Distillation, quantization, and pruning

CS 5624: Natural Language Processing

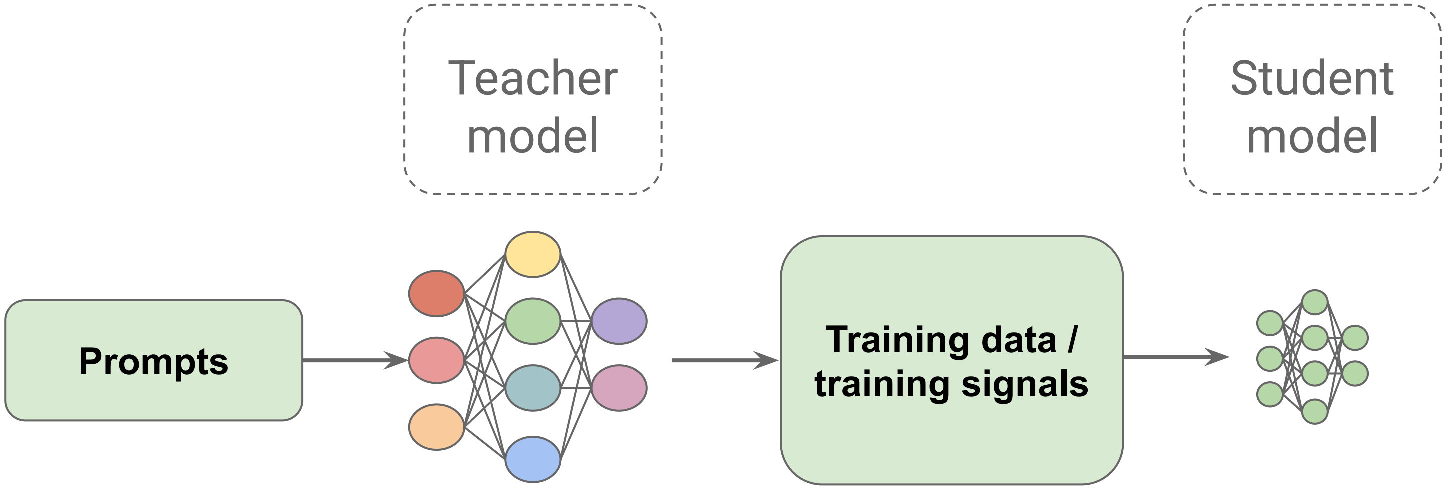
Spring 2025

https://tuvllms.github.io/nlp-spring-2025

Tu Vu

VIRGINIA TECH

Knowledge distillation



Pros and cons of knowledge distillation

Distilling the Knowledge in a Neural Network

Geoffrey Hinton[*†]
Google Inc.
Mountain View
geoffhinton@google.com

Oriol Vinyals[†]
Google Inc.
Mountain View
vinyals@google.com

Jeff Dean
Google Inc.
Mountain View
jeff@google.com

DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter

Victor SANH, Lysandre DEBUT, Julien CHAUMOND, Thomas WOLF
Hugging Face
{victor,lysandre,julien,thomas}@huggingface.co

$$\text{Loss} = \lambda_{\text{ce}} \cdot \mathcal{L}_{\text{ce}} + \lambda_{\text{kd}} \cdot \mathcal{L}_{\text{kd}}$$

$$\text{Loss} = \lambda_{\text{ce}} \cdot \left(-\sum_{i=1}^{N} y_i \log(p_i) \right) + \lambda_{\text{kd}} \cdot D_{\text{KL}}(q_{\text{teacher}}(x) \| q_{\text{student}}(x))$$

Where:

- y_i is the true label for token i,
- p_i is the predicted probability for the correct token for token i,
- N is the number of tokens,
- $D_{\text{KL}}(q_{\text{teacher}}(x) \| q_{\text{student}}(x))$ is the Kullback-Leibler divergence between the teacher and student models' probability distributions,
- $q_{\text{teacher}}(x)$ and $q_{\text{student}}(x)$ are the output probability distributions from the teacher and student models, respectively,
- λ_{ce} and λ_{kd} are the weighting hyperparameters for the cross-entropy and knowledge distillation losses, respectively.

Assume two different distributions for predicting the next word:

- P (from Model 1):
  - *mat* → 0.7
  - *floor* → 0.2
  - *chair* → 0.1
- Q (from Model 2):
  - *mat* → 0.5
  - *floor* → 0.3
  - *chair* → 0.2

Kullback–Leibler (KL) Divergence Calculation

KL divergence measures how much P diverges from Q:

$$D_{KL}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}$$

Substituting the values:

$$D_{KL}(P||Q) = 0.7 \log \frac{0.7}{0.5} + 0.2 \log \frac{0.2}{0.3} + 0.1 \log \frac{0.1}{0.2}$$

Training loss The student is trained with a distillation loss over the soft target probabilities of the teacher: $L_{ce} = \sum_i t_i * \log(s_i)$ where t_i (resp. s_i) is a probability estimated by the teacher (resp. the student). This objective results in a rich training signal by leveraging the full teacher distribution. Following Hinton et al. [2015] we used a *softmax-temperature*: $p_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$ where T controls the smoothness of the output distribution and z_i is the model score for the class i. The same temperature T is applied to the student and the teacher at training time, while at inference, T is set to 1 to recover a standard *softmax*.

The final training objective is a linear combination of the distillation loss L_{ce} with the supervised training loss, in our case the *masked language modeling* loss L_{mlm} [Devlin et al., 2018]. We found it beneficial to add a *cosine embedding* loss (L_{cos}) which will tend to align the directions of the student and teacher hidden states vectors.

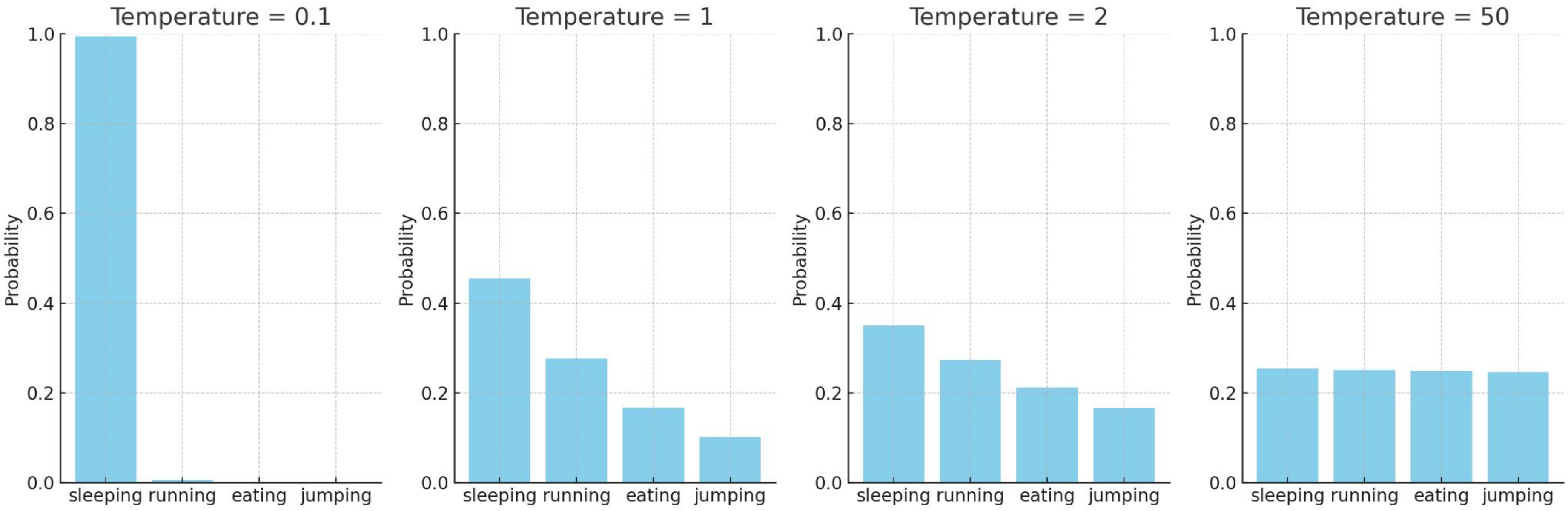
Temperature

$$P(y_i|\mathbf{x}) = \frac{\exp\left(\frac{z_i}{T}\right)}{\sum_j \exp\left(\frac{z_j}{T}\right)}$$

where:

- $P(y_i|\mathbf{x})$ is the probability of token y_i given the input $\mathbf{x}$
- z_i is the logit (raw score before softmax) for token y_i
- T is the temperature (where $T = 1$ is the default, and $T < 1$ reduces randomness while $T > 1$ increases randomness)
- The summation in the denominator is over all possible tokens j

Temperature (cont'd)



peaked distribution (more deterministic)

flatter distribution (more randomness)

DistillBERT reduces BERT's size by 40%, while retaining 97% of its performance and being 60% faster

Model	**Score**	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST-2	STS-B	WNLI
ELMo	68.7	44.1	68.6	76.6	71.1	86.2	53.4	91.5	70.4	56.3
BERT-base	79.5	56.3	86.7	88.6	91.8	89.6	69.3	92.7	89.0	53.5
DistilBERT	77.0	51.3	82.2	87.5	89.2	88.5	59.9	91.3	86.9	56.3

Model	IMDb (acc.)	SQuAD (EM/F1)
BERT-base	93.46	81.2/88.5
DistilBERT	92.82	77.7/85.8
DistilBERT (D)	-	79.1/86.9

Model	# param. (Millions)	Inf. time (seconds)
ELMo	180	895
BERT-base	110	668
DistilBERT	66	410

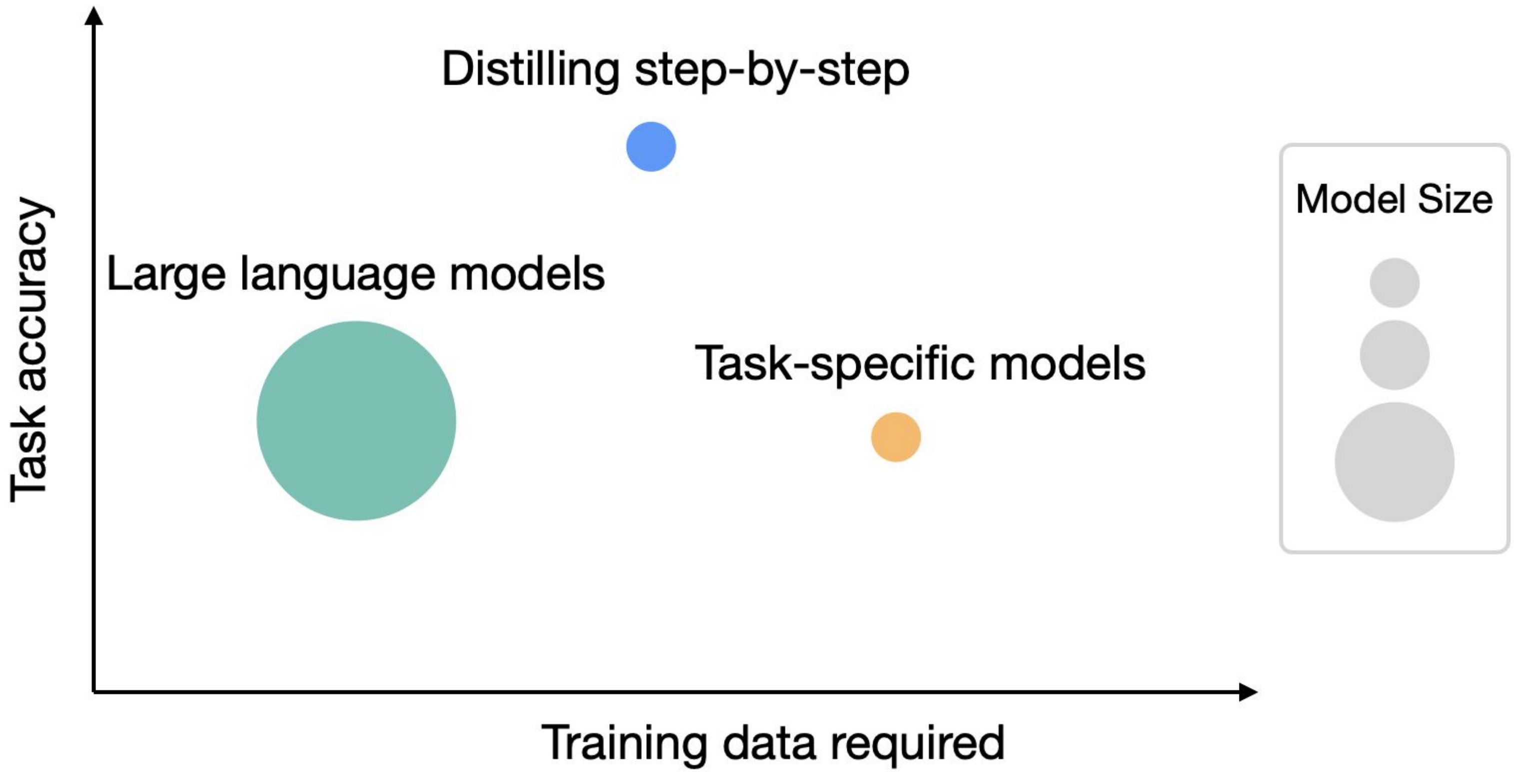
Distilling Step-by-Step! Outperforming Larger Language Models with Less Training Data and Smaller Model Sizes

Cheng-Yu Hsieh[1,*], Chun-Liang Li[2], Chih-Kuan Yeh[3], Hootan Nakhost[2], Yasuhisa Fujii[3], Alexander Ratner[1], Ranjay Krishna[1], Chen-Yu Lee[2], Tomas Pfister[2]
[1]University of Washington, [2]Google Cloud AI Research, [3]Google Research
cydhsieh@cs.washington.edu

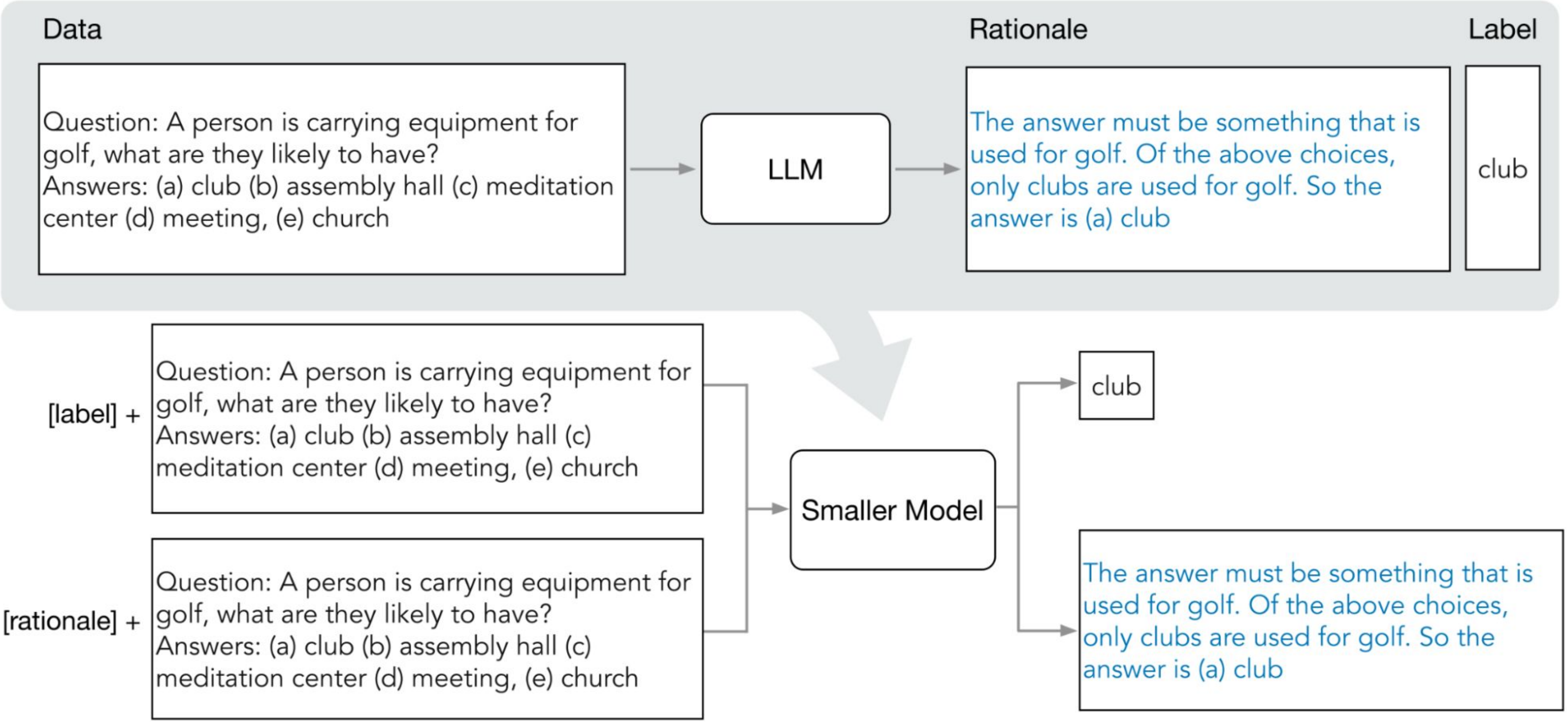
Distilling Step-by-Step! Outperforming Larger Language Models with Less Training Data and Smaller Model Sizes

Cheng-Yu Hsieh[1,*], **Chun-Liang Li**[2], **Chih-Kuan Yeh**[3], **Hootan Nakhost**[2], **Yasuhisa Fujii**[3], **Alexander Ratner**[1], **Ranjay Krishna**[1], **Chen-Yu Lee**[2], **Tomas Pfister**[2]
[1]University of Washington, [2]Google Cloud AI Research, [3]Google Research
cydhsieh@cs.washington.edu

Enabling a 770M parameter T5 model to outperform the few-shot prompted 540B PaLM model



Distilling step-by-step



Leveraging few-shot CoT prompting to extract rationales from LLMs

Few-shot CoT

Question: Sammy wanted to go to where the people are. Where might he go?
Answer Choices: (a) populated areas, (b) race track, (c) desert, (d) apartment, (e) roadblock
Answer: The answer must be a place with a lot of people. Of the above choices, only populated areas have a lot of people. So the answer is (a) populated areas.

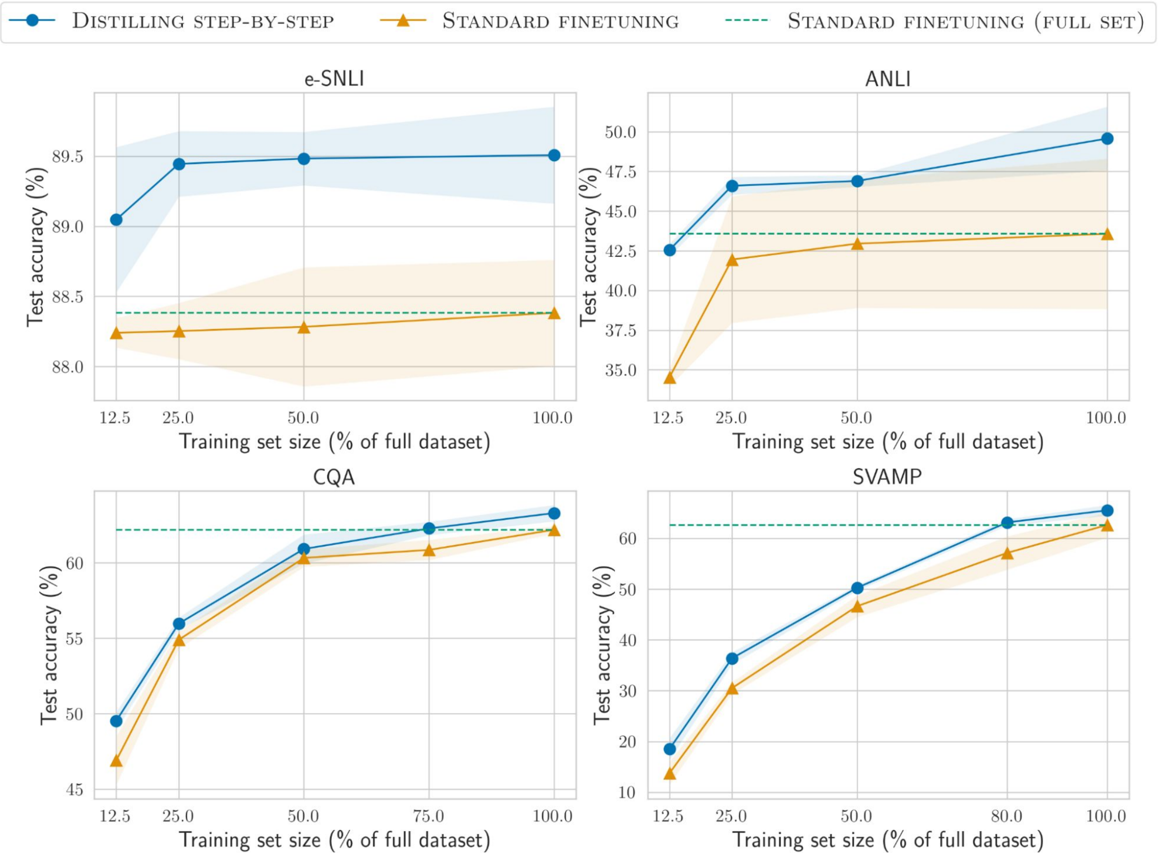
Input

Question: A person is carrying equipment for golf. What are they likely to have?
Answer Choices: (a) club, (b) assembly hall, (c) meditation center, (d) meeting, (e) church
Answer:

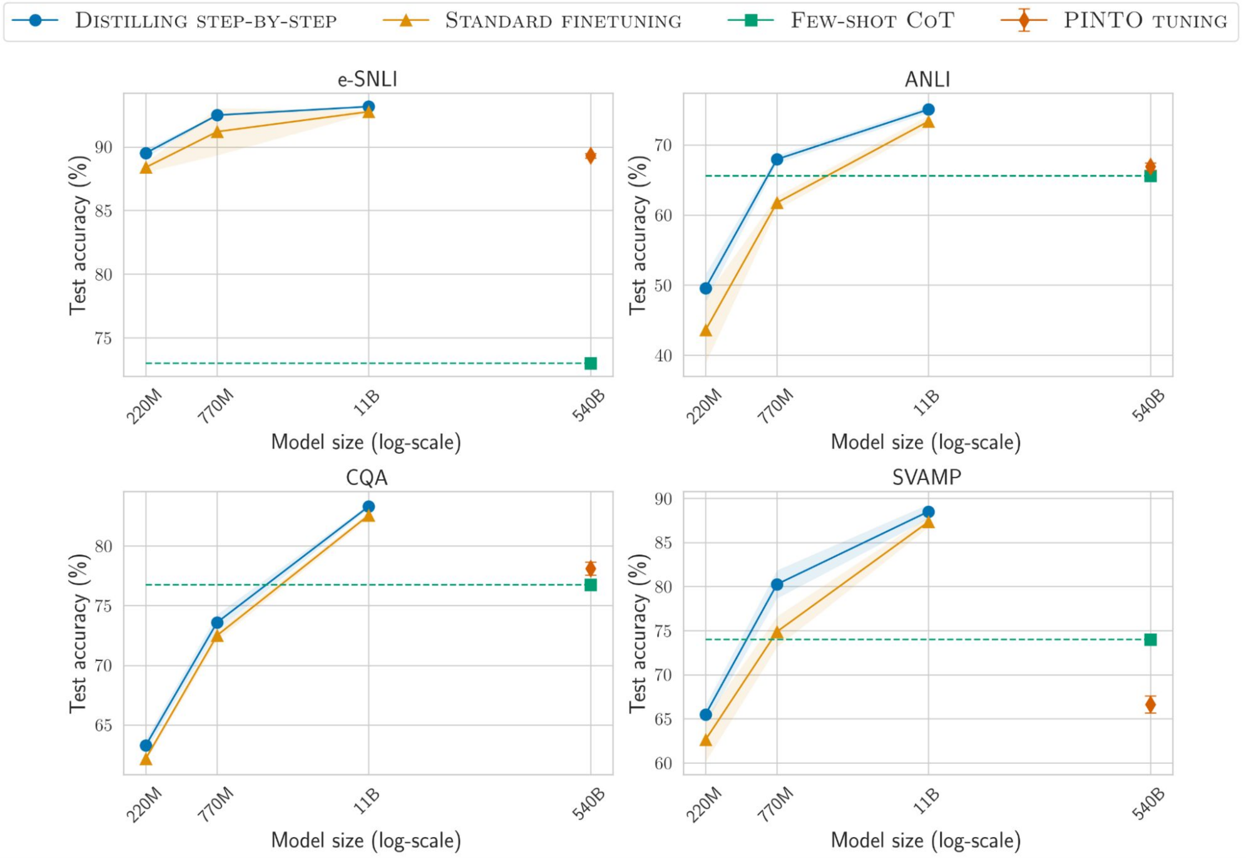
Output

The answer must be something that is used for golf. Of the above choices, only clubs are used for golf. So the answer is (a) club.

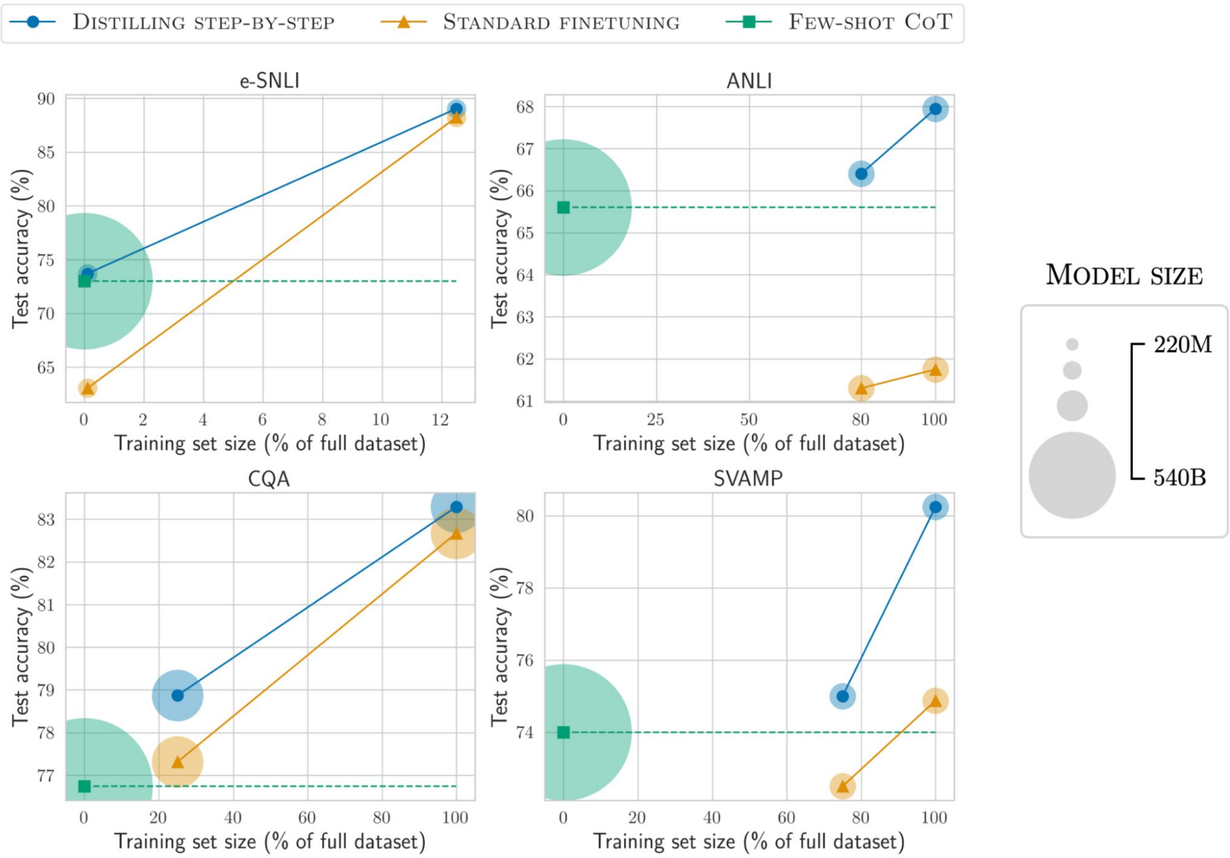
Less training data



Smaller deployed model size



Distilling step-by-step outperforms few-shot LLMs with smaller models using less data



DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning

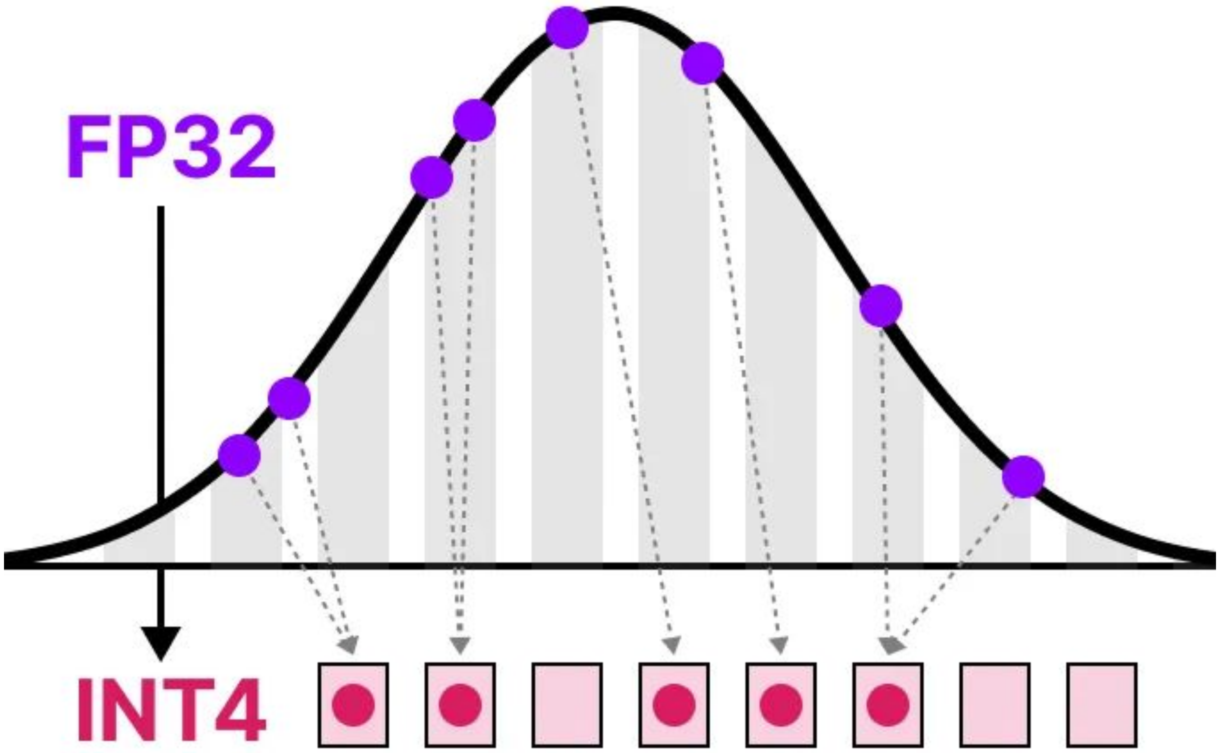
DeepSeek-AI

research@deepseek.com

Model	AIME 2024		MATH-500	GPQA Diamond	LiveCode Bench	CodeForces
	pass@1	cons@64	pass@1	pass@1	pass@1	rating
GPT-4o-0513	9.3	13.4	74.6	49.9	32.9	759
Claude-3.5-Sonnet-1022	16.0	26.7	78.3	65.0	38.9	717
OpenAI-o1-mini	63.6	80.0	90.0	60.0	53.8	**1820**
QwQ-32B-Preview	50.0	60.0	90.6	54.5	41.9	1316
DeepSeek-R1-Distill-Qwen-1.5B	28.9	52.7	83.9	33.8	16.9	954
DeepSeek-R1-Distill-Qwen-7B	55.5	83.3	92.8	49.1	37.6	1189
DeepSeek-R1-Distill-Qwen-14B	69.7	80.0	93.9	59.1	53.1	1481
DeepSeek-R1-Distill-Qwen-32B	**72.6**	83.3	94.3	62.1	57.2	1691
DeepSeek-R1-Distill-Llama-8B	50.4	80.0	89.1	49.0	39.6	1205
DeepSeek-R1-Distill-Llama-70B	70.0	**86.7**	**94.5**	**65.2**	**57.5**	1633

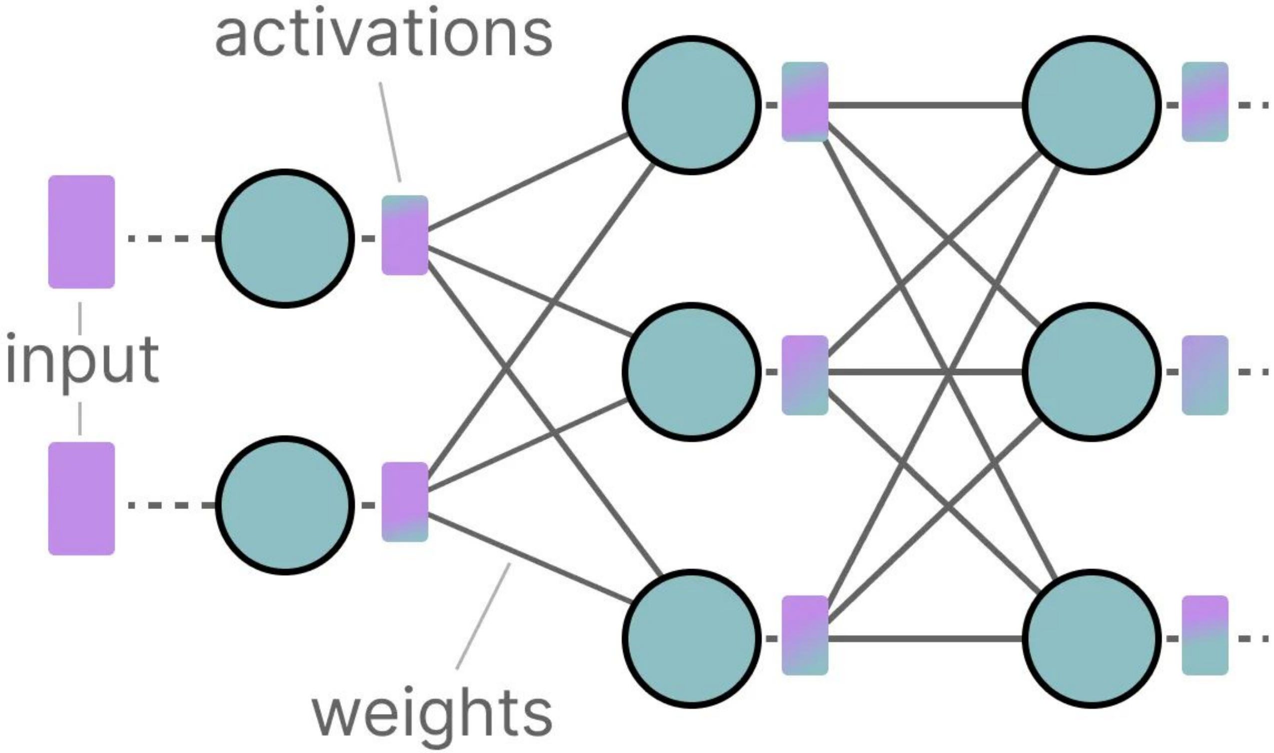
Table 5 | Comparison of DeepSeek-R1 distilled models and other comparable models on reasoning-related benchmarks.

Quantization



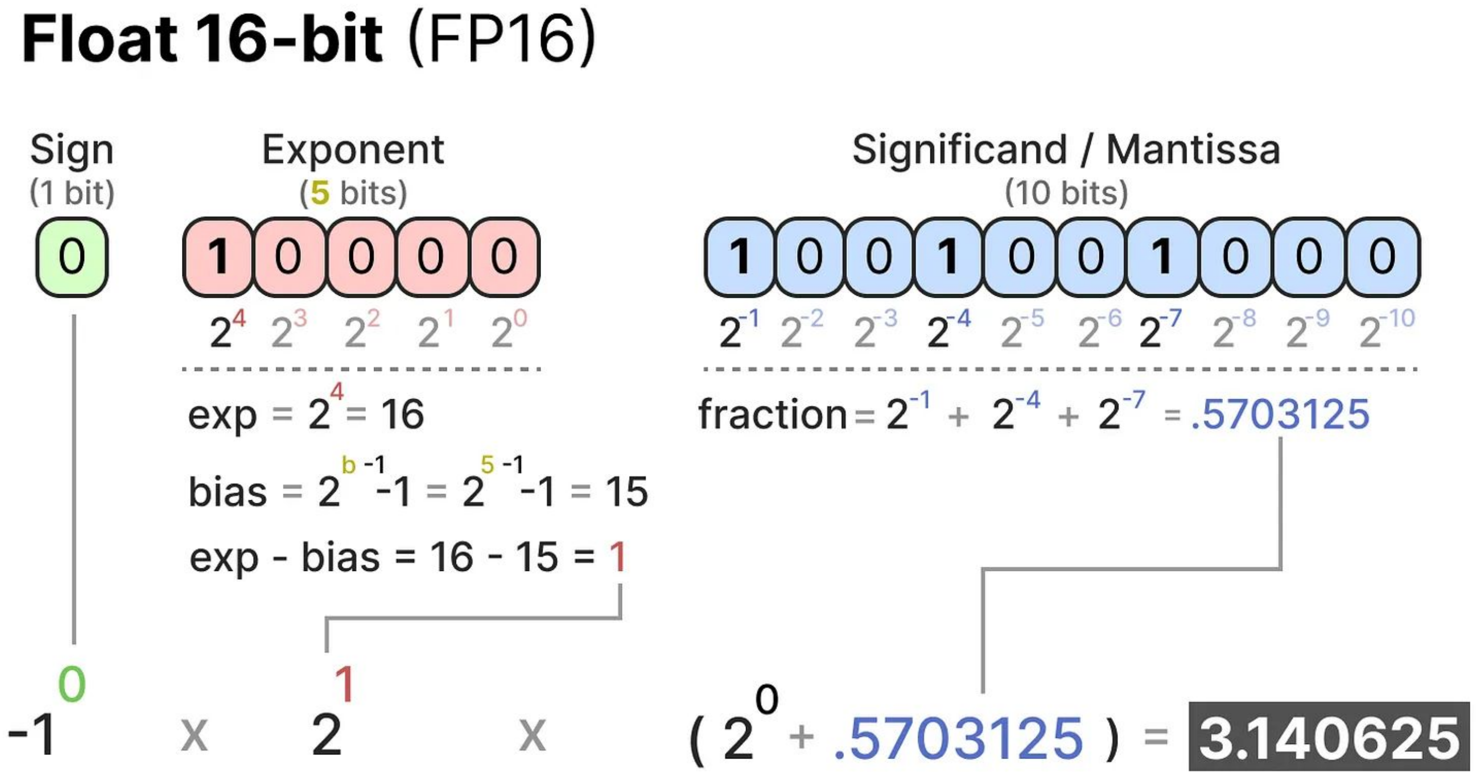
Maarten Grootendorst's blog: A Visual Guide to Quantization

Quantizing both the weights and activations



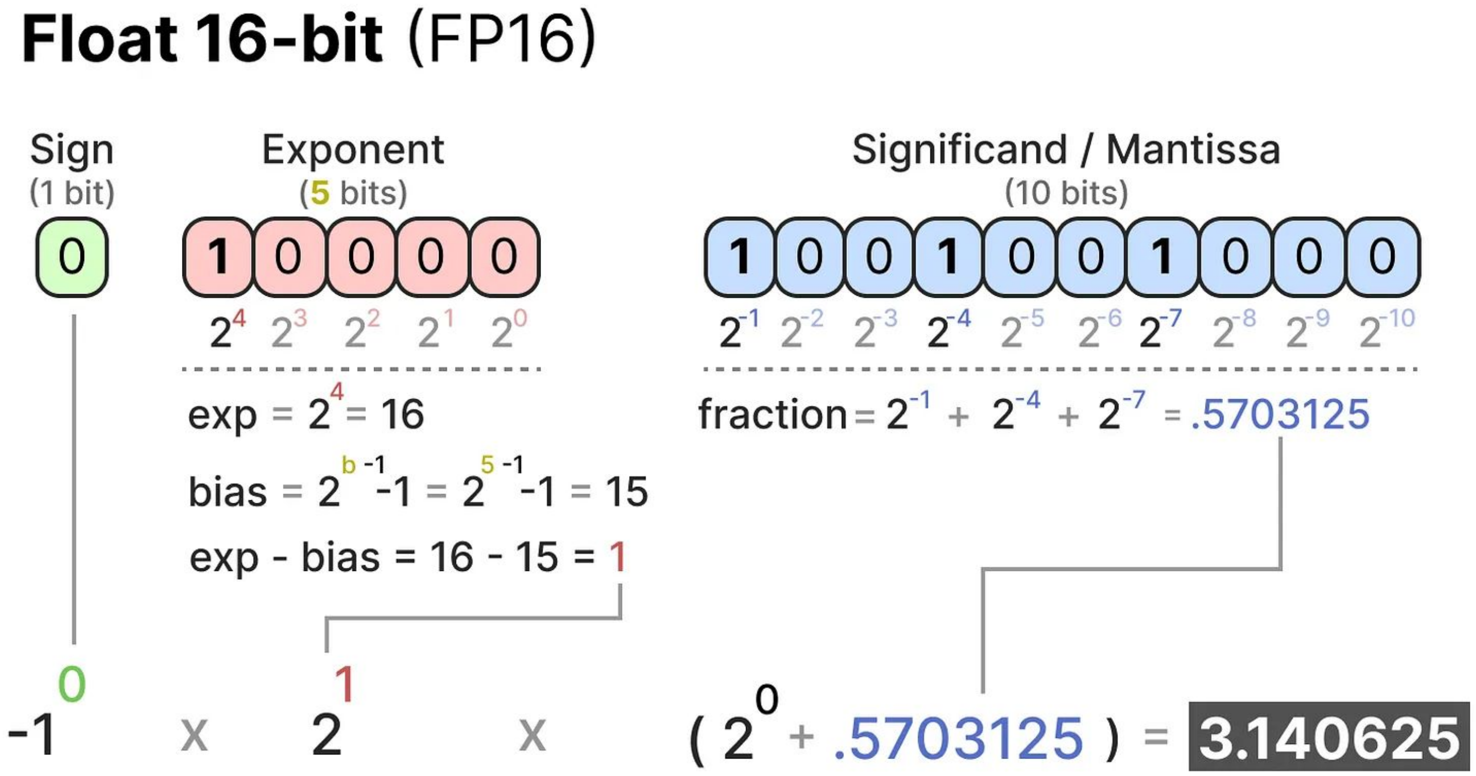
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

How to represent numerical values



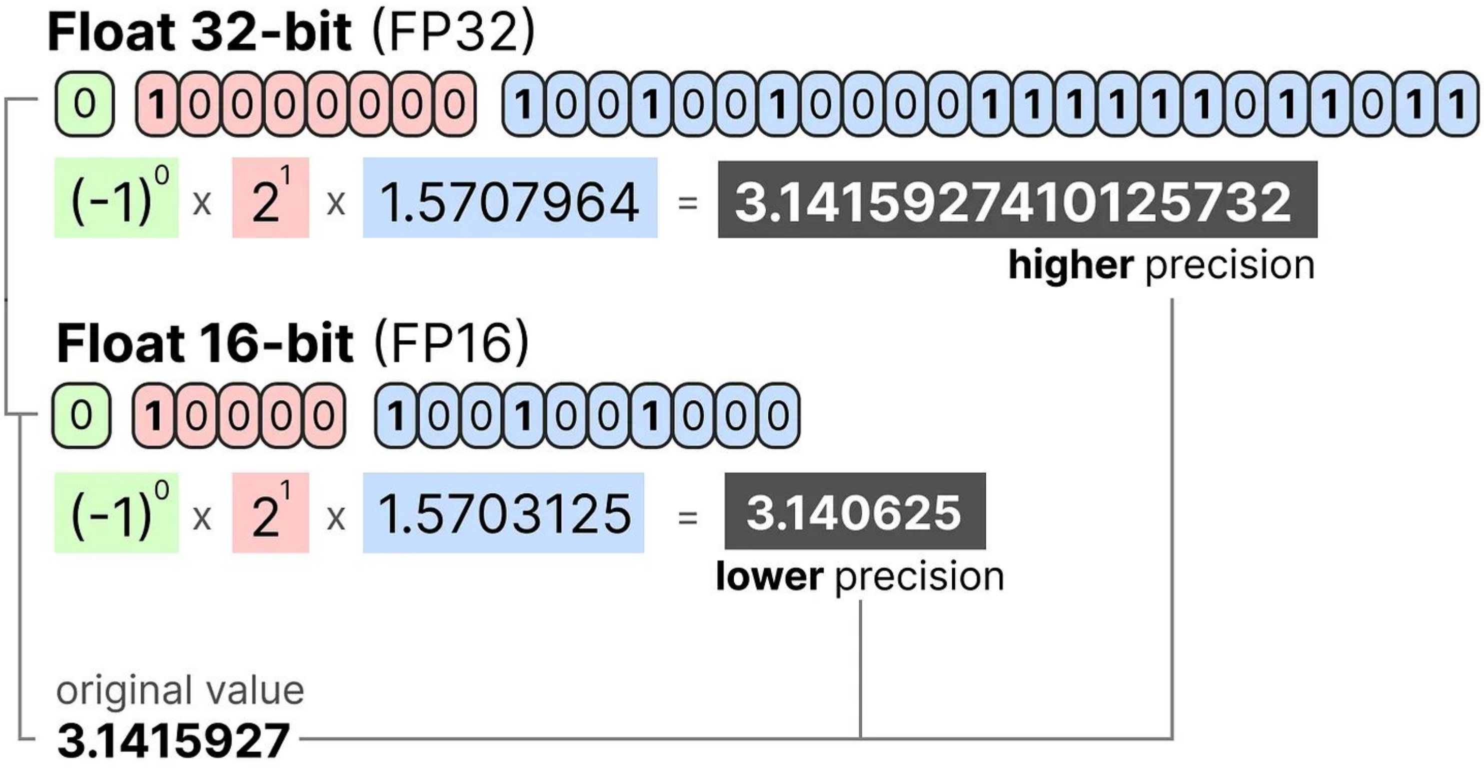
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

How to represent numerical values (cont'd)



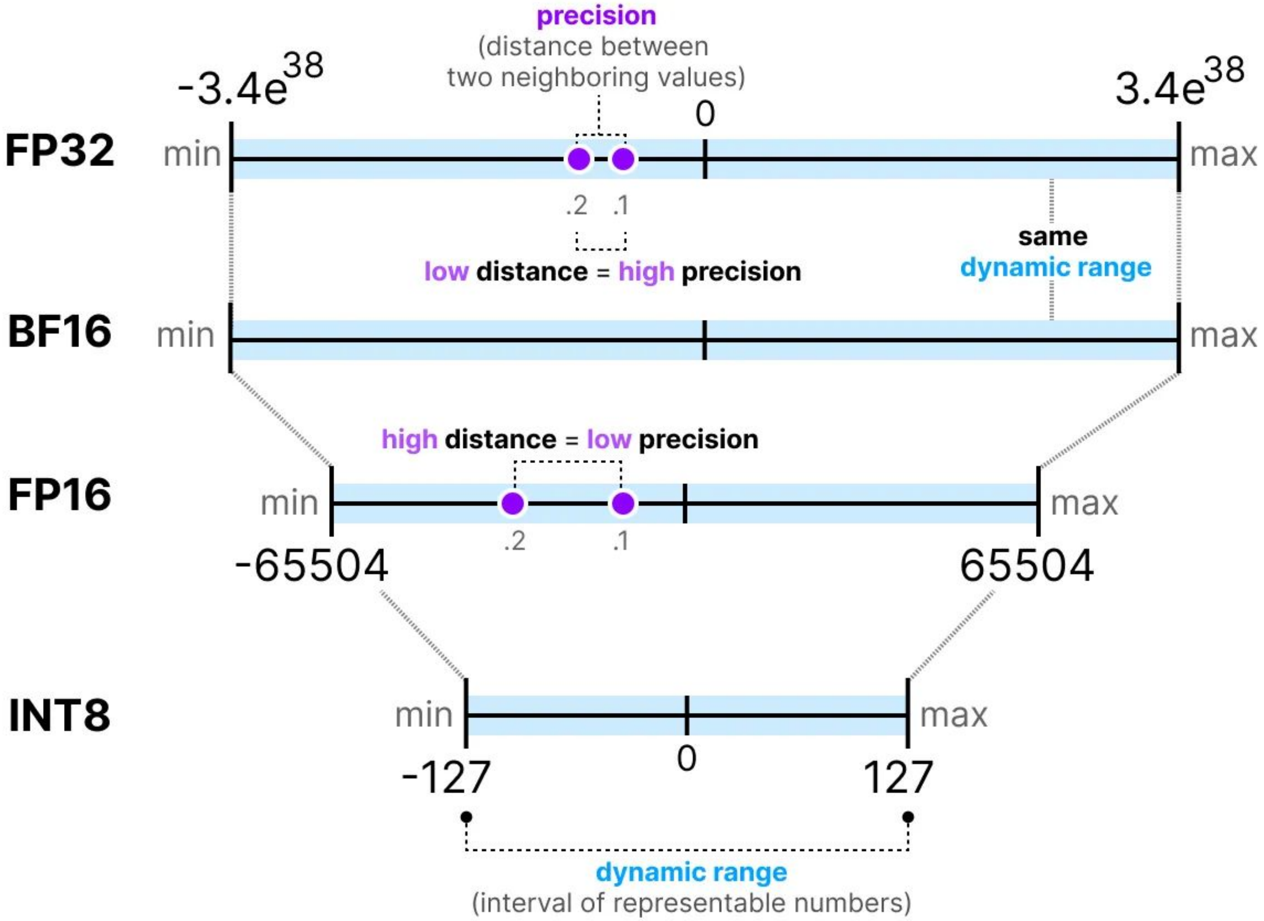
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

How to represent numerical values (cont'd)



Maarten Grootendorst's blog: A Visual Guide to Quantization

Memory constraints



[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Memory constraints (cont'd)

$$\text{memory} = \frac{\text{nr\_bits}}{8} \times \text{nr\_params}$$

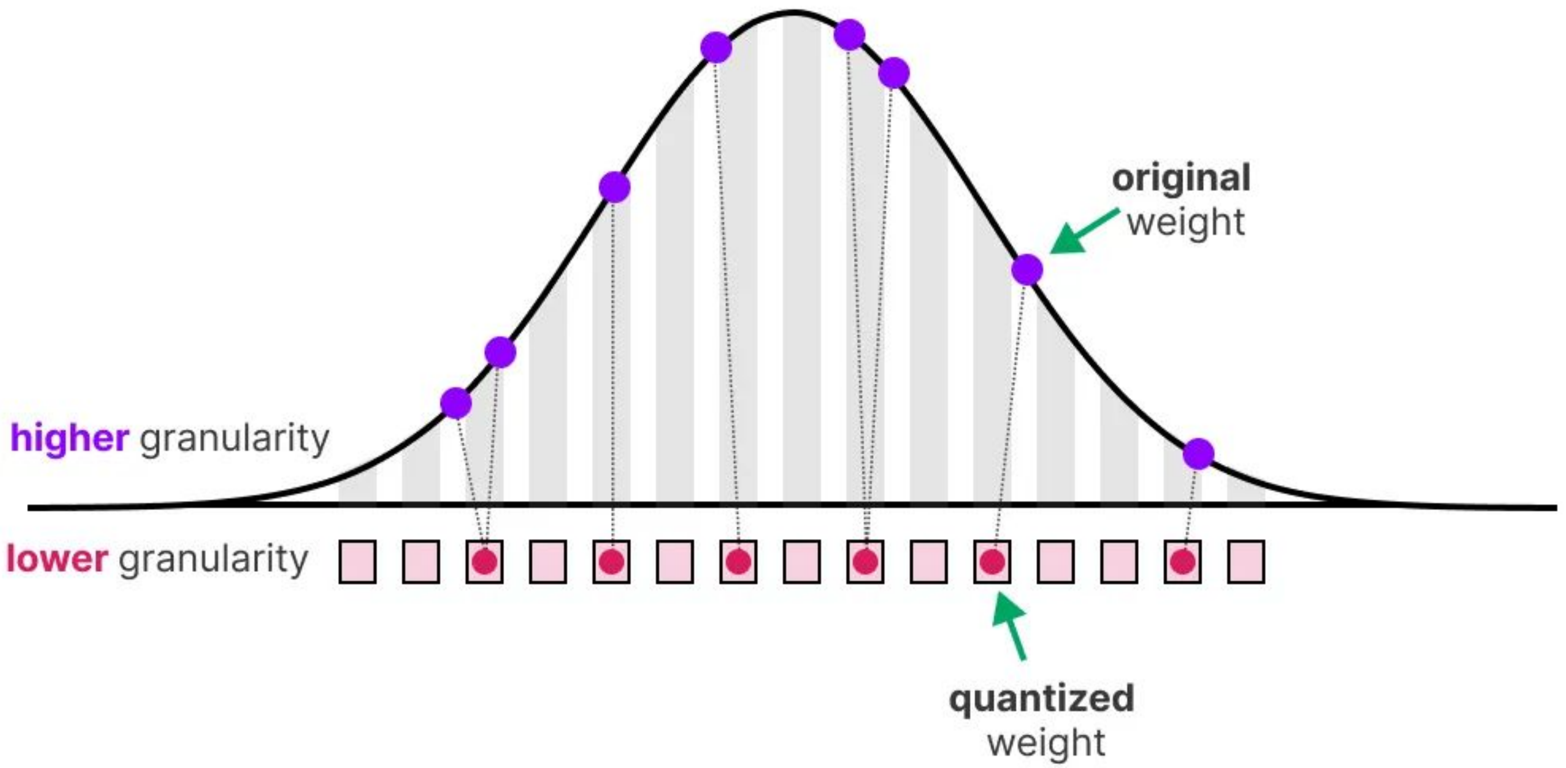
$$\textbf{64-bits} = \frac{64}{8} \times 70\text{B} \approx \textbf{560 GB}$$

$$\textbf{32-bits} = \frac{32}{8} \times 70\text{B} \approx \textbf{280 GB}$$

$$\textbf{16-bits} = \frac{16}{8} \times 70\text{B} \approx \textbf{140 GB}$$

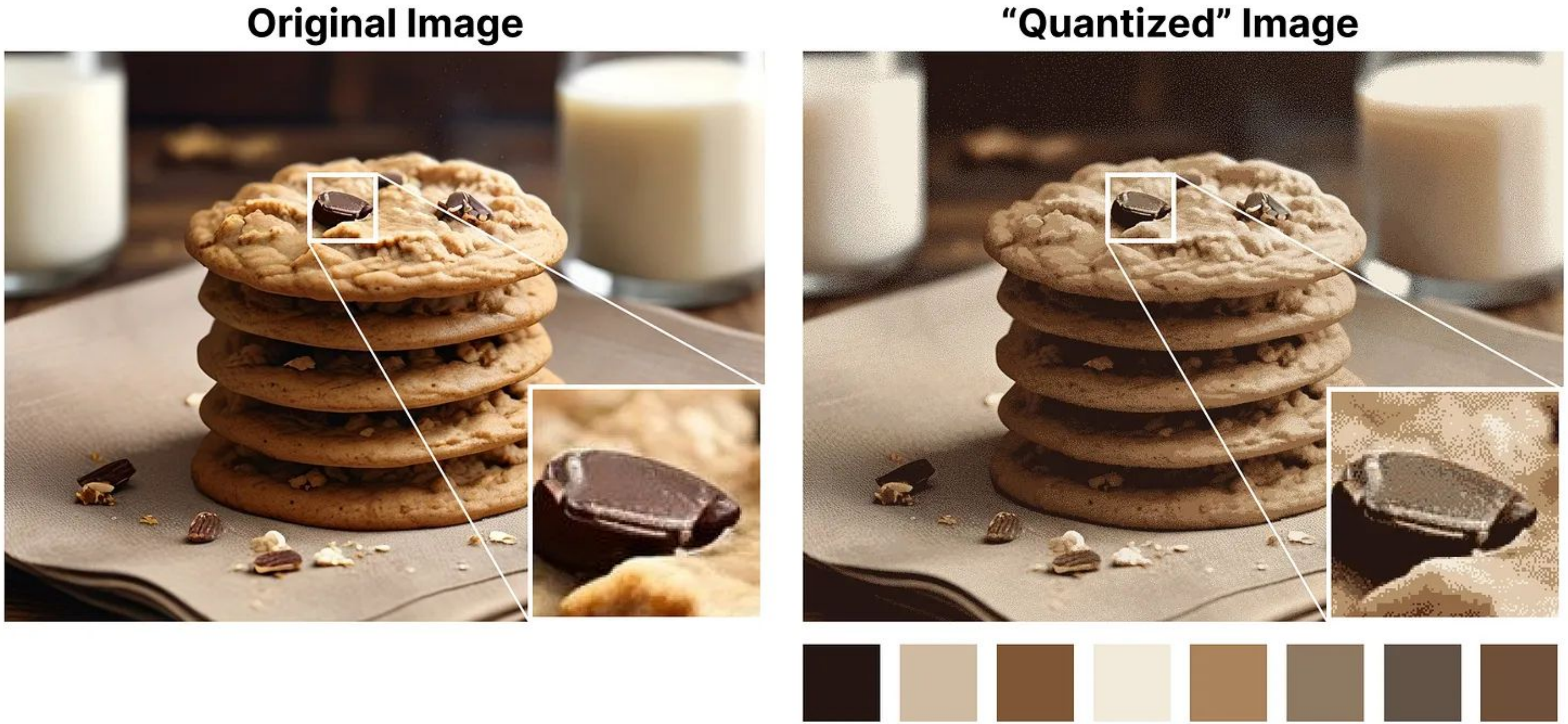
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Quantization



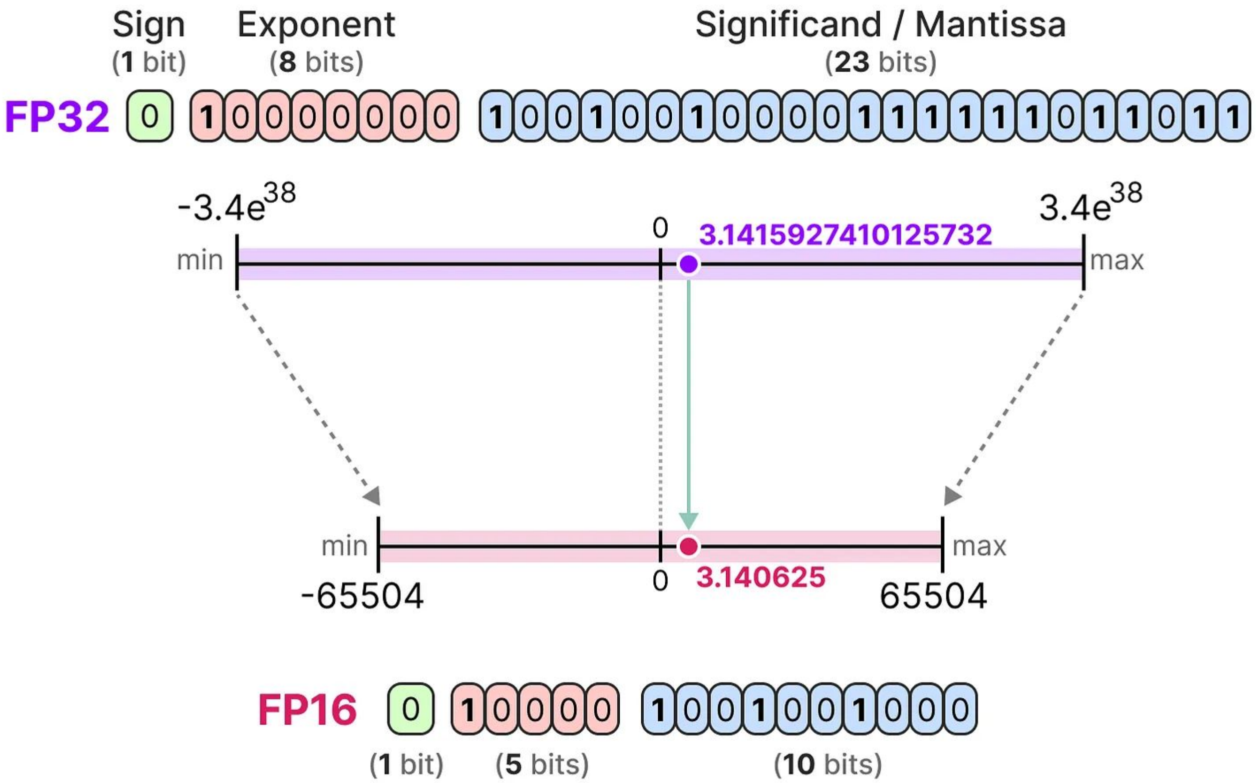
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Quantization



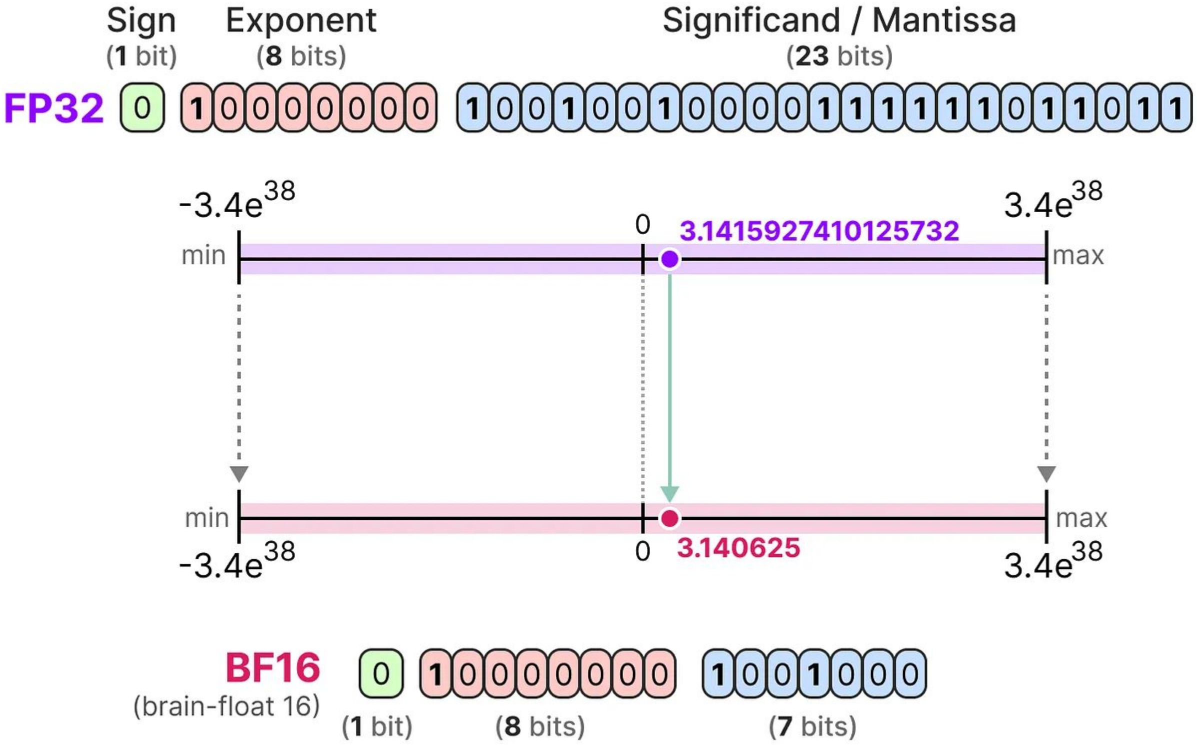
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Common data types: FP16 (half precision)



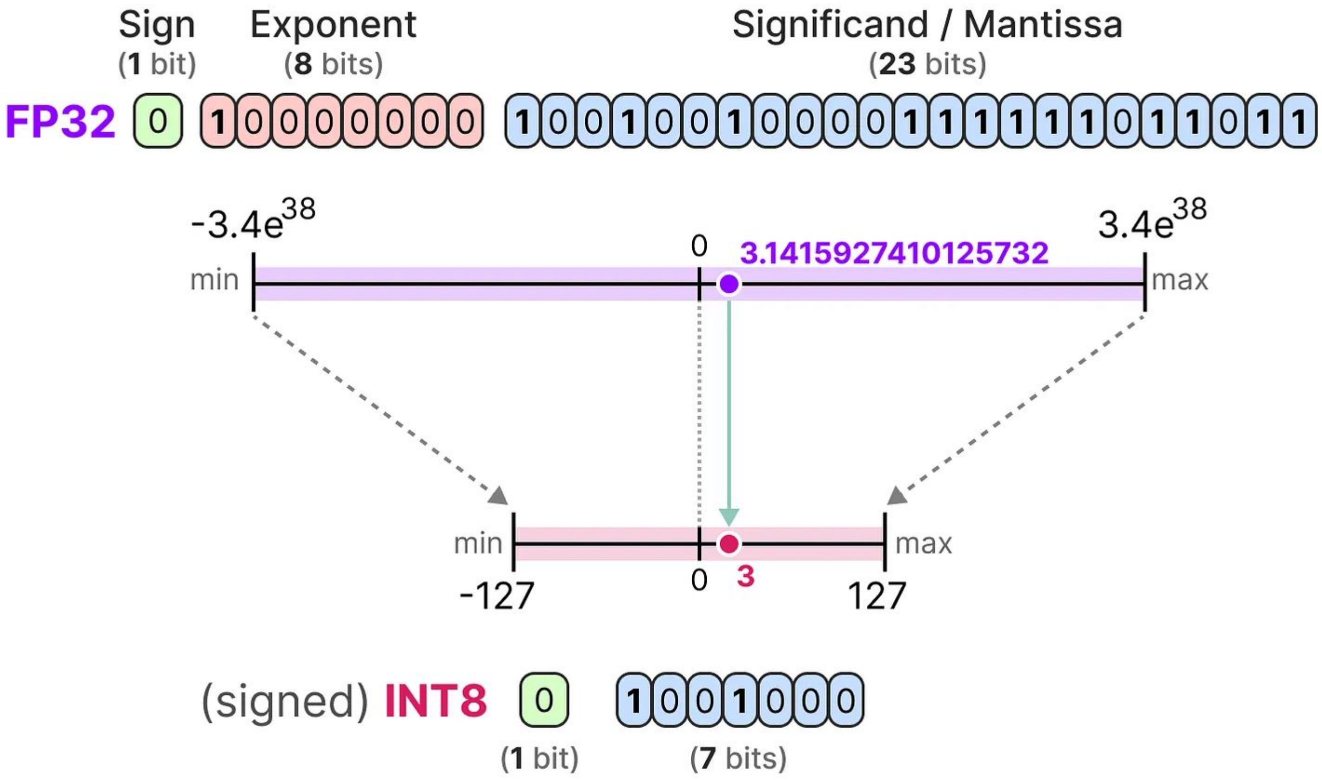
[Maarten Grootendorst's blog: A Visual Guide to Quantization]()

Common data types: BF16



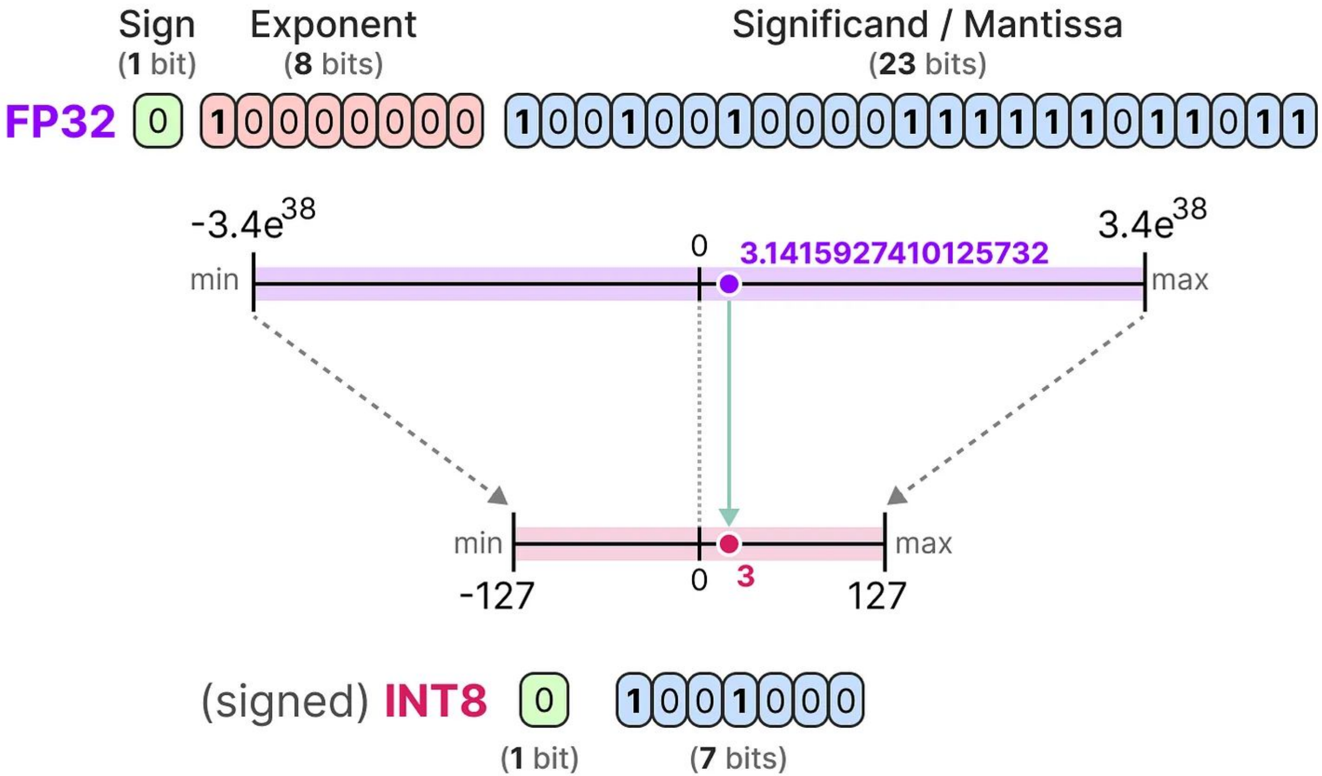
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Common data types: INT8



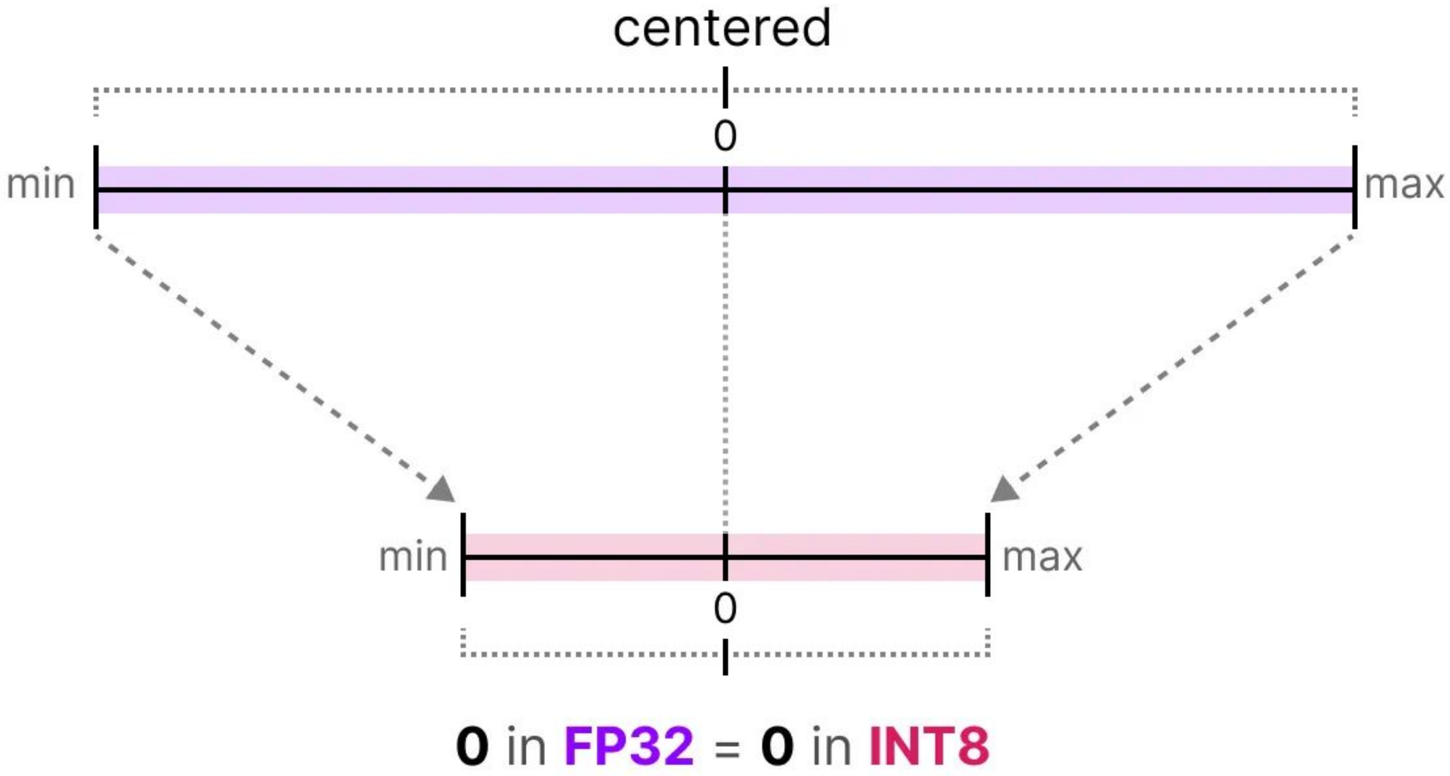
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Common data types: INT8



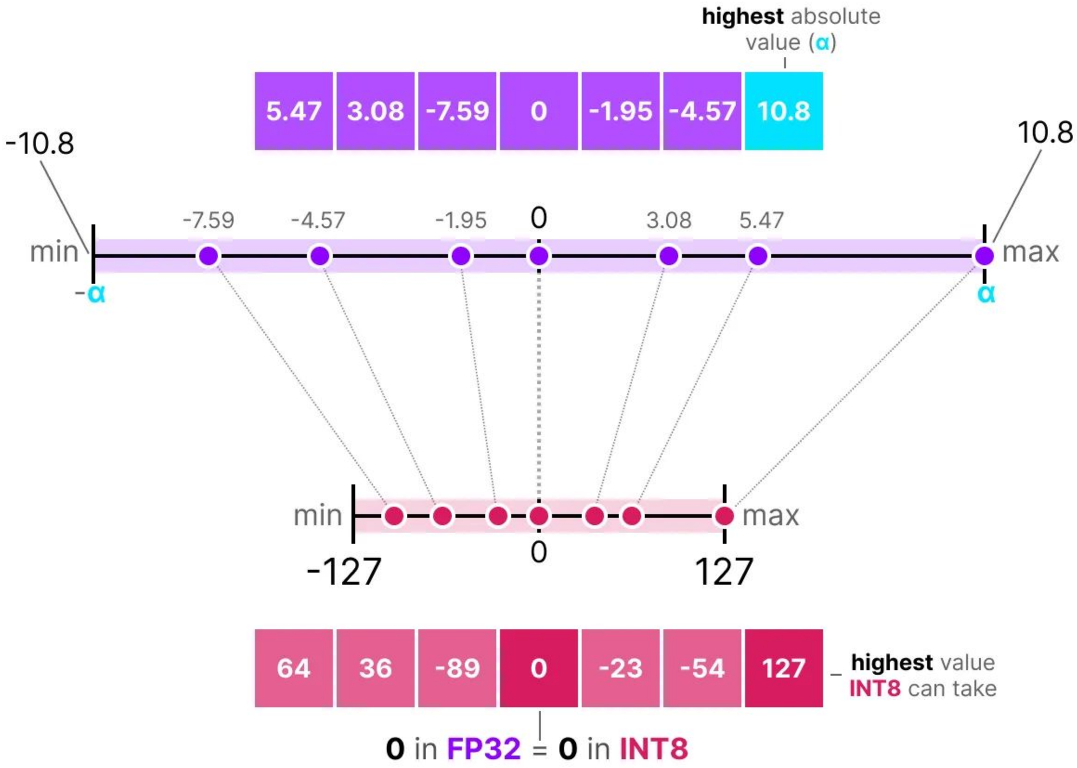
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Symmetric quantization



[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Absolute maximum (absmax) quantization



[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Absolute maximum (absmax) quantization

We first calculate a scale factor (*s*) using:

- ***b*** is the number of bytes that we want to quantize to (8),
- **α** is the *highest* absolute value,

Then, we use the ***s*** to quantize the input ***x***:

$$s = \frac{2^{b-1} - 1}{\alpha} \qquad \text{(scale factor)}$$

$$x_{\text{quantized}} = \text{round}\left(s \cdot x\right) \qquad \text{(quantization)}$$

Filling in the values would then give us the following:

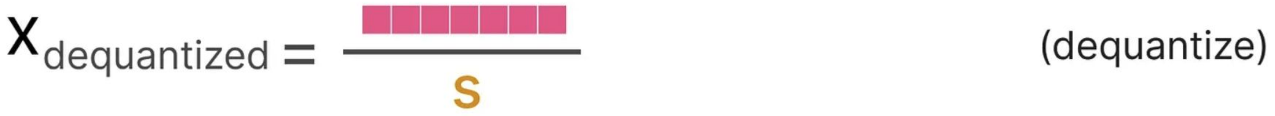
$$s = \frac{127}{10.8} = 11.76 \qquad \text{(scale factor)}$$

$$x_{\text{quantized}} = \text{round}\left(11.76 \cdot \right) \qquad \text{(quantization)}$$

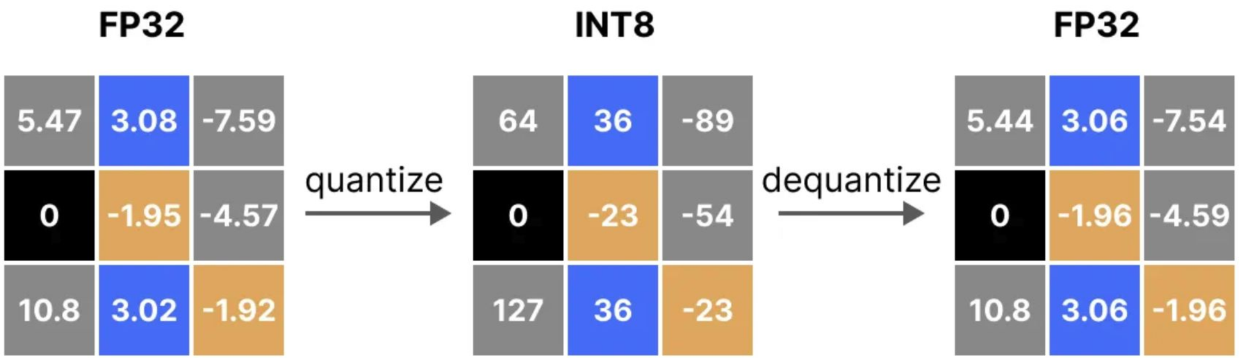
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Dequantization

To retrieve the original FP32 values, we can use the previously calculated *scaling factor* (*s*) to *dequantize* the quantized values.



Applying the quantization and then dequantization process to retrieve the original looks as follows:



[Maarten Grootendorst's blog: A Visual Guide to Quantization]()

Dequantization

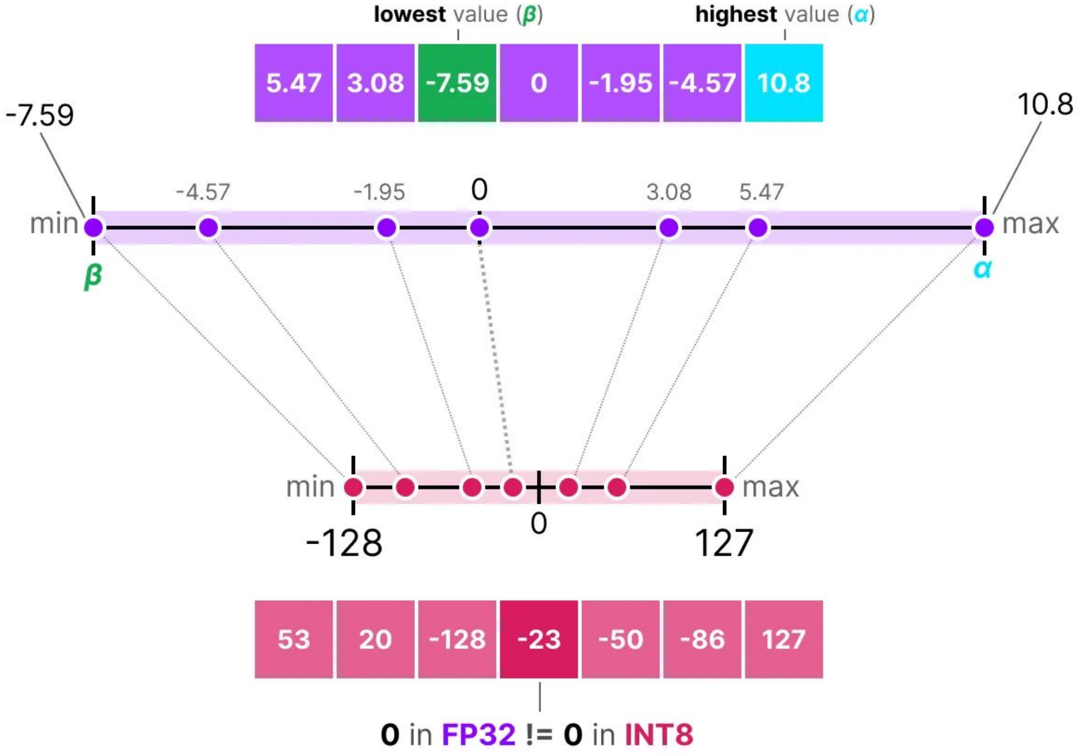
$$\underset{\text{(original)}}{\overset{\textbf{FP32}}{\begin{bmatrix} 5.47 & 3.08 & -7.59 \\ 0 & -1.95 & -4.57 \\ 10.8 & 3.02 & -1.92 \end{bmatrix}}} - \underset{\text{(dequantized)}}{\overset{\textbf{FP32}}{\begin{bmatrix} 5.44 & 3.06 & -7.54 \\ 0 & -1.96 & -4.59 \\ 10.8 & 3.06 & -1.96 \end{bmatrix}}} = \underset{\text{error}}{\overset{\text{Quantization}}{\begin{bmatrix} .03 & .02 & .05 \\ 0 & -.01 & -.02 \\ 0 & -.04 & -.04 \end{bmatrix}}}$$

[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Dequantization

[Maarten Grootendorst's blog: A Visual Guide to Quantization](#)

Asymmetric quantization



[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Asymmetric quantization (cont'd)

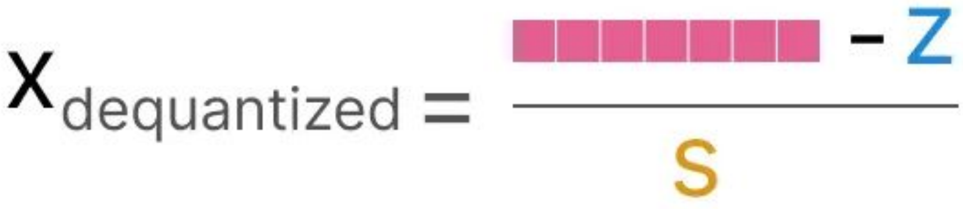
$$s = \frac{128 - -127}{\alpha - \beta} \qquad \text{(scale factor)}$$

$$z = \text{round}\left(-s \cdot \beta\right) - 2^{b-1} \qquad \text{(zeropoint)}$$

$$X_{\text{quantized}} = \text{round}\left(s \cdot X + z\right) \qquad \text{(quantization)}$$

[Maarten Grootendorst's blog: A Visual Guide to Quantization]

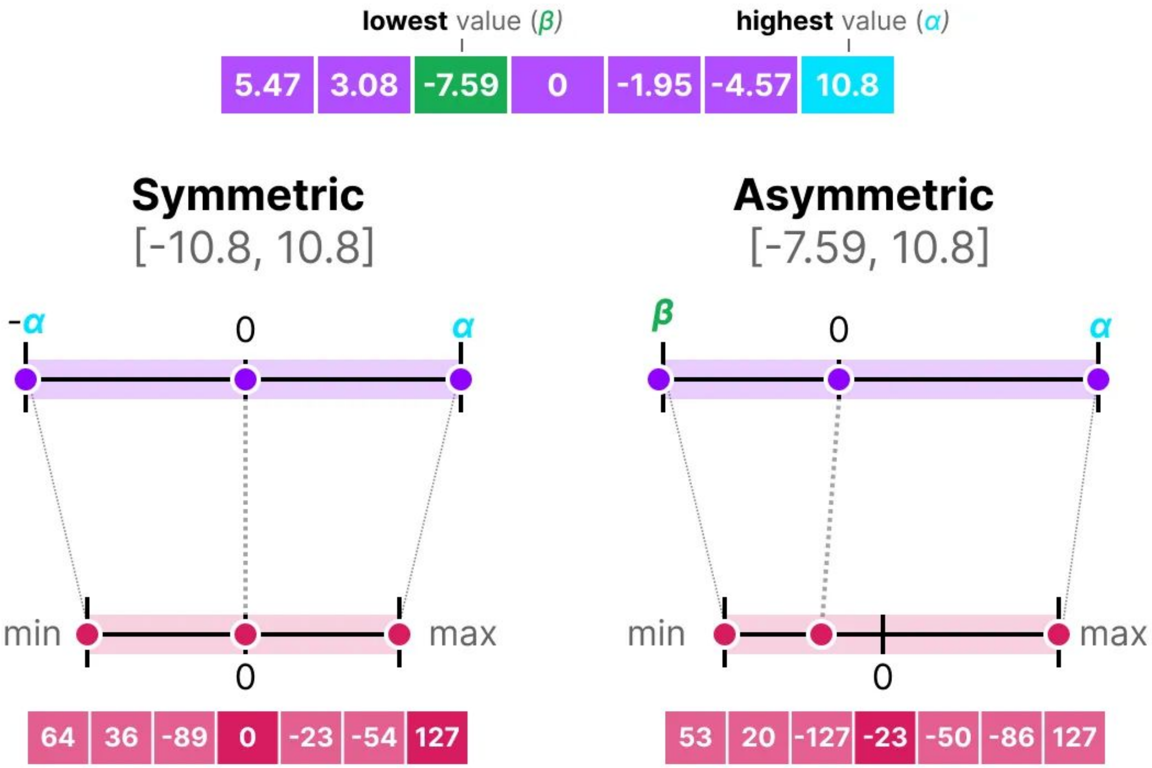
Asymmetric quantization (cont'd)



(dequantize)

[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Symmetric vs. Asymmetric quantization



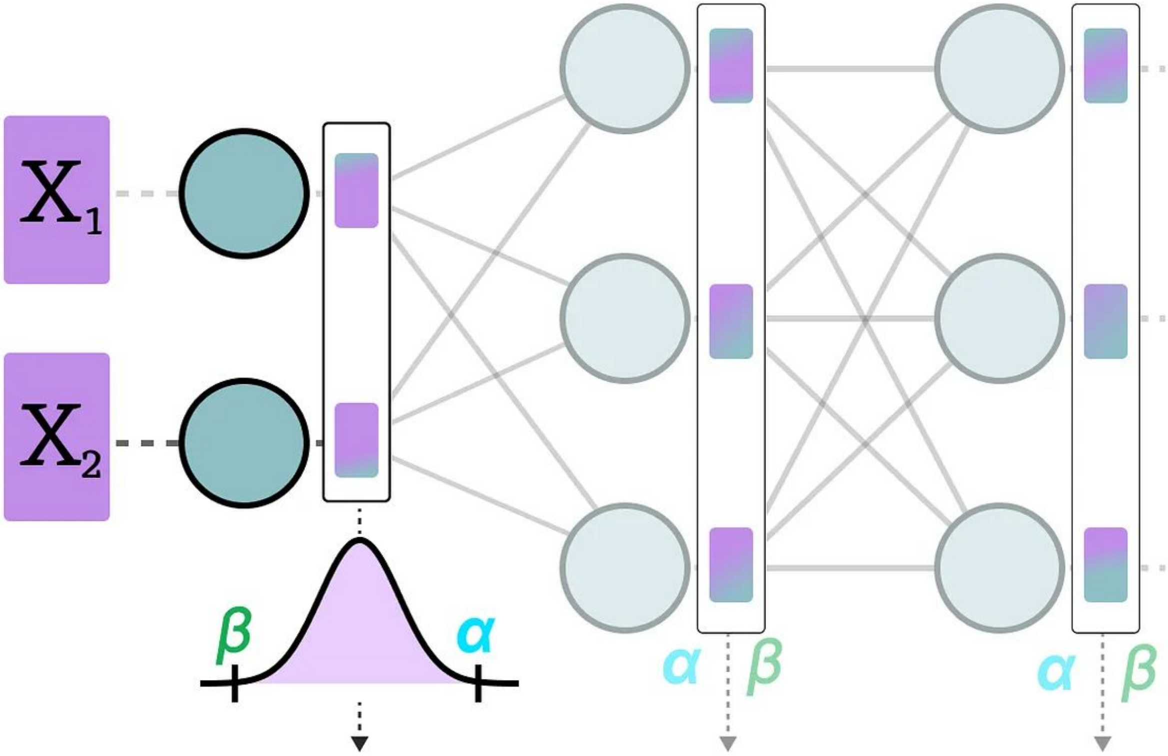
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Post-training quantization

- Dynamic Quantization
- Static Quantization

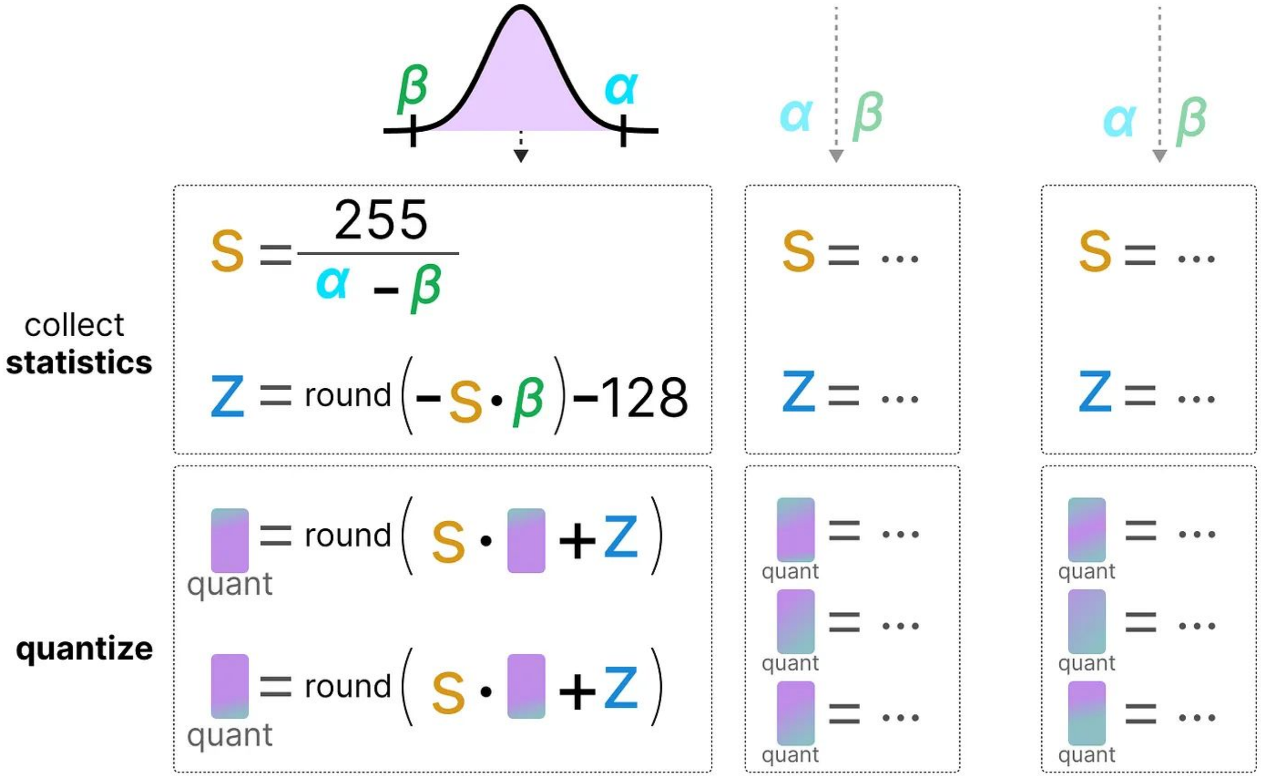
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Dynamic quantization



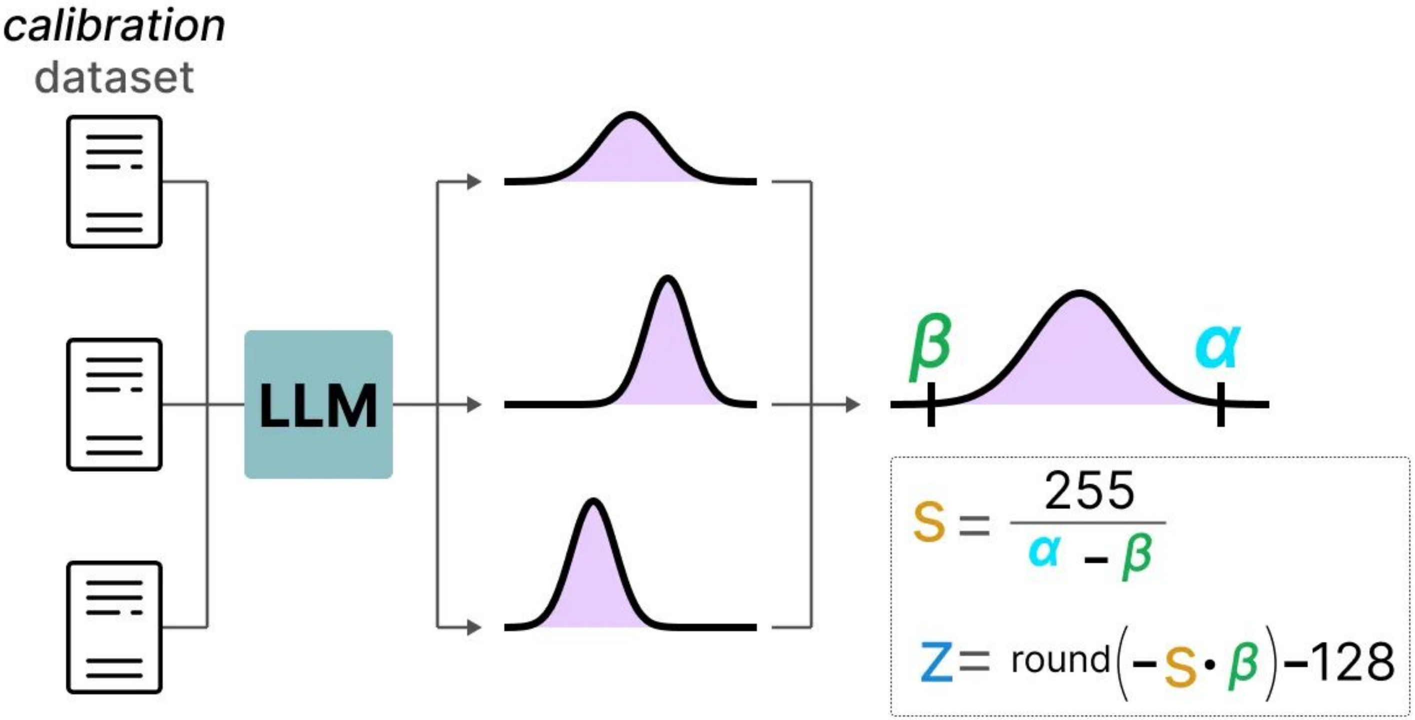
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Dynamic quantization (cont'd)



[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Static quantization



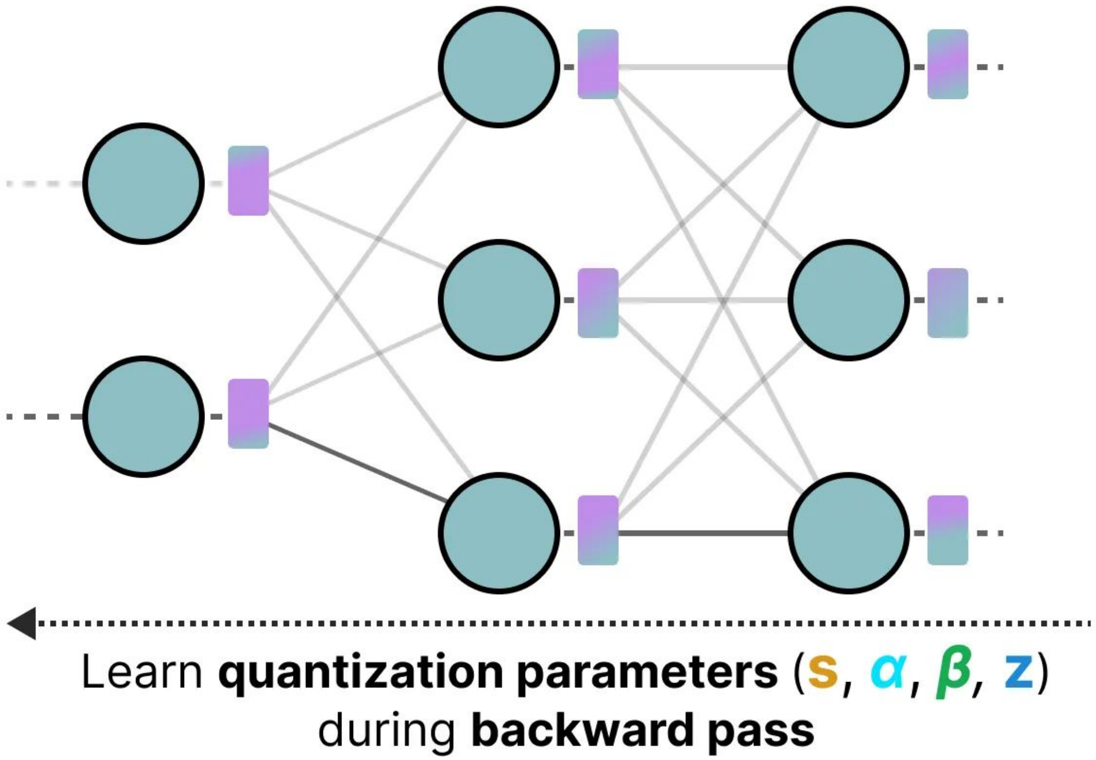
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

The realm of 4-bit quantization

- GPTQ (full model on GPU)
- GGUF (potentially offload layers on the CPU)

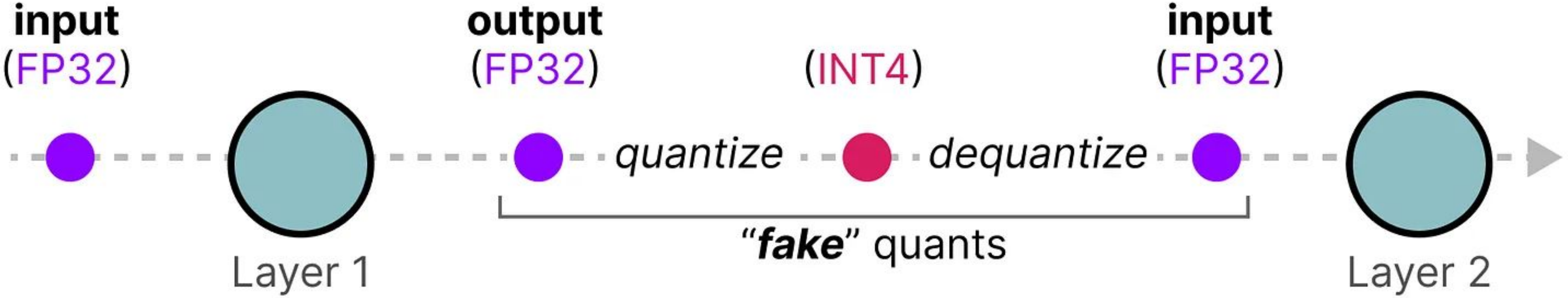
Maarten Grootendorst's blog: A Visual Guide to Quantization

Quantization aware training



Maarten Grootendorst's blog: A Visual Guide to Quantization

Quantization aware training (cont'd)



[Maarten Grootendorst's blog: A Visual Guide to Quantization]

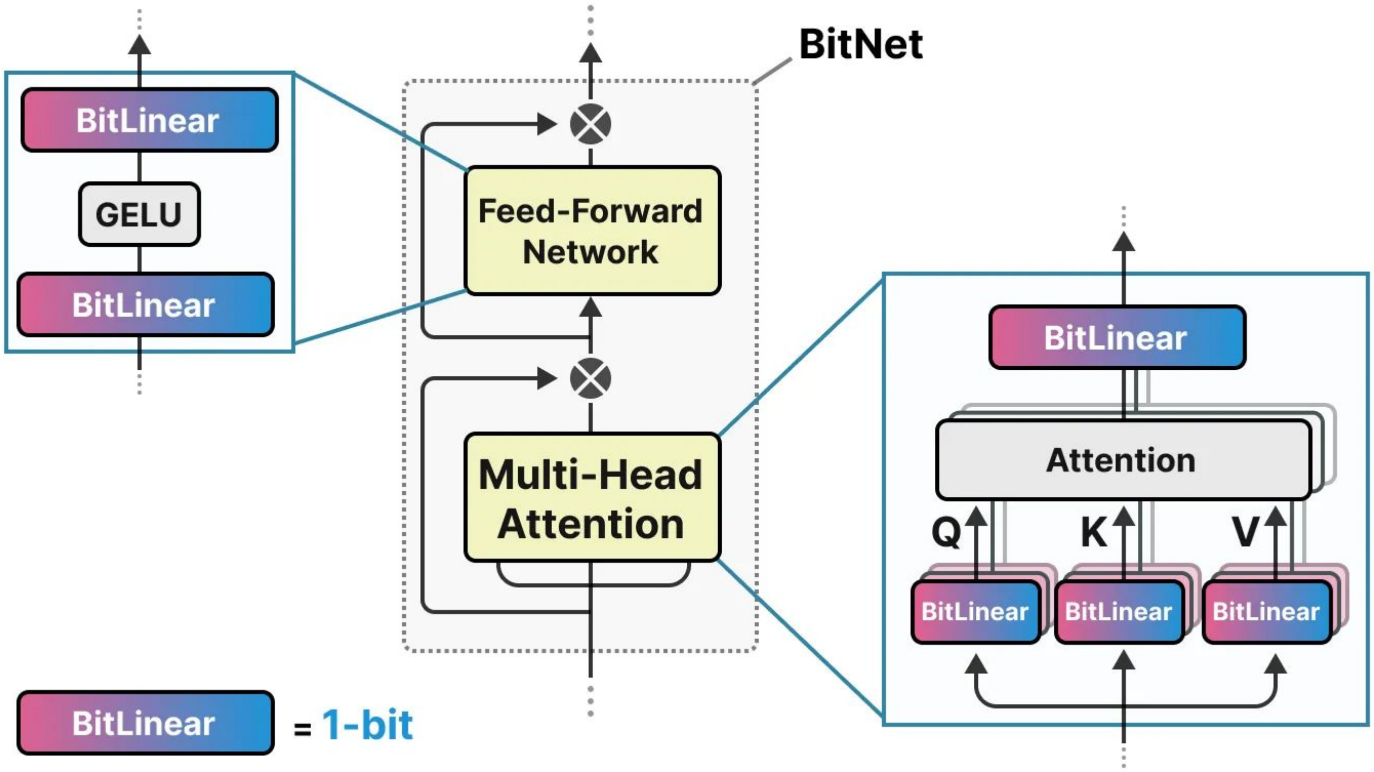
QLoRA: Efficient Finetuning of Quantized LLMs

Tim Dettmers*   **Artidoro Pagnoni***   **Ari Holtzman**

Luke Zettlemoyer

University of Washington
{dettmers,artidoro,ahai,lsz}@cs.washington.edu

The era of 1-bit LLMs: BitNet



[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Pruning

- Remove parameters from the model after training

A SIMPLE AND EFFECTIVE PRUNING APPROACH FOR LARGE LANGUAGE MODELS

Mingjie Sun[1*] **Zhuang Liu**[2*] **Anna Bair**[1] **J. Zico Kolter**[1,3]
[1]Carnegie Mellon University [2]Meta AI Research [3]Bosch Center for AI

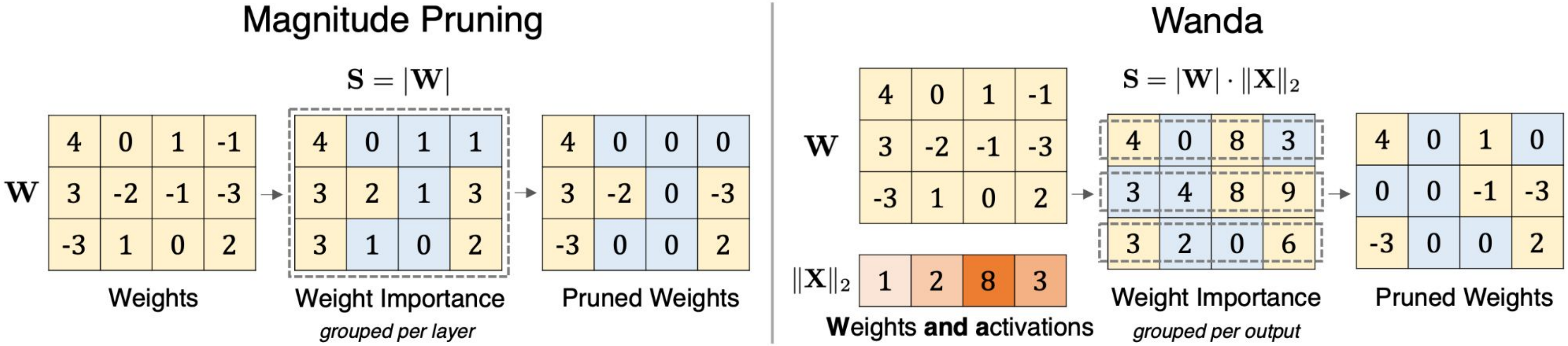


Figure 1: Illustration of our proposed method Wanda (Pruning by **W**eights **and a**ctivations), compared with the magnitude pruning approach. Given a weight matrix $\mathbf{W}$ and input feature activations $\mathbf{X}$, we compute the weight importance as the elementwise product between the weight magnitude and the norm of input activations ($|\mathbf{W}| \cdot \|\mathbf{X}\|_2$). Weight importance scores are compared on a *per-output* basis (within each row in $\mathbf{W}$), rather than globally across the entire matrix.

Are Sixteen Heads Really Better than One?

Paul Michel
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA
pmichel1@cs.cmu.edu

Omer Levy
Facebook Artificial Intelligence Research
Seattle, WA
omerlevy@fb.com

Graham Neubig
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA
gneubig@cs.cmu.edu

Published as a conference paper at ICLR 2019

The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks

Jonathan Frankle
MIT CSAIL
jfrankle@csail.mit.edu

Michael Carbin
MIT CSAIL
mcarbin@csail.mit.edu

Training a pruned randomly-initialized networks can be better than training the full randomly-initialized network

Thank you!