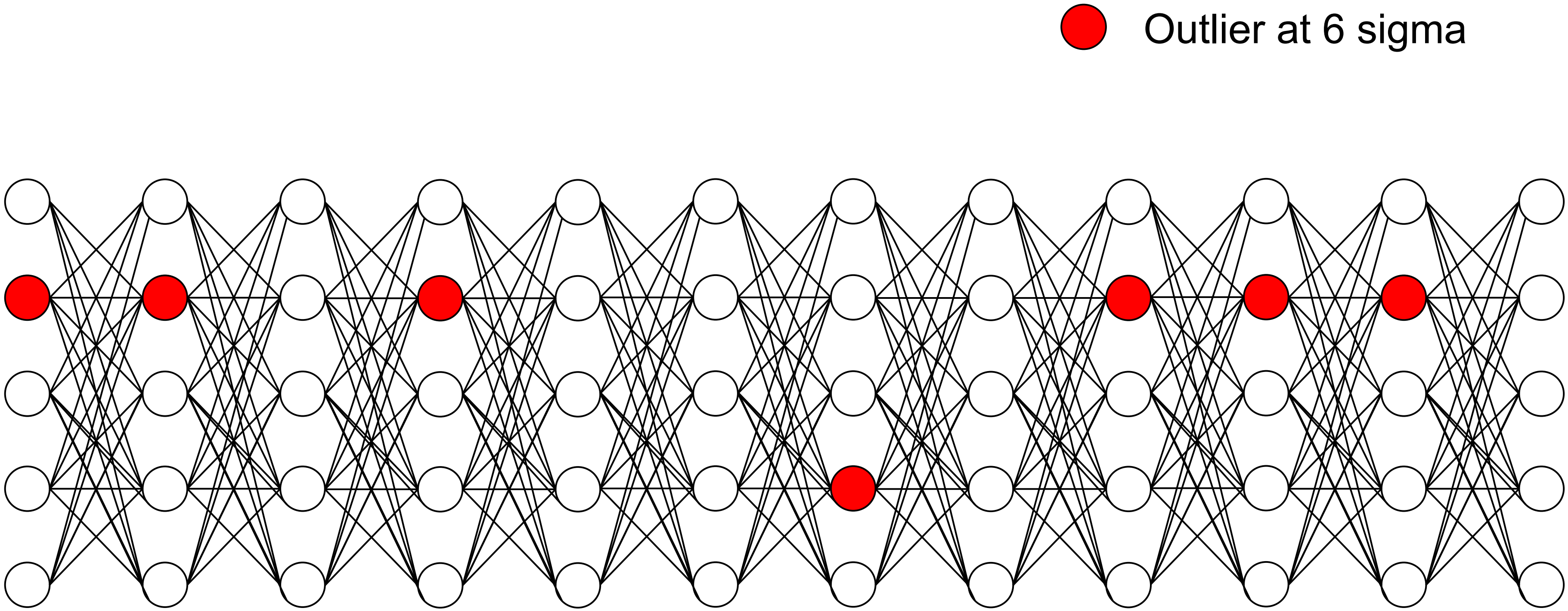
Outlier at 6 sigma



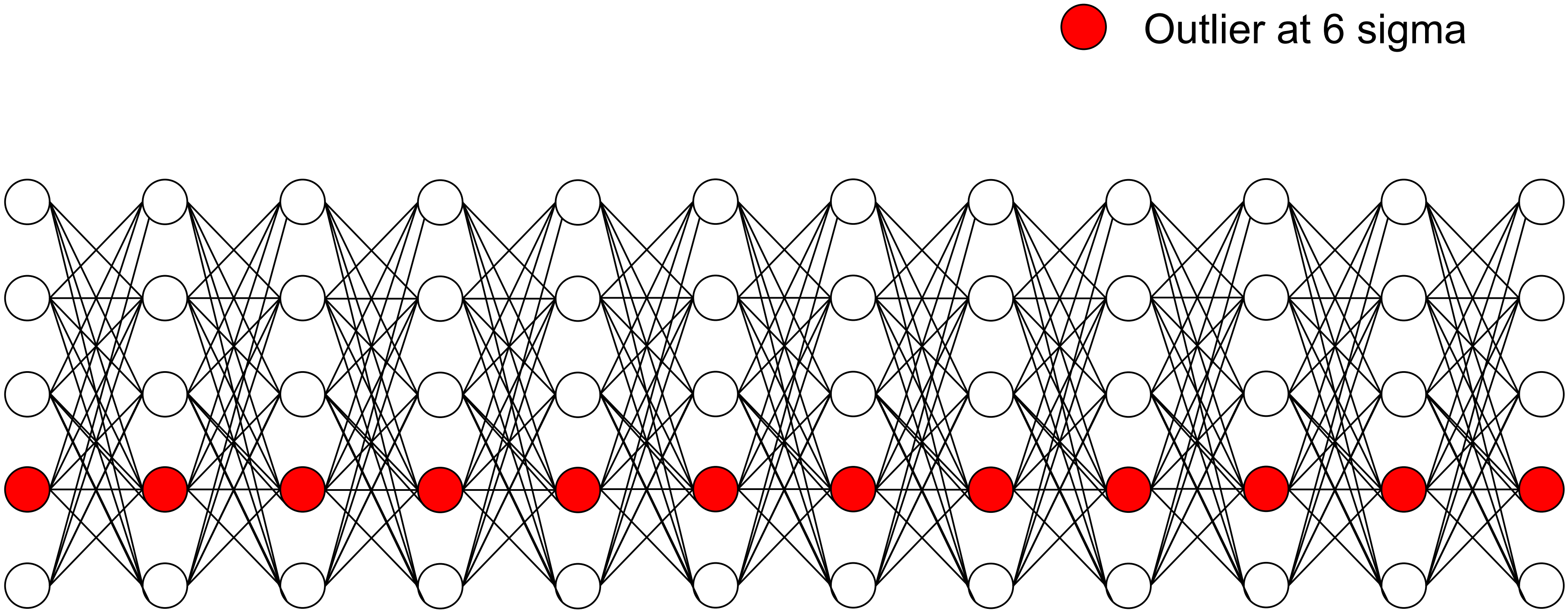
Outlier patterns in small neural networks (2.7B parameters)



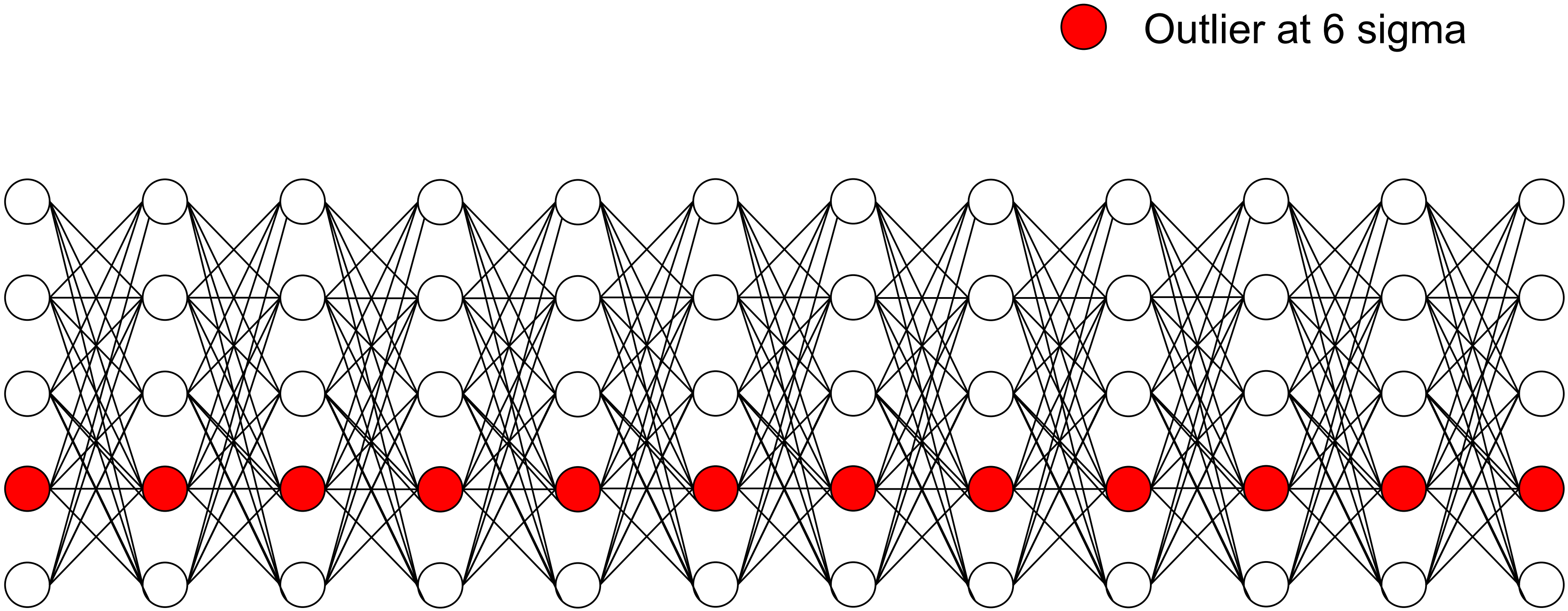
Outlier patterns in small neural networks (2.7B parameters)



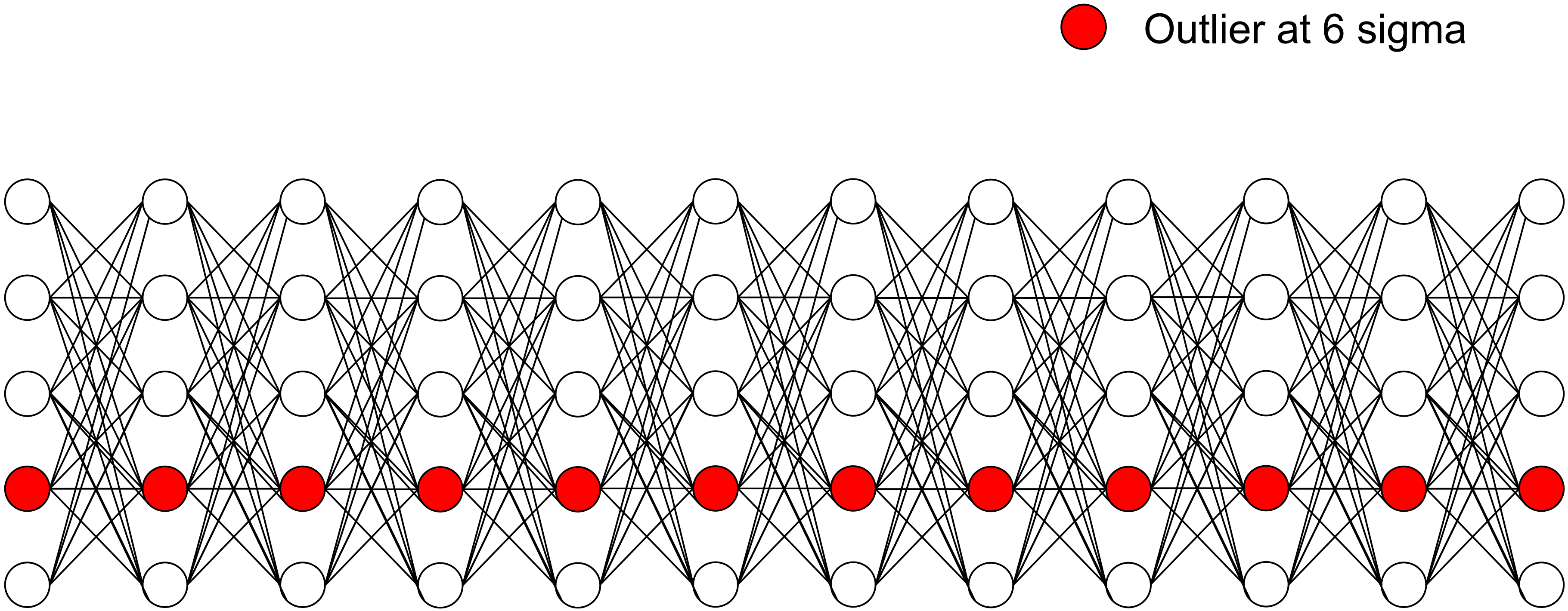
Outlier patterns in **large** neural networks (6.7B parameters)



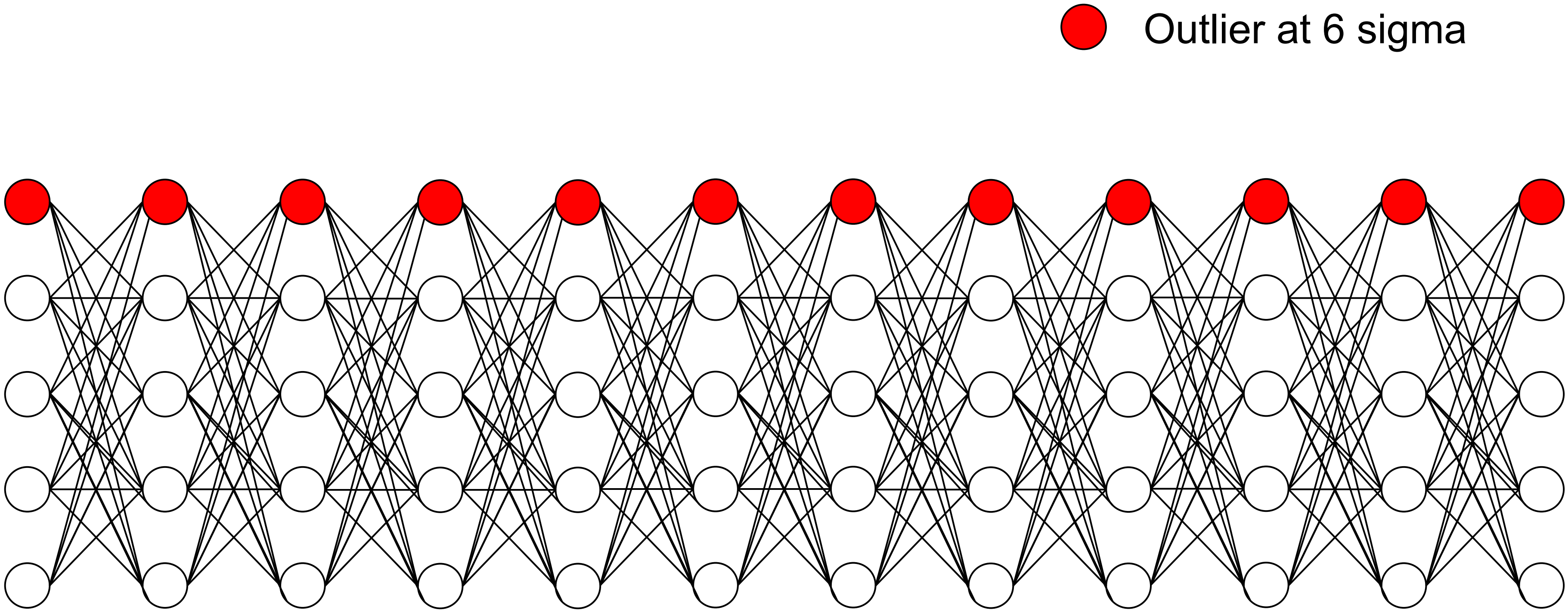
Outlier patterns in **large** neural networks (6.7B parameters)



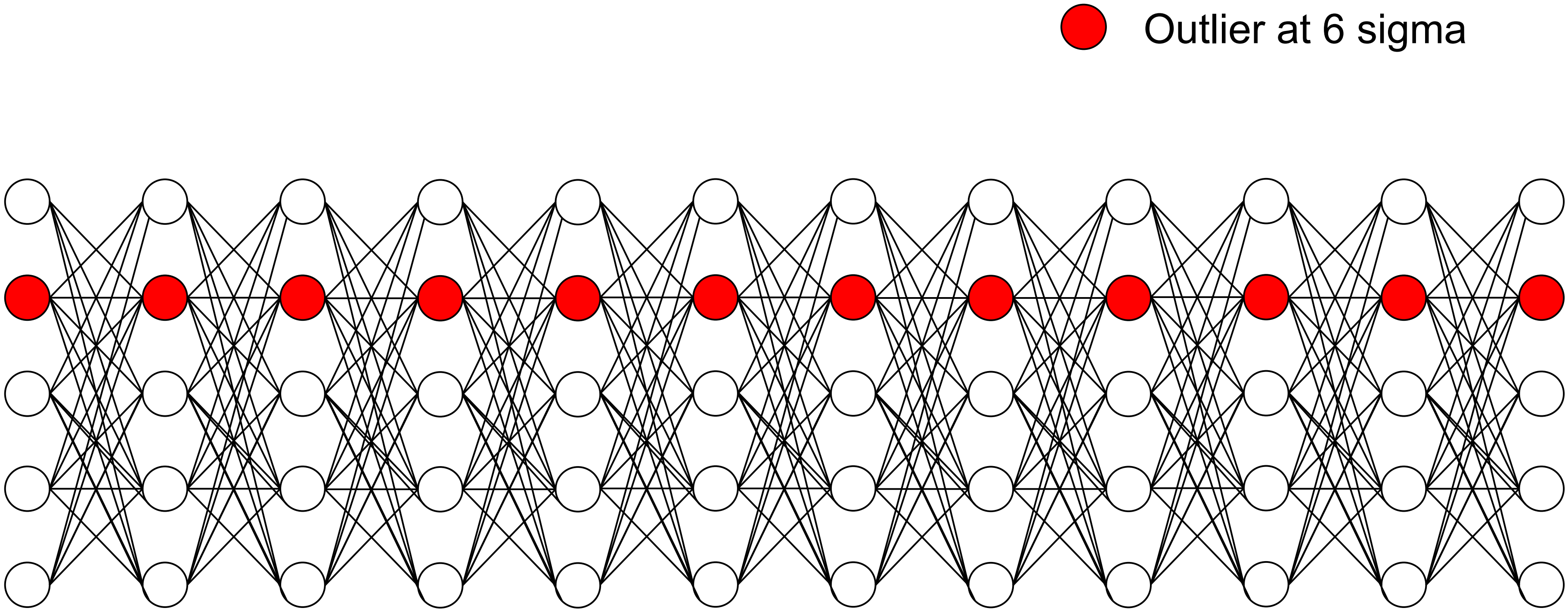
Outlier patterns in **large** neural networks (6.7B parameters)



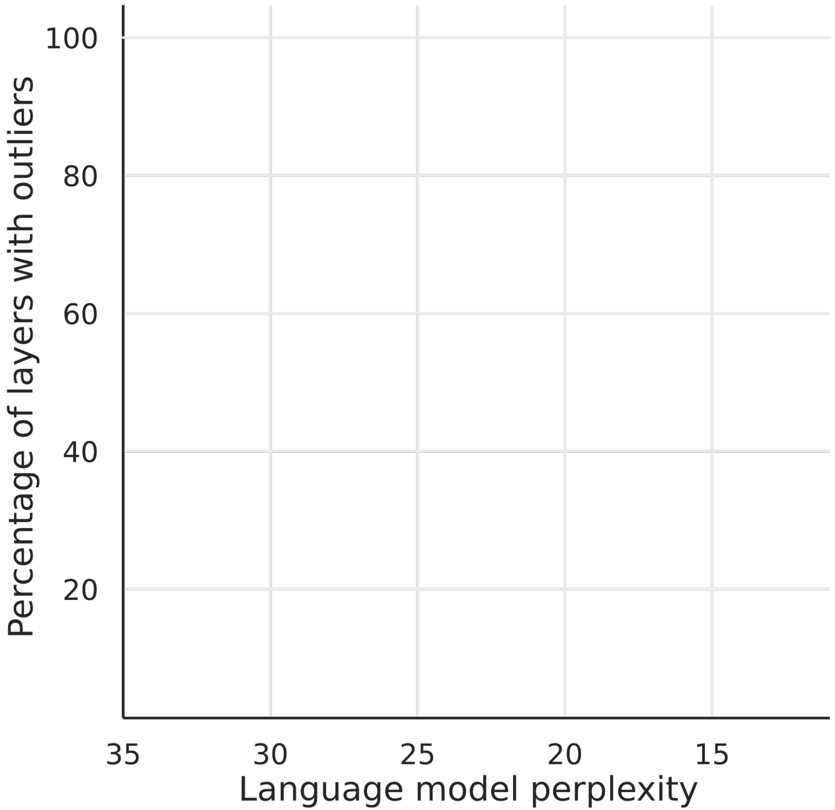
Outlier patterns in **large** neural networks (6.7B parameters)



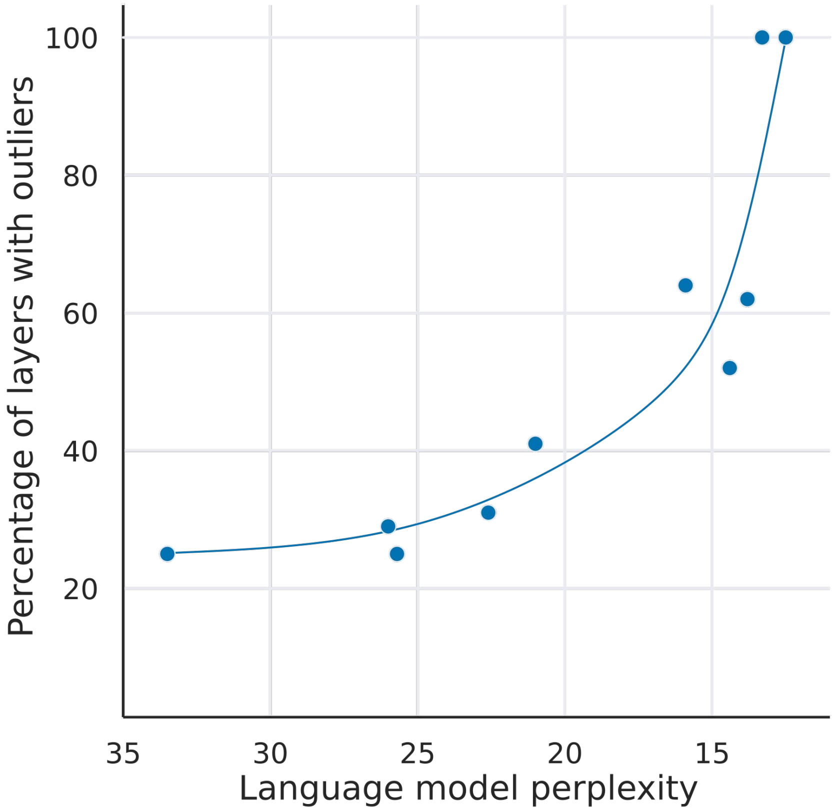
Outlier patterns in **large** neural networks (13B parameters)



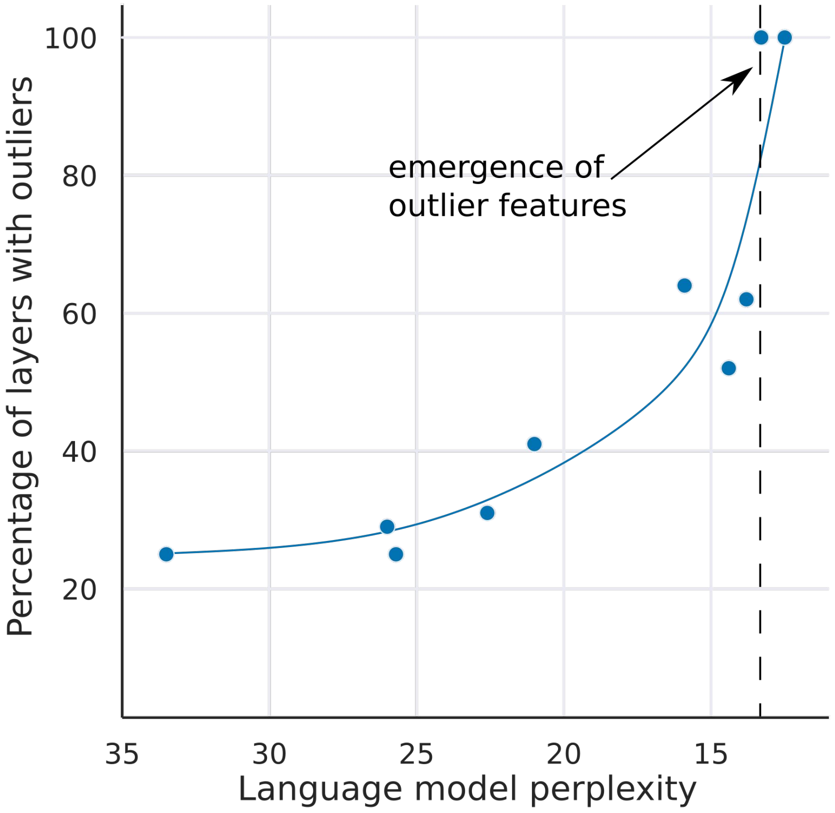
Emergent outliers vs language model performance



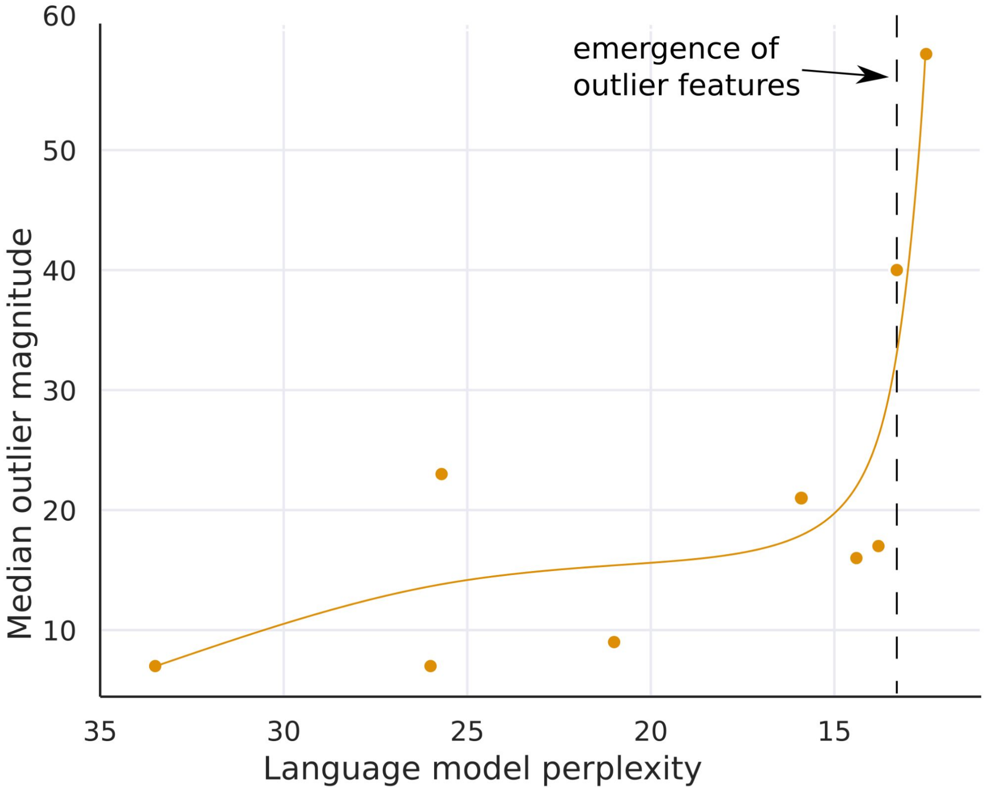
Emergent outliers vs language model performance



Emergent outliers vs language model performance



Emergent outliers vs outlier magnitude



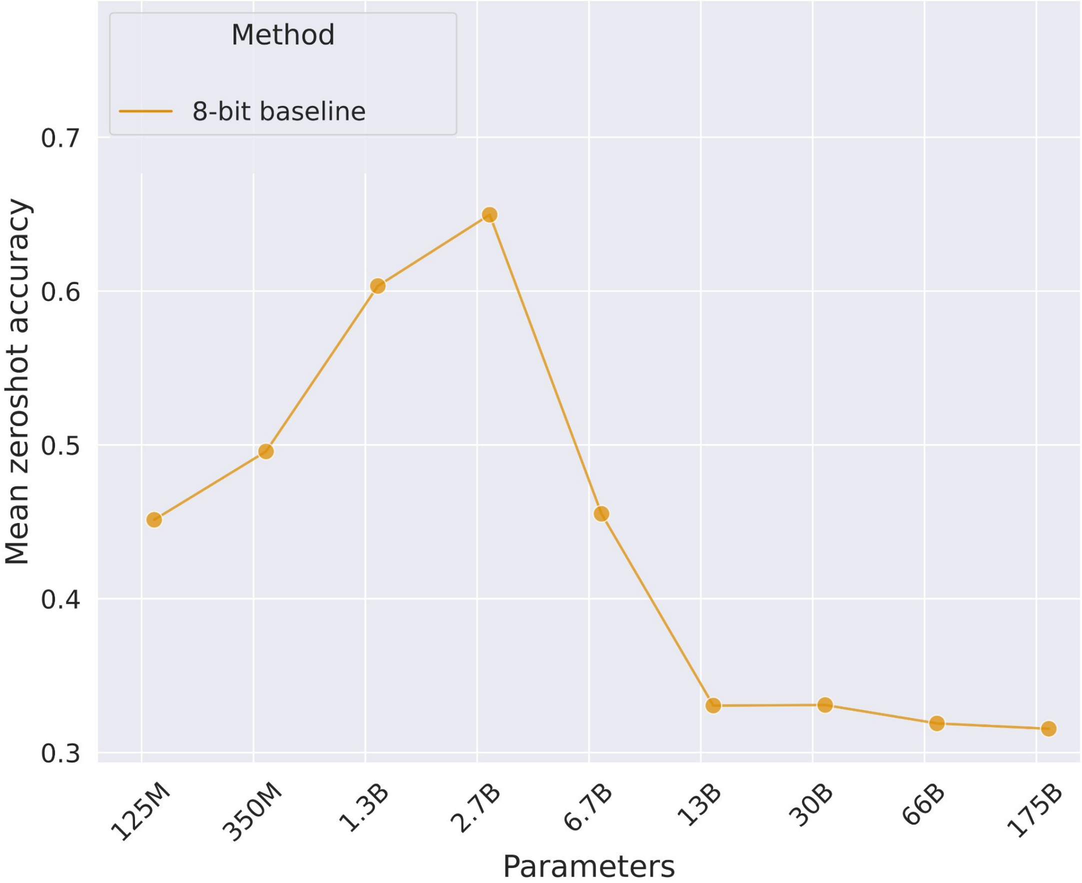
Mixed precision decomposition

Matrix multiply outliers (0.1%) in 16-bit.

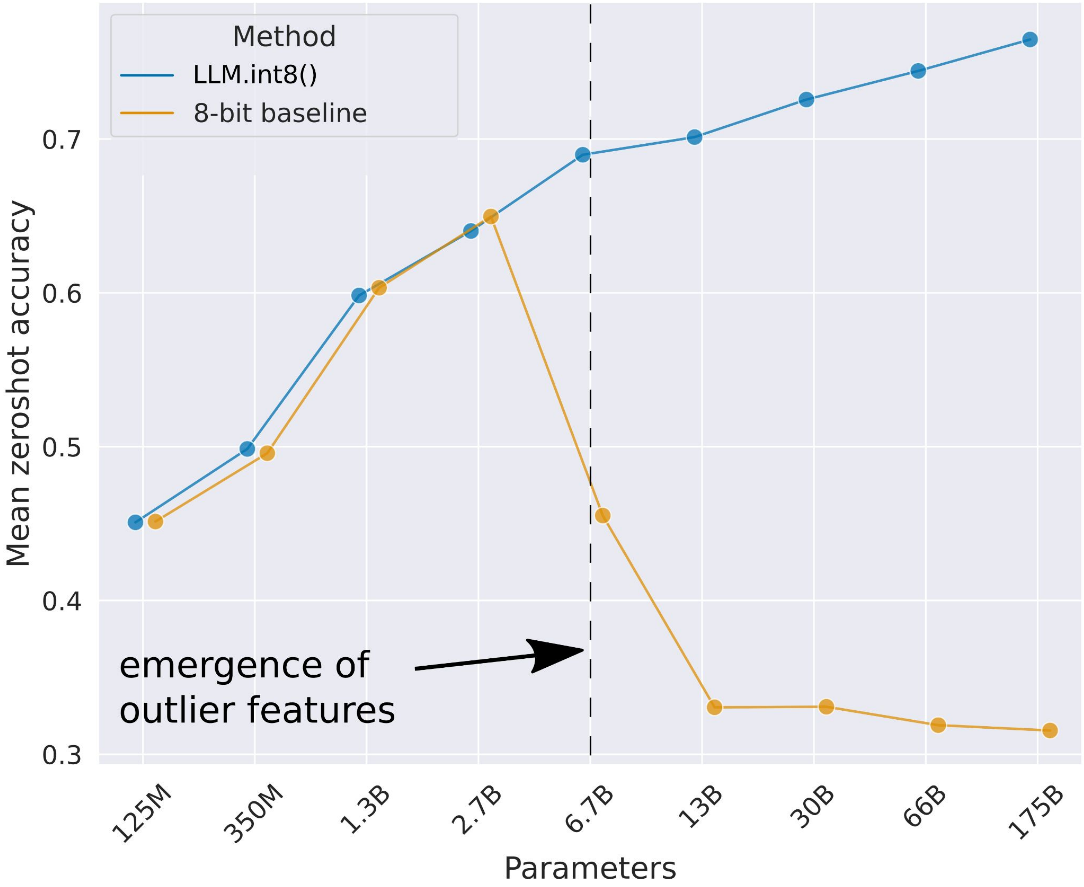
Matrix multiply other values (99.9%) in 8-bit.

$$\mathbf{C}_{f16} \approx \sum_{h \in O} \mathbf{X}^h_{f16} \mathbf{W}^h_{f16} + \mathbf{S}_{f16} \cdot \sum_{h \notin O} \mathbf{X}^h_{i8} \mathbf{W}^h_{i8}$$

8-bit Foundation Models Fail at Scale



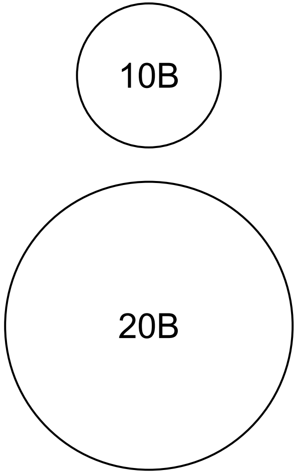
Our LLM.int8() method is the first method that works at scale



How can we maximize performance density per bit?

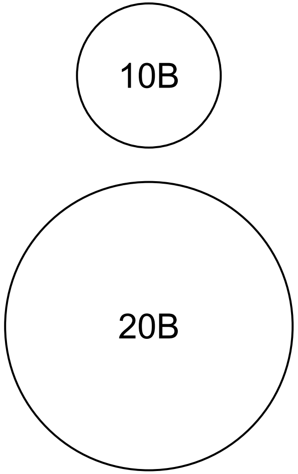
Maximizing performance density in foundation models

Parameters



Maximizing performance density in foundation models

Parameters

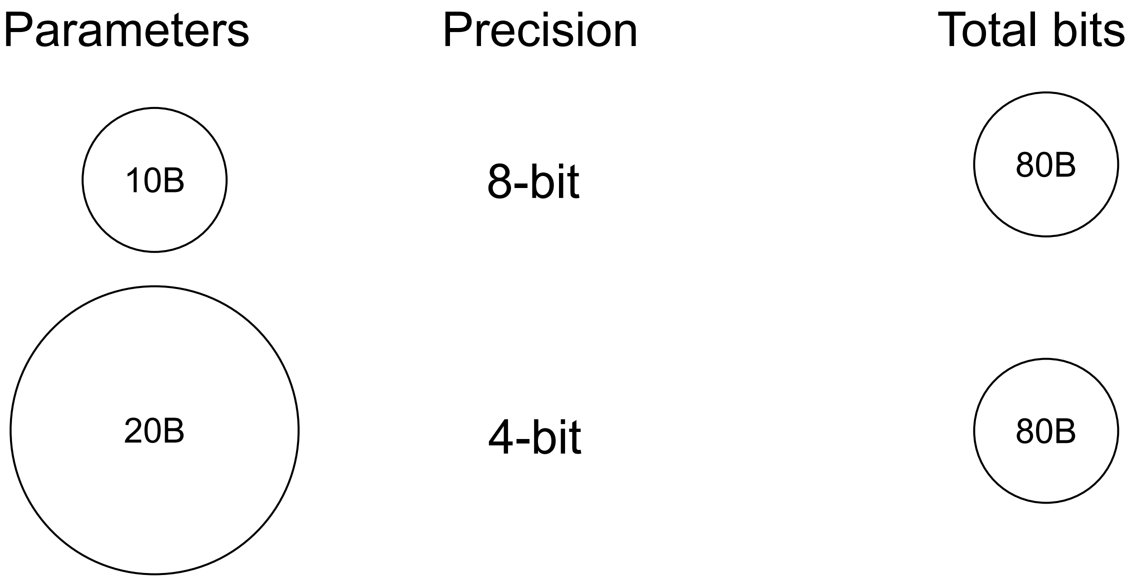


Precision

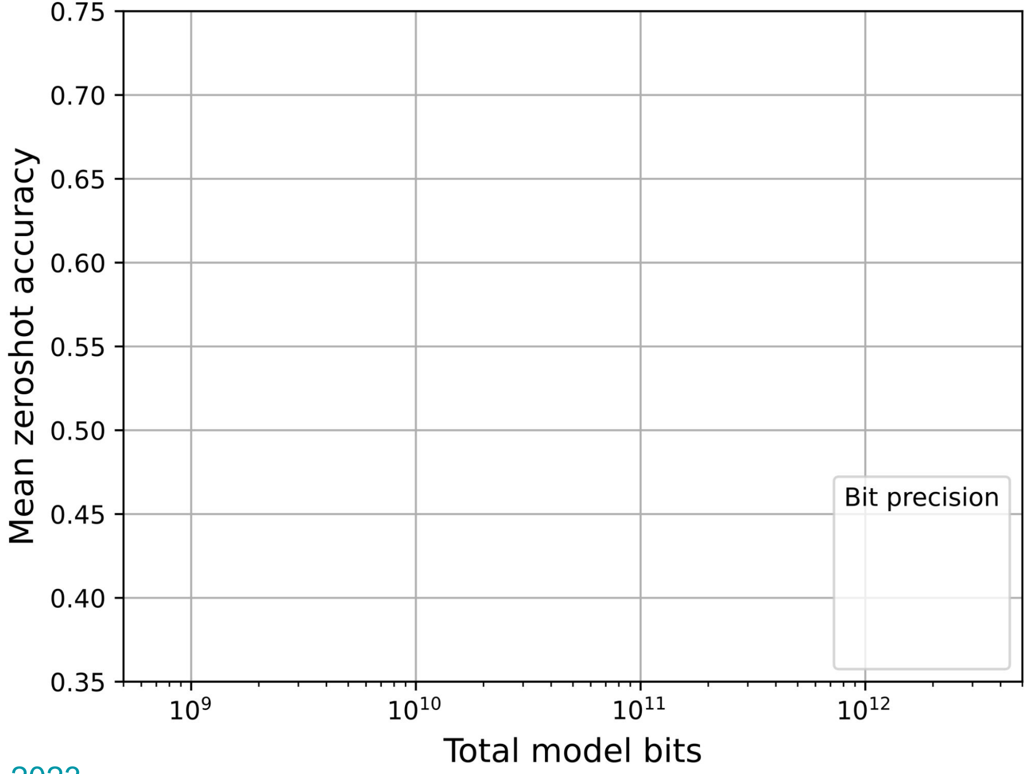
8-bit

4-bit

Maximizing performance density in foundation models

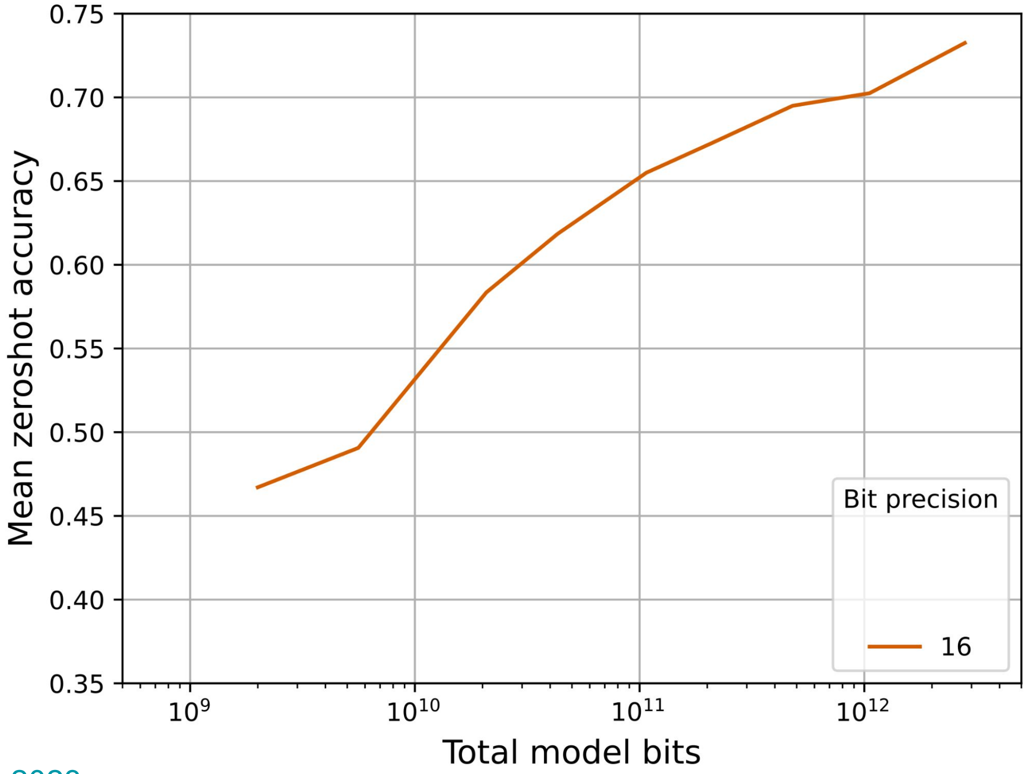


Maximizing performance density in foundation models



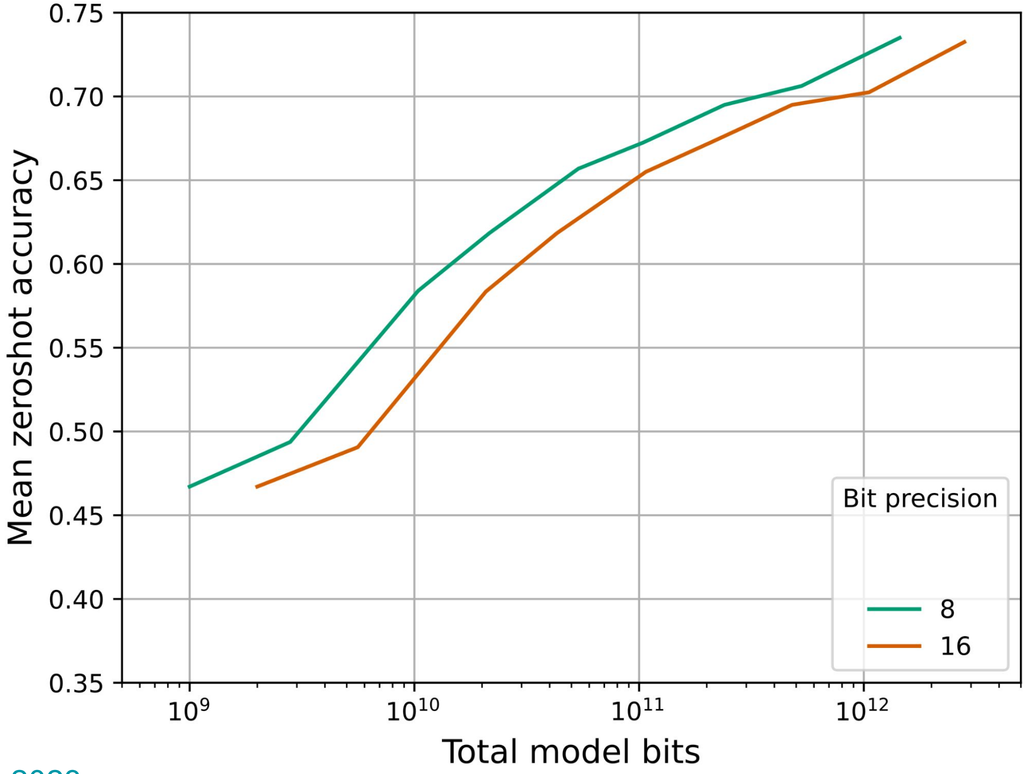
Dettmers & Zettlemoyer, 2023

Maximizing performance density in foundation models



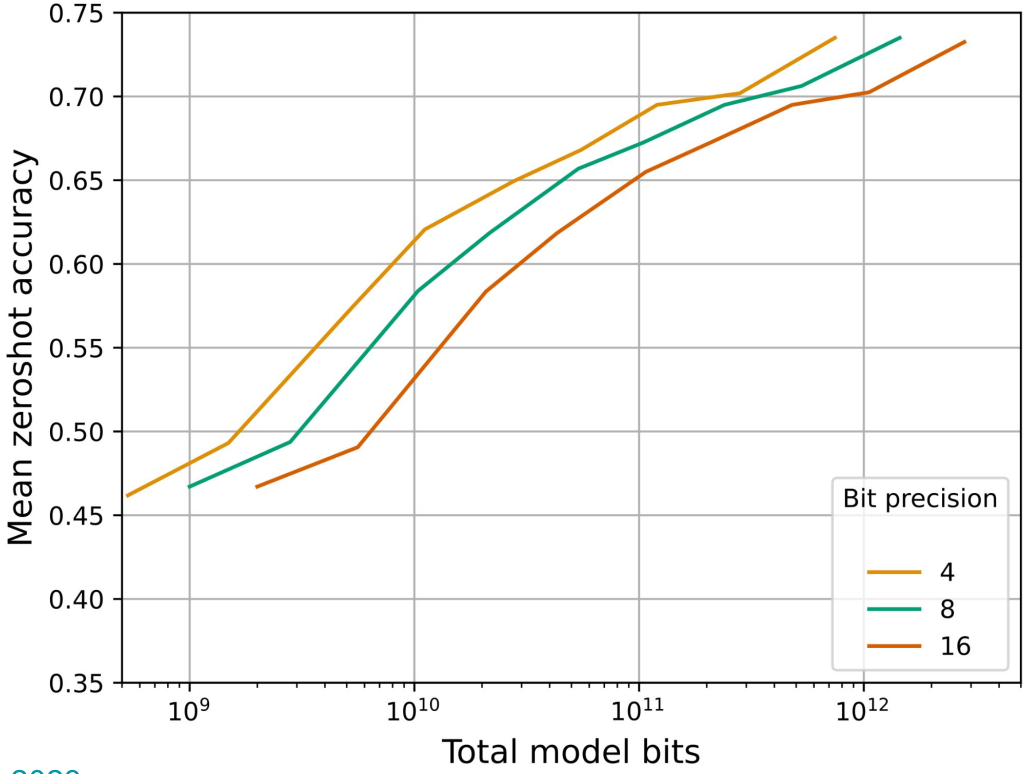
Dettmers & Zettlemoyer, 2023

Maximizing performance density in foundation models



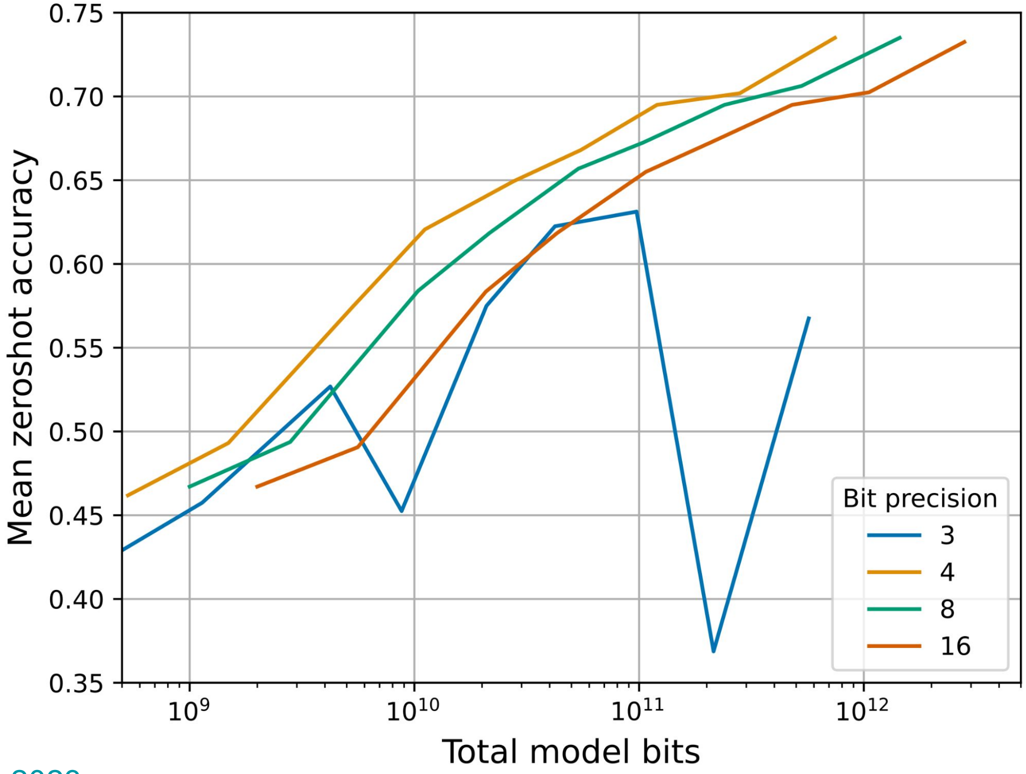
[Dettmers & Zettlemoyer, 2023]

Maximizing performance density in foundation models



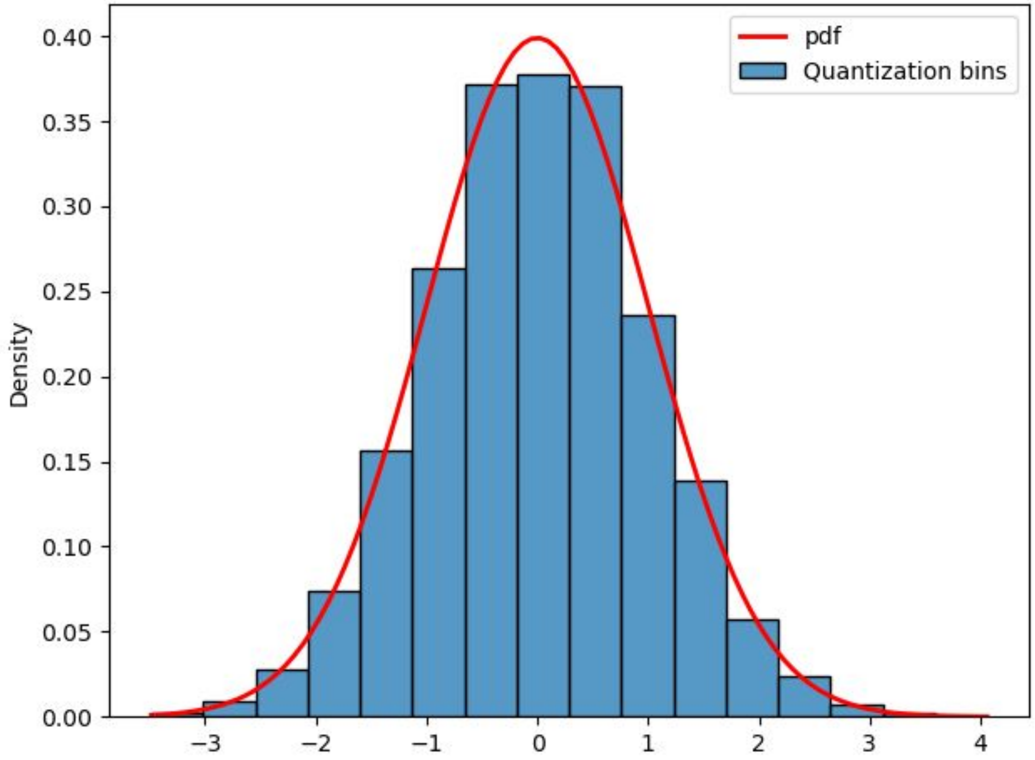
Dettmers & Zettlemoyer, 2023

Maximizing performance density in foundation models

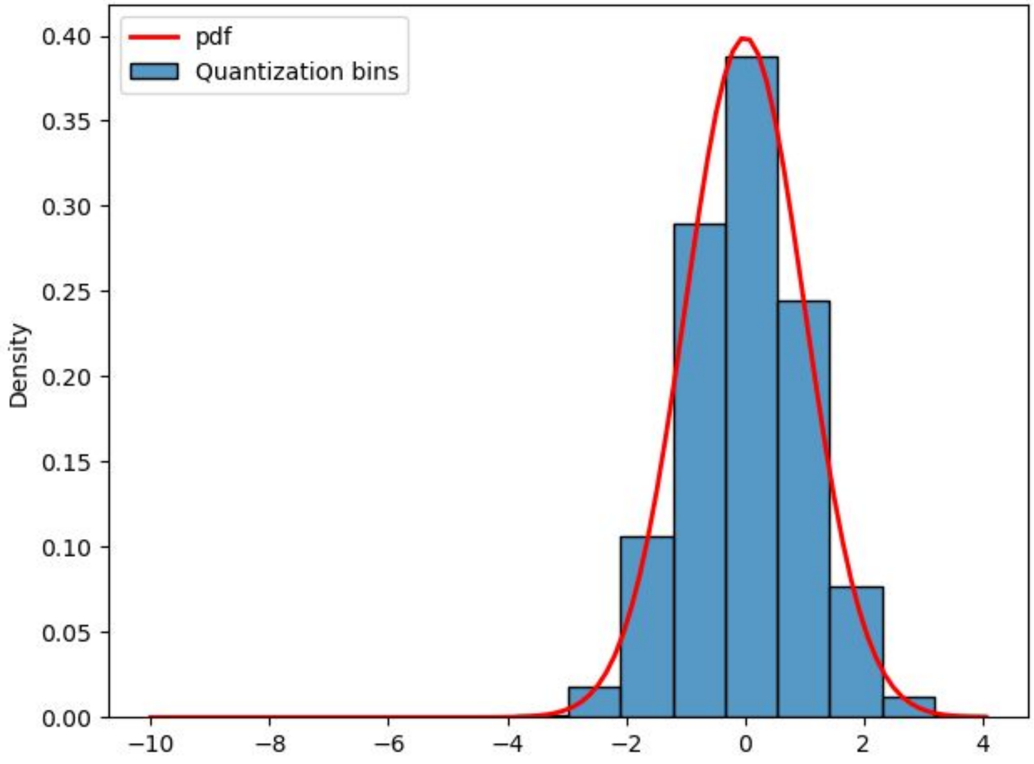


Dettmers & Zettlemoyer, 2023

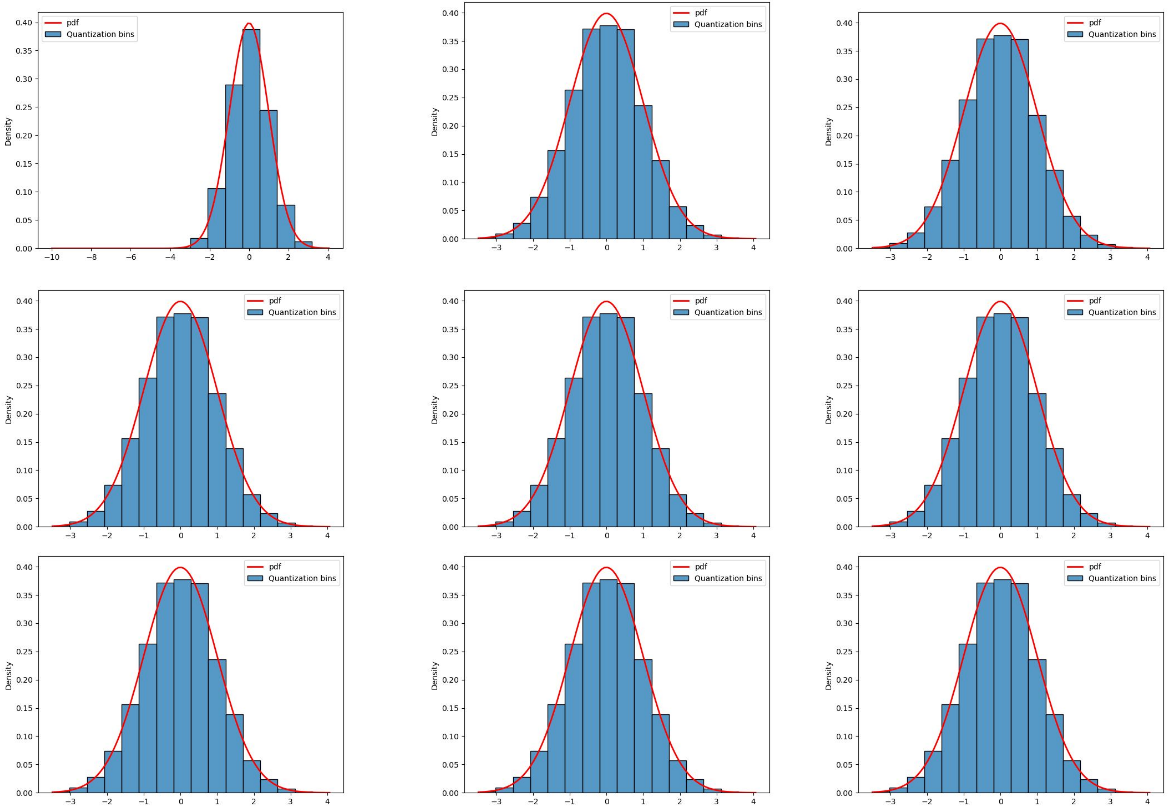
Integer quantization is similar to histogram binning



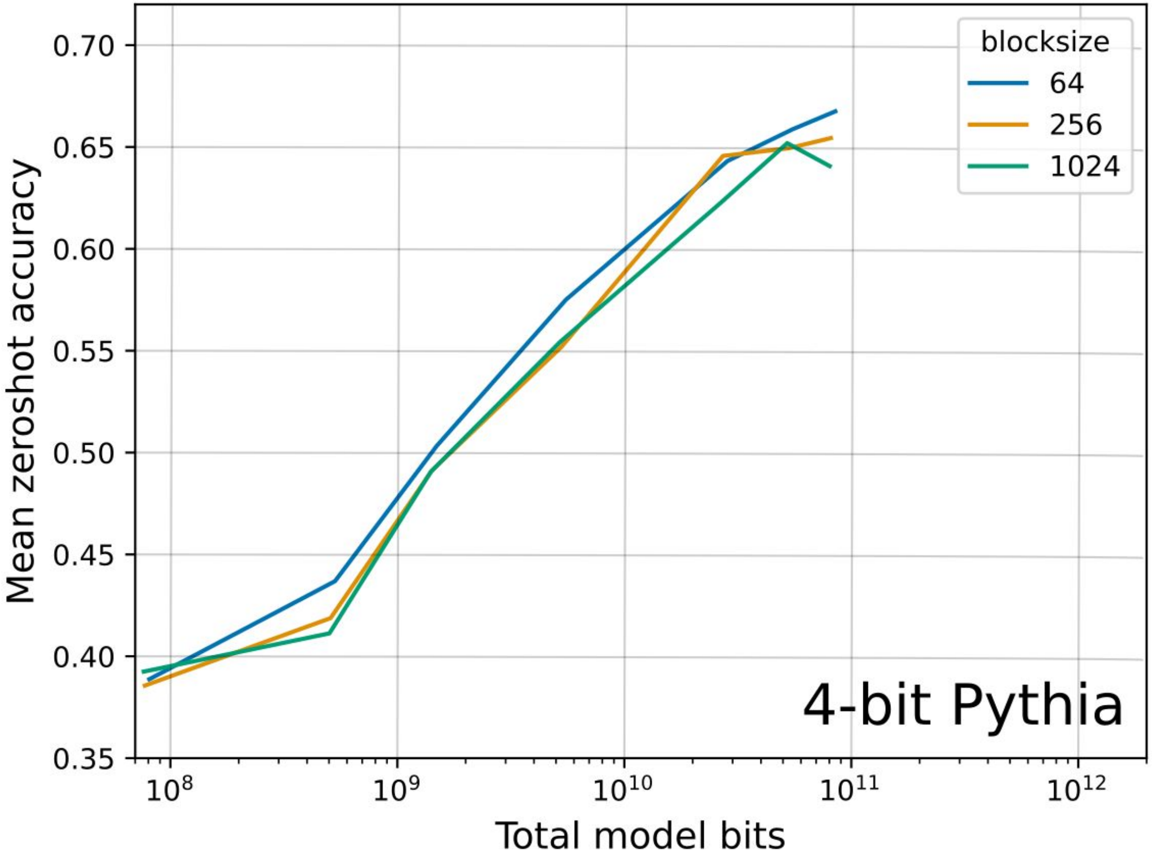
What do outliers in quantization look like?



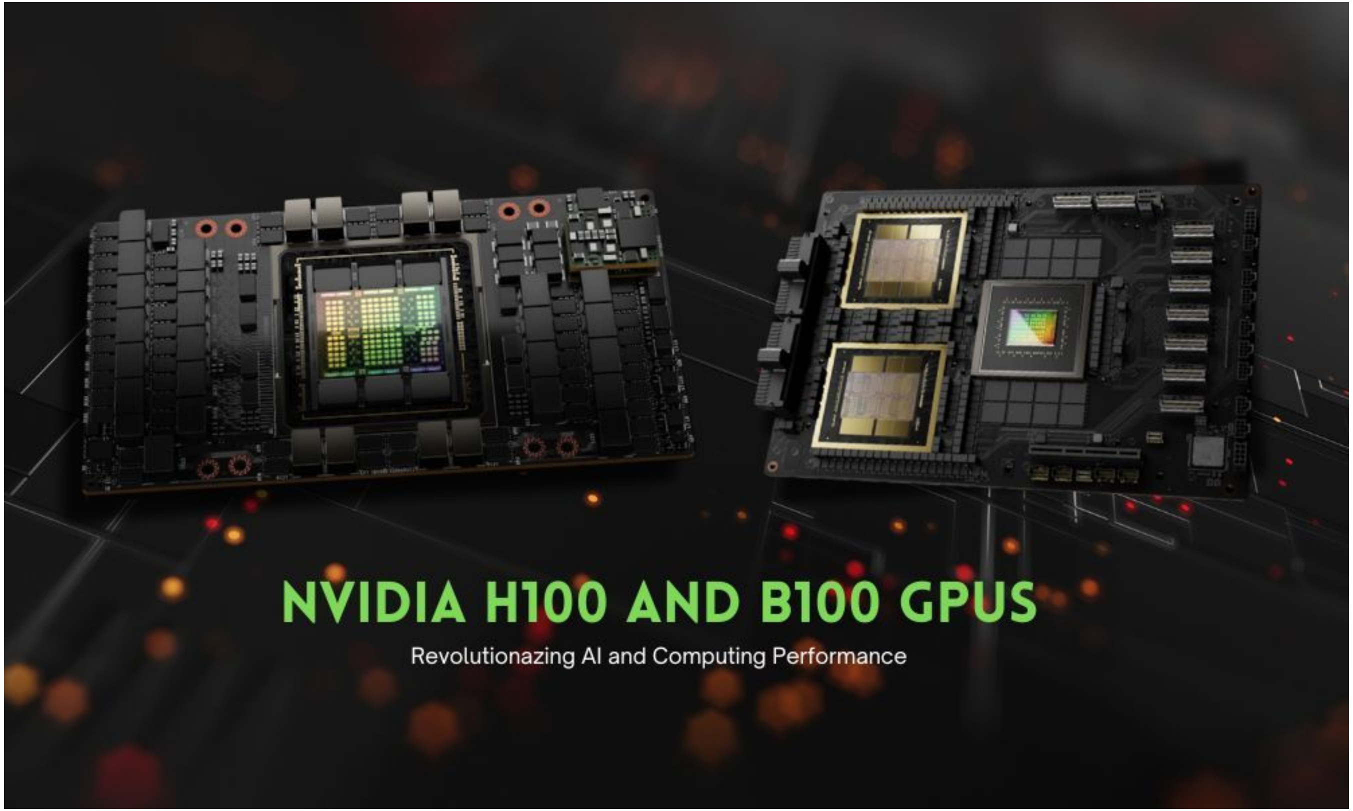
Block-wise quantization



What does help to improve scaling? Block size



Hardware-based block-wise quantization with Blackwell B100/B200

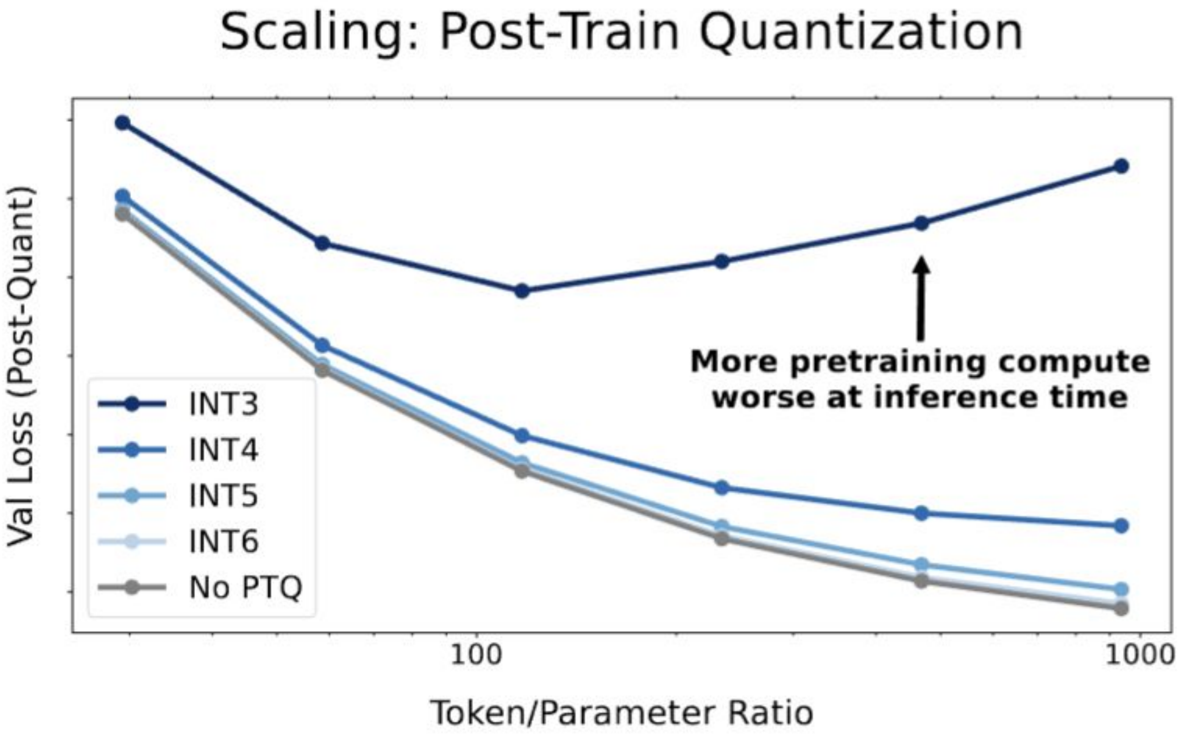


Scaling Laws for Precision

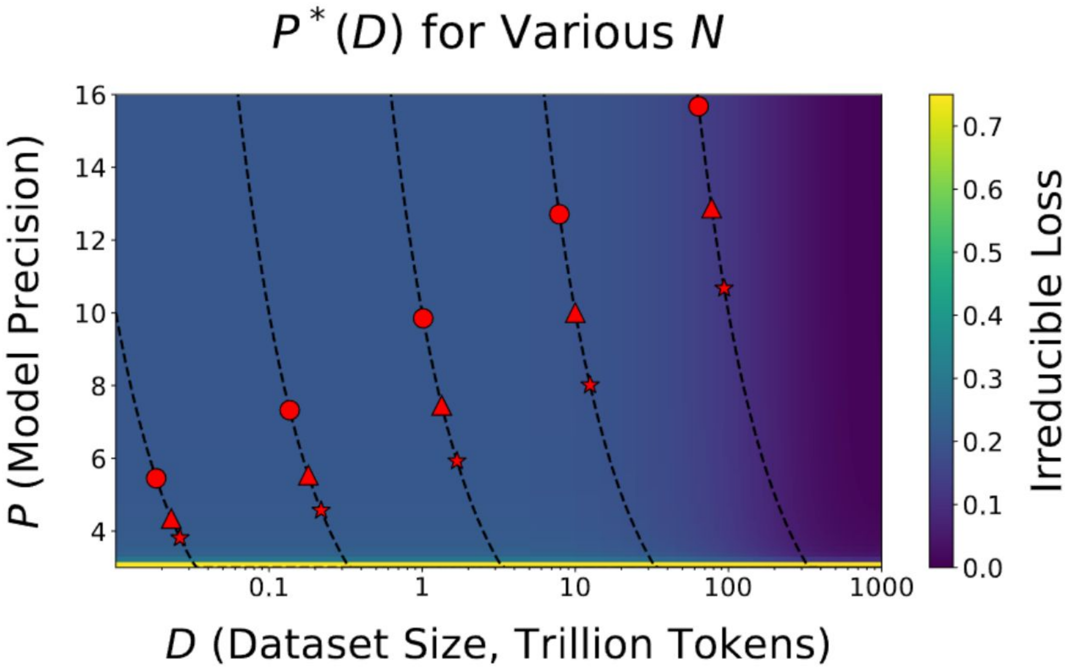
Tanishq Kumar*[1] Zachary Ankner*[3, 4] Benjamin F. Spector[2] Blake Bordelon[1] Niklas Muennighoff[2]
Mansheej Paul[4] Cengiz Pehlevan[1] Christopher Ré[2] Aditi Raghunathan[5]

[1]Harvard University
[2]Stanford University
[3]MIT
[4]Databricks
[5]Carnegie Mellon University

Quantization precision optimality depends on data per parameter



Optimal precision for pretraining/post-training quantization

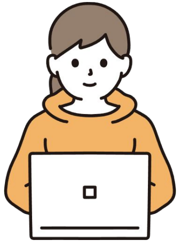


Circle: 8B
Triangle: 70B
Star: 405B

Take-away

Fundamental insights into foundation models information processing enables efficiency and accessibility

Accessibility challenges of foundation models

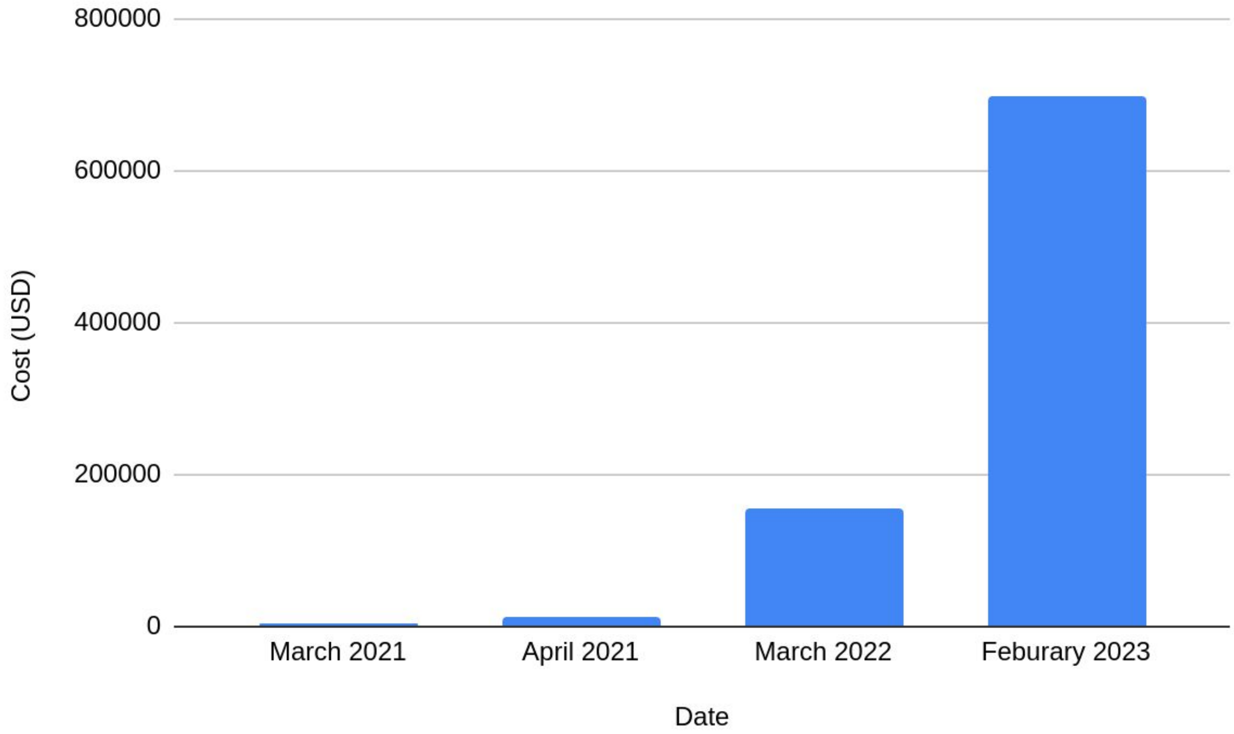


Using foundation models

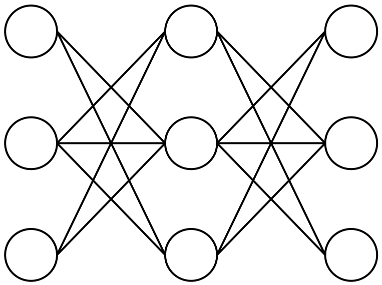
Finetuning foundation models

Quantization: companies vs users

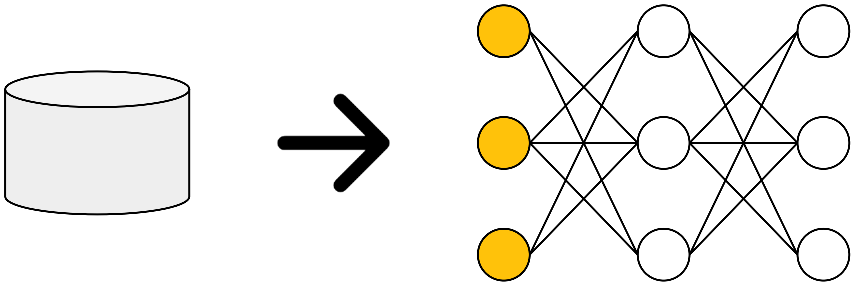
Finetuning is expensive due to GPU memory requirements



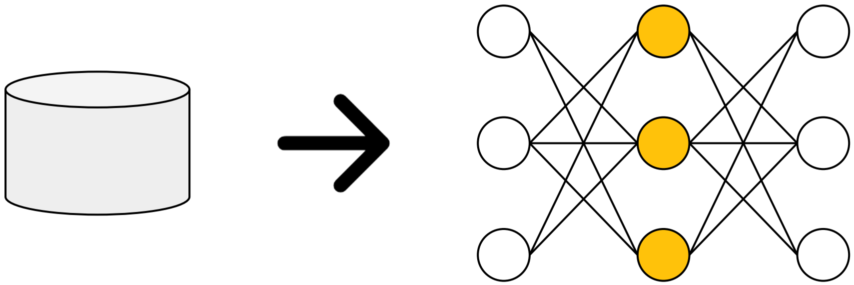
How to finetune a model



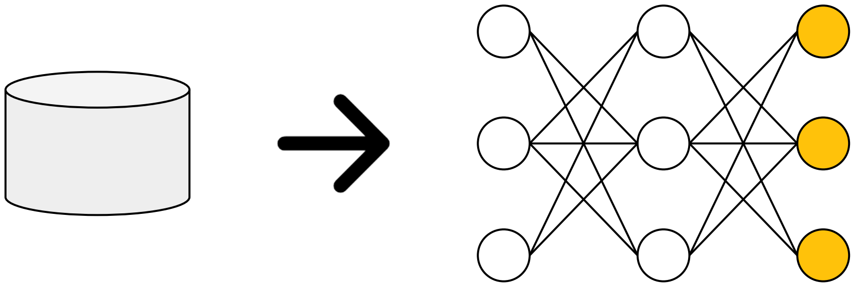
How to finetune a model



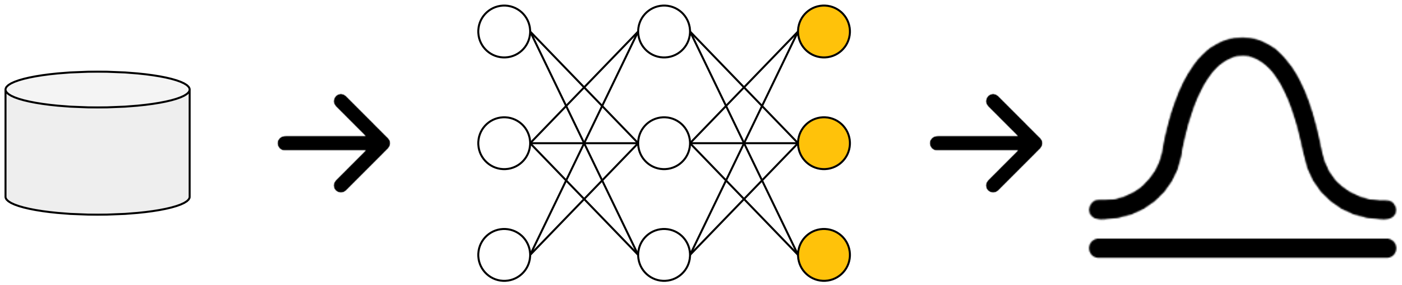
How to finetune a model



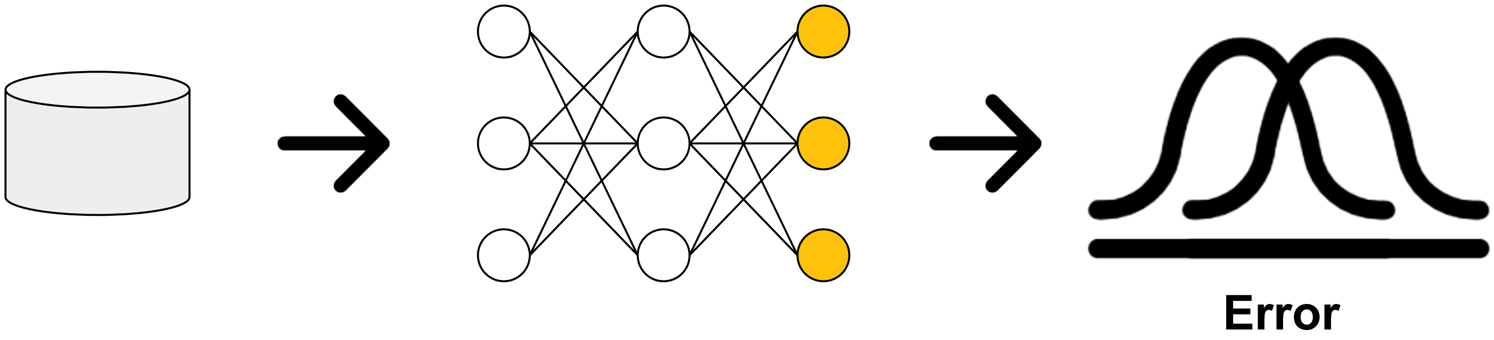
How to finetune a model



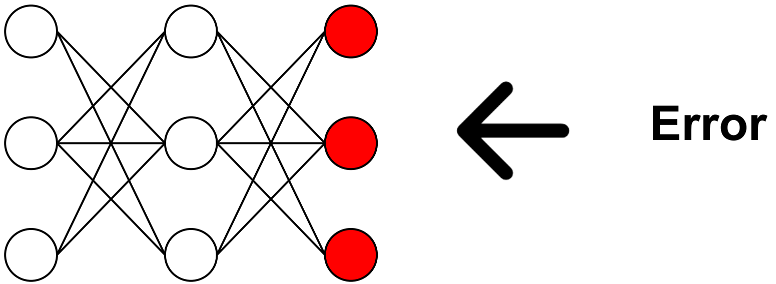
How to finetune a model



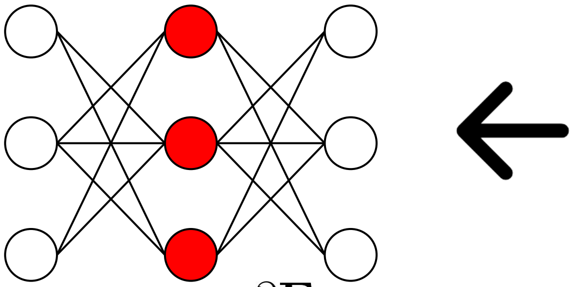
How to finetune a model



How to finetune a model



How to finetune a model

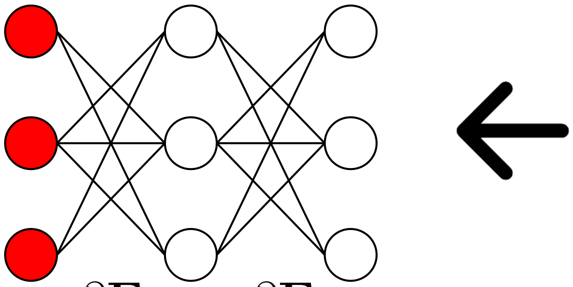


Error

$$\frac{\partial \mathbf{E}}{\partial \mathbf{W}_2}$$

Weight gradients

How to finetune a model



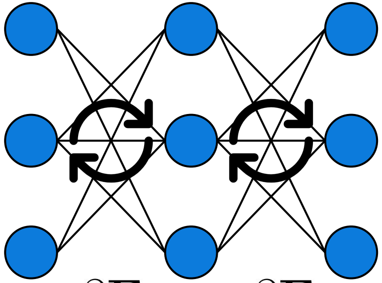
Error

$$\frac{\partial \mathbf{E}}{\partial \mathbf{W}_1} \quad \frac{\partial \mathbf{E}}{\partial \mathbf{W}_2}$$

Weight gradients

Background: How to finetune a model

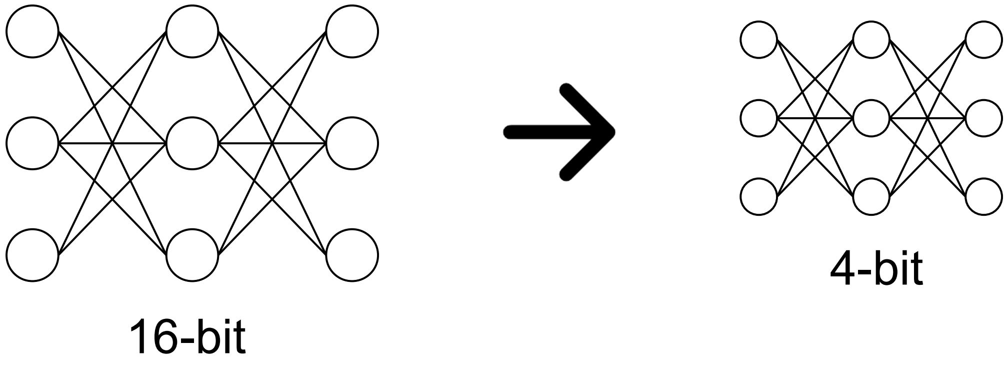
Update the weights



$$\frac{\partial \mathbf{E}}{\partial \mathbf{W}_1} \quad \frac{\partial \mathbf{E}}{\partial \mathbf{W}_2}$$

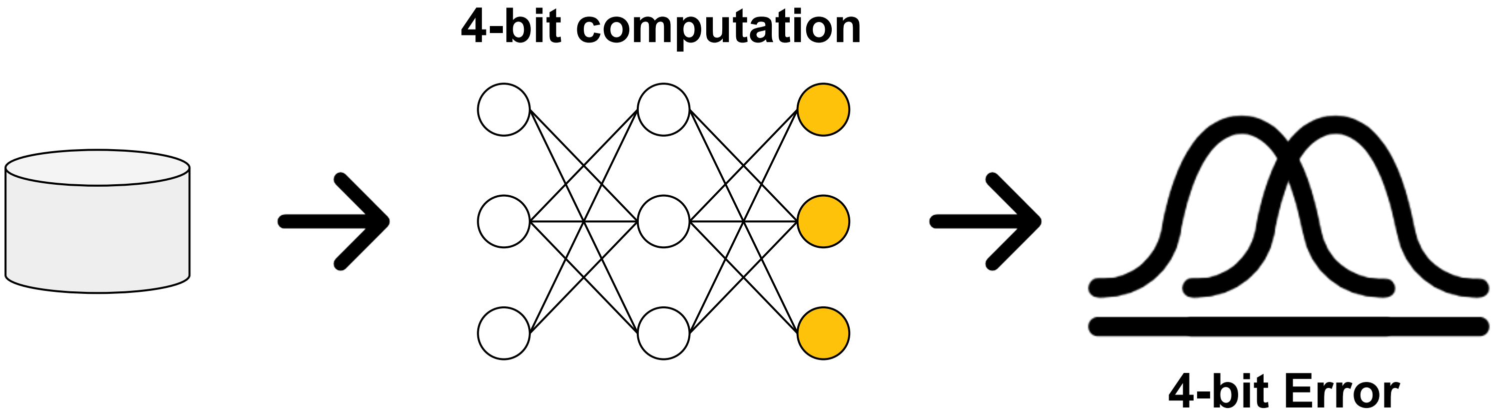
Quality 

Finetuning a 4-bit model with our insights …



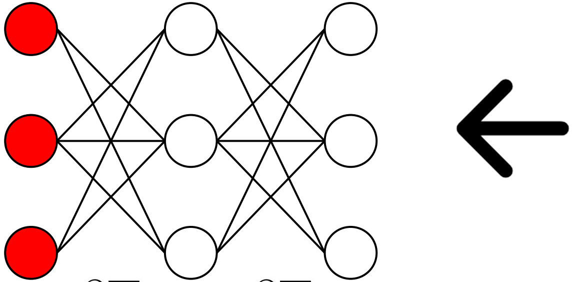
Quality 

Finetuning a 4-bit model with our insights …



Finetuning a 4-bit model with our insights …

4-bit backpropagation



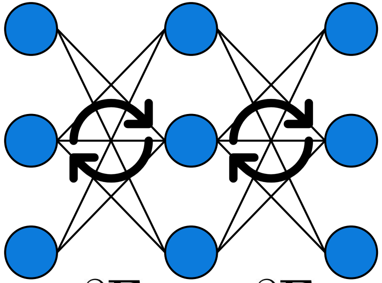
4-bit Error

$$\frac{\partial \mathbf{E}}{\partial \mathbf{W}_1} \qquad \frac{\partial \mathbf{E}}{\partial \mathbf{W}_2}$$

4-bit Weight gradients

Finetuning a 4-bit model with our insights …

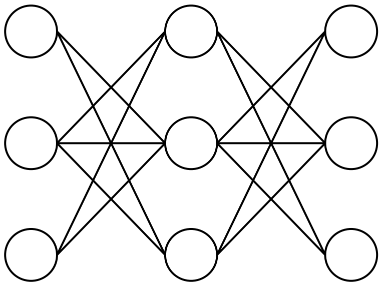
Update the weights



$$\frac{\partial \mathbf{E}}{\partial \mathbf{W}_1} \quad \frac{\partial \mathbf{E}}{\partial \mathbf{W}_2}$$

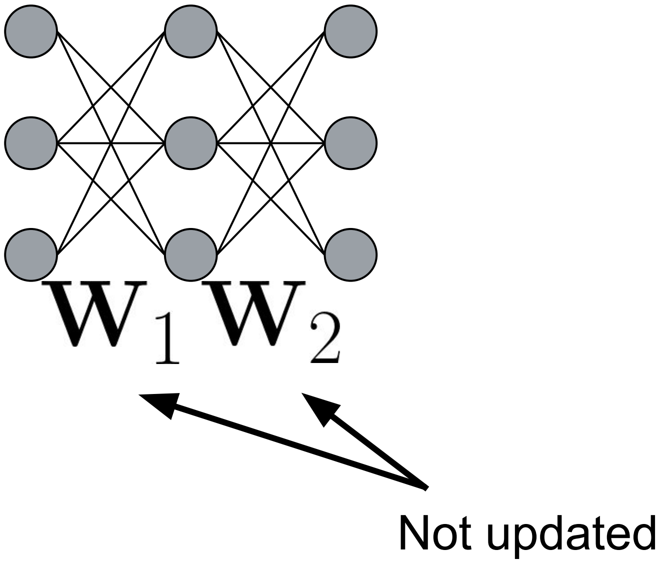
Quality

Low-rank Adaptation (LoRA)



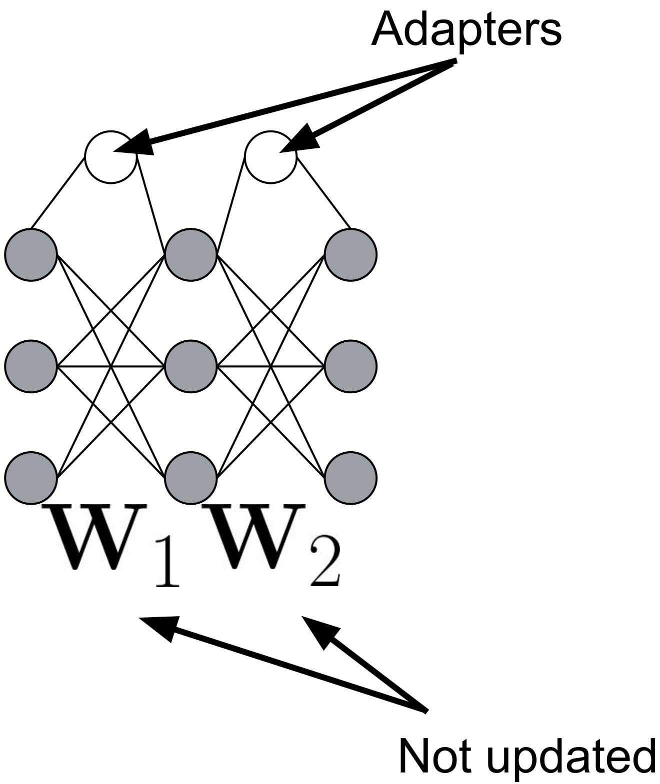
[Hu et al., 2021](Hu et al., 2021)

Low-rank Adaptation (LoRA)



[Hu et al., 2021]

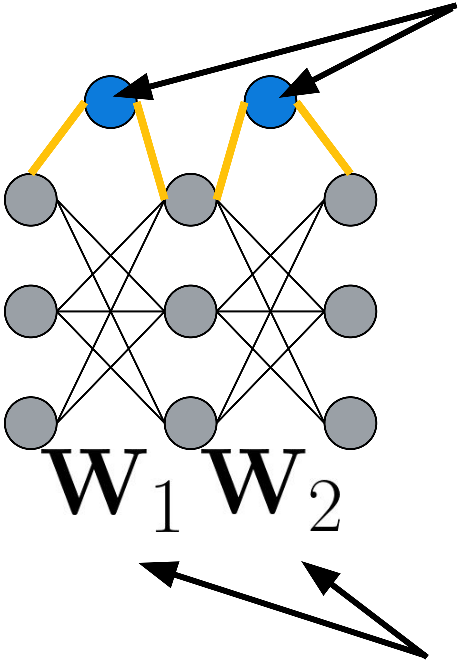
Low-rank Adaptation (LoRA)



Hu et al., 2021

Low-rank Adaptation (LoRA)

Adapters

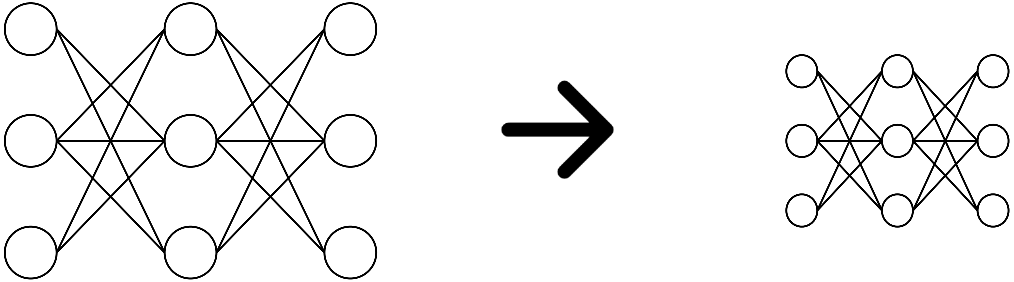


Only update
adapter weights

Not updated

[Hu et al., 2021](#)

Quantized Low-rank Adaptation (QLoRA)

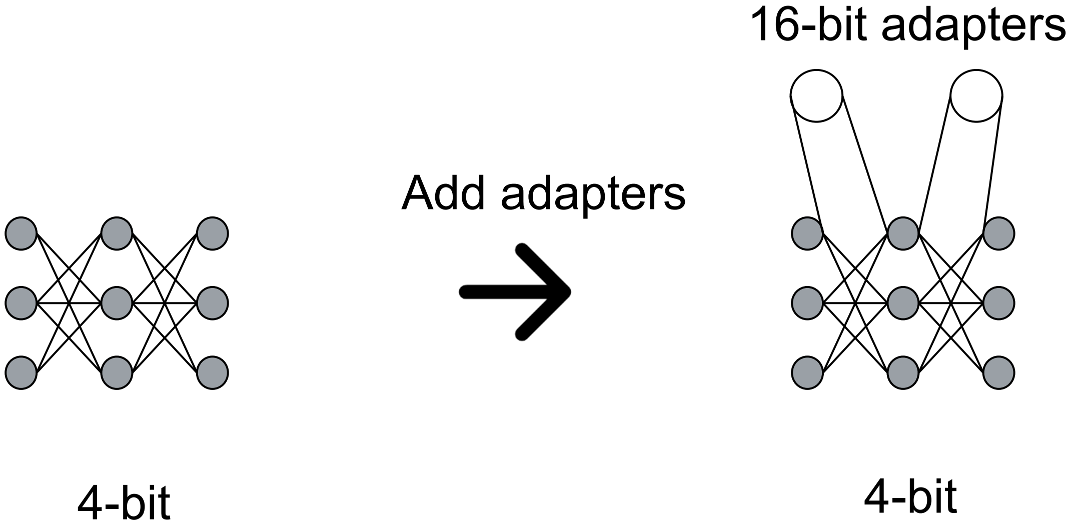


16-bit

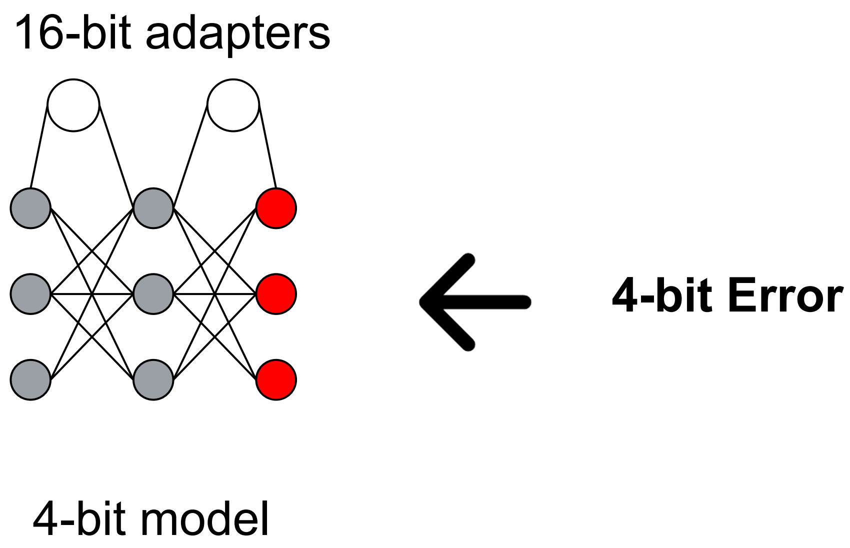
4-bit

Quality 

Quantized Low-rank Adaptation (QLoRA)

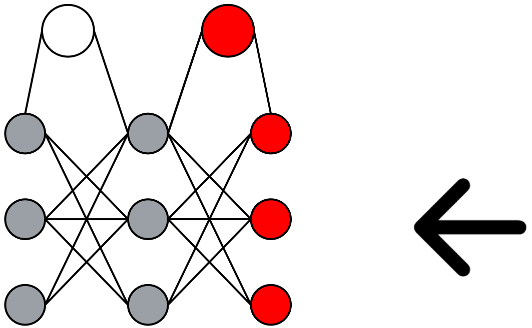


Quantized Low-rank Adaptation (QLoRA)



Quantized Low-rank Adaptation (QLoRA)

16-bit adapters

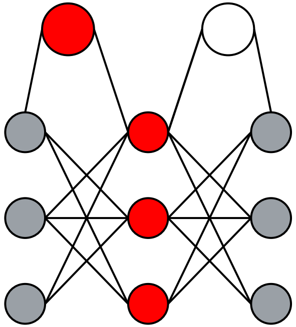


4-bit Error

4-bit model

Quantized Low-rank Adaptation (QLoRA)

16-bit adapters

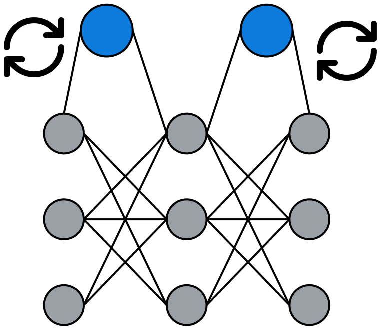


4-bit model

4-bit Error

Quantized Low-rank Adaptation (QLoRA)

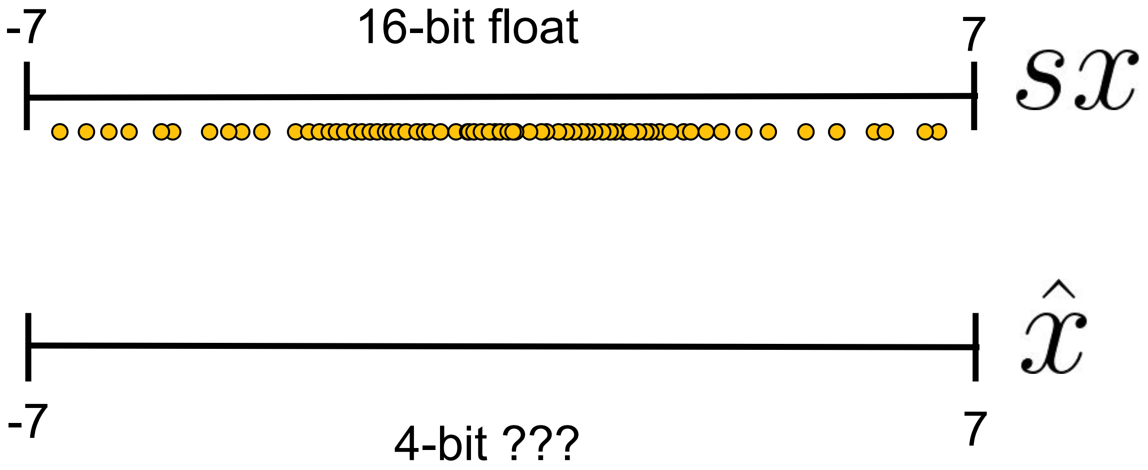
16-bit adapters



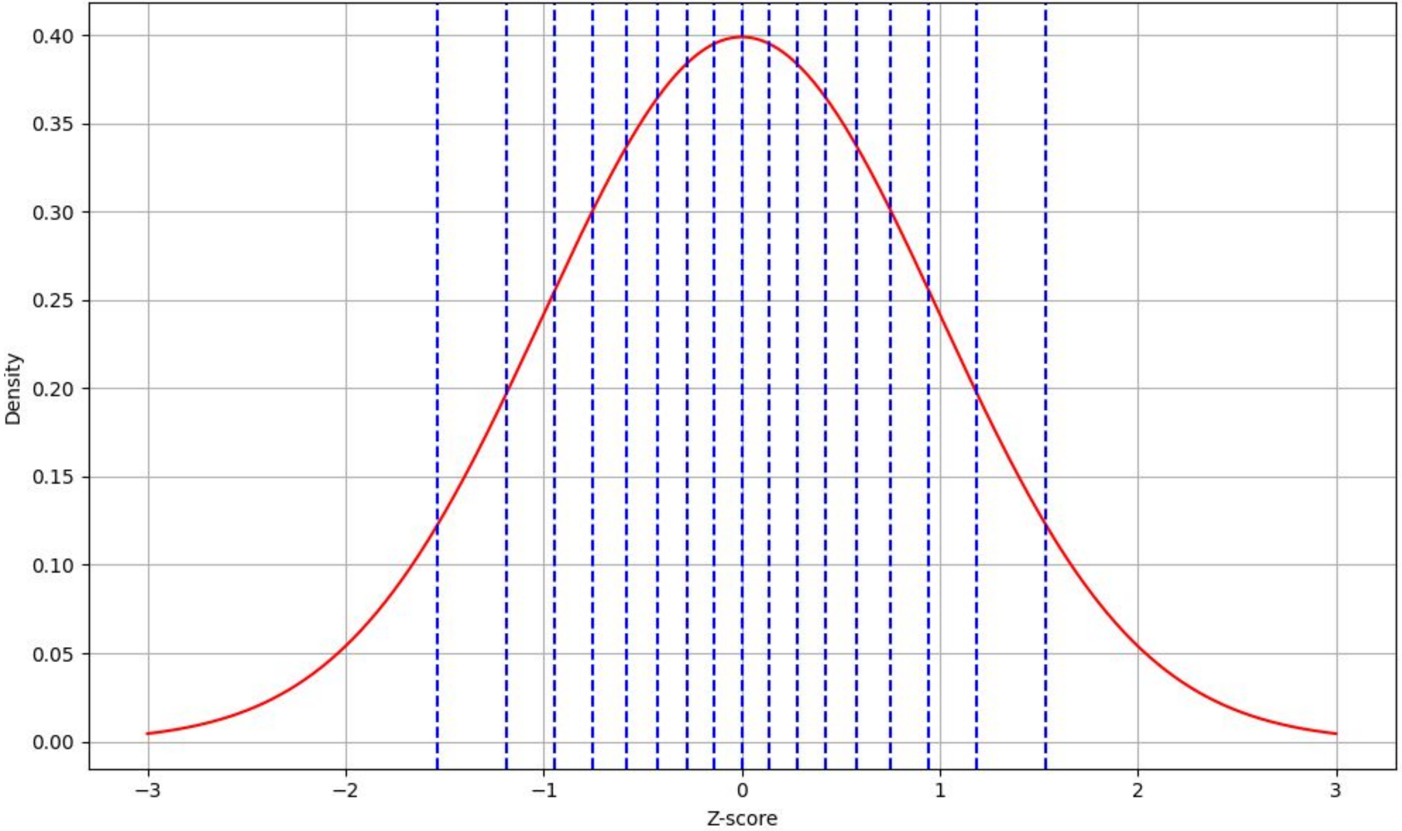
4-bit model

Quality 

What 4-bit data type is information theoretically optimal?



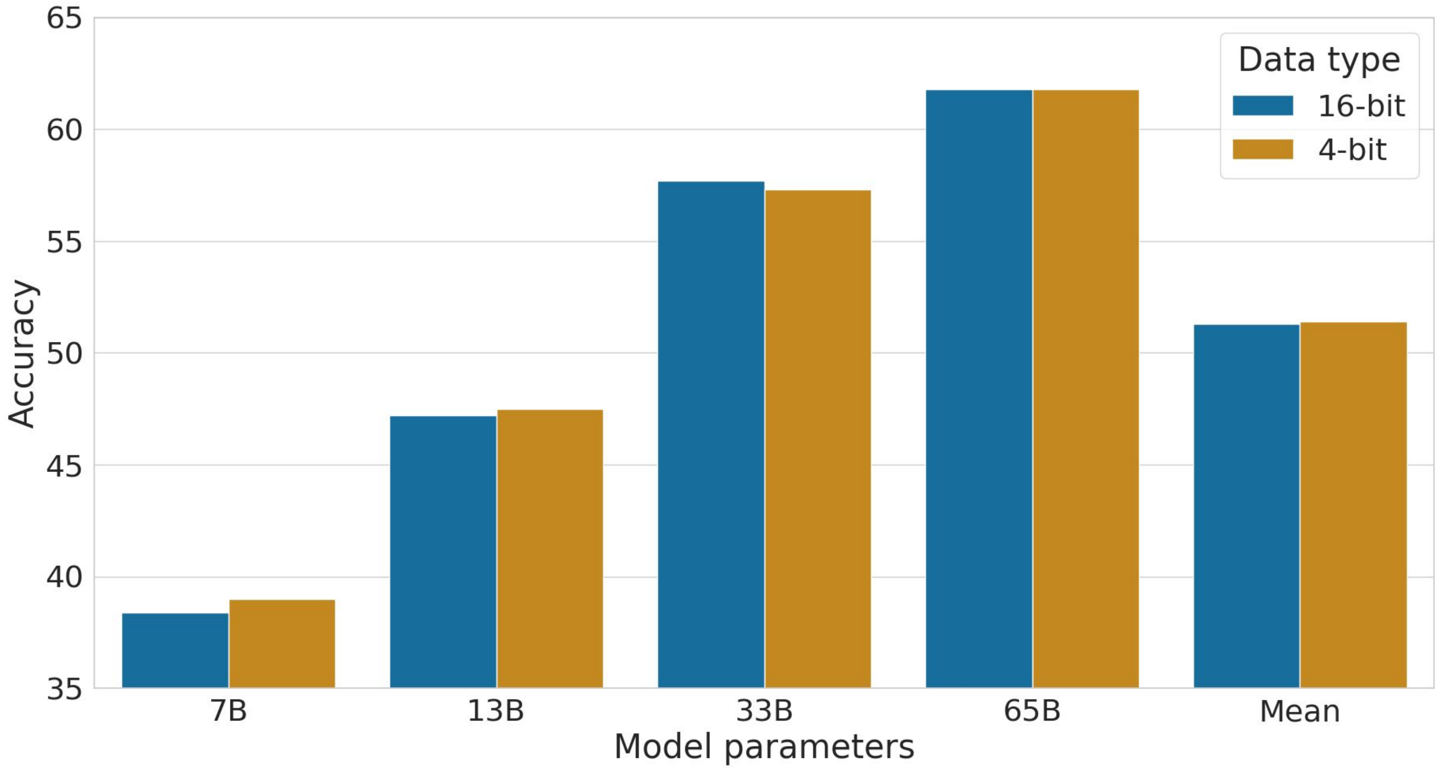
4-bit NormalFloat (NF4) an information-theoretically optimal data type for normal distributions



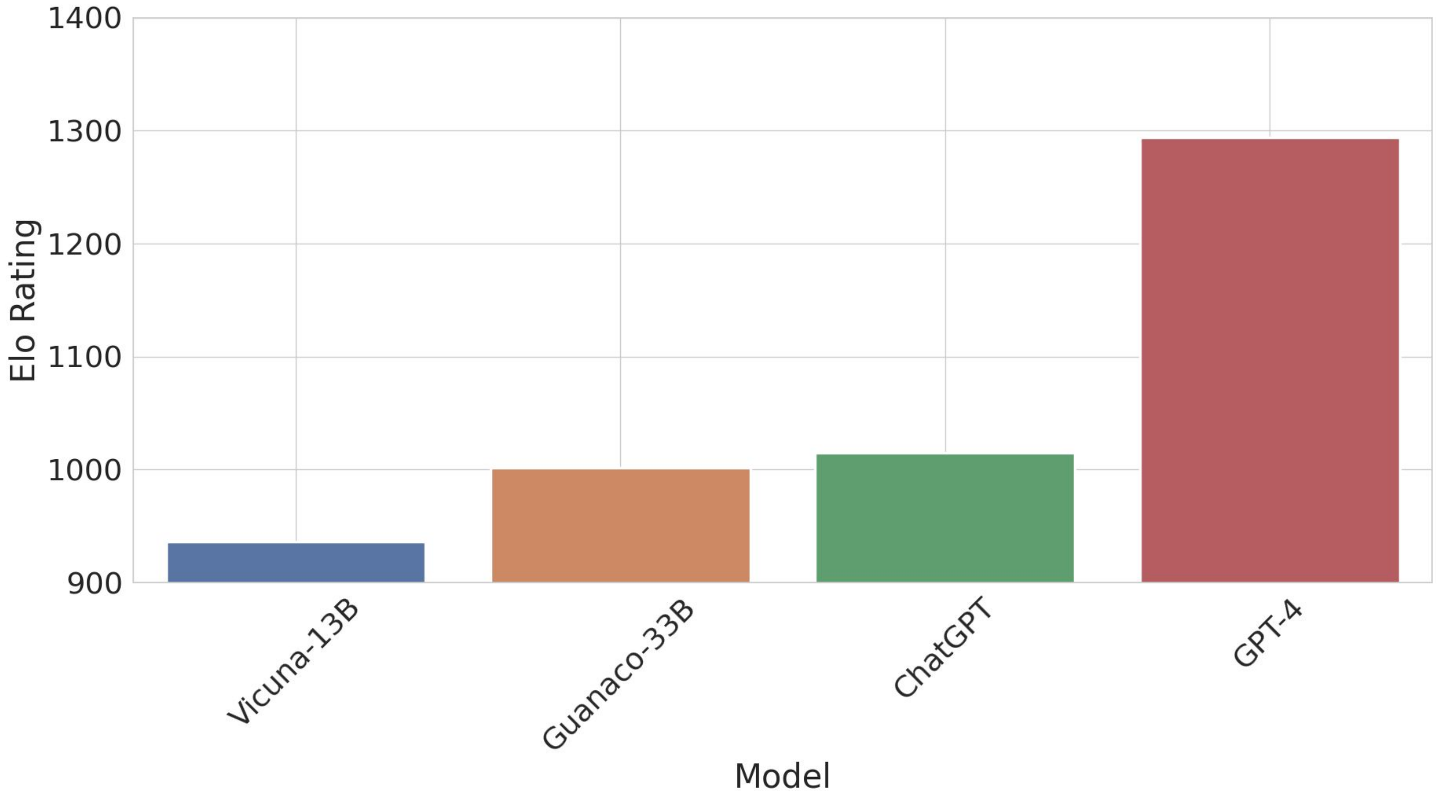
QLoRA systems contributions

Results

QLoRA recovers lost performance through fine-tuning



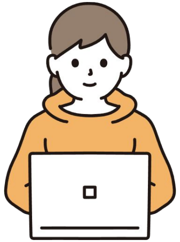
4-bit Guanaco: A ChatGPT-quality 4-bit chatbot finetuned in 24h on a single GPU



Take-away

4-bit finetuning is possible by passing gradients through a 4-bit neural network to 16-bit adapters.

Accessibility challenges of foundation models



Using foundation models

Finetuning foundation models

Quantization: companies vs users

Inference efficiency

Prefill: multiple tokens are multiplied with each weight matrix. Efficient even at small batch sizes

Decoding: generating token-by-token. Only Efficient at large batch sizes.

Companies optimize for decoding tokens/$. Users optimize token/second.

Decoding speed interactive learning

1. Batch size 1, fast for one user, but very wasteful in terms of GPU resources
2. Larger batch sizes yield about the same tokens/sec/user, but more efficient
3. MFU only very high with large batch sizes, but difficult to achieve for a single user. KV-cache size is a limiting factor.
4. Multiple GPUs only fast if one has a fast connection between GPUs
5. Quantization only efficient for small batch sizes