Efficient training & inference

CS 4804: Introduction to AI

Fall 2025

https://tuvllms.github.io/ai-fall-2025/

Tu Vu

VIRGINIA TECH

Logistics

- HW 2 released due 11/18
- Final presentations: 12/4 & 12/9

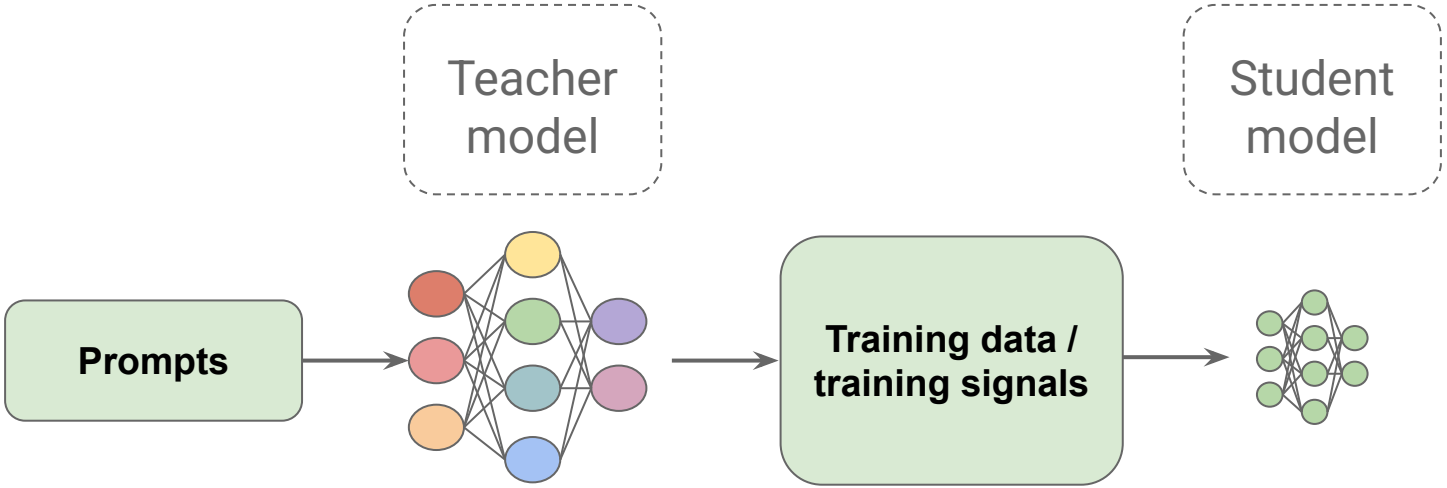
Today's lecture

- Knowledge distillation
- Model merging / Model editing
- Multi-query attention / Grouped-query attention
- KV caching
- Steering vectors
- Pruning
- Quantization

Future lectures

- Agents & Compound AI systems (2 lectures)
  - Retrieval-Augmented Generation (RAG)
  - Tool-use models
  - Modular AI software with DSPy
  - Mixture of agents
  - Agentic memory
  - Model Context Protocol (MCP) & Agentic AI workflows
- Diffusion models
- Ethics and safety

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.

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

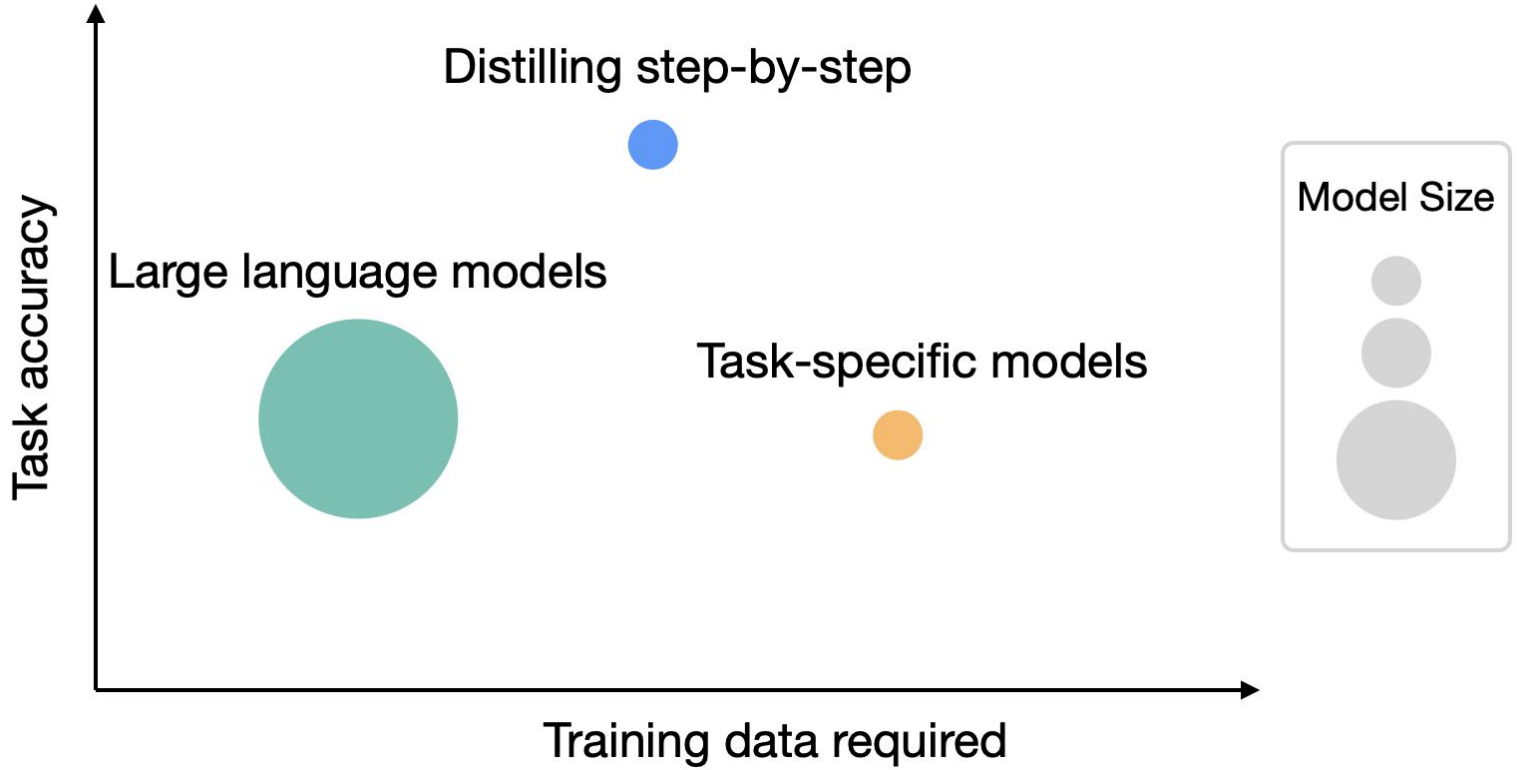
Distilling Step-by-Step! Outperforming Larger Language Models with Less Training Data and Smaller Model Sizes

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

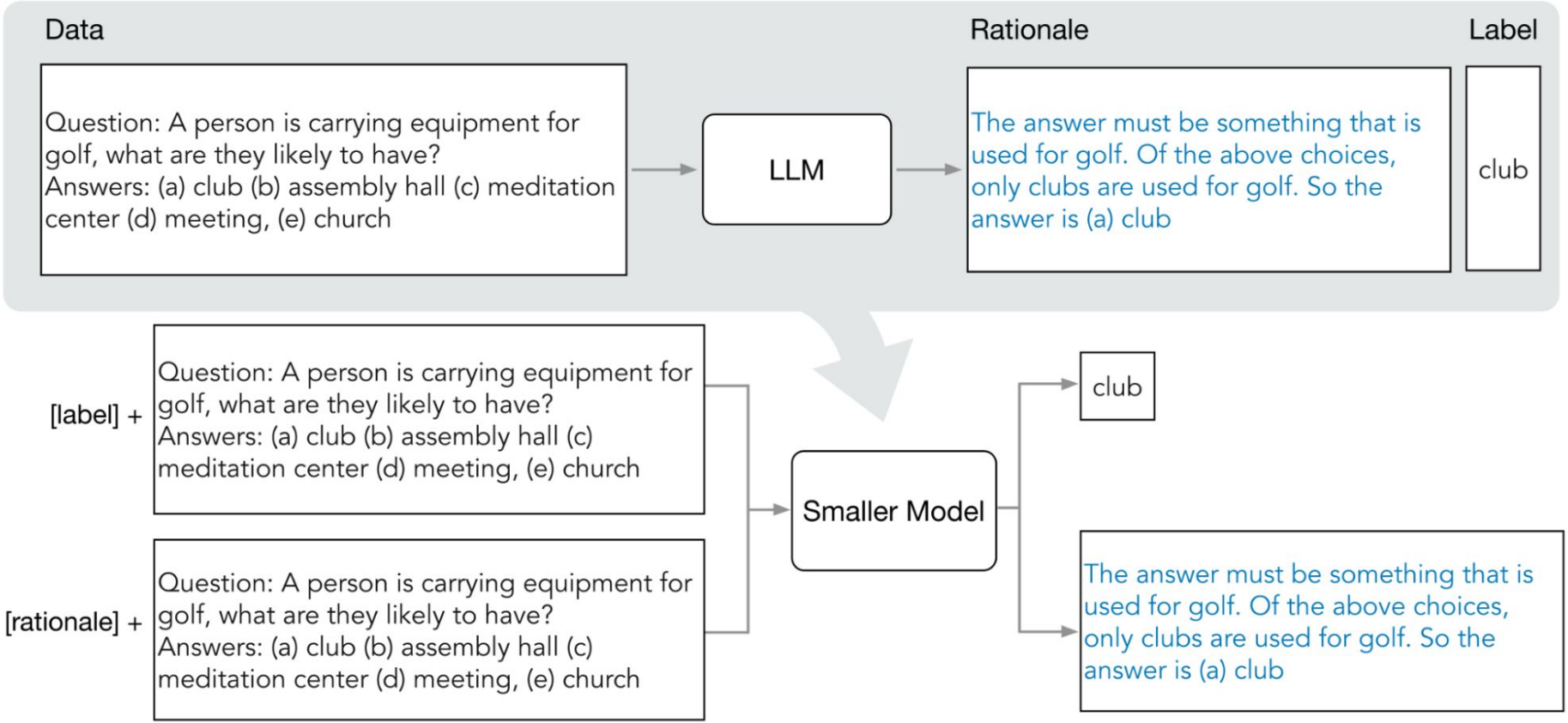
Distilling Step-by-Step! Outperforming Larger Language Models with Less Training Data and Smaller Model Sizes

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.

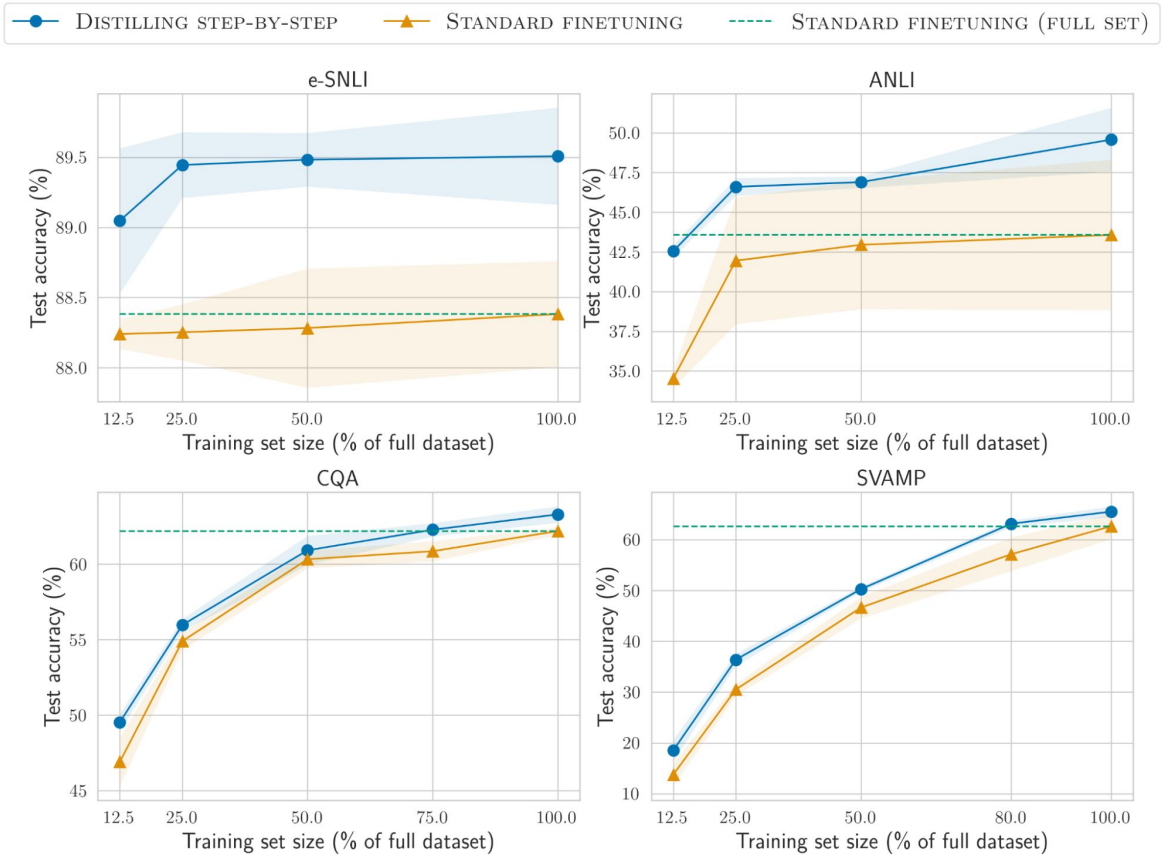
Input

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:

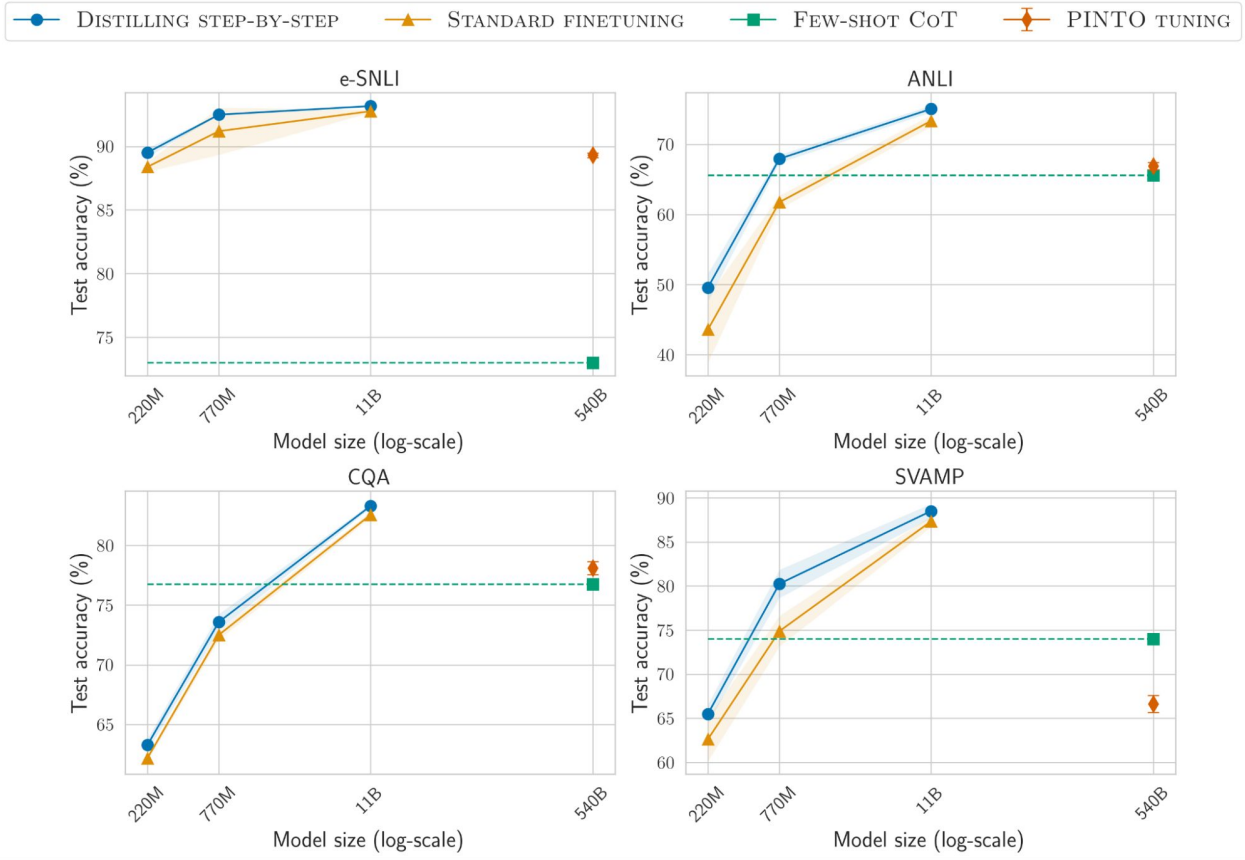
Output

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.

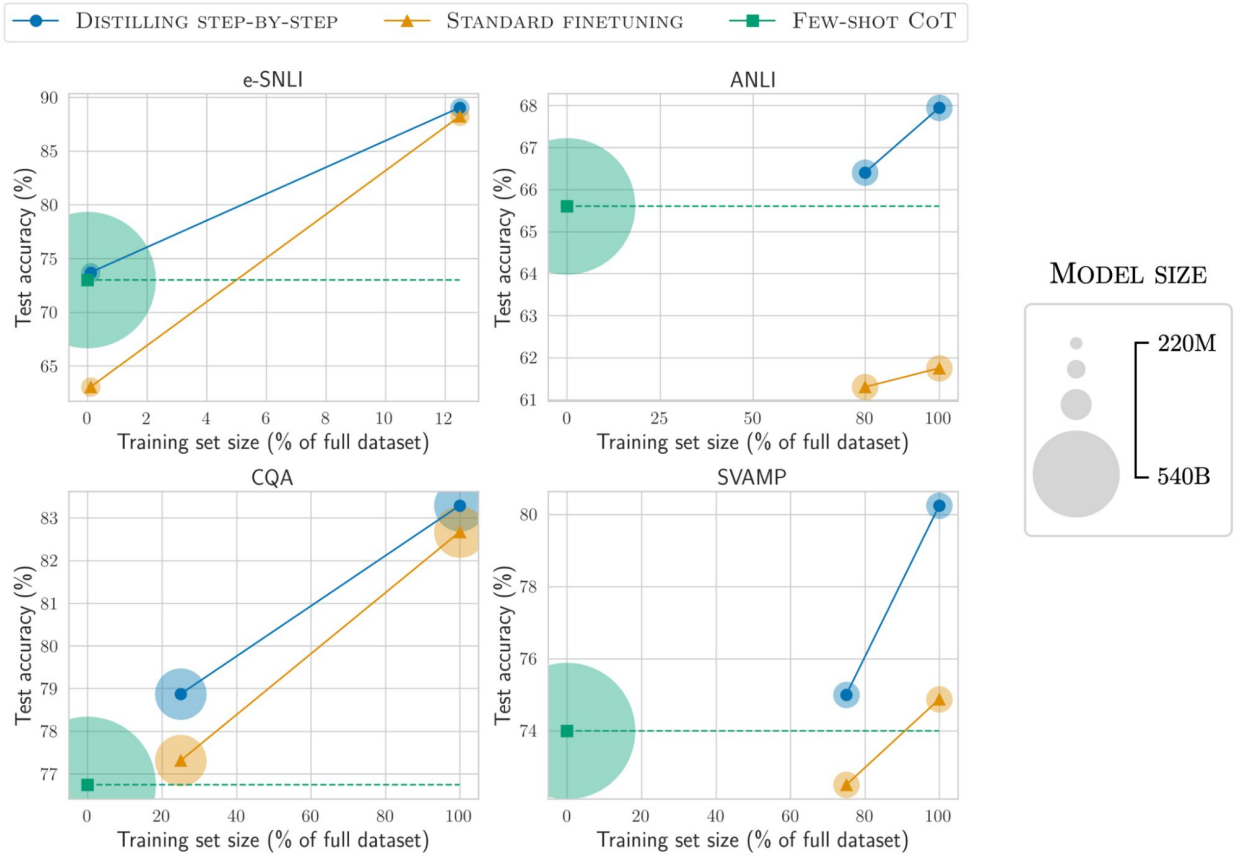
Less training data



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.

Published as a conference paper at ICLR 2023

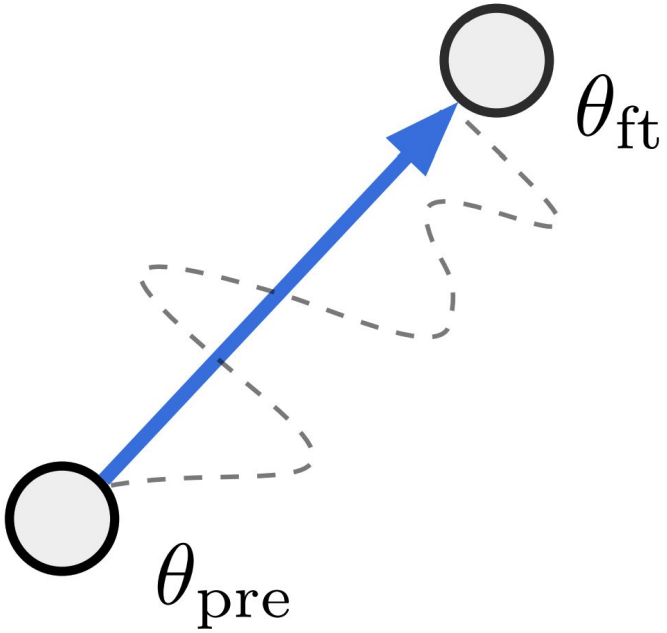
EDITING MODELS WITH TASK ARITHMETIC

Gabriel Ilharco[*1] **Marco Tulio Ribeiro**[2] **Mitchell Wortsman**[1] **Suchin Gururangan**[1]
Ludwig Schmidt[1,3] **Hannaneh Hajishirzi**[1,3] **Ali Farhadi**[1]
[1]University of Washington [2]Microsoft Research [3]Allen Institute for AI

Why do we want to edit LLMs?

- improve performance on downstream tasks
- mitigate biases or unwanted behavior
- align models with human preferences
- update models with new information

The notion of task vectors



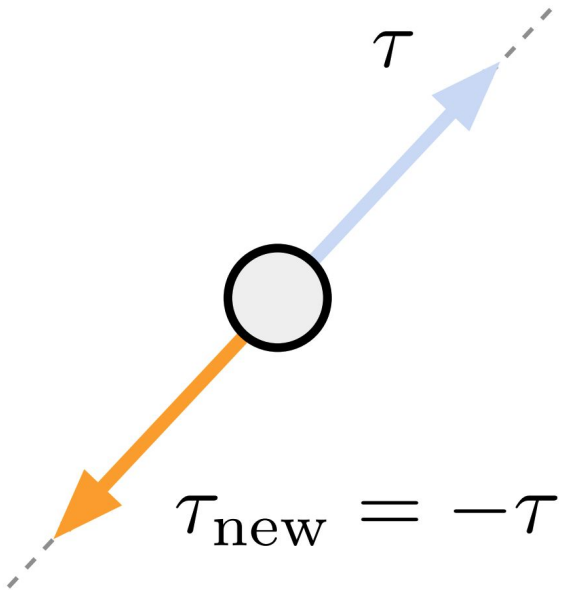
$$\tau = \theta_{\text{ft}} - \theta_{\text{pre}}$$

$$\theta_{\text{new}} = \theta + \tau$$

In practice, we have an optional scaling term λ

$$\theta_{\text{new}} = \theta + \lambda\tau$$

Forgetting via negation



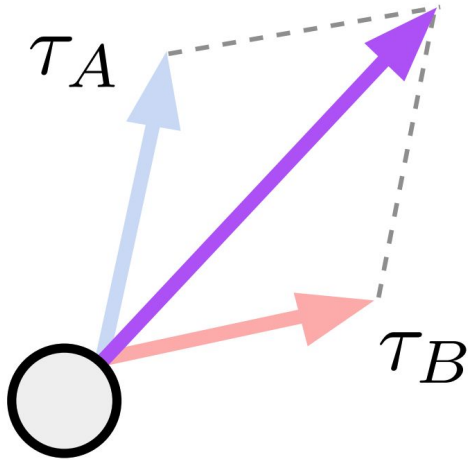
Example: making a language model produce less toxic content

$$\theta_{\text{new}} = \theta - \tau = \theta - (\theta_{ft} - \theta)$$

In practice, we have an optional scaling term λ

Learning via addition

$$\tau_{\text{new}} = \tau_A + \tau_B$$



Example: building a multi-task model

$$\theta_{\text{new}} = \theta + \tau = \theta + (\tau_A + \tau_B)$$

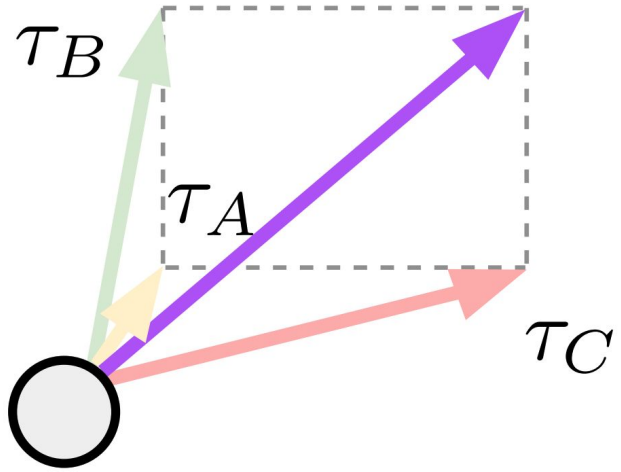
$$= \theta + (\theta_A - \theta) + (\theta_B - \theta)$$

In practice, we have optional scaling terms λ_A, λ_B

Task analogies

"A is to B as C is to D"

$$\tau_{\text{new}} = \tau_C + (\tau_B - \tau_A)$$



$$\tau_B - \tau_A = \tau_D - \tau_C$$

$$\tau_D = \tau_C + (\tau_B - \tau_A)$$

$$\theta_{\text{new}} = \theta + \tau_C + (\tau_B - \tau_A)$$

$$= \theta + (\theta_C - \theta) + (\theta_B - \theta) - (\theta_A - \theta)$$

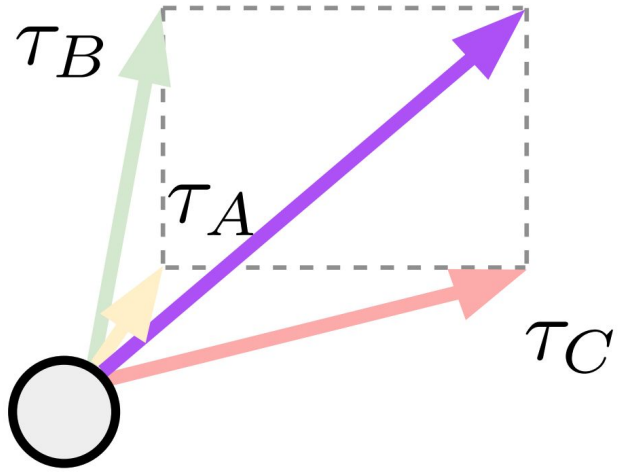
Example: improving domain generalization

In practice, we have optional scaling terms λ_A, λ_B, λ_C

Task analogies

"A is to B as C is to D"

$$\tau_{\text{new}} = \tau_C + (\tau_B - \tau_A)$$



$$\tau_B - \tau_A = \tau_D - \tau_C$$

$$\tau_D = \tau_C + (\tau_B - \tau_A)$$

$$\theta_{\text{new}} = \theta + \tau_C + (\tau_B - \tau_A)$$

$$= \theta + (\theta_C - \theta) + (\theta_B - \theta) - (\theta_A - \theta)$$

Example: improving domain generalization

In practice, we have optional scaling terms λ_A, λ_B, λ_C

Making language models less toxic with negative task vectors

Method	% toxic generations (↓)	Avg. toxicity score (↓)	WikiText-103 perplexity (↓)
Pre-trained	4.8	0.06	16.4
Fine-tuned	57	0.56	16.6
Gradient ascent	0.0	0.45	$>10^{10}$
Fine-tuned on non-toxic	1.8	0.03	17.2
Random vector	4.8	0.06	16.4
Negative task vector	0.8	0.01	16.9

What Matters for Model Merging at Scale?

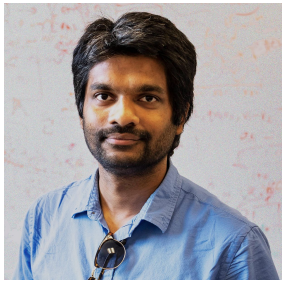
Prateek Yadav

Tu Vu

Jonathan Lai

Alexandra Chronopoulou

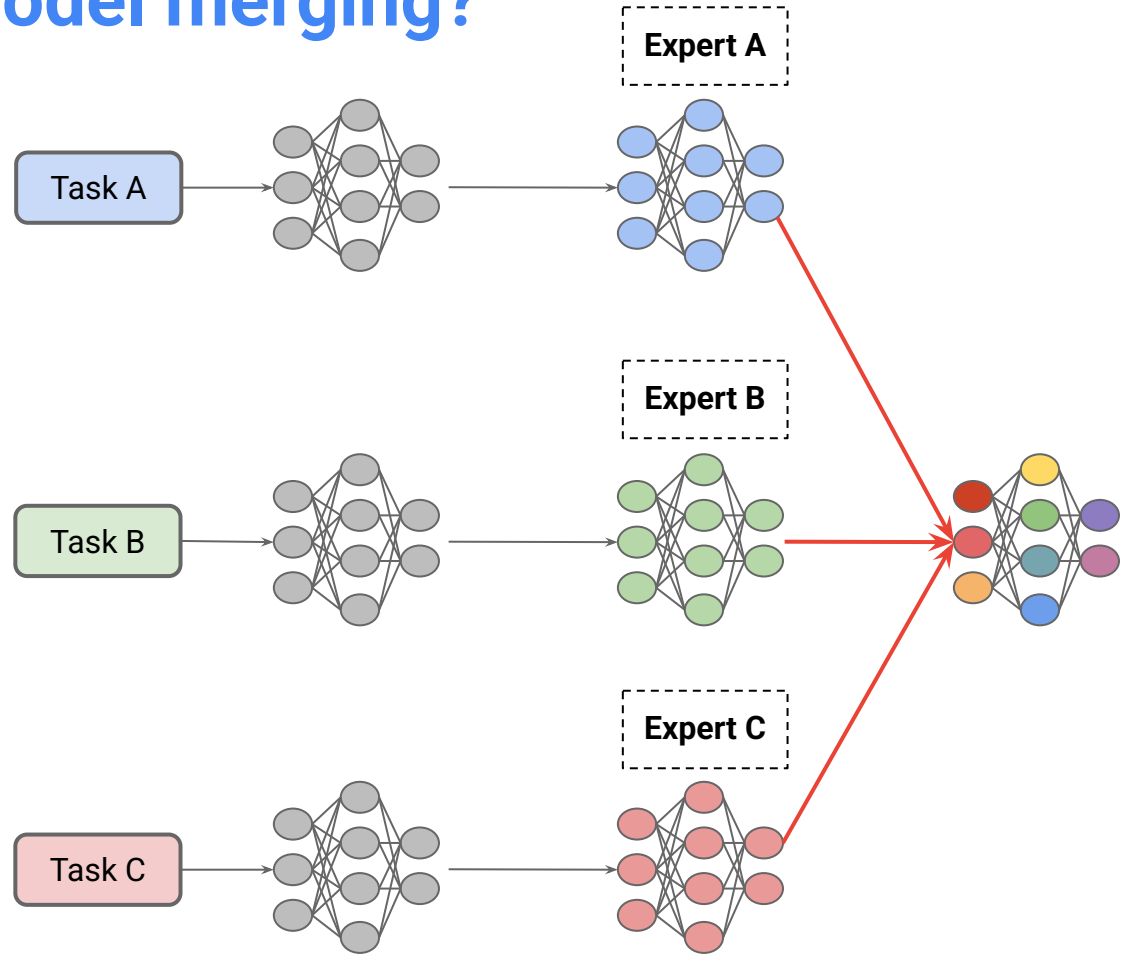
Manaal Faruqui



Mohit Bansal

Tsendsuren Munkhdalai

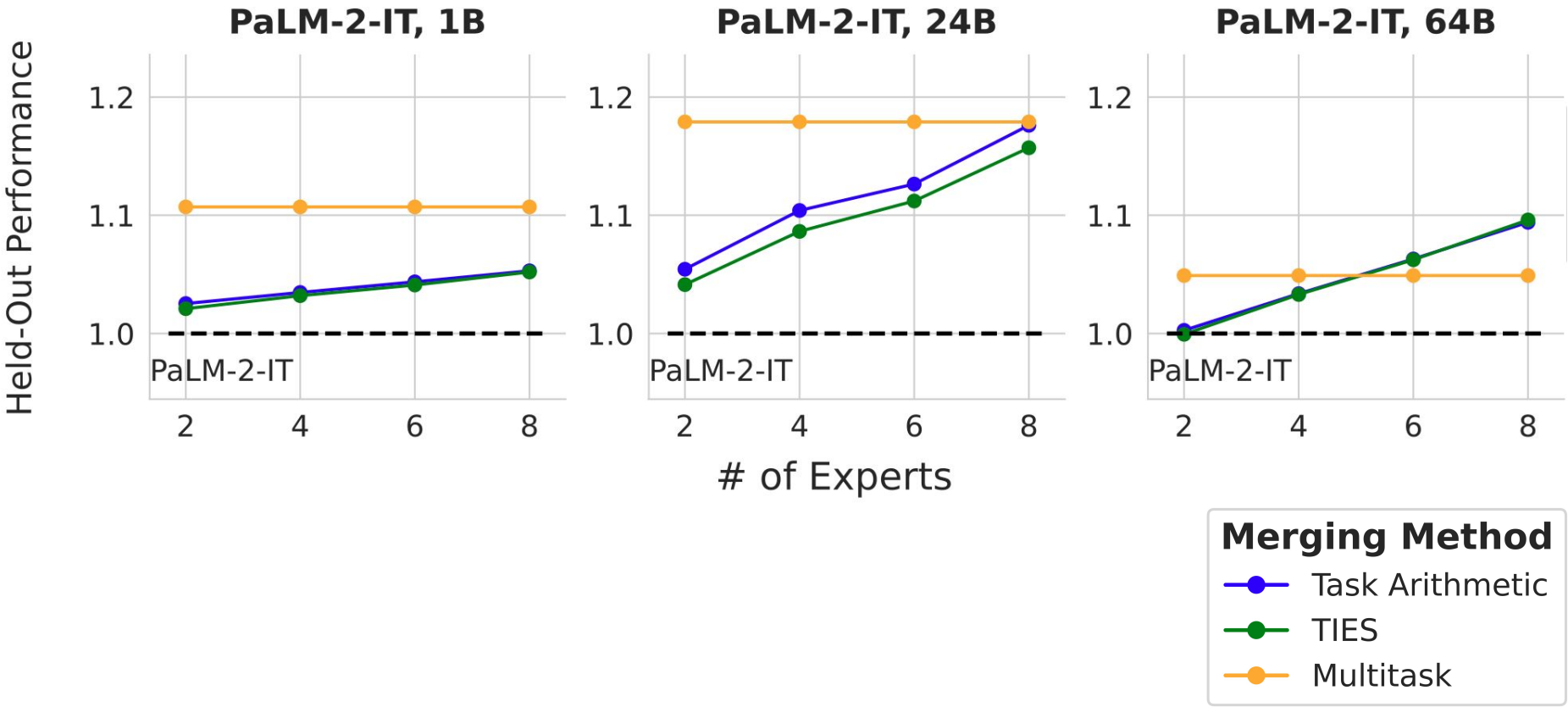
What is model merging?



Why model merging?

- dramatically reduces storage and serving costs by reusing a single model across tasks
- enables compositional combination of capabilities from expert models, which can improve generalization to novel tasks
- supports decentralized and modular model development by allowing multiple contributors to independently build models and later combine them together

Merging boosts zero-shot generalization



Efficient Model Development through Fine-tuning Transfer

Pin-Jie Lin[1] *pinjie@vt.edu*

Rishab Balasubramanian[1] *rishbb@vt.edu*

Fengyuan Liu[2] *fy.liu@mail.utoronto.ca*

Nikhil Kandpal[2] *nkandpa2@cs.toronto.edu*

Tu Vu[1] *tuvu@vt.edu*

[1] *Virginia Tech* [2] *University of Toronto & Vector Institute*

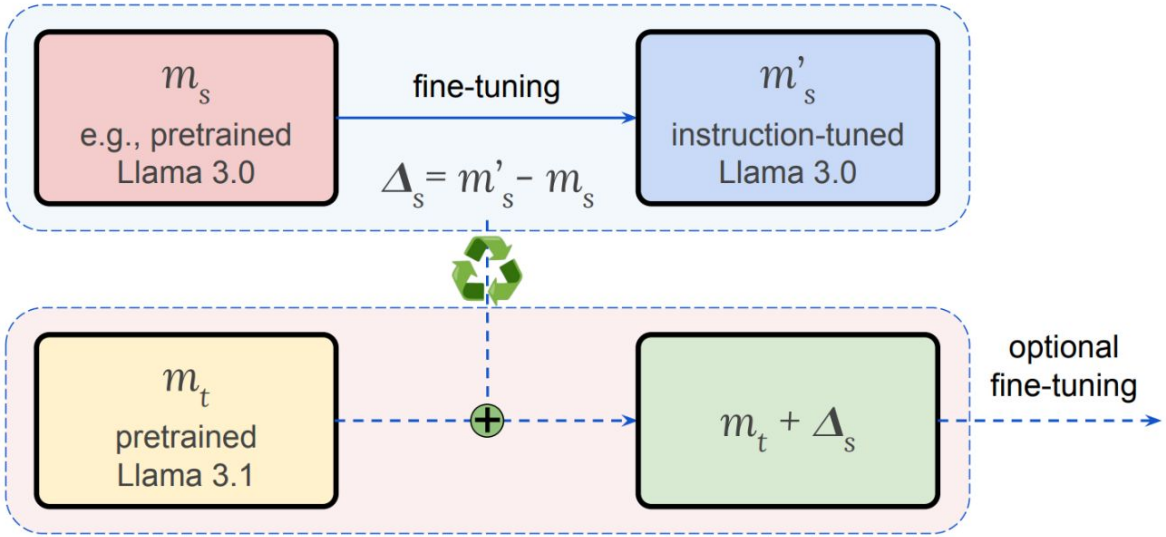


Figure 1: To transfer fine-tuning (e.g., instruction tuning) from a *source* model version s (e.g., Llama 3.0) to a *target* version t (Llama 3.1), we first compute the diff vector $\Delta_s = m'_s - m_s$ from version s, where m'_s is the fine-tuned model (instruction-tuned Llama 3.0) and m_s is the base model (pretrained Llama 3.0). Then, we add Δ_s to the target base model (pretrained Llama 3.1) to approximate the fine-tuned model in version t (instruction-tuned Llama 3.1). We explore two scenarios: (1) *recycling*—transferring from an older model version to a newer one to reduce retraining, and (2) *backporting*—transferring from a newer version to an older one to take advantage of the newer fine-tuning while maintaining optimization for specific use cases.

Transferring fine-tuning updates

Model	GSM8K	MATH	ARC_C	GPQA	MMLU	IFEval
Llama 3.0 8B Instruct	81.1	28.8	82.4	**31.5**	64.9	**76.6**
Llama 3.0 8B	55.6	17.3	79.7	22.3	66.7	34.5
+ $\Delta_{3.1}$	**82.8**	**44.7**	**83.0**	25.9	**70.0**	**76.6**
Llama 3.1 8B Instruct	**86.5**	**50.3**	**83.8**	31.3	**72.9**	80.5
Llama 3.1 8B	56.6	19.3	79.2	21.9	66.8	36.4
+ $\Delta_{3.0}$	79.8	29.9	82.9	**32.6**	65.1	**83.3**

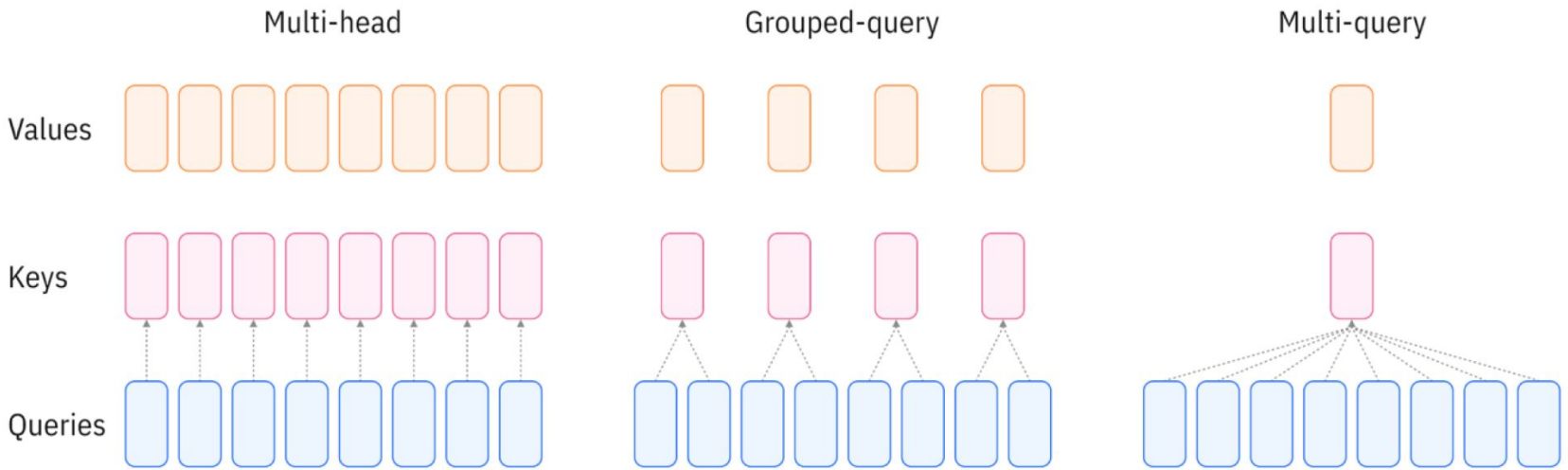
Table 1: Fine-tuning transfer significantly improves the performance of the target base model across various tasks, achieving results comparable to its fine-tuned counterpart in many cases. Here, $\Delta_{3.0}$ and $\Delta_{3.1}$ represent the diff vectors between Llama Instruct and Llama for versions 3.0 and 3.1, respectively. Notably, adding the diff vector Δ_s from a different model version can effectively transform a non-instruction-tuned model (e.g., Llama 3.0 or Llama 3.1) into one that follows instructions well (Llama 3.0 + $\Delta_{3.1}$ or Llama 3.1 + $\Delta_{3.0}$) without further training. Additional results for OLMo and Tülu can be found in Appendix A, where we additionally find that advanced LLM capabilities, attained through alignment tuning stages such as Supervised Fine-Tuning (SFT), Direct Preference Optimization (DPO), or Group Relative Policy Optimization (GRPO), can be successfully transferred across different model versions.

Multilingual model development

Model	Malagasy	Sinhala	Turkish
Llama 3.0 8B Instruct	23.1	23.3	30.8
+ FT	30.8	29.0	43.2
Llama 3.1 8B Instruct	27.6	**33.0**	27.7
+ $\Delta_{3.0}$	**32.3**	32.3	**43.2**

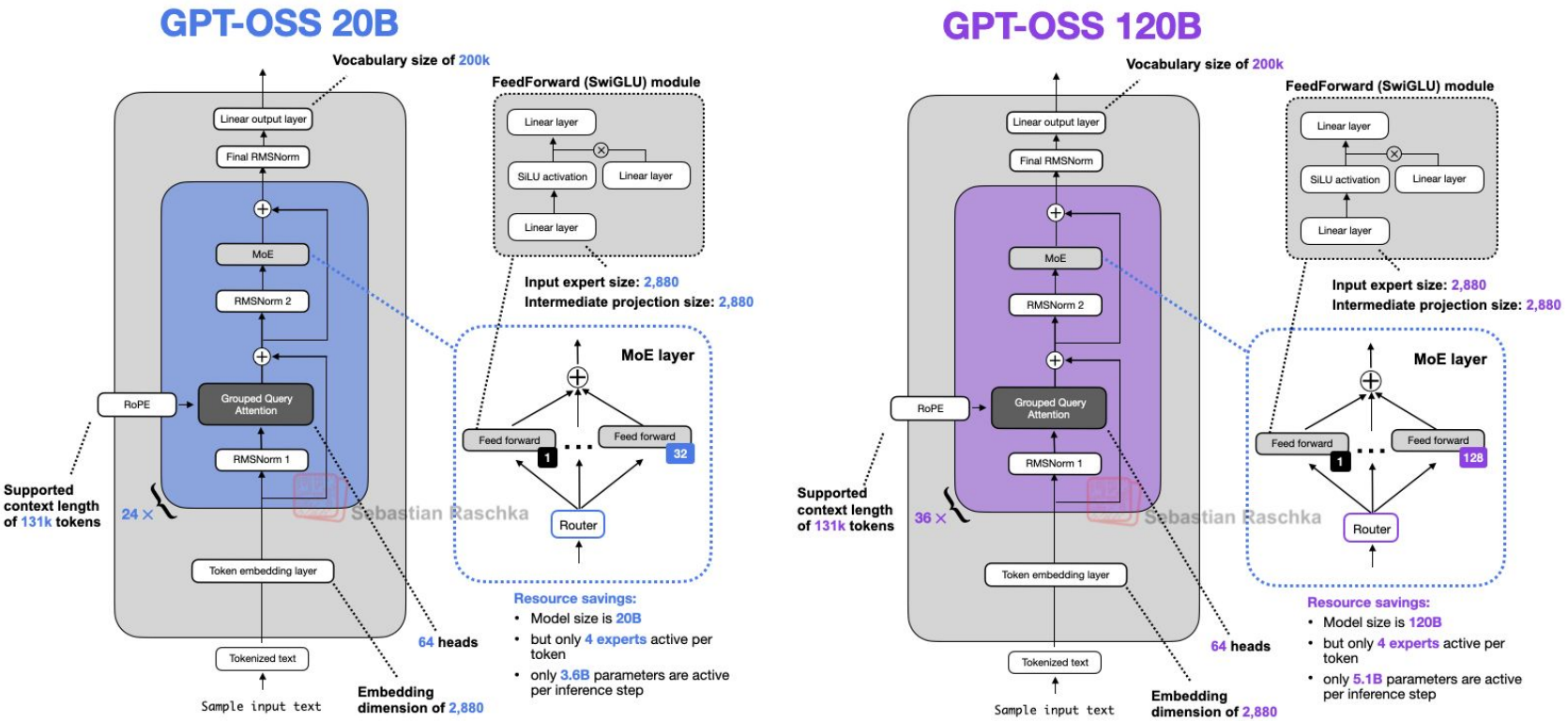
Table 2: Recycling fine-tuning updates improves multilingual performance on Global MMLU without re-training, yielding a 4.7% and 15.5% absolute improvement for Malagasy and Turkish, respectively, compared to Llama 3.1 8B Instruct. $\Delta_{3.0}$ represents the diff vector between Llama 3.0 Instruct and its monolingual fine-tuned (FT) version.

Multi-query attention / grouped-query attention



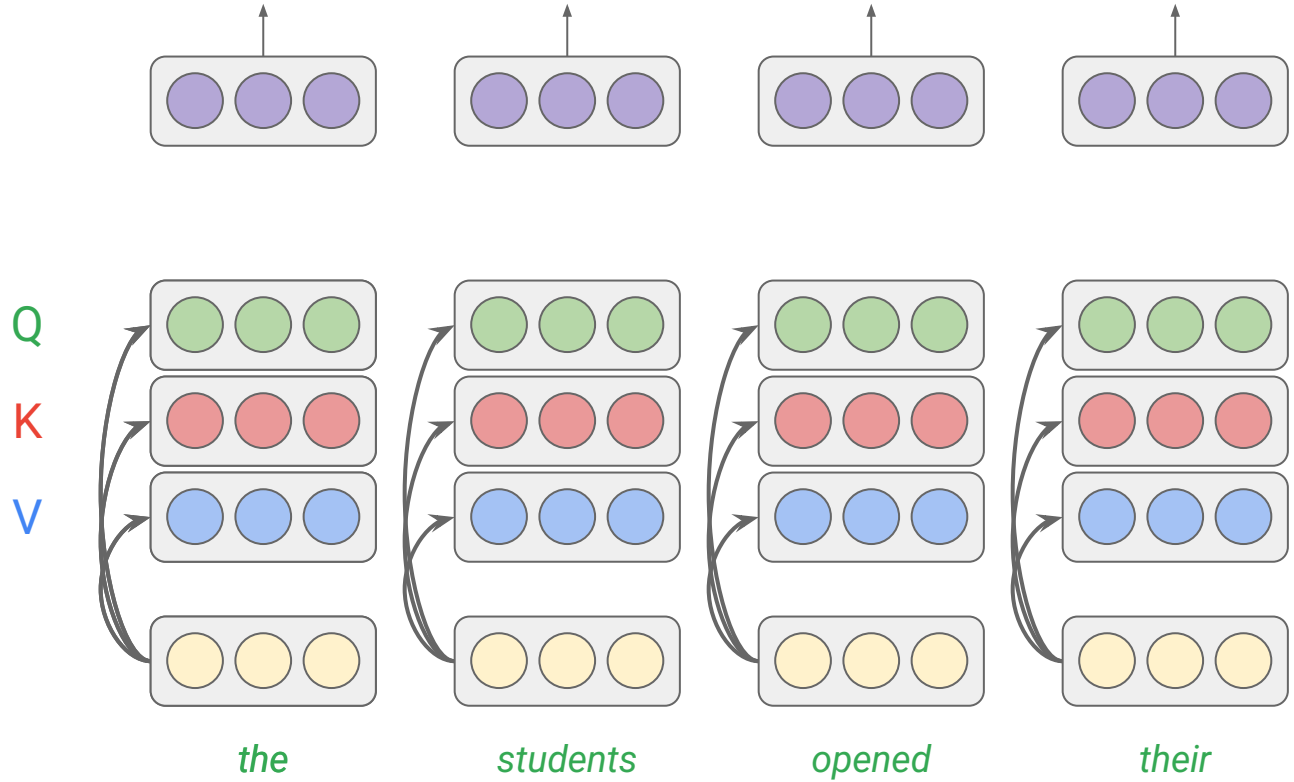
https://www.ibm.com/think/topics/grouped-query-attention

Multi-query attention / grouped-query attention



https://magazine.sebastianraschka.com/p/from-gpt-2-to-gpt-oss-analyzing-the

KV caching



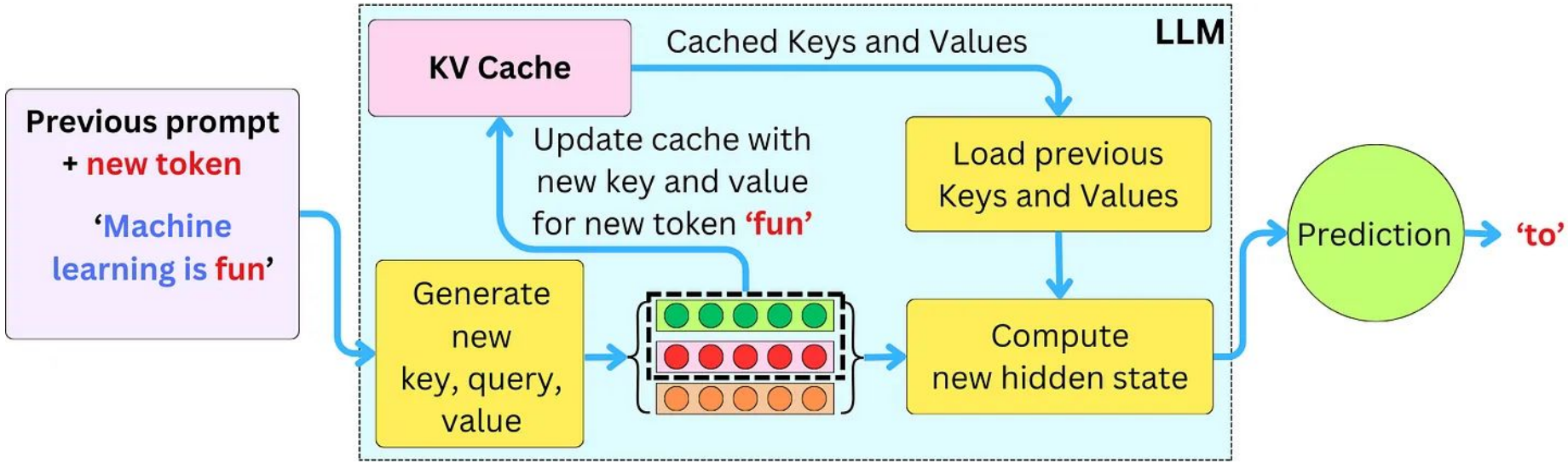
$$Q = X \cdot W_Q$$

$$K = X \cdot W_K$$

$$V = X \cdot W_V$$

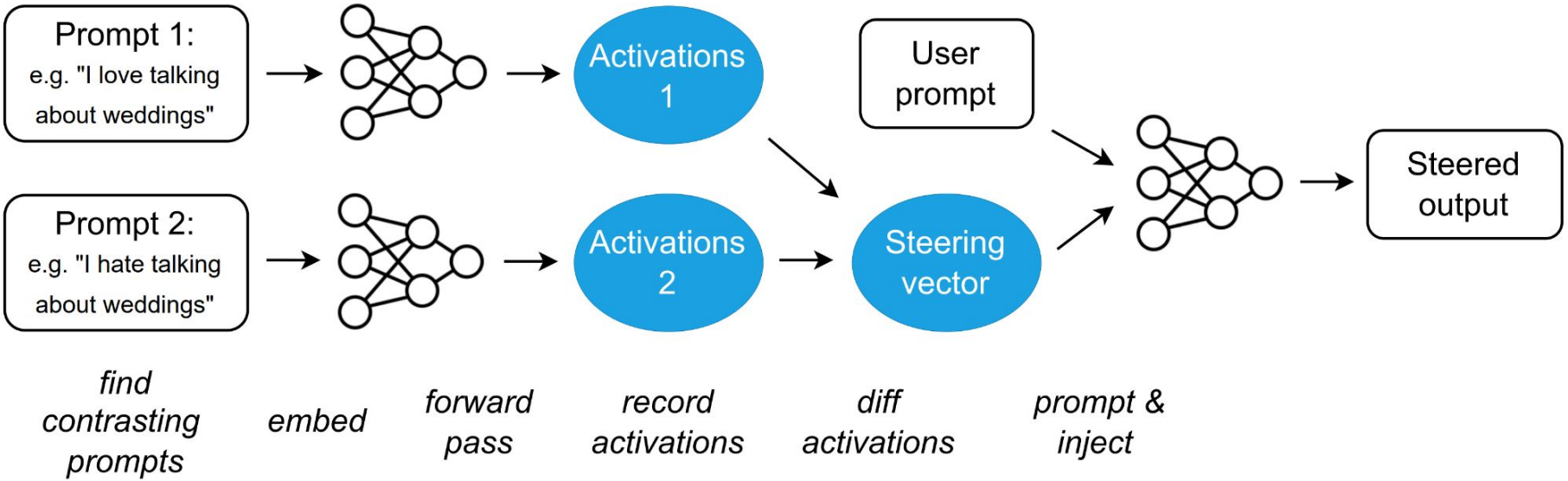
linear projections

KV caching (cont'd)



https://newsletter.theaiedge.io/p/how-to-reduce-llm-decoding-time-with

Steering vectors



https://arxiv.org/abs/2308.10248

Steering vectors (cont'd)

Prompt	+	steering	=	completion
I hate you because...		[None]		...you are the most disgusting thing I have ever seen.
		ActAdd (love)		...you are so beautiful and I want to be with you forever.
I went up to my friend and said...		[None]		..."I'm sorry, I can't help you." "No," he said. "You're not."
		ActAdd (weddings)		..."I'm going to talk about the wedding in this episode of Wedding Season. I think it's a really good episode. It's about how you're supposed to talk about weddings."

https://arxiv.org/abs/2308.10248

Pruning

- Remove parameters from the model after training

Published as a conference paper at ICLR 2024

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

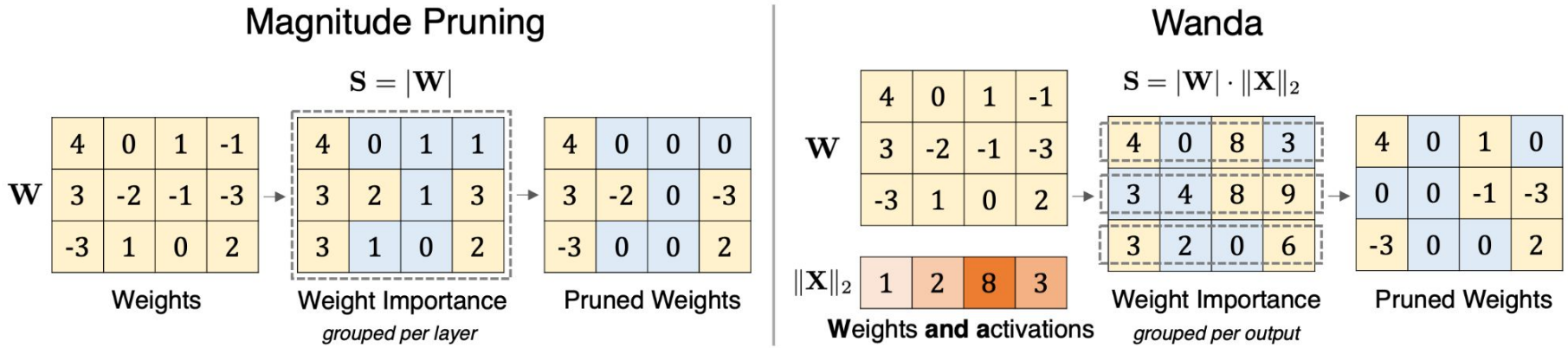


Figure 1: Illustration of our proposed method Wanda (Pruning by **W**eights **and** **a**ctivations), compared with the magnitude pruning approach. Given a weight matrix $\mathbf{W}$ and input feature activations $\mathbf{X}$, we compute the weight importance as the elementwise product between the weight 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

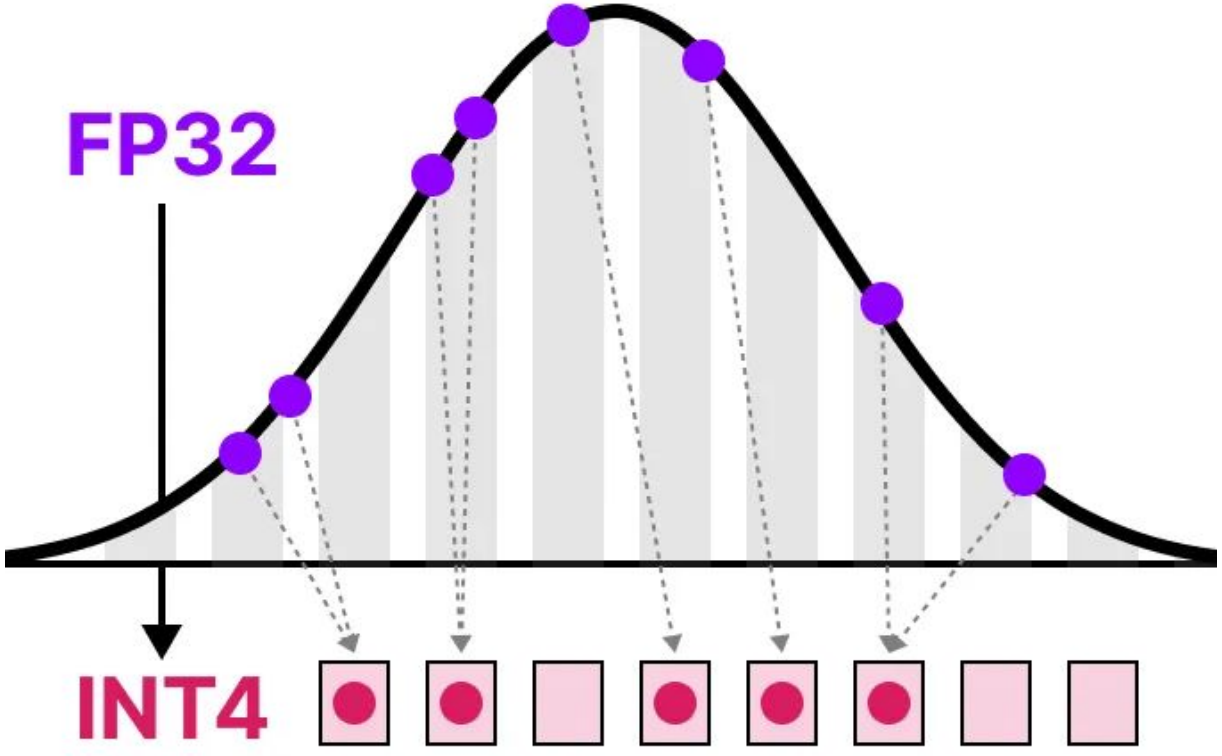
THE LOTTERY TICKET HYPOTHESIS: FINDING SPARSE, TRAINABLE NEURAL NETWORKS

Jonathan Frankle
MIT CSAIL
jfrankle@csail.mit.edu

Michael Carbin
MIT CSAIL
mcarbin@csail.mit.edu

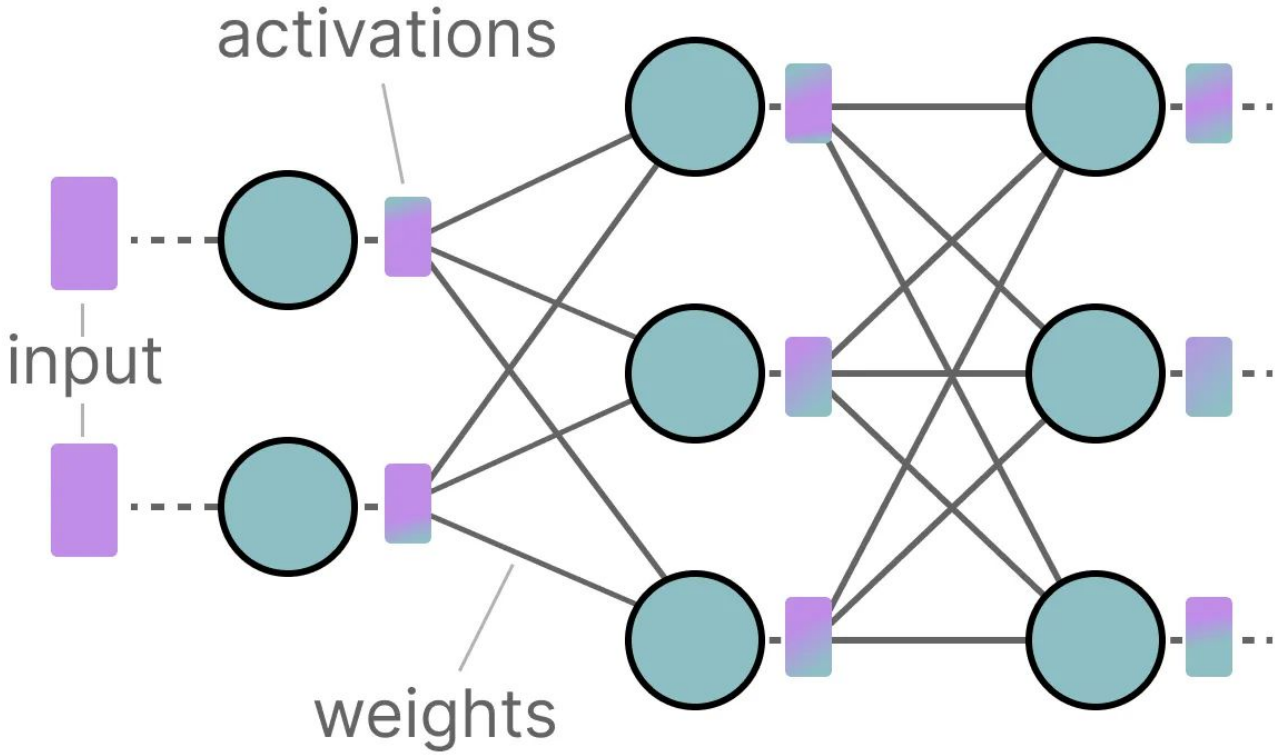
Training a pruned randomly-initialized networks can be better than training the full randomly-initialized network

Quantization



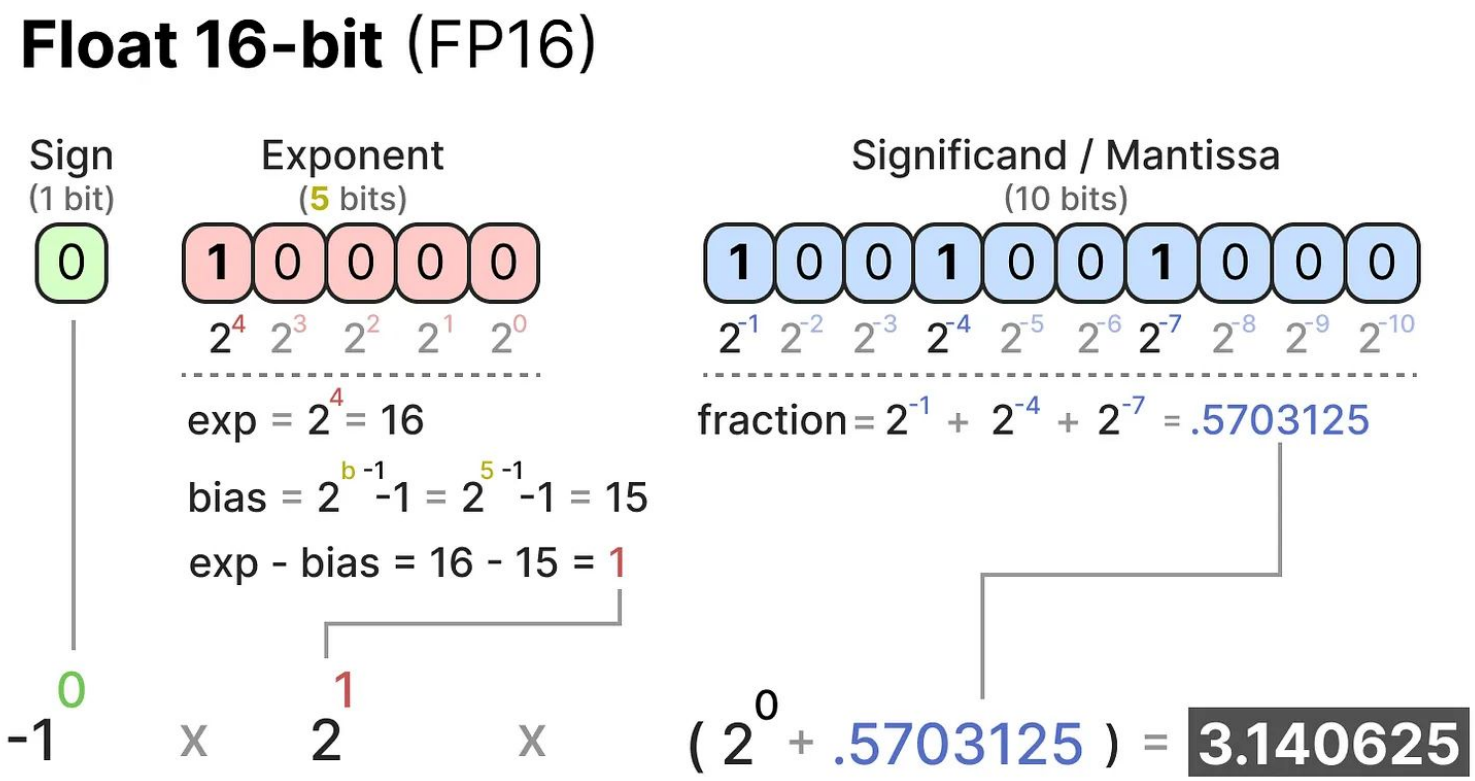
Maarten Grootendorst's blog: A Visual Guide to Quantization

Quantizing both the weights and activations



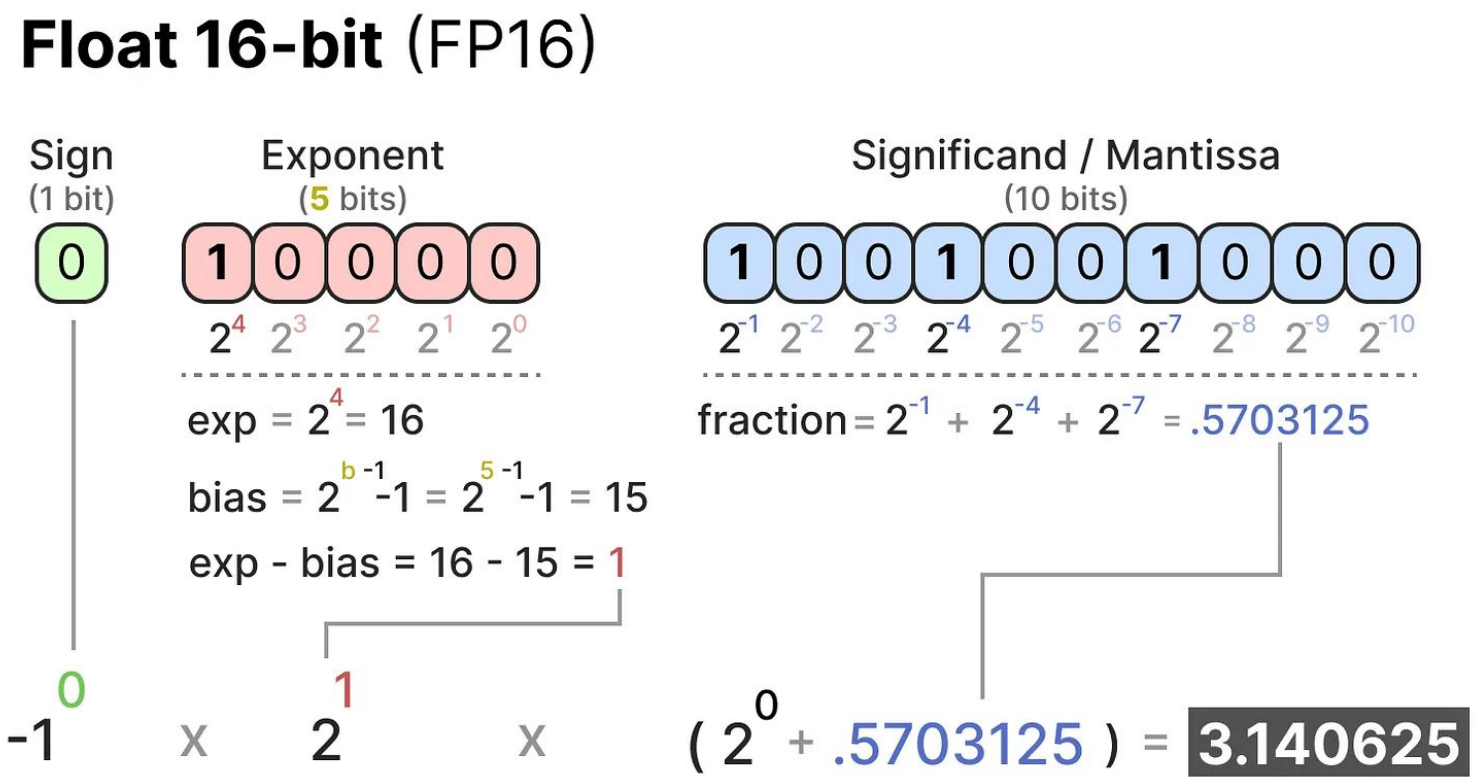
Maarten Grootendorst's blog: A Visual Guide to Quantization

How to represent numerical values



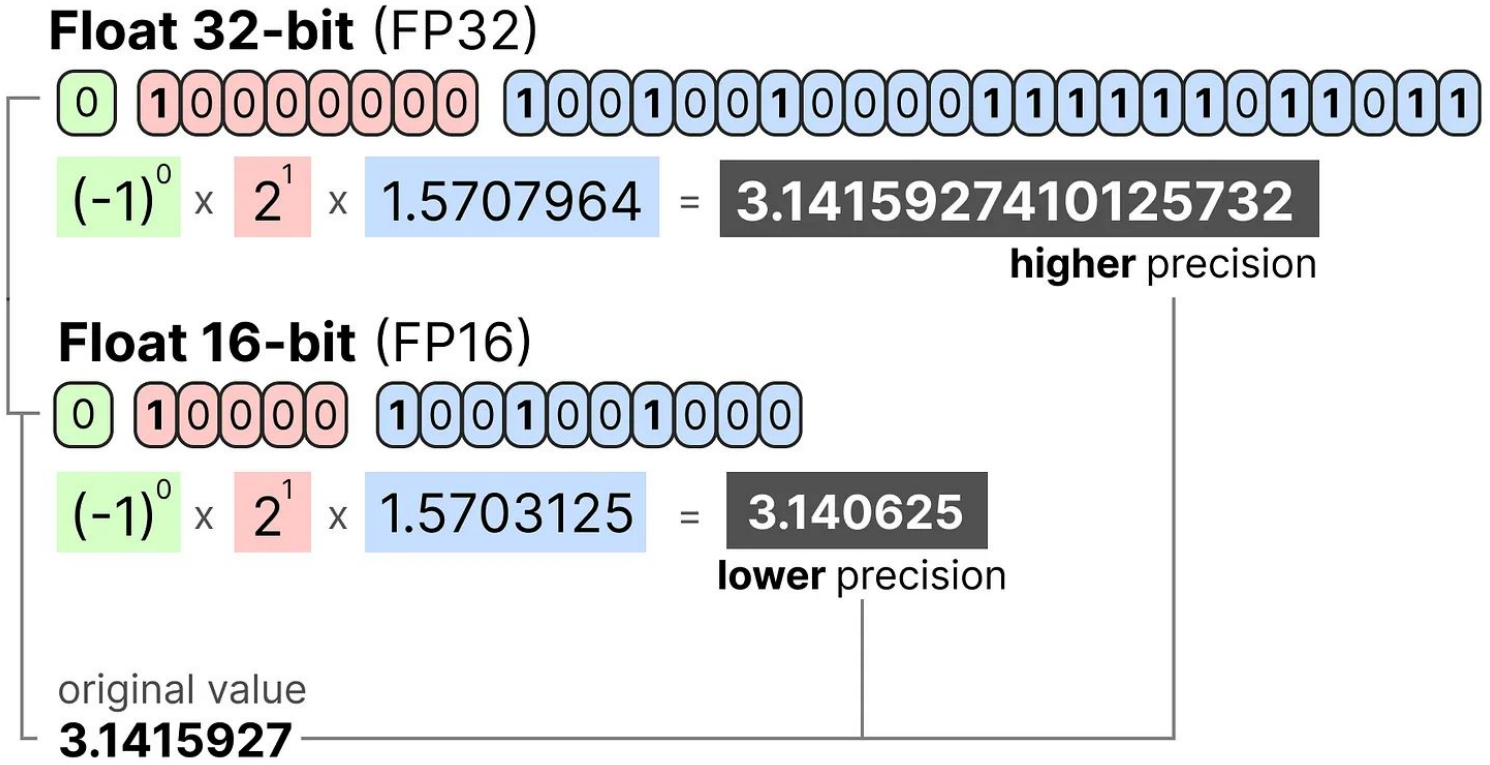
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

How to represent numerical values (cont'd)



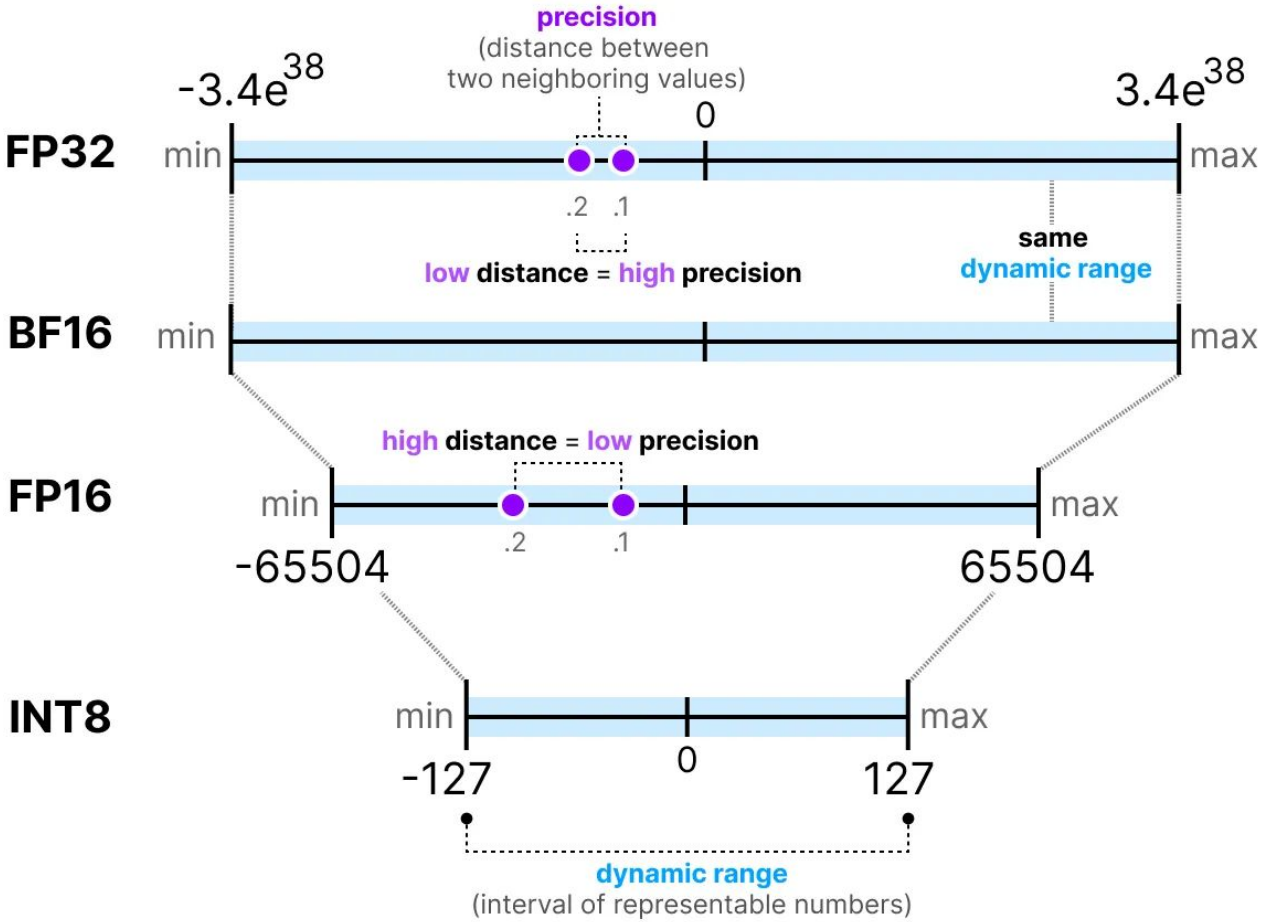
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

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)

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

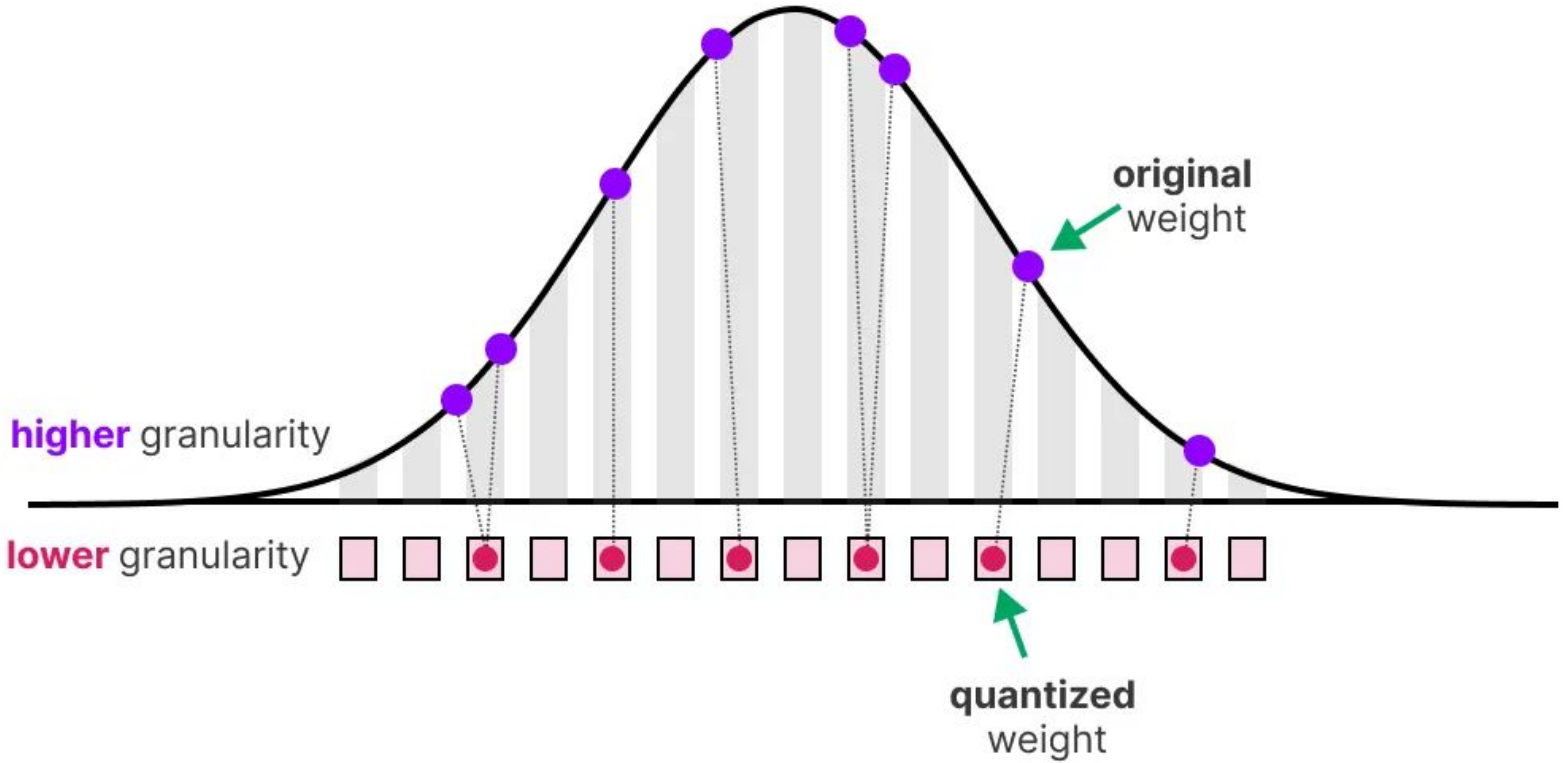
64-bits $= \frac{64}{8} \times 70\text{B} \approx$ **560** GB

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

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

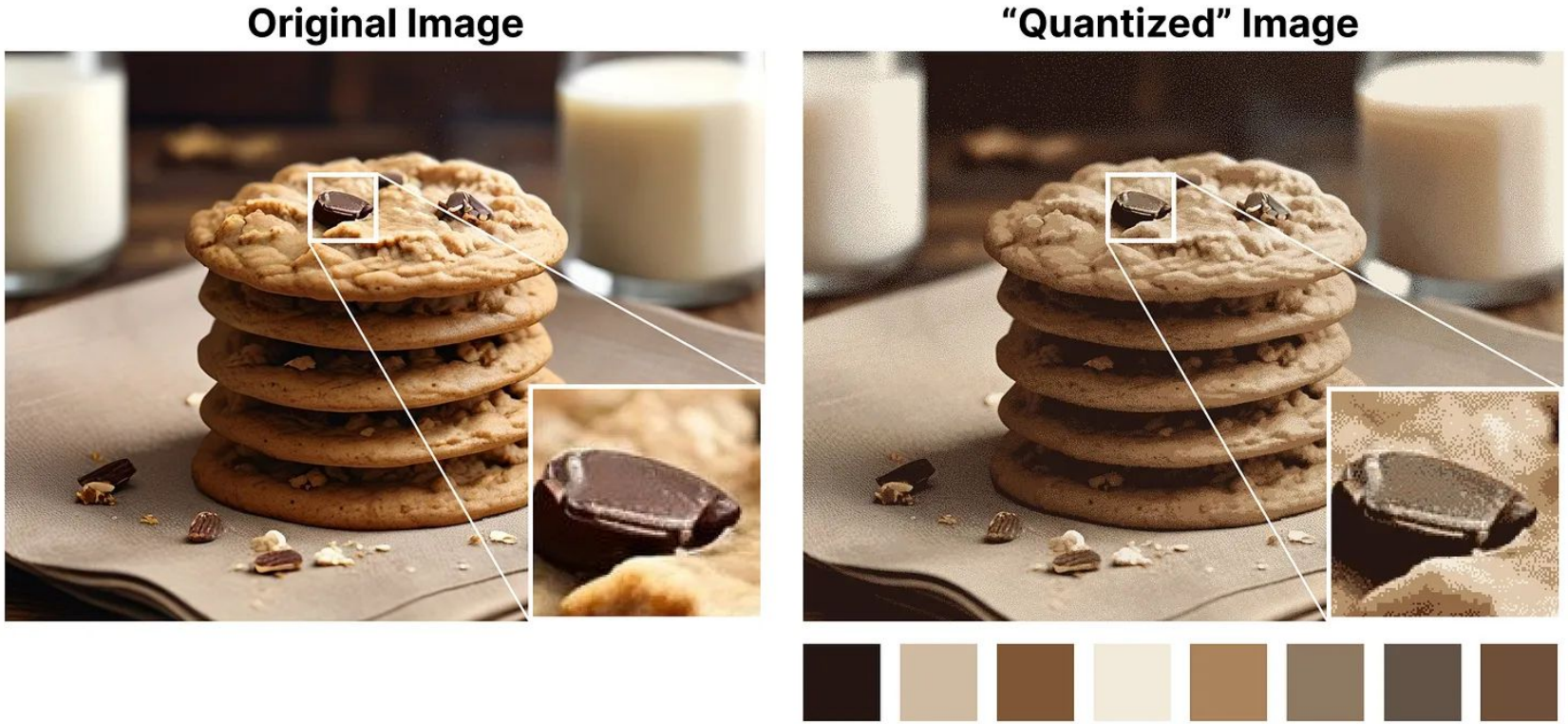
Maarten Grootendorst's blog: A Visual Guide to Quantization

Quantization



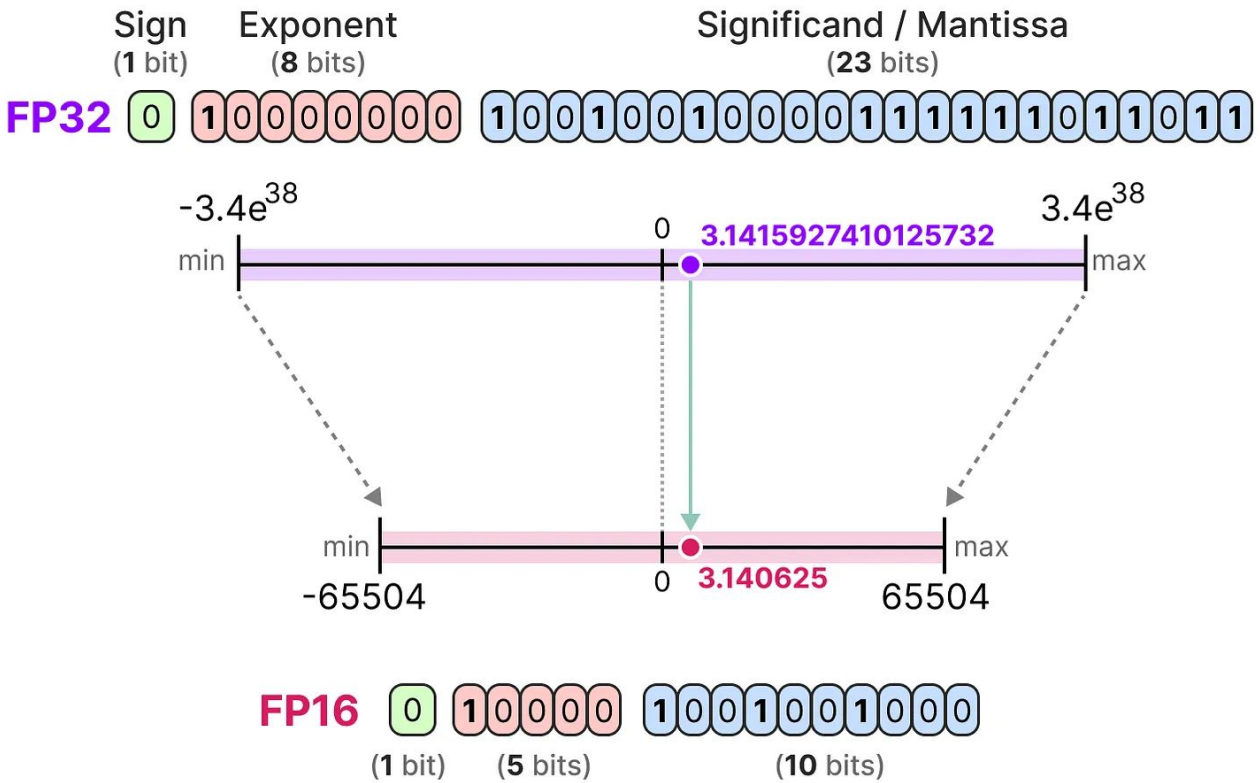
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Quantization



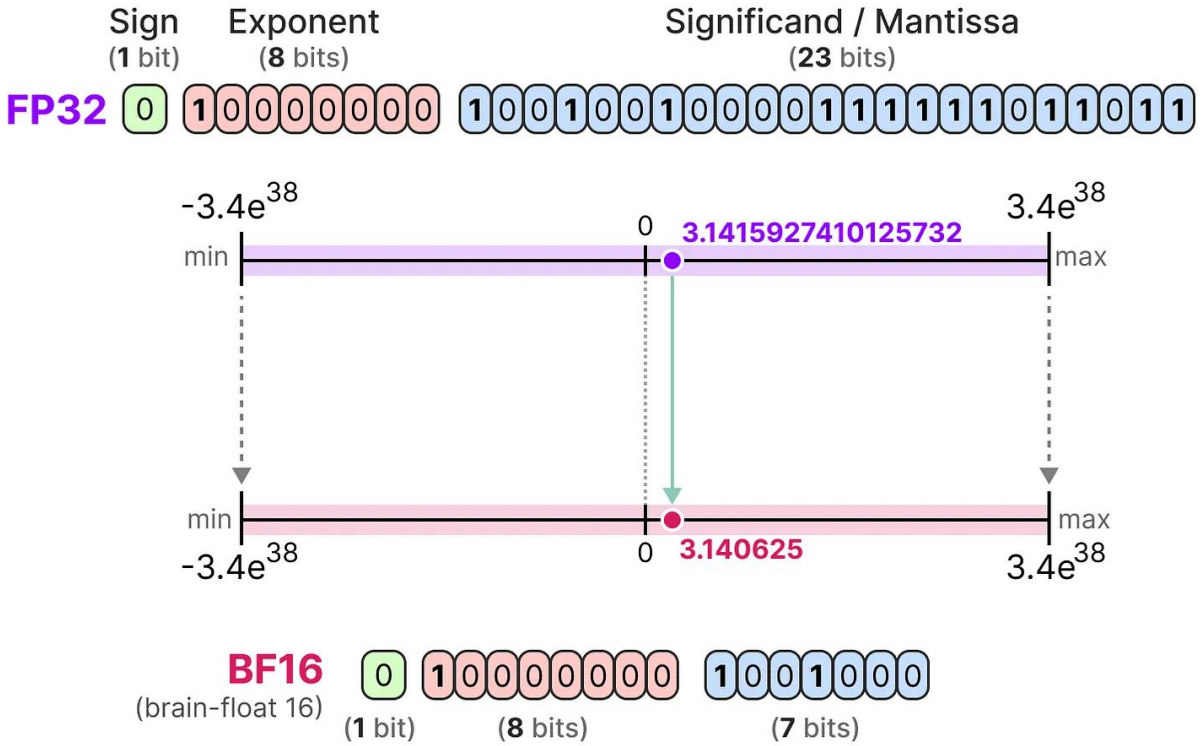
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Common data types: FP16 (half precision)



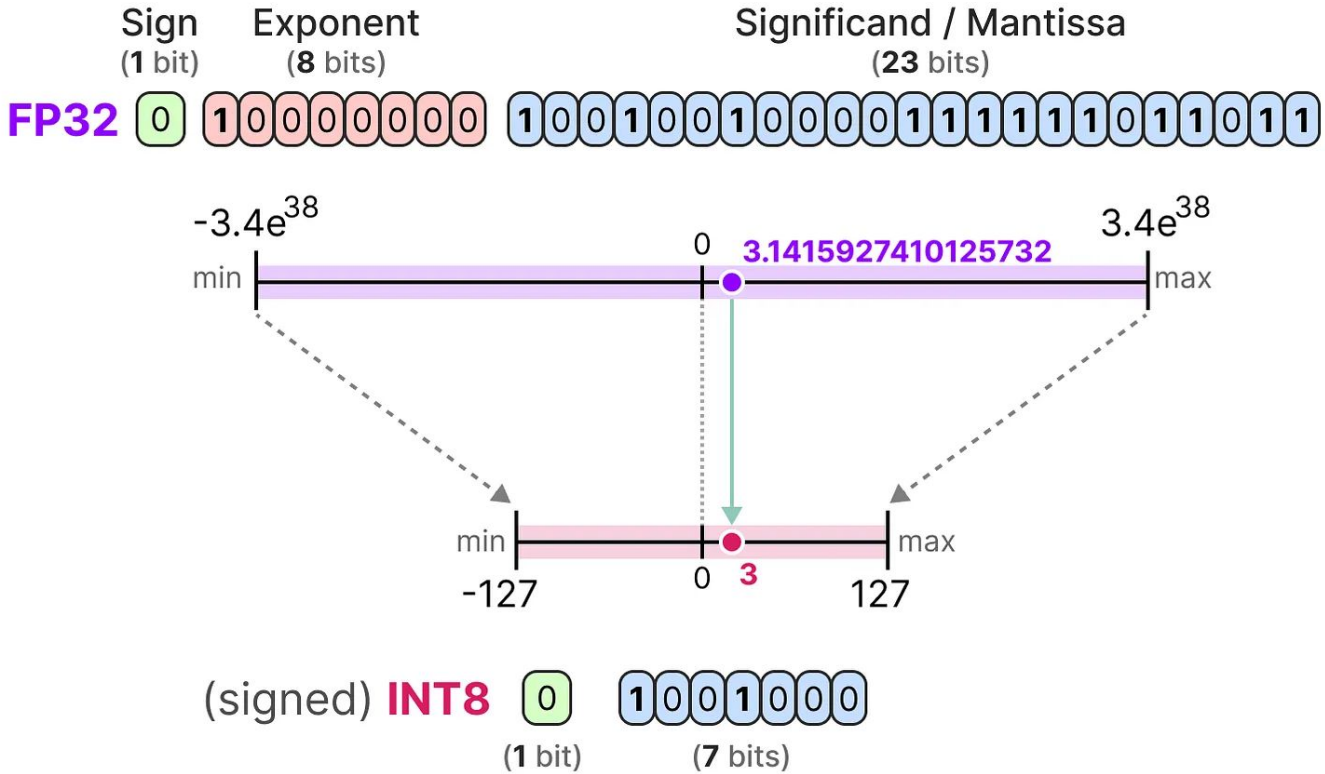
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Common data types: BF16



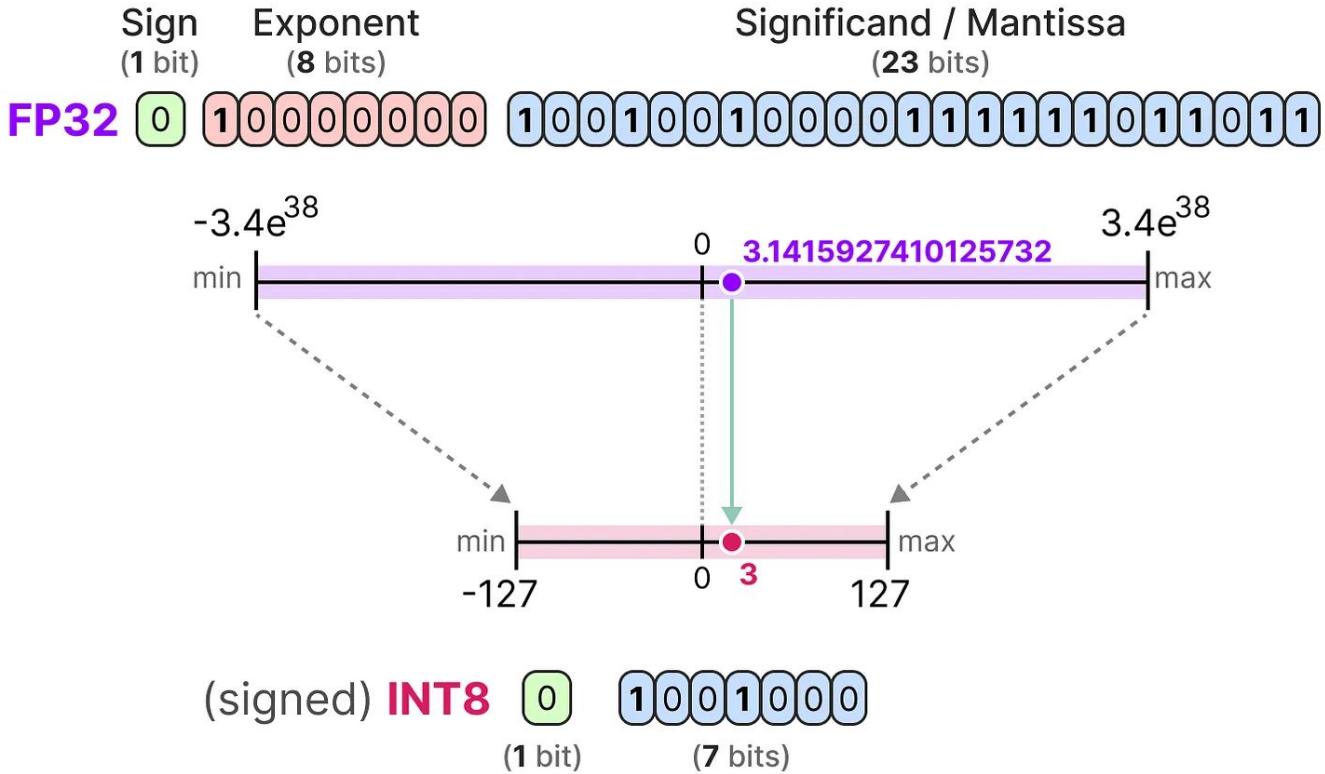
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Common data types: INT8



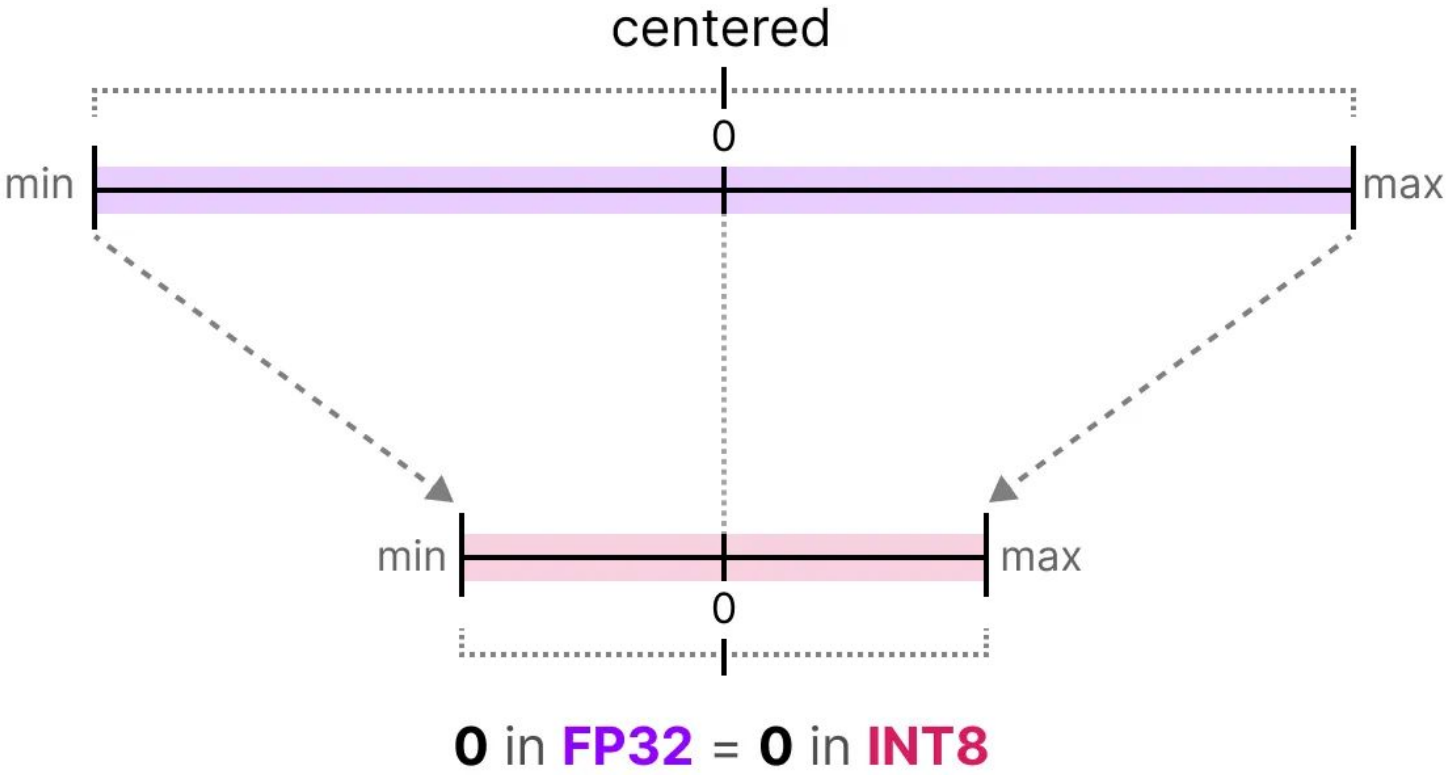
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Common data types: INT8



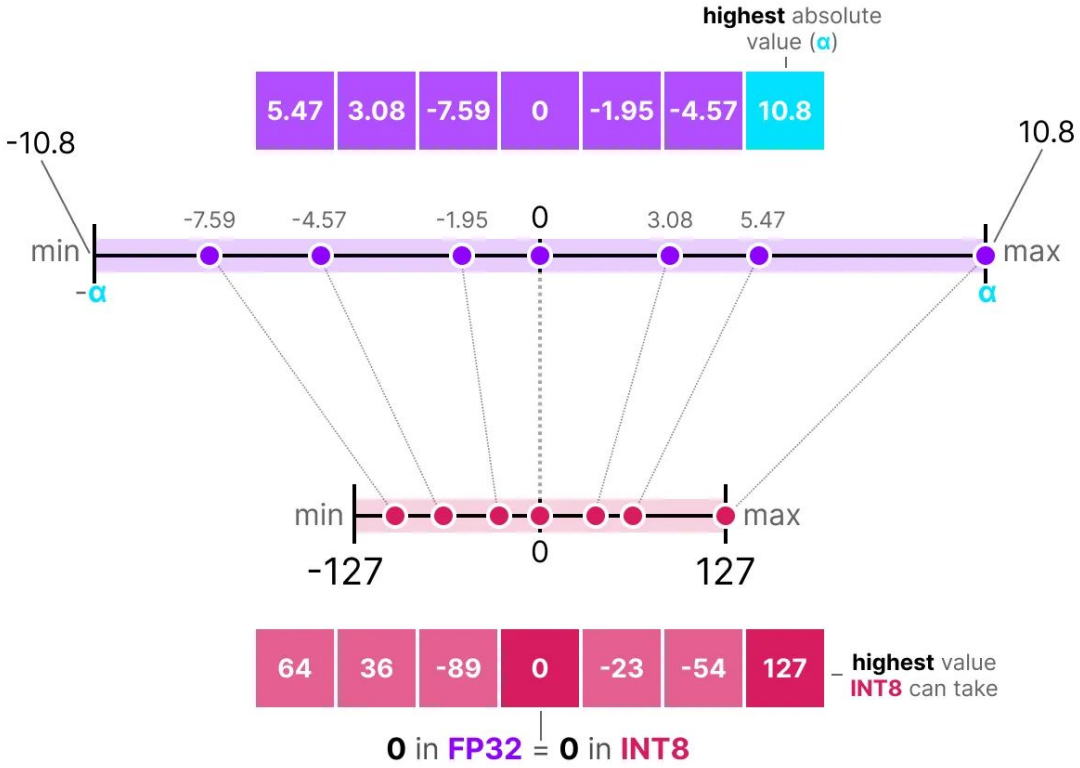
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

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:

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

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

Filling in the values would then give us the following:

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

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

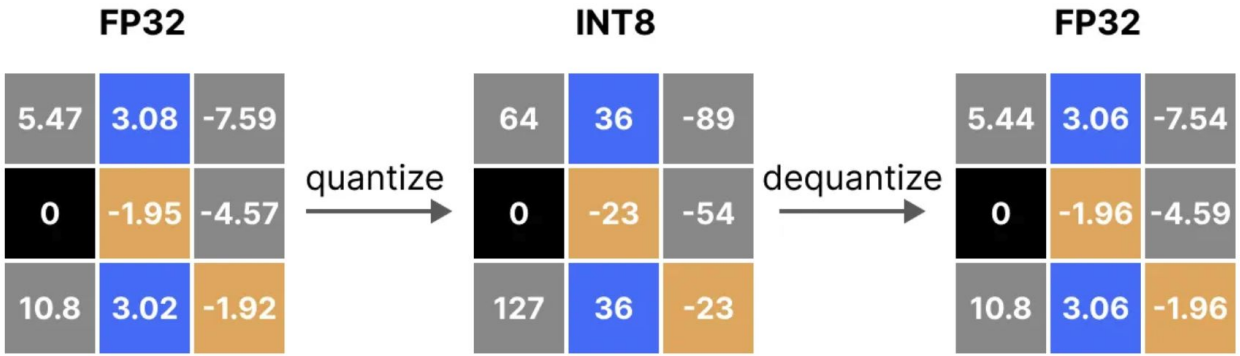
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Dequantization

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

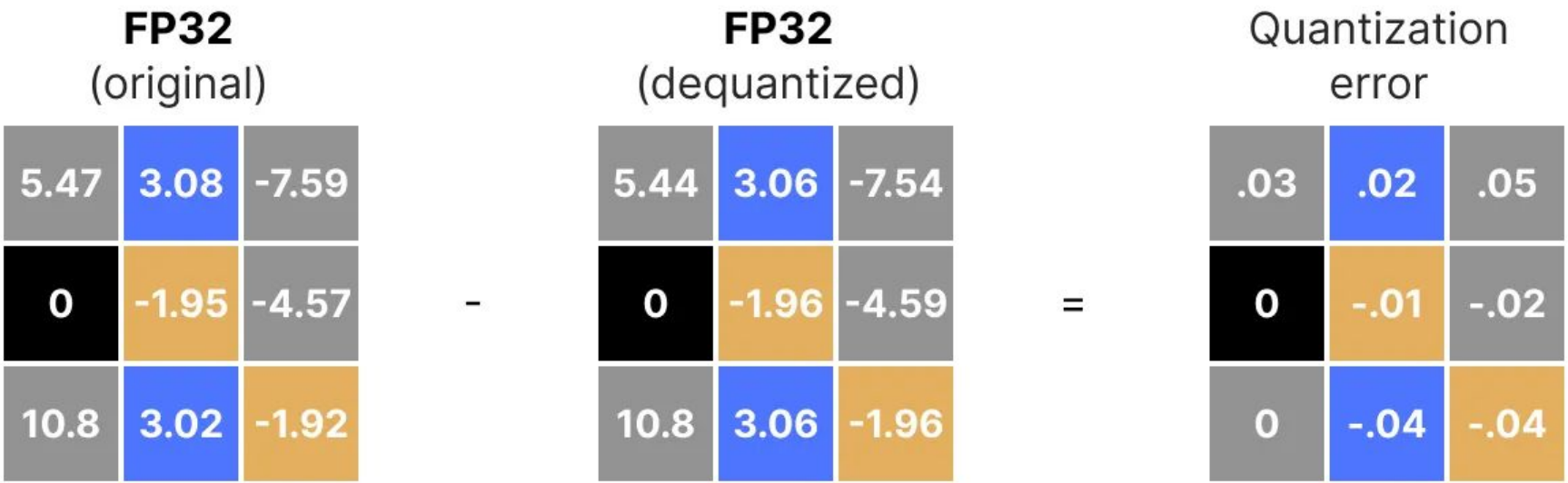
$$X_{dequantized} = \frac{\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare}{s} \qquad \text{(dequantize)}$$

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



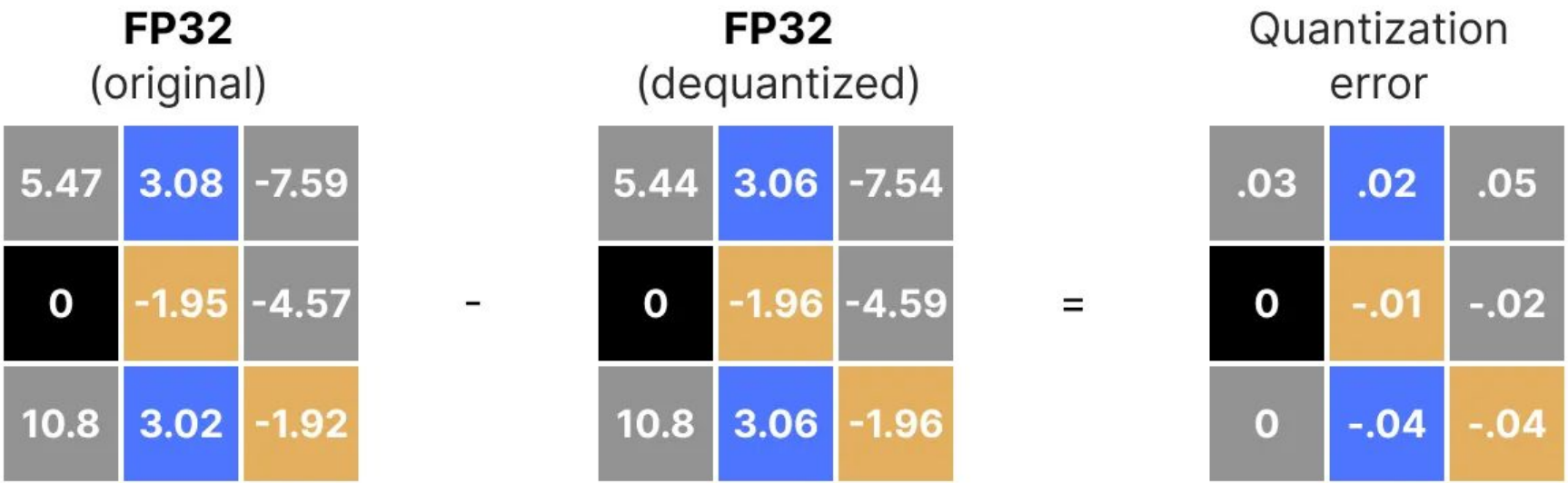
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Dequantization



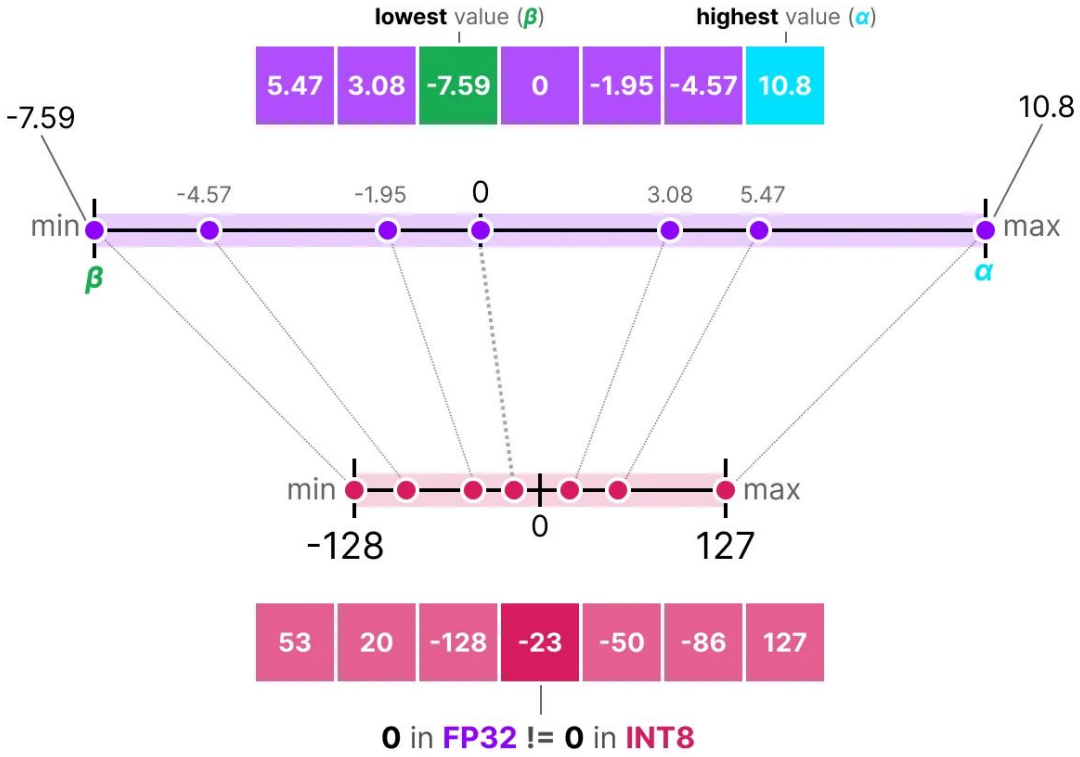
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

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)

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

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

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

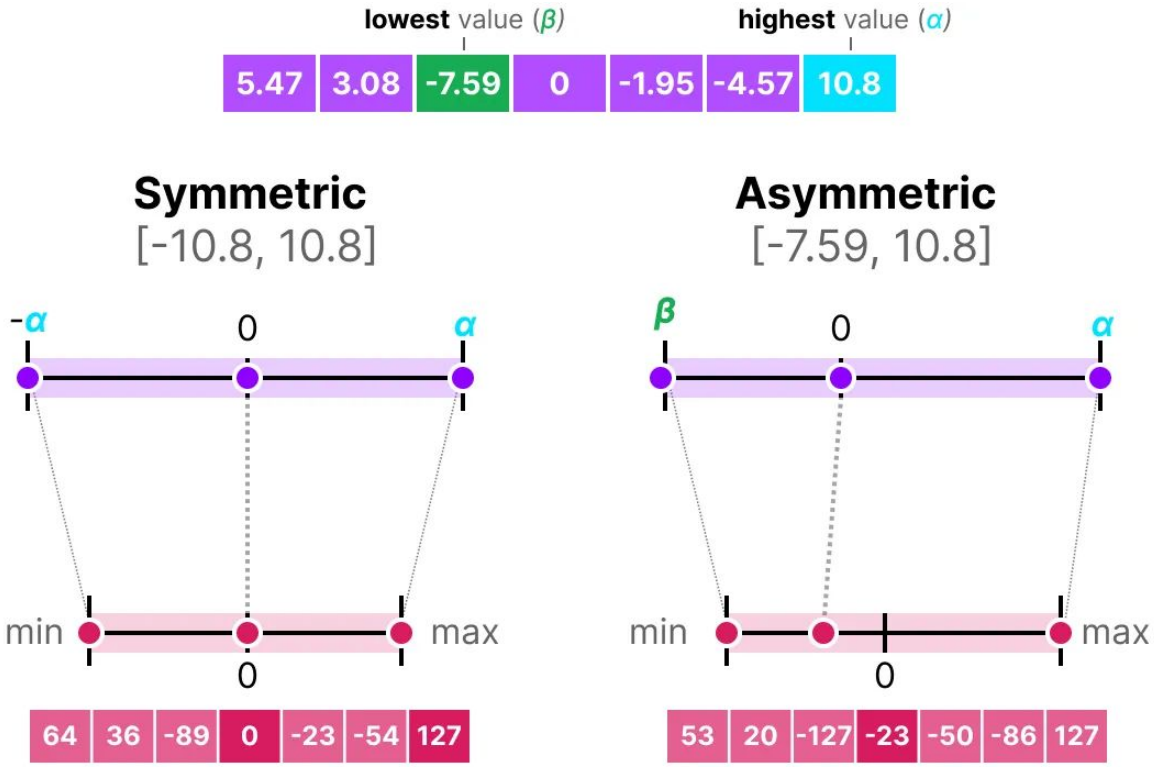
Maarten Grootendorst's blog: A Visual Guide to Quantization

Asymmetric quantization (cont'd)

$$X_{\text{dequantized}} = \frac{\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare - Z}{S} \qquad \text{(dequantize)}$$

Maarten Grootendorst's blog: A Visual Guide to Quantization

Symmetric vs. Asymmetric quantization



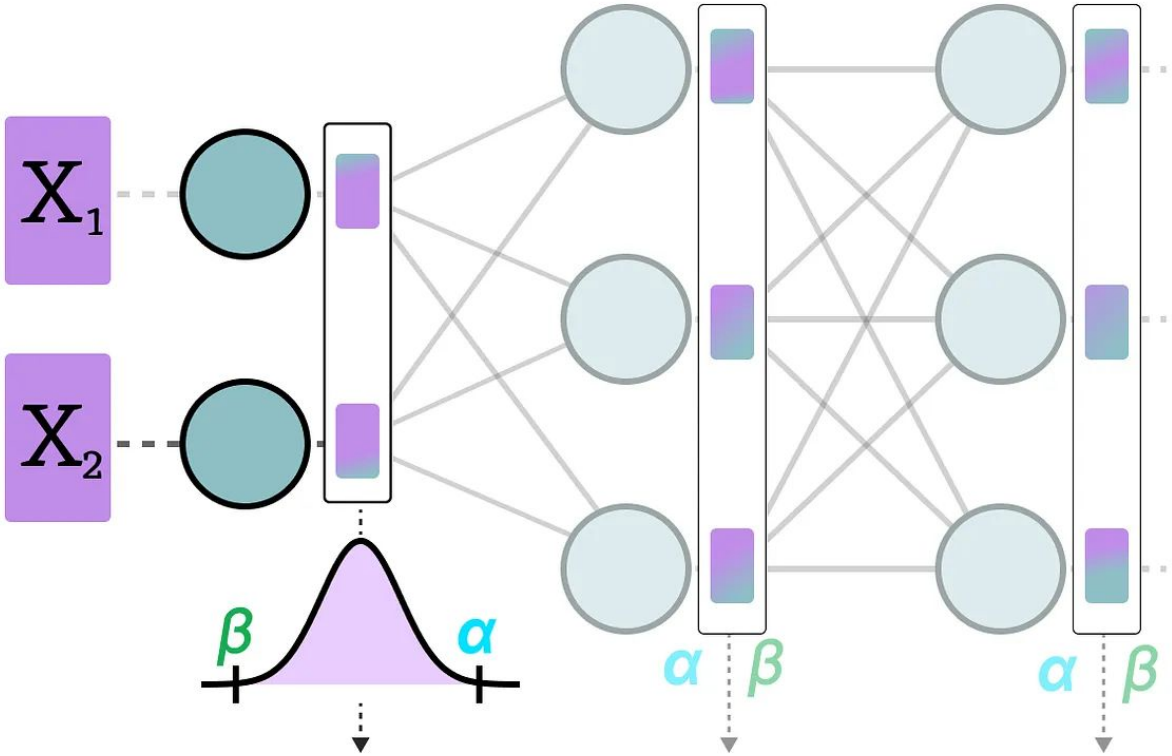
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Post-training quantization

- Dynamic Quantization
- Static Quantization

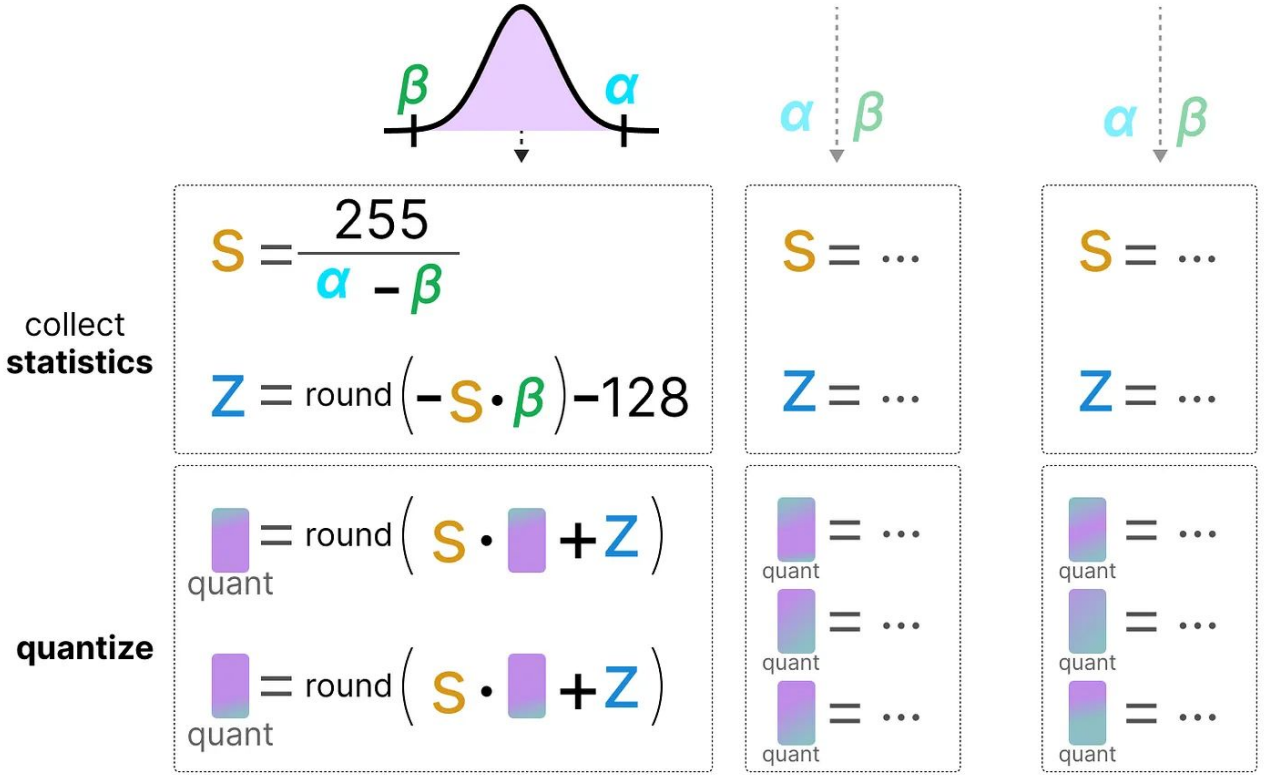
Maarten Grootendorst's blog: A Visual Guide to Quantization

Dynamic quantization



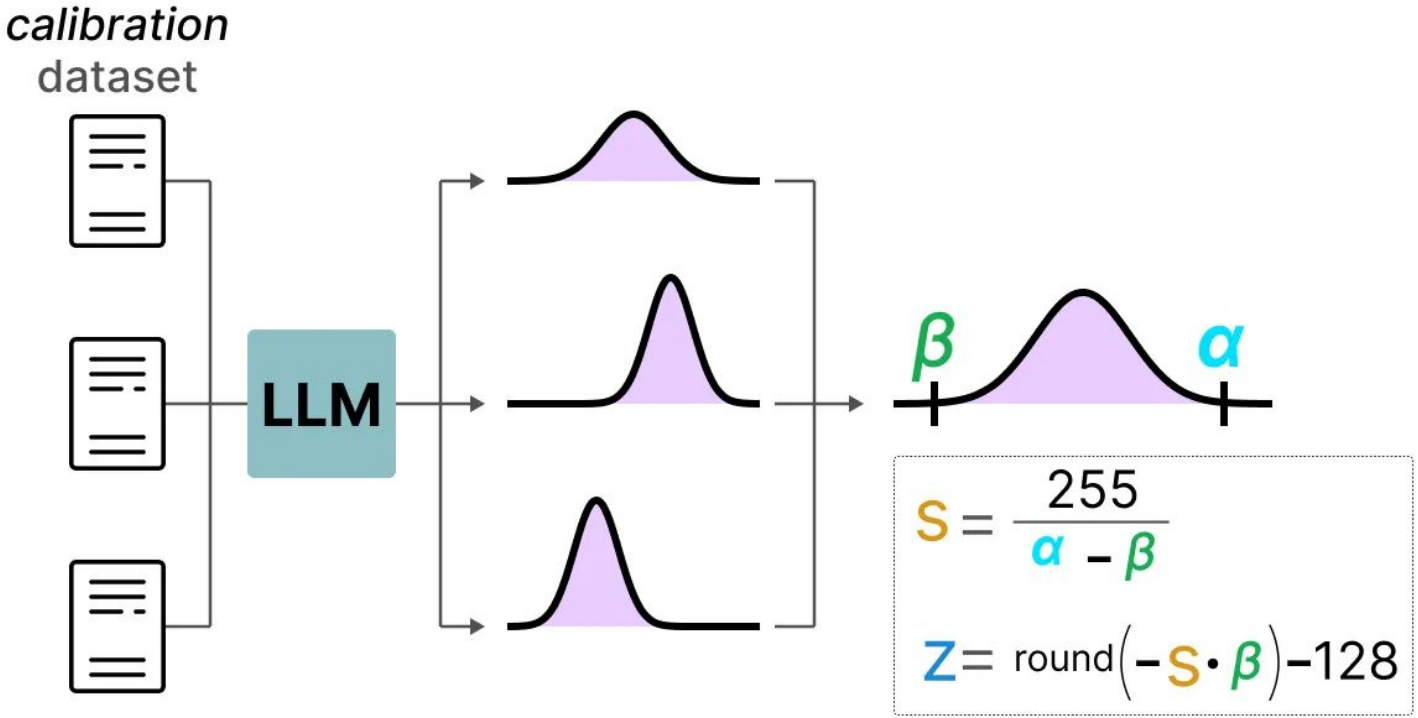
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

Dynamic quantization (cont'd)



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

Static quantization



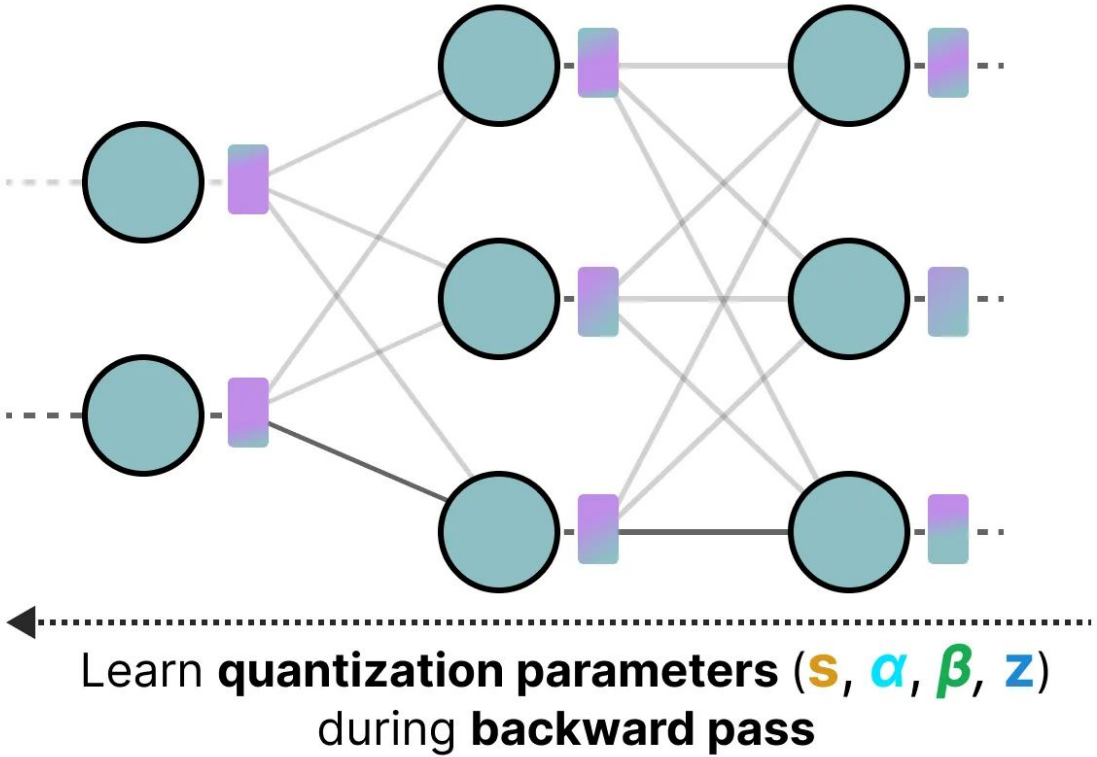
[Maarten Grootendorst's blog: A Visual Guide to Quantization]

The realm of 4-bit quantization

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

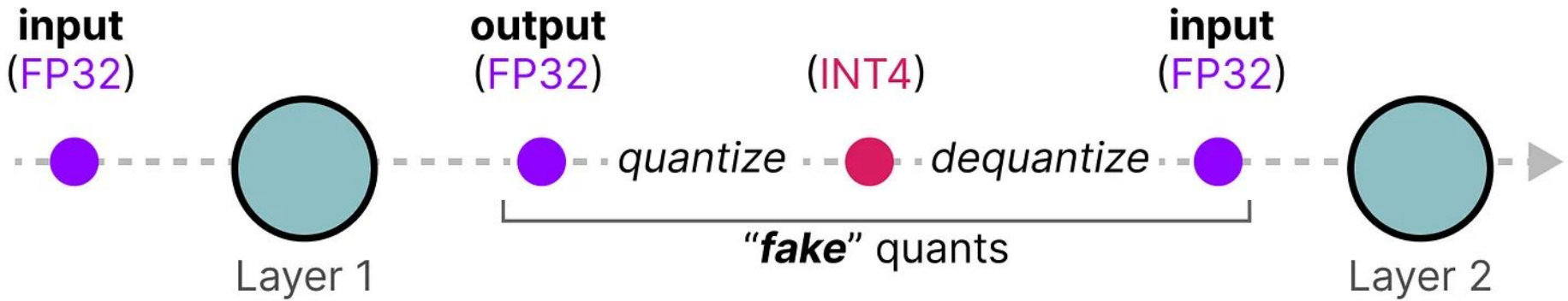
Maarten Grootendorst's blog: A Visual Guide to Quantization

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

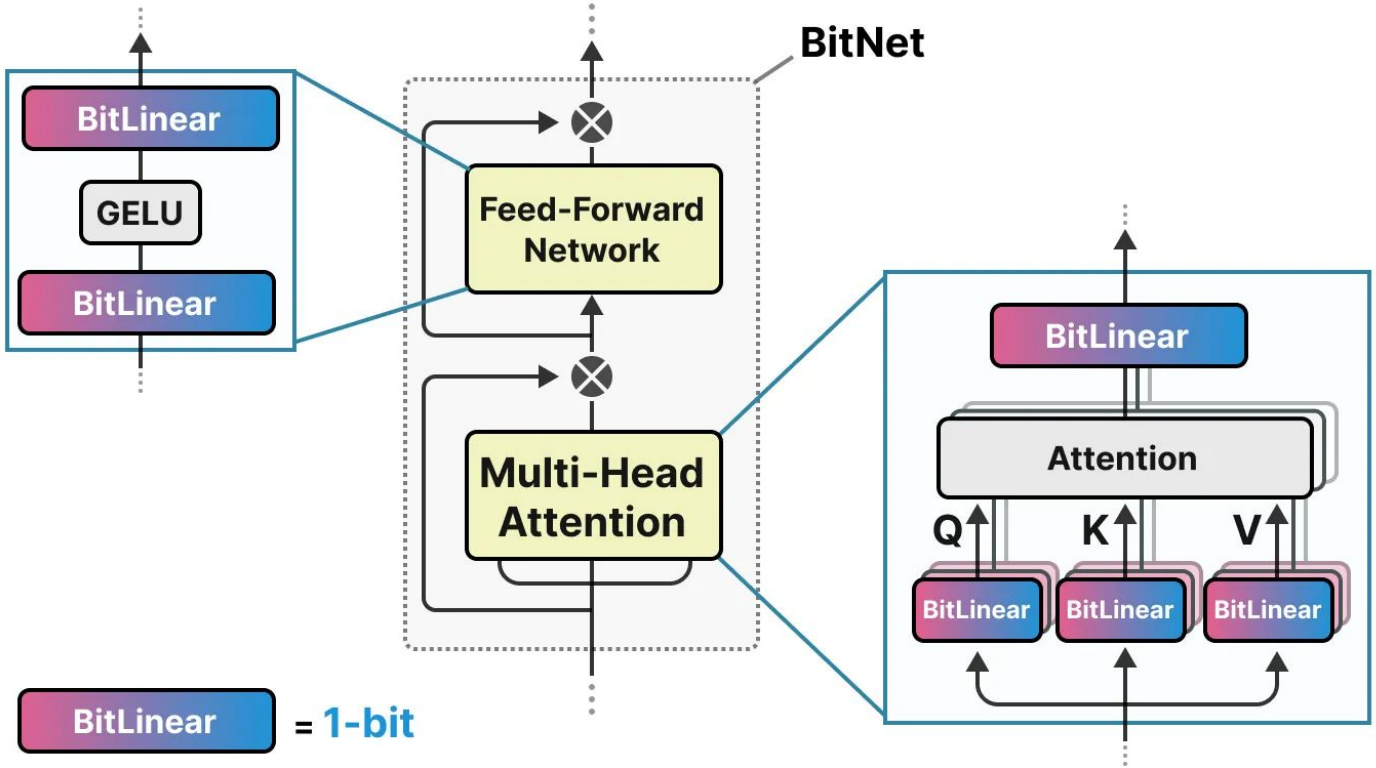
QLoRA: Efficient Finetuning of Quantized LLMs

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

Thank you!