Model Quantization

Revisiting number formats, challenges with quantization, and defocusing from weight sizes

Required

Lin et al., AWQ: Activation-aware Weight Quantization · arXiv:2306.00978

Optional

Frantar et al., GPTQ · arXiv:2210.17323 Dettmers et al., LLM.int8() · arXiv:2208.07339

Xiao et al., SmoothQuant · arXiv:2211.10438

DeepSeek-AI, DeepSeek-V3 Technical Report, the FP8 sections · arXiv:2412.19437

Christoffer Heckman

Department of Computer Science, University of Colorado Boulder

October 13, 2026

Quick overview

Bytes per weight. Last Tuesday’s decoding sequence but with 1/4 size weights.

Floats, integers, blocks, and scale factor effects

Outlier channels. Why a language model is harder to quantize than a vision model.

Measured on Qwen3-1.7B on my RTX 4090; how the SmoothQuant identity is derived

Weight-only quantization. Round to nearest, then GPTQ and AWQ.

What a 4-bit kernel gains at batch 1

Beyond the weights. The KV cache, FP8 training, and training with the quantizer in the loop.

Not going to give away what Mark's doing though! ;)

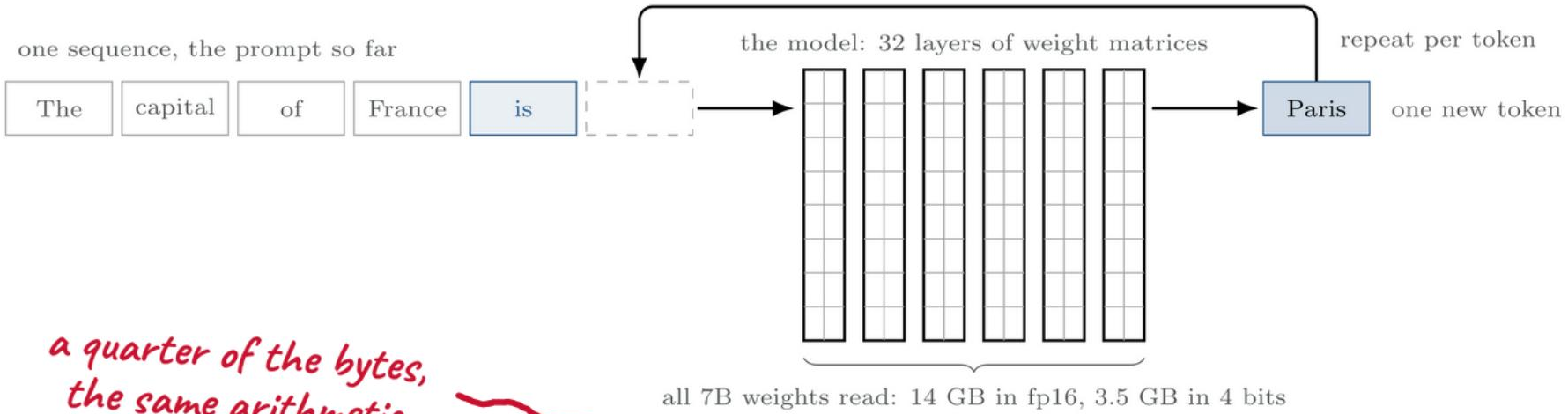
What quantization loses. Let’s play a little game!

Perplexity against task accuracy, 16 bits down to 2

1 THE RECAP

Bytes per Weight

One Read per Token



a quarter of the bytes, the same arithmetic

per decode step, $N = 7 \times 10^9$ weights	batch 1	4-bit weights		
bytes read	$2N \approx 14$ GB	$N/2 \approx 3.5$ GB		fp16 stores 2 bytes per weight
FLOPs	$2N \approx 14$ GFLOP	$2N$	the same arithmetic	a multiply and an add per weight, per sequence
FLOPs per byte	≈ 1	≈ 4		

an H100 can do about 300 FLOPs per byte it reads (989 TFLOPS dense bf16 against 3.35 TB/s; approximate)

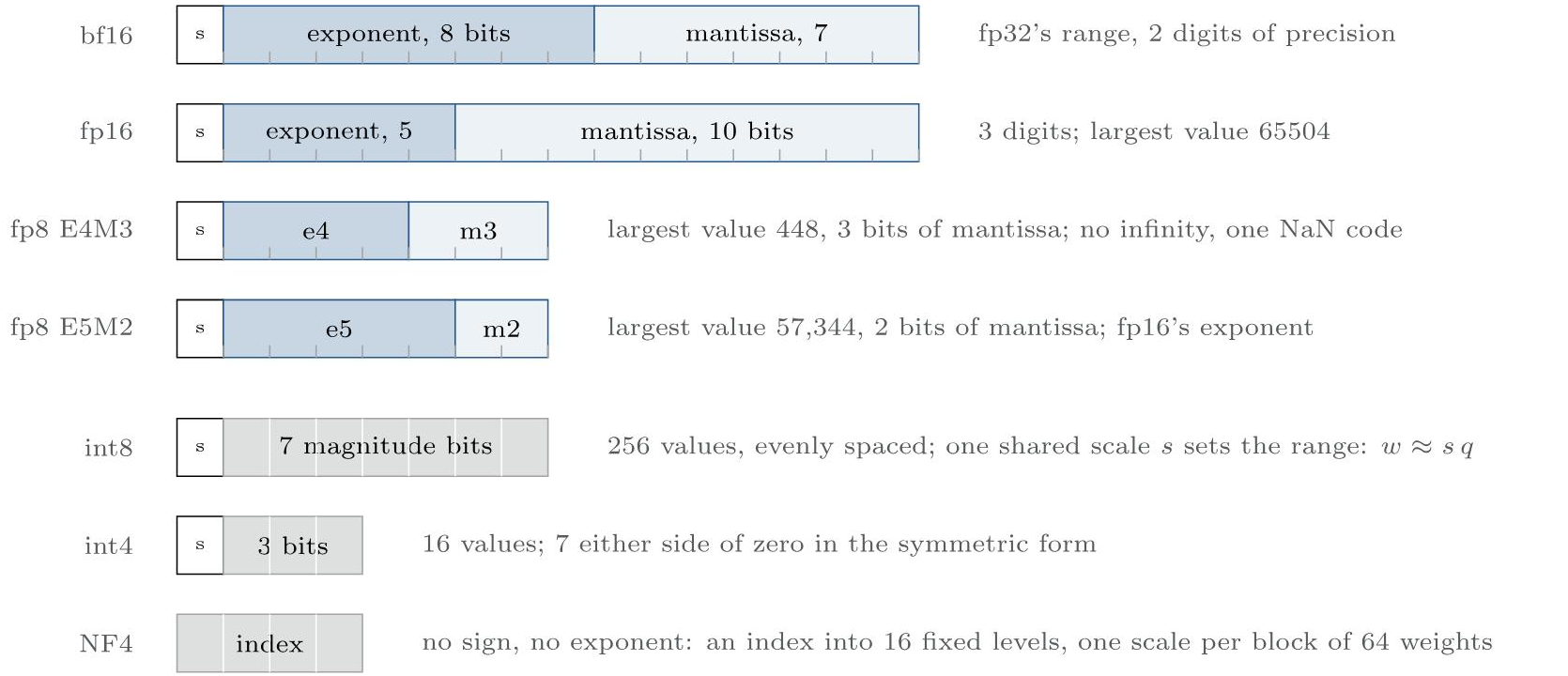
```
the same step with 4-bit weights:  3.5 GB at 3.35 TB/s, about 1 ms
```

Schematic for a 7B model. H100 figures from NVIDIA's data sheet (989 TFLOPS dense bf16, 3.35 TB/s HBM3), approximate. The 4-bit column counts weight bytes only; scales add about 3 percent.

2 NUMBER FORMATS

Range and Precision

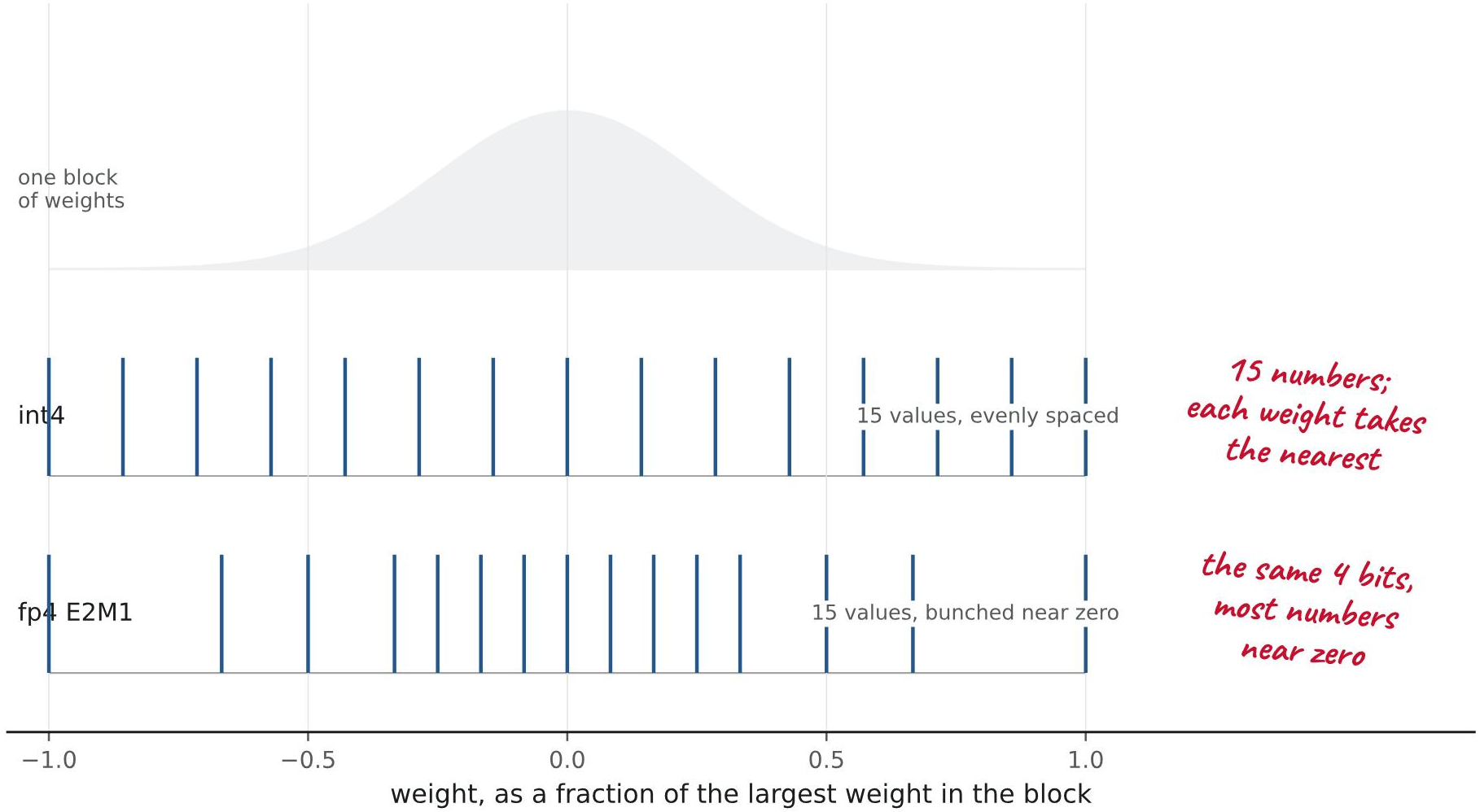
Floats and Integers



an integer has no exponent; the scale s chosen per tensor, per channel or per group supplies the range

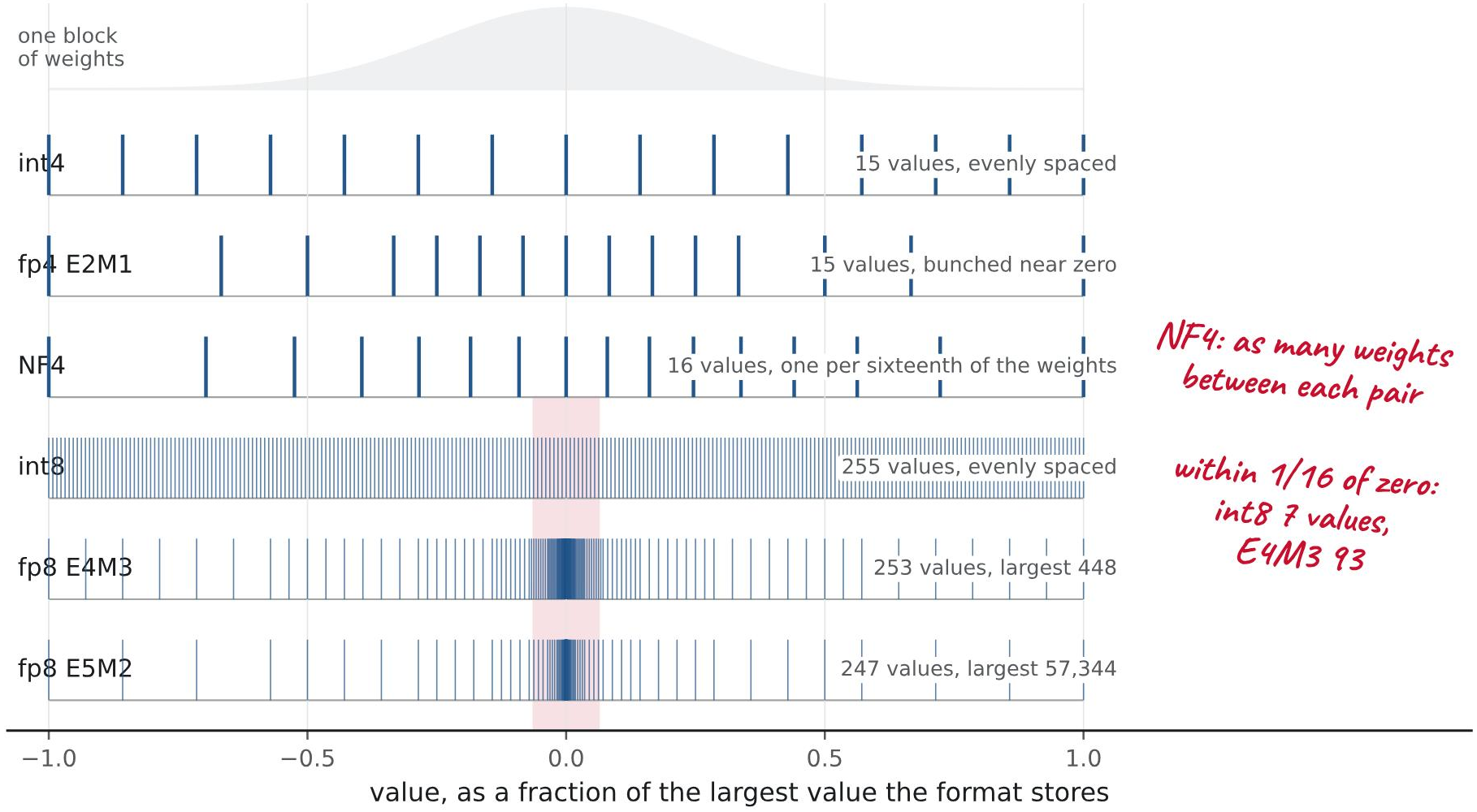
fp8 E4M3 and E5M2 after Micikevicius et al., "FP8 Formats for Deep Learning," 2022, arXiv:2209.05433. NF4 after Dettmers et al., "QLoRA," 2023, arXiv:2305.14314.

Sixteen Values



fp4 E2M1 after Rouhani et al., "Microscaling Data Formats for Deep Learning," 2023, arXiv:2310.10537.

The Number Line



NF4: as many weights between each pair

within 1/16 of zero: int8 7 values, E4M3 93

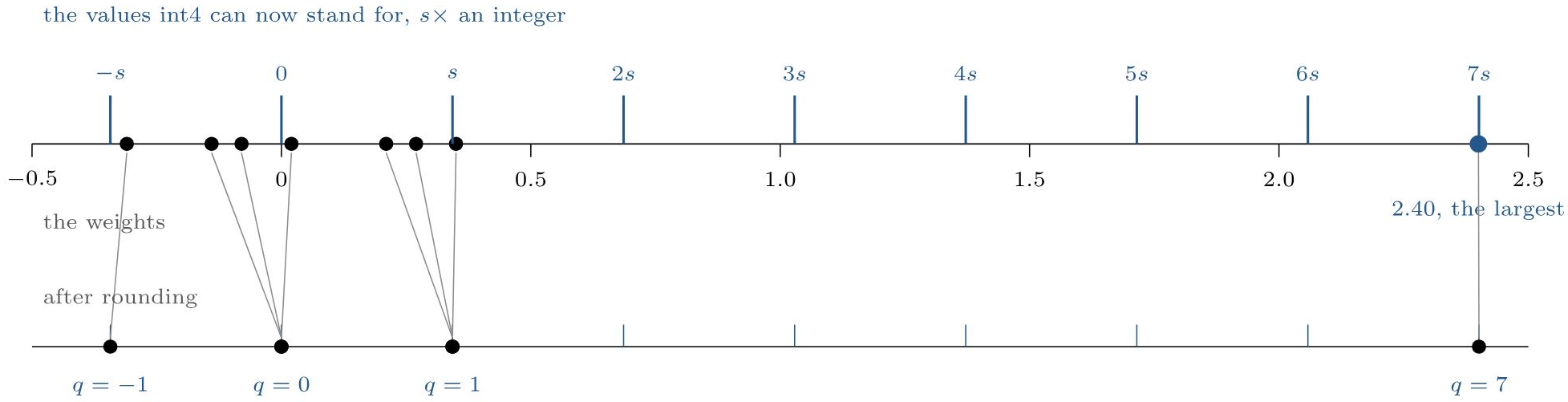
fp8 after Micikevicius et al., "FP8 Formats for Deep Learning," 2022, arXiv:2209.05433; NF4 after Dettmers et al., "QLoRA," 2023, arXiv:2305.14314.

Absmax Scaling

eight fp16 weights from one row of W; int4 can store only the integers $-7, -6, \ldots, 6, 7$

choose s so the largest weight maps to 7: $7s = \max_i |w_i| = 2.40$, so $s = 2.40/7 = 0.343$

for b bits the top integer is $2^{b-1} - 1$, so $s = \max_i |w_i| \,/\, (2^{b-1} - 1)$



round each weight to the nearest one: $q_i = \text{round}(w_i/s)$, and the model reads back $\hat{w}_i = s\, q_i$

Schematic of symmetric absmax quantization at 4 bits.

Scale and Round

eight weights from one row of W

the weights, fp16	w	0.21	−0.08	0.35	−0.14	0.02	0.27	−0.31	2.40
4-bit integers	q	1	0	1	0	0	1	−1	7
read back, $s \times q$	$\hat{w}$	0.34	0.00	0.34	0.00	0.00	0.34	−0.34	2.40
rounding error	$\lvert w - \hat{w} \rvert$	0.13	0.08	0.01	0.14	0.02	0.07	0.03	0.00

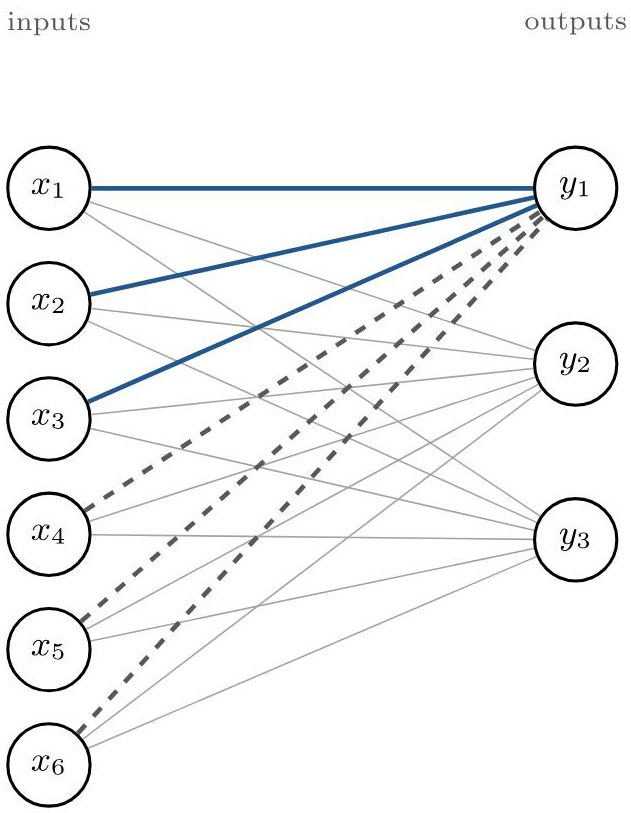
$$s = \max_i |w_i| \,/\, (2^{b-1} - 1) = 2.40/7 = 0.343$$ the scale: the largest weight maps to 7

$$q_i = \text{round}(w_i / s)$$ the integer stored, −7 to 7

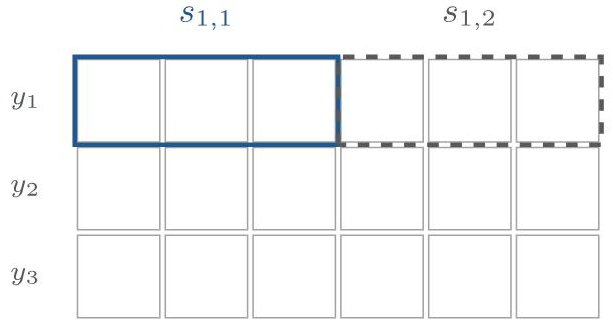
$$\hat{w}_i = s\, q_i$$ what the model reads back, in fp16

Symmetric absmax quantization at $b = 4$.

Blocks of Weights



the weight matrix W: a row per output, a column per input



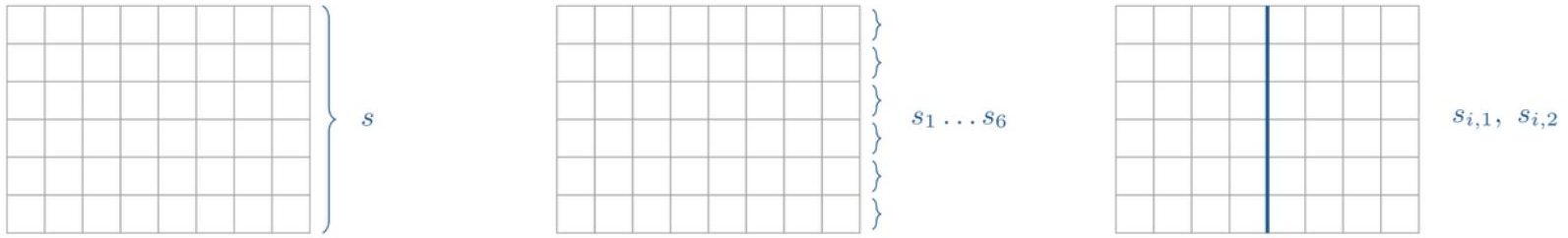
a block: g neighboring weights in one row, sharing one scale

here $g = 3$; in practice $g = 128$, so a row of 2048 weights has 16 scales

a scale per block keeps one outlier from setting the step for the whole row

Schematic of one linear layer.

Block Formats



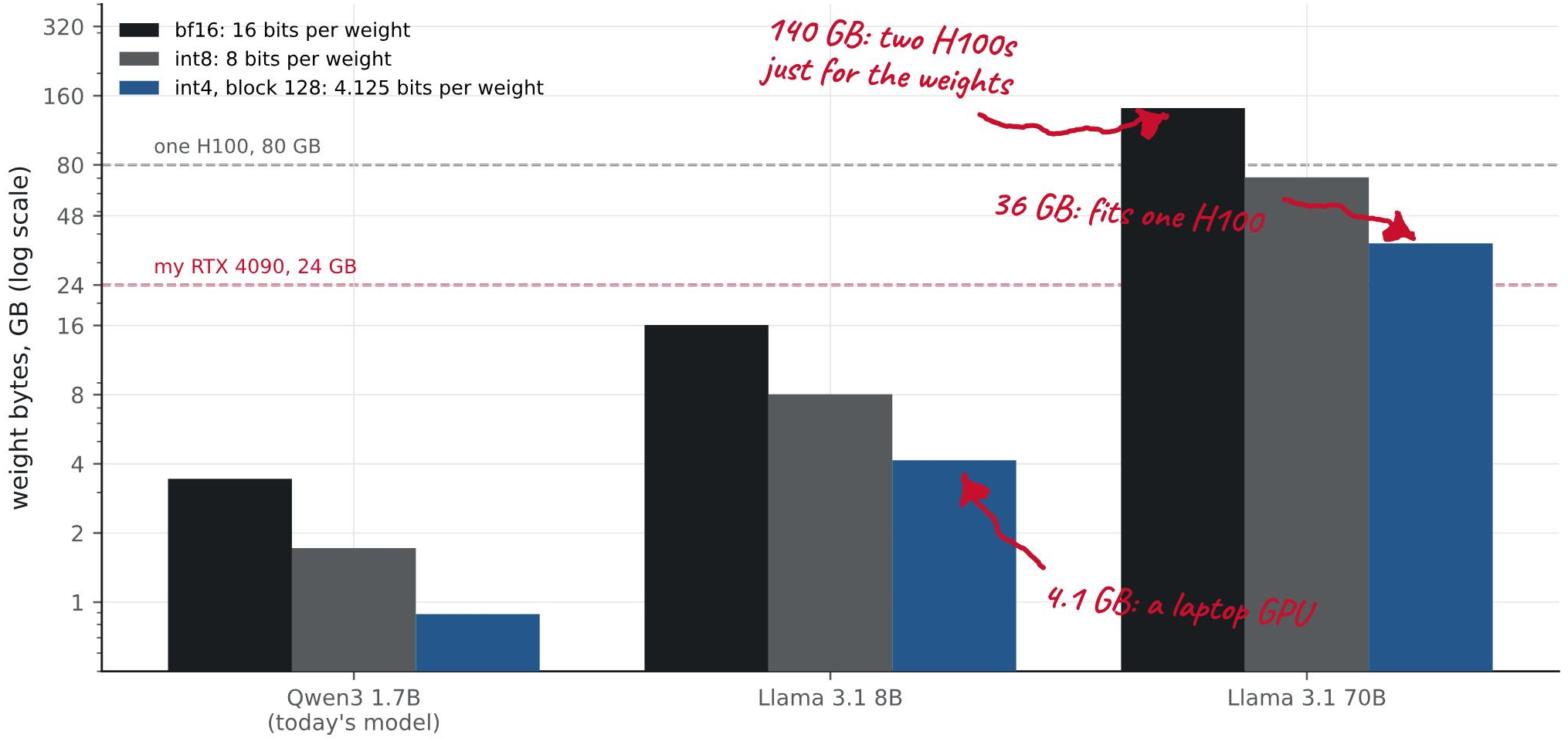
finer scales track the weights better and take more bits per weight

int4, block 128 the common 4-bit choice	4 b | 4 b | 4 b | 4 b | 4 b · · · 128 weights × 4 bits · · · 4 b | 4 b	fp16 scale, 16 b; 4-bit zero, 4 b	$4 + 20/128 = 4.16$ bits per weight
NF4, block 64 QLoRA	4 b | 4 b | 4 b | 4 b | 4 b · · · 64 weights × 4 bits · · · 4 b | 4 b	fp32 scale, 32 b	$4 + 32/64 = 4.5$ double quantization of the scales: 4.13
MXFP4 (OCP, 2023) FP4 E2M1 elements	4 b | 4 b | 4 b | 4 b | 4 b · · · 32 weights × 4 bits · · · 4 b | 4 b	E8M0 scale, 8 b: a power of two	$4 + 8/32 = 4.25$
NVFP4 (Blackwell, 2025) FP4 E2M1 elements	4 b | 4 b | 4 b | 4 b | 4 b · · · 16 weights × 4 bits · · · 4 b | 4 b	E4M3 scale, 8 b and one fp32 scale per tensor	$4 + 8/16 = 4.5$

some papers say group, others say block: the same thing

Blocks of 128 after Frantar et al., 2022, arXiv:2210.17323, and Lin et al., 2023, arXiv:2306.00978; NF4 and double quantization after QLoRA (arXiv:2305.14314); MXFP4 after Rouhani et al., "Microscaling Data Formats for Deep Learning," 2023, arXiv:2310.10537; NVFP4 after NVIDIA, "Pretraining Large Language Models with NVFP4," 2025, arXiv:2509.25149.

Fits on the Card

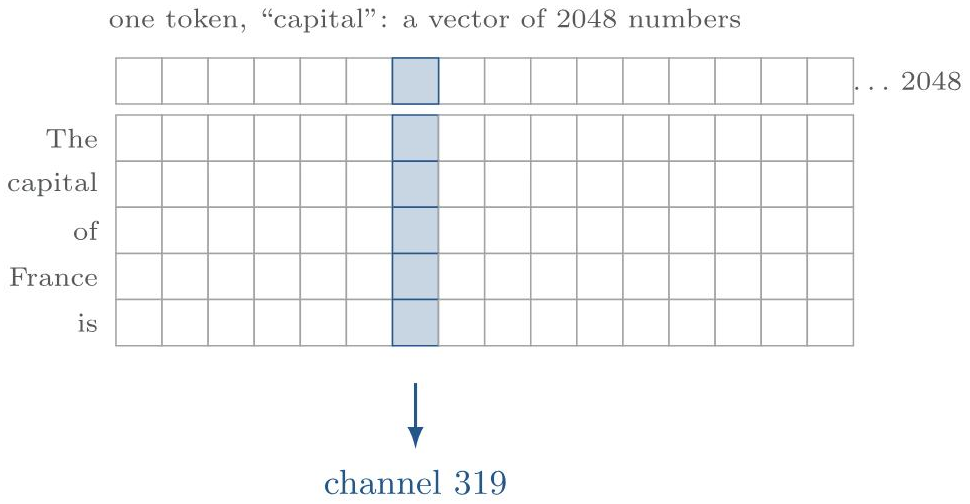


Weights only, no KV cache or activations: $\text{bytes} = \text{parameters} \times \text{bits}/8$. int4 shares one 16-bit scale across 128 weights: $4 + 16/128 = 4.125$ bits per weight.

3 ACTIVATION OUTLIERS

Outlier Channels

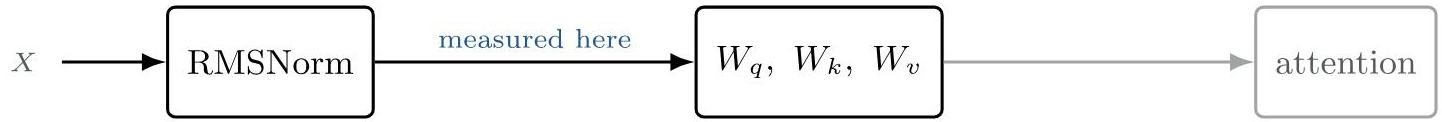
Channels



X: one row per token, one column per channel

a channel is one position in the vector, the same position in every token

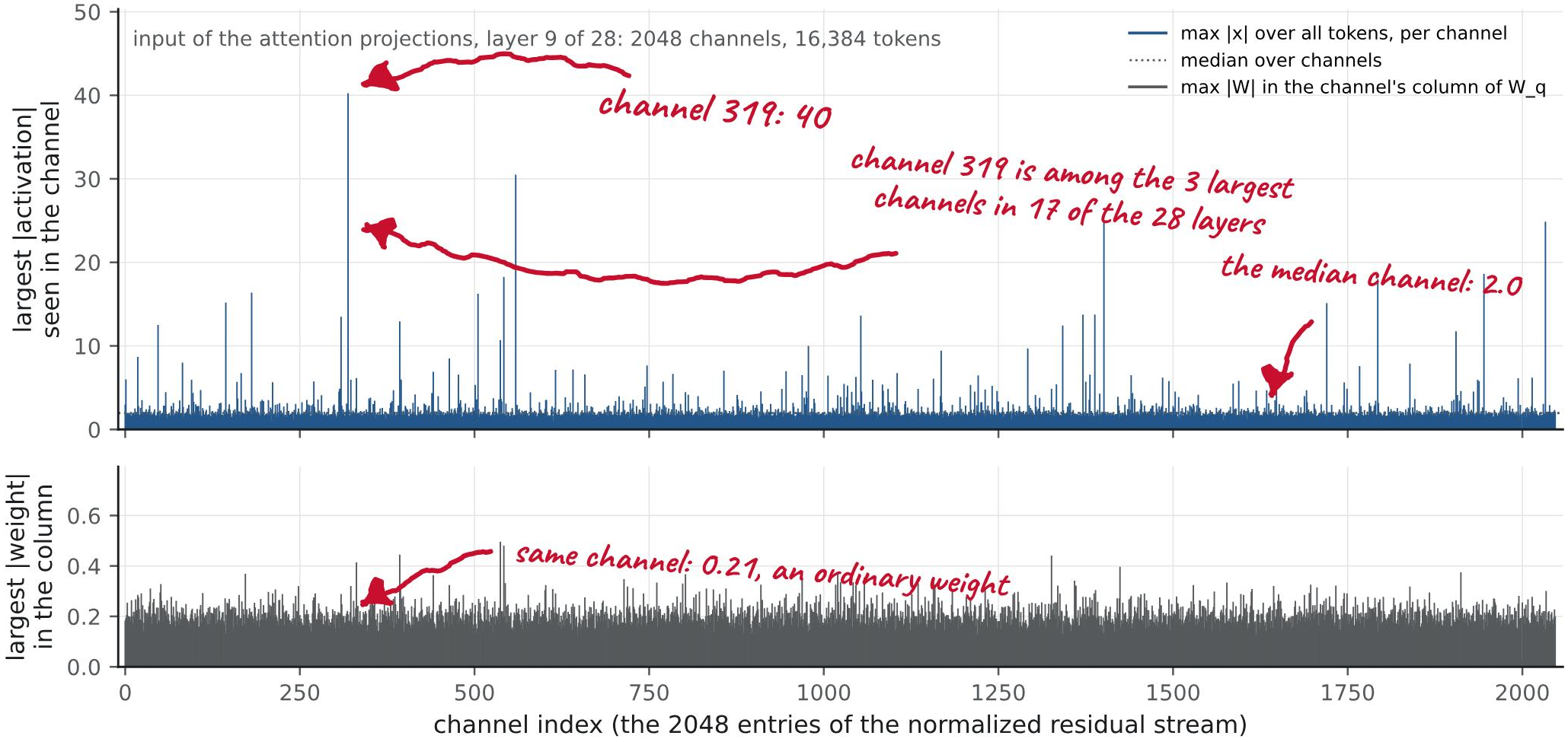
next: for each of the 2048 channels, the largest value it reaches over all tokens



layer 9 of 28 in Qwen3-1.7B-Base

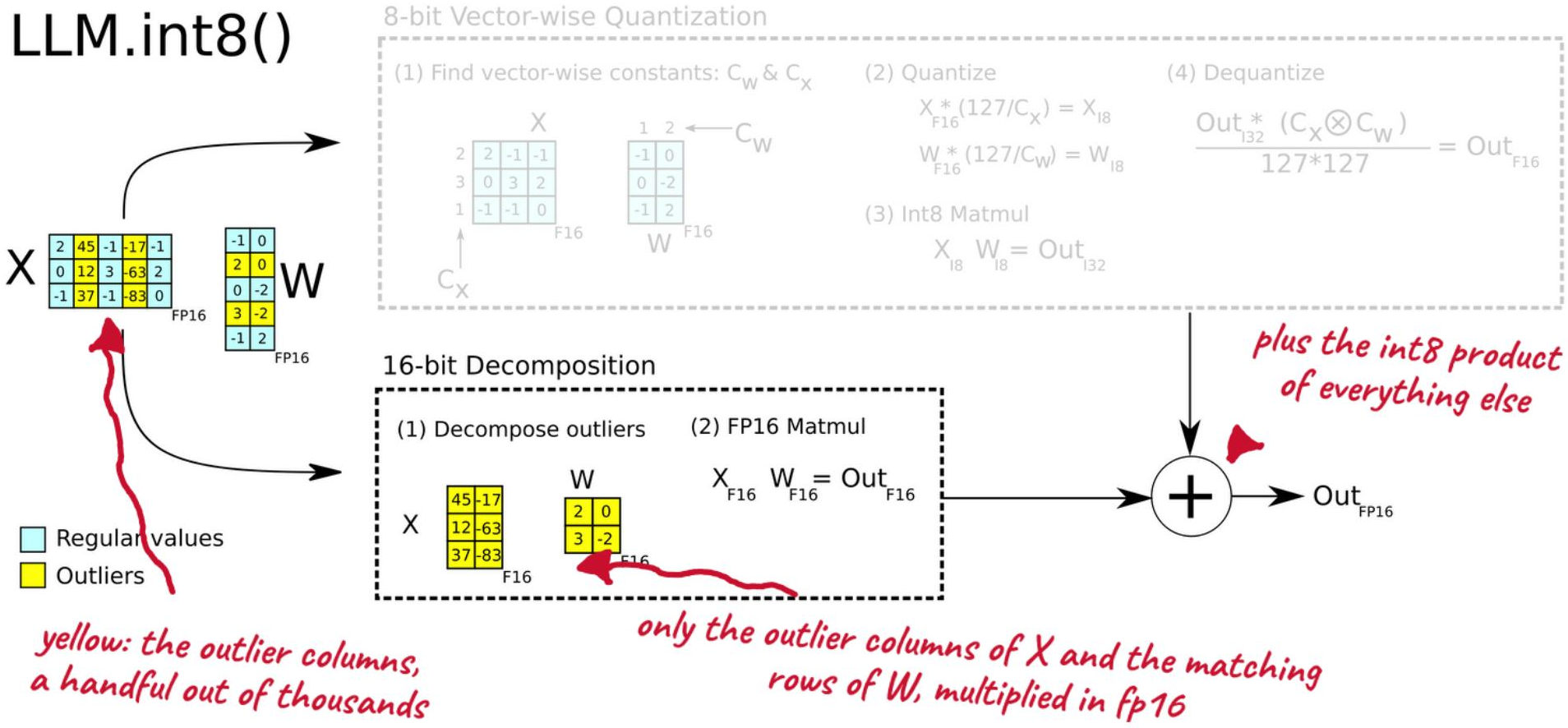
Schematic of the input to one attention layer in Qwen3-1.7B-Base.

Qwen3 Activations



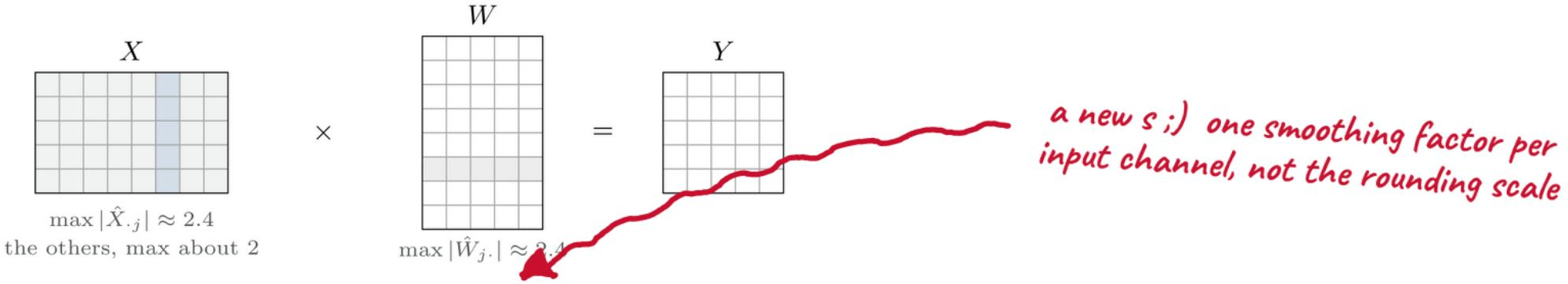
My runs on Qwen3-1.7B-Base in bf16 on my RTX 4090, WikiText-2 test text.

Outlier Features



Dettmers et al., "LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale," 2022, arXiv:2208.07339.

SmoothQuant on One Channel

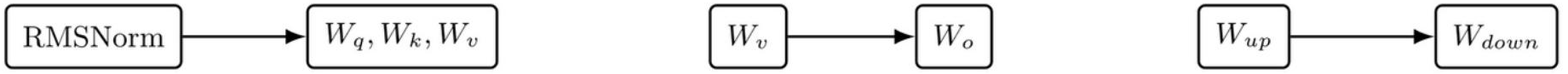


$$Y = XW = \left(X\,\mathrm{diag}(s)^{-1}\right)\left(\mathrm{diag}(s)\,W\right) = \hat{X}\,\hat{W}$$ exact for any $s > 0$, one entry per input channel

$$s_j = \frac{\max_i |X_{ij}|^{\alpha}}{\max_k |W_{jk}|^{1-\alpha}}, \qquad \alpha = 0.5$$

$$s_j = \sqrt{60/0.1} \approx 24, \quad \max|\hat{X}_{\cdot j}| = 60/24 \approx 2.4, \quad \max|\hat{W}_{j\cdot}| = 0.1 \times 24 \approx 2.4$$

the range moves from the activation (quantized per tensor at run time) to the weight (quantized once, offline)



$1/s$ folded into the norm's weight vector

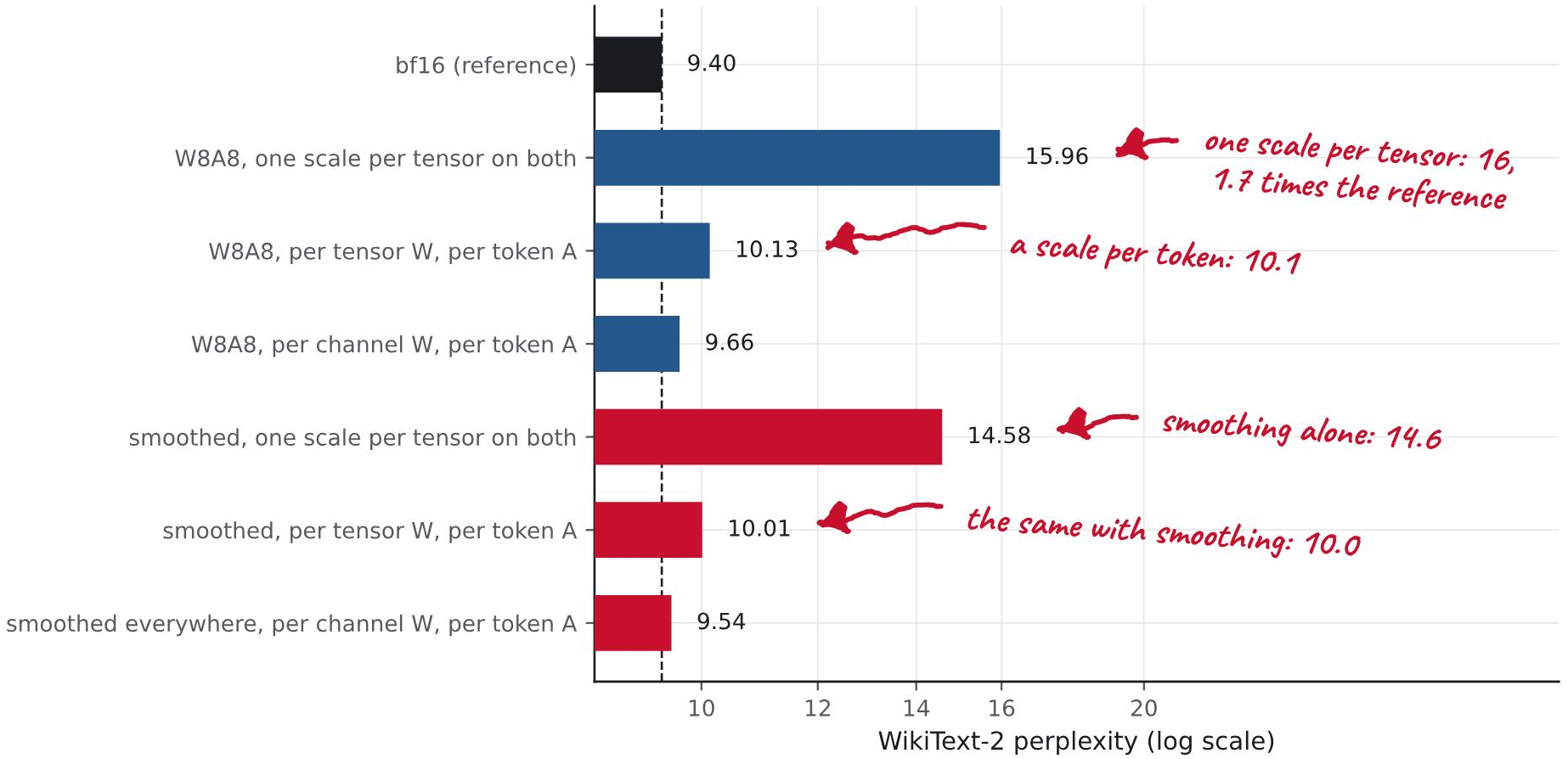
folded into W_v's output channels

folded into W_{up}'s output channels

no extra operation at inference

Xiao et al., "SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models," 2023, arXiv:2211.10438.

SmoothQuant on Qwen3



W8A8: weights (W) and activations (A) both in 8 bits. Perplexity: roughly how many next tokens the model is torn between; lower is better.

My runs on Qwen3-1.7B-Base on my RTX 4090, WikiText-2.

4 WEIGHT-ONLY QUANTIZATION

GPTQ and AWQ

Three Ways to Round

a layer computes $Y = WX$; quantizing replaces W with $\hat{W}$

the goal: $\min_{\hat{W}} \|WX - \hat{W}X\|^2$ X: activations from a few hundred calibration sentences

---

round to nearest

round every weight on its own; X plays no part

GPTQ

Frantar et al., 2022

round one column of W at a time, and spread its rounding error over the columns not yet rounded

$$\delta = -\frac{w_q - \mathrm{quant}(w_q)}{[H^{-1}]_{qq}} (H^{-1})_{:,q}, \qquad H = 2XX^\top$$

δ: the nudge to the columns not yet rounded. H, the Hessian: which inputs move together

AWQ

Lin et al., 2023

find the input channels with the largest activations, about 1 in 100;

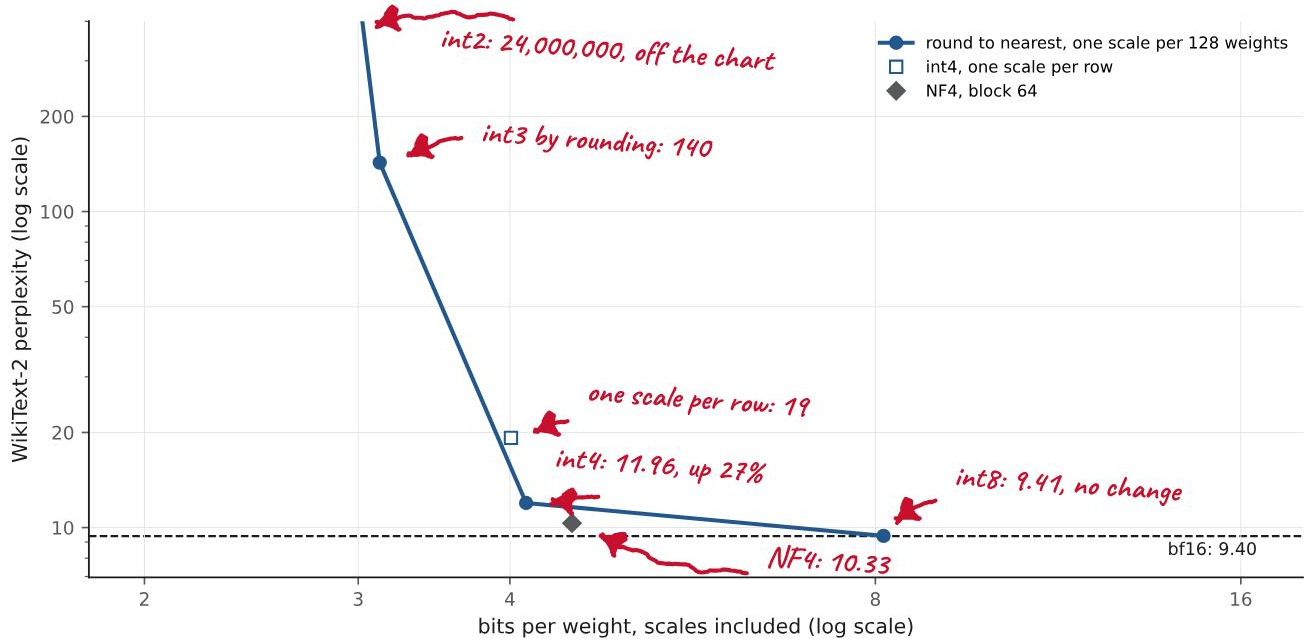
multiply their weights by s before rounding and divide the input by s: $Q(w\,s)\,(x/s)$

the same product, with a smaller rounding error on the weights that matter most

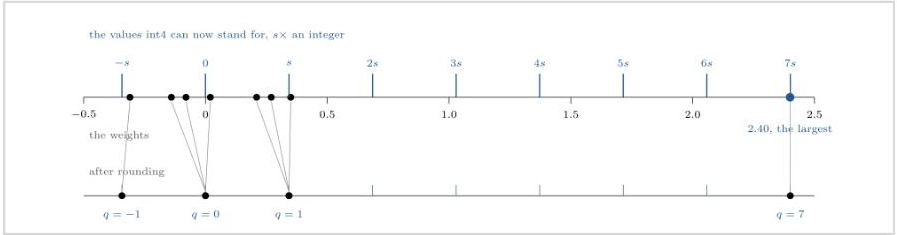
no training and no gradients: each needs only a forward pass over the calibration sentences

Frantar et al., "GPTQ," 2022, arXiv:2210.17323; Lin et al., "AWQ," 2023, arXiv:2306.00978.

Round to Nearest



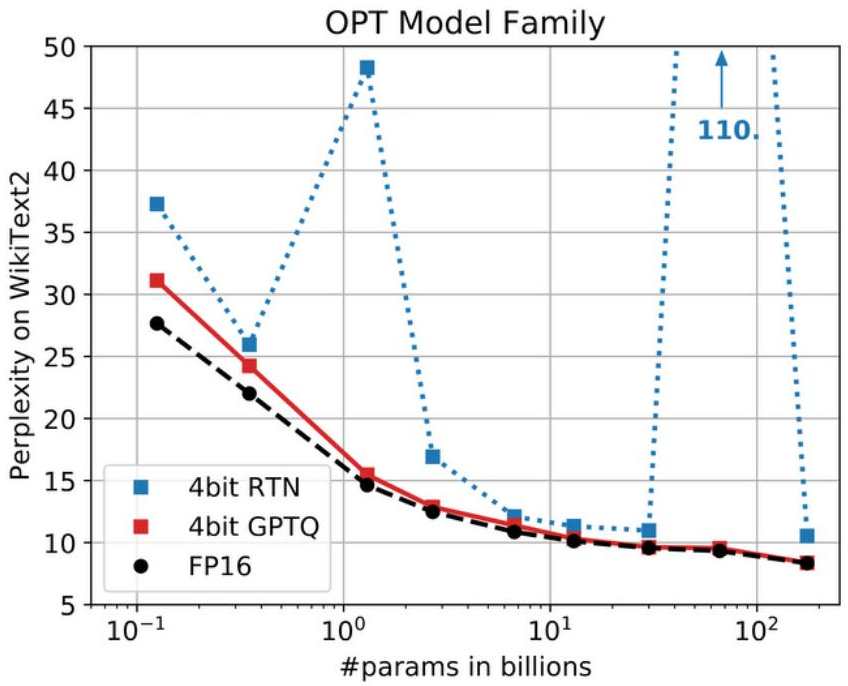
one scale per row is the absmax picture from earlier, across a whole row: one big weight sets the step for all 2048



My runs on Qwen3-1.7B-Base on my RTX 4090, WikiText-2.

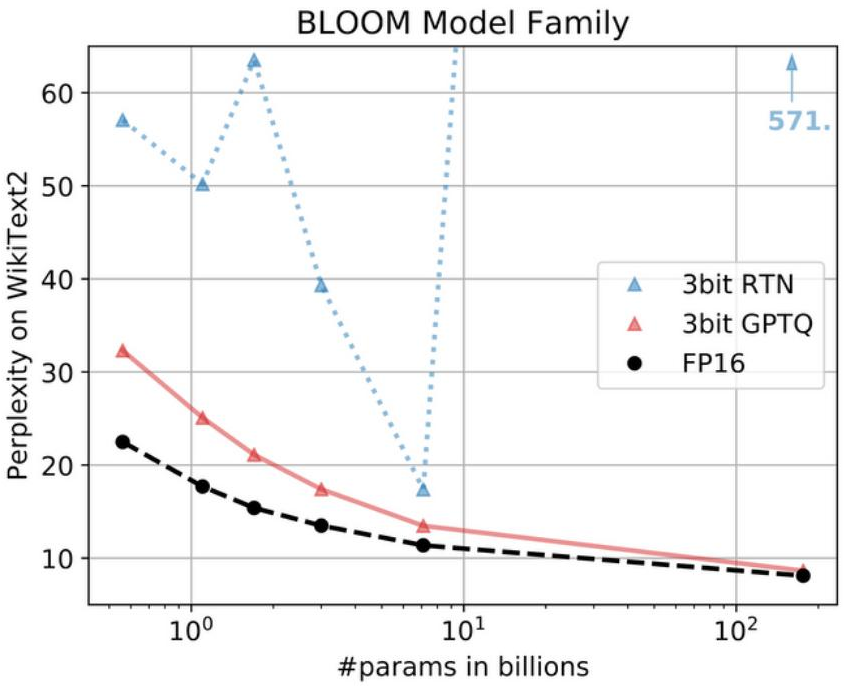
GPTQ

OPT: Meta's open GPT-style models, 125M to 175B, here at 4 bits



dotted blue, round to nearest: the old way, it blows up

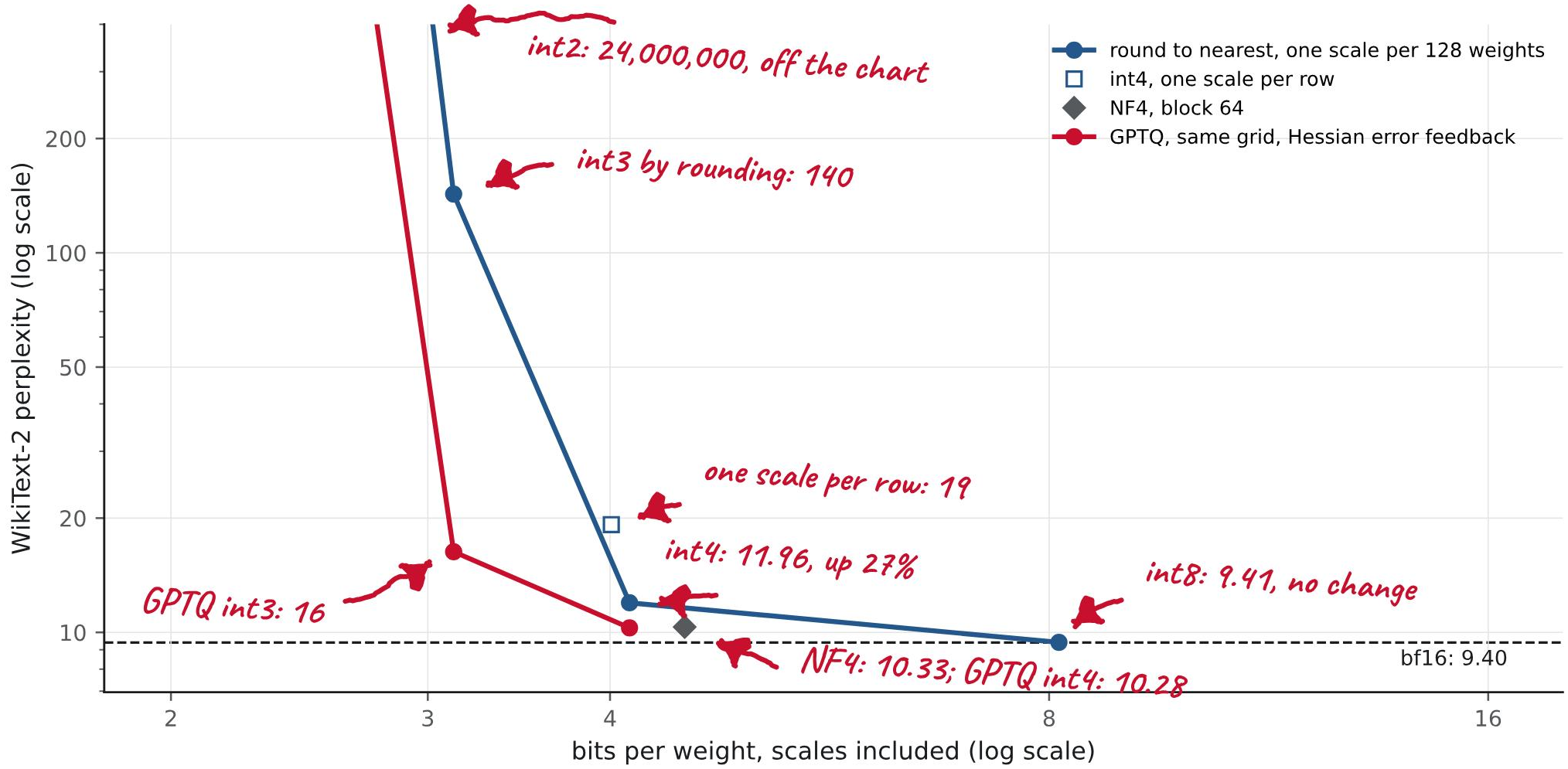
BLOOM: BigScience's open models, 560M to 176B, here at 3 bits



red, GPTQ: the new way, close to fp16 in black

Frantar et al., "GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers," 2022, arXiv:2210.17323.

GPTQ on Qwen3



My runs on Qwen3-1.7B-Base on my RTX 4090, WikiText-2.

Salient Weights

1% chosen by activation: almost back to fp16

chosen by weight or at random: no better than rounding

PPL ↓	FP16	RTN (w3-g128)	FP16% (based on act.)			FP16% (based on W)			FP16% (random)		
			0.1%	1%	3%	0.1%	1%	3%	0.1%	1%	3%
OPT-1.3B	14.62	119.00	25.03	16.91	16.68	108.71	98.55	98.08	119.76	109.38	61.49
OPT-6.7B	10.86	23.54	11.58	11.39	11.36	23.41	22.37	22.45	23.54	24.23	24.22
OPT-13B	10.13	46.04	10.51	10.43	10.42	46.07	48.96	54.49	44.87	42.00	39.71

Lin et al., "AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration," 2023, arXiv:2306.00978.

Scale Then Round

$$Q(w)\,x = \Delta \cdot \mathrm{Round}\left(\tfrac{w}{\Delta}\right) x, \qquad \Delta = \frac{\max|w|}{2^{b-1}}$$

rounding error at most Δ/2, whatever *w*

$$Q(w\,s)\,\frac{x}{s} = \Delta' \cdot \mathrm{Round}\left(\tfrac{ws}{\Delta'}\right) x \cdot \frac{1}{s}$$

same product; this weight's error shrinks by (Δ′/Δ)(1/*s*)

$$s^* = \arg\min_s \left\| Q(W\,\mathrm{diag}(s))\,\mathrm{diag}(s)^{-1} X - WX \right\|$$

$s = s_X^{\alpha}$, s_X the mean activation magnitude per channel; one α per layer from a grid of 20

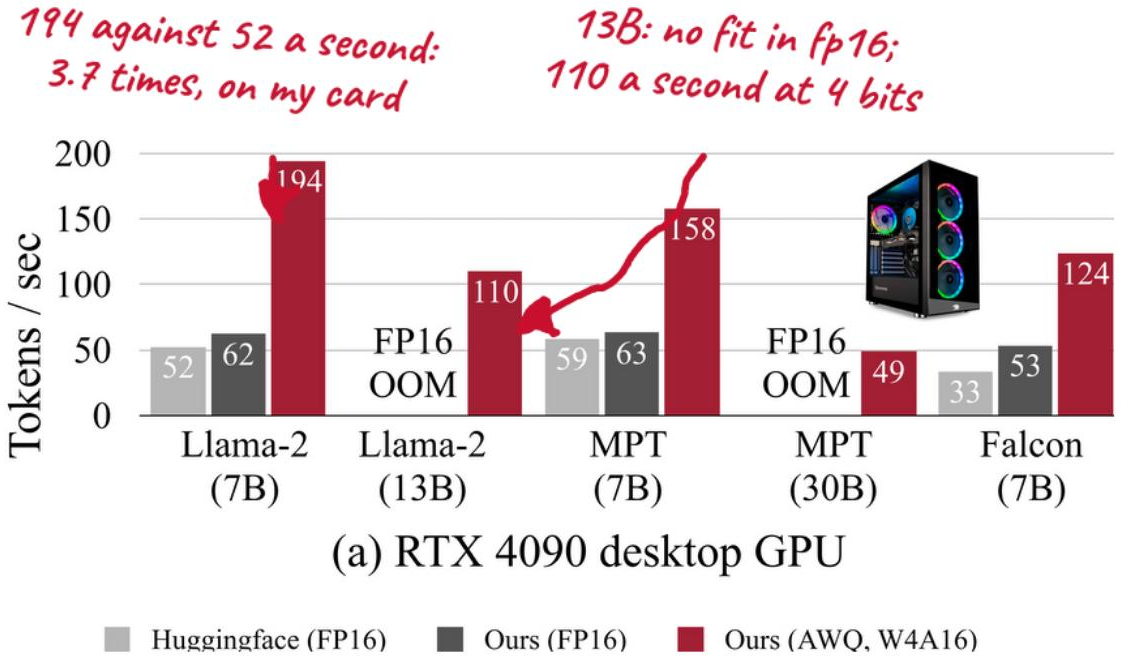
up to s = 2 the step barely grows

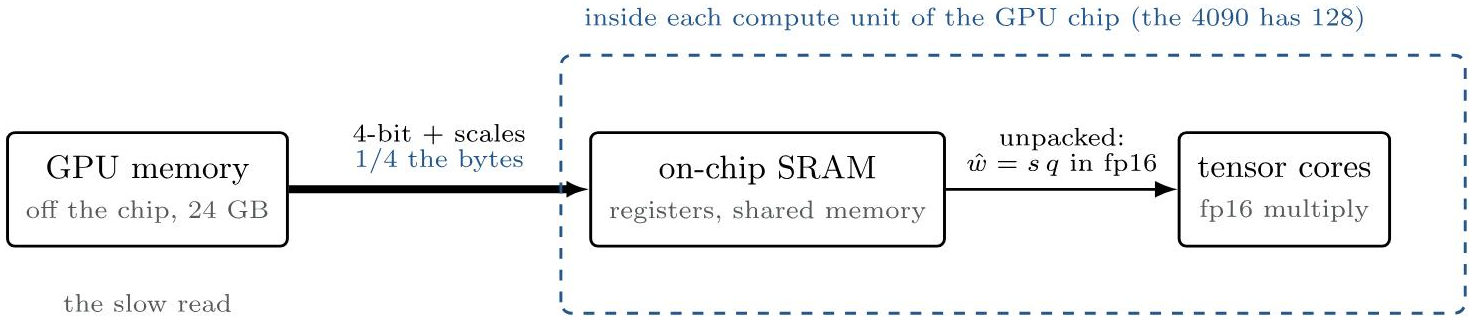
OPT-6.7B	$s=1$	$s=1.25$	$s=1.5$	$s=2$	$s=4$
proportion of $\Delta' \neq \Delta$	0%	2.8%	4.4%	8.2%	21.2%
average Δ'/Δ	1	1.005	1.013	1.038	1.213
average $\frac{\Delta'}{\Delta} \cdot \frac{1}{s}$	1	0.804	0.676	0.519	**0.303**
Wiki-2 PPL	23.54	12.87	12.48	**11.92**	12.36

best at s = 2; push to s = 4 and the whole group gets a coarser step

Lin et al., 2023, arXiv:2306.00978.

Dequantize on Chip



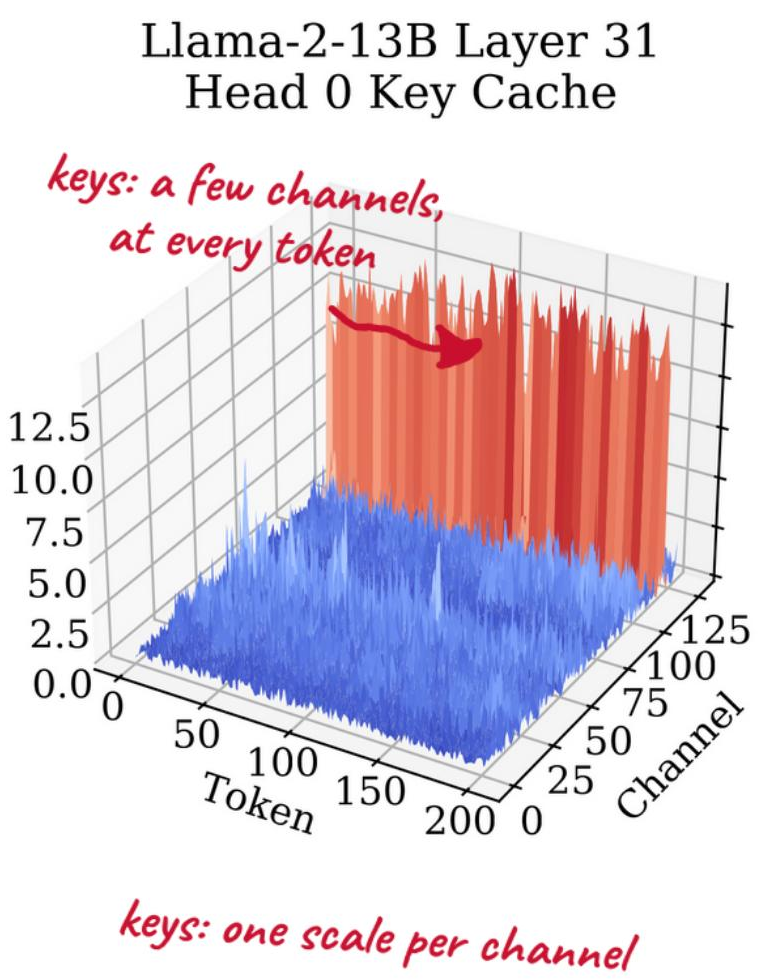


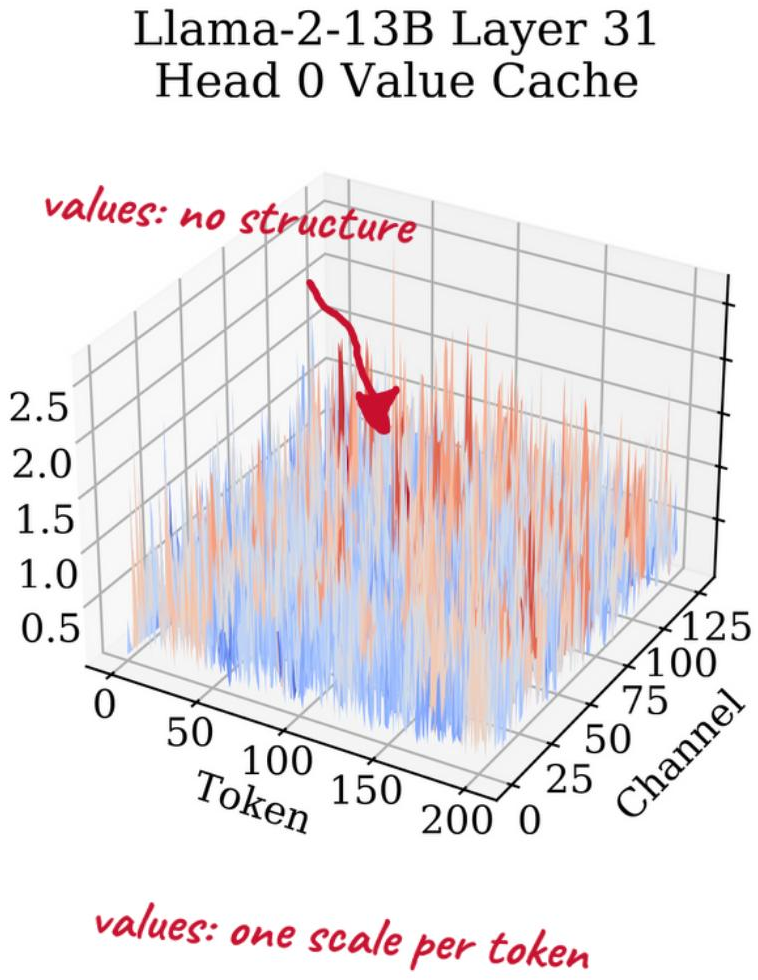
Lin et al., 2023, arXiv:2306.00978.

5 BEYOND THE WEIGHTS

Cache and Gradients

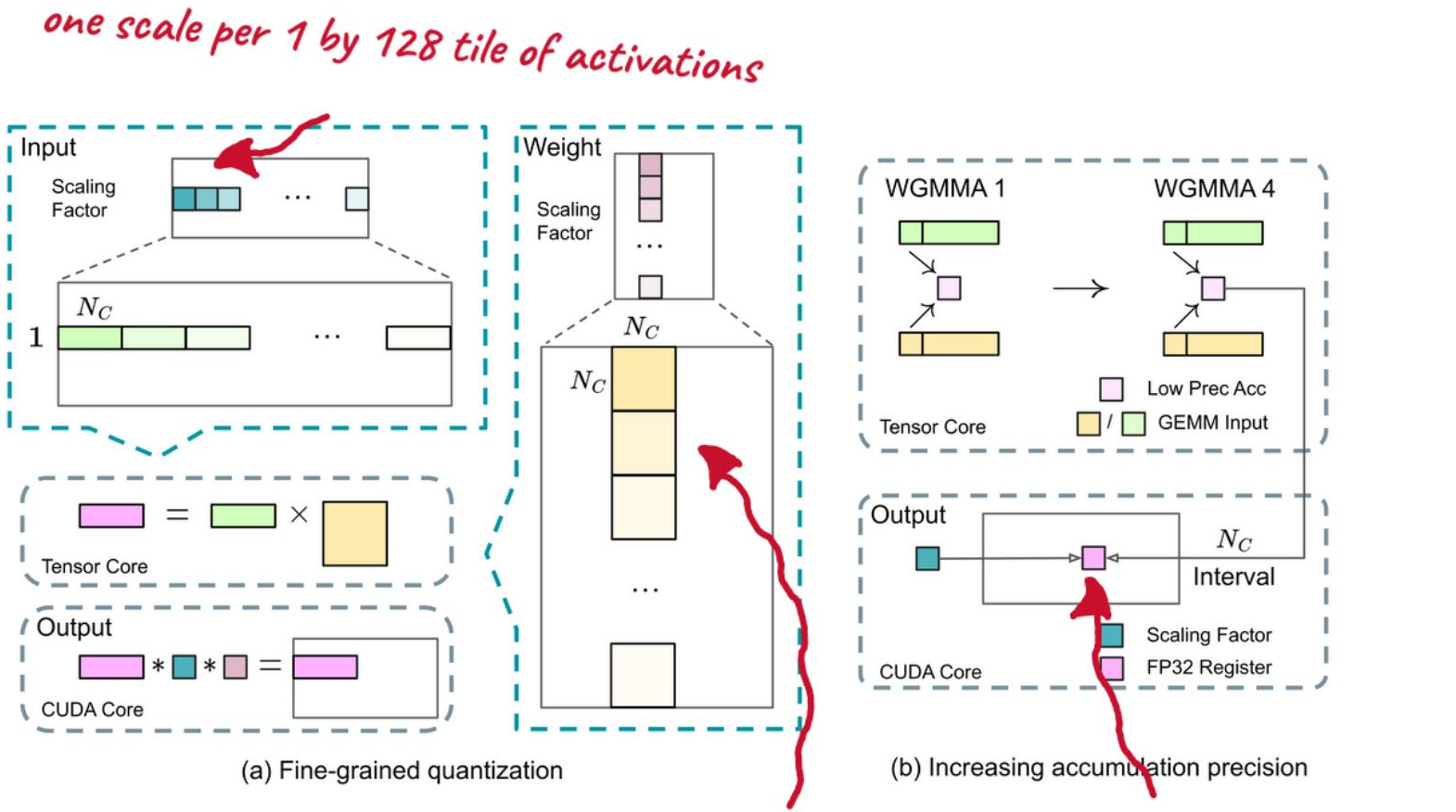
Quantizing the Cache





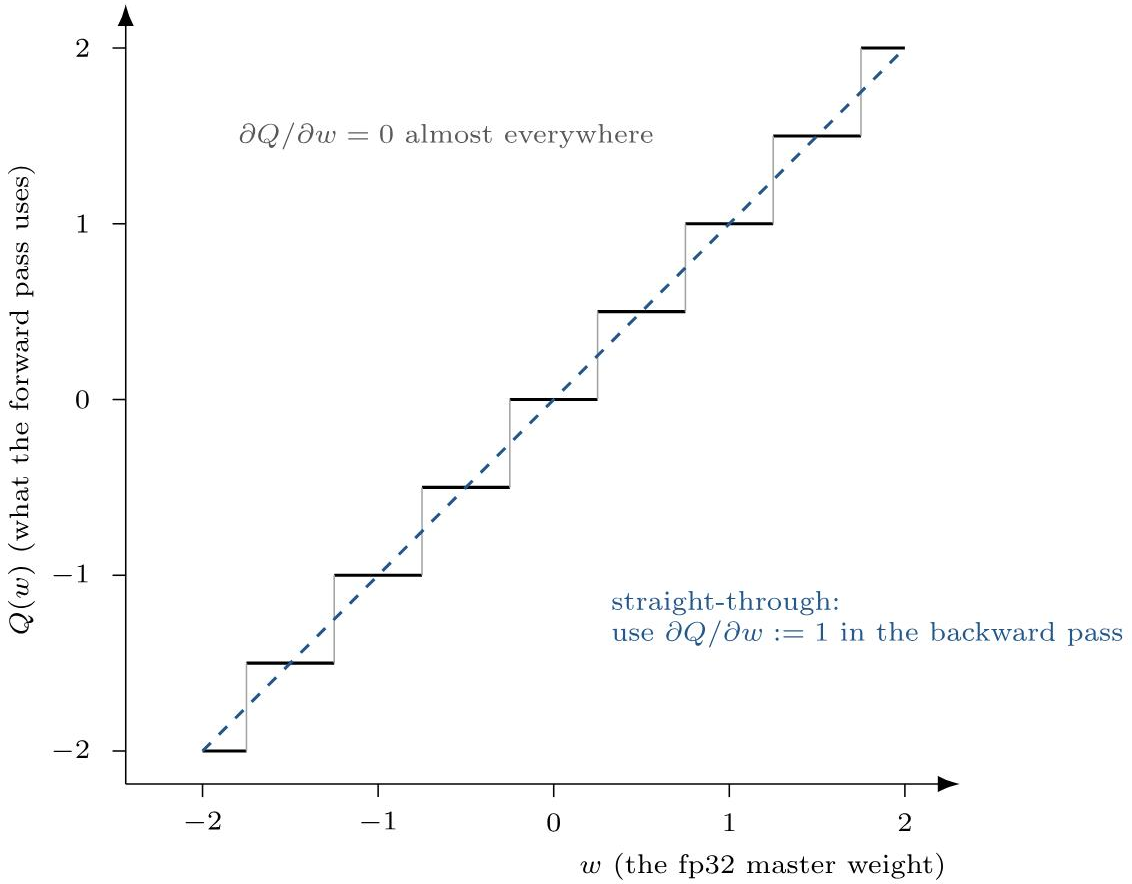
Liu et al., "KIVI: A Tuning-Free Asymmetric 2bit Quantization for KV Cache," 2024, arXiv:2402.02750.

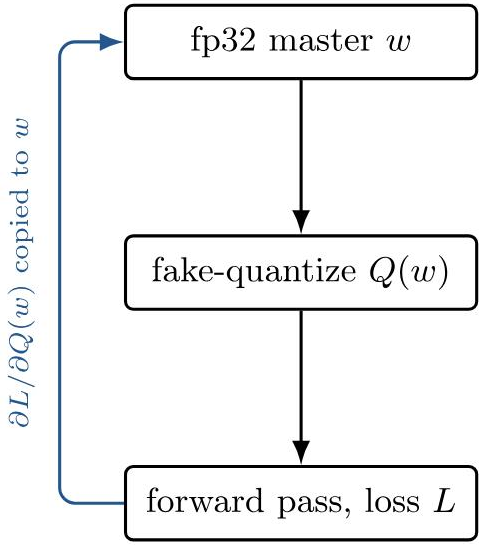
FP8 Training



DeepSeek-AI, "DeepSeek-V3 Technical Report," 2024, arXiv:2412.19437.

Quantization-Aware Training





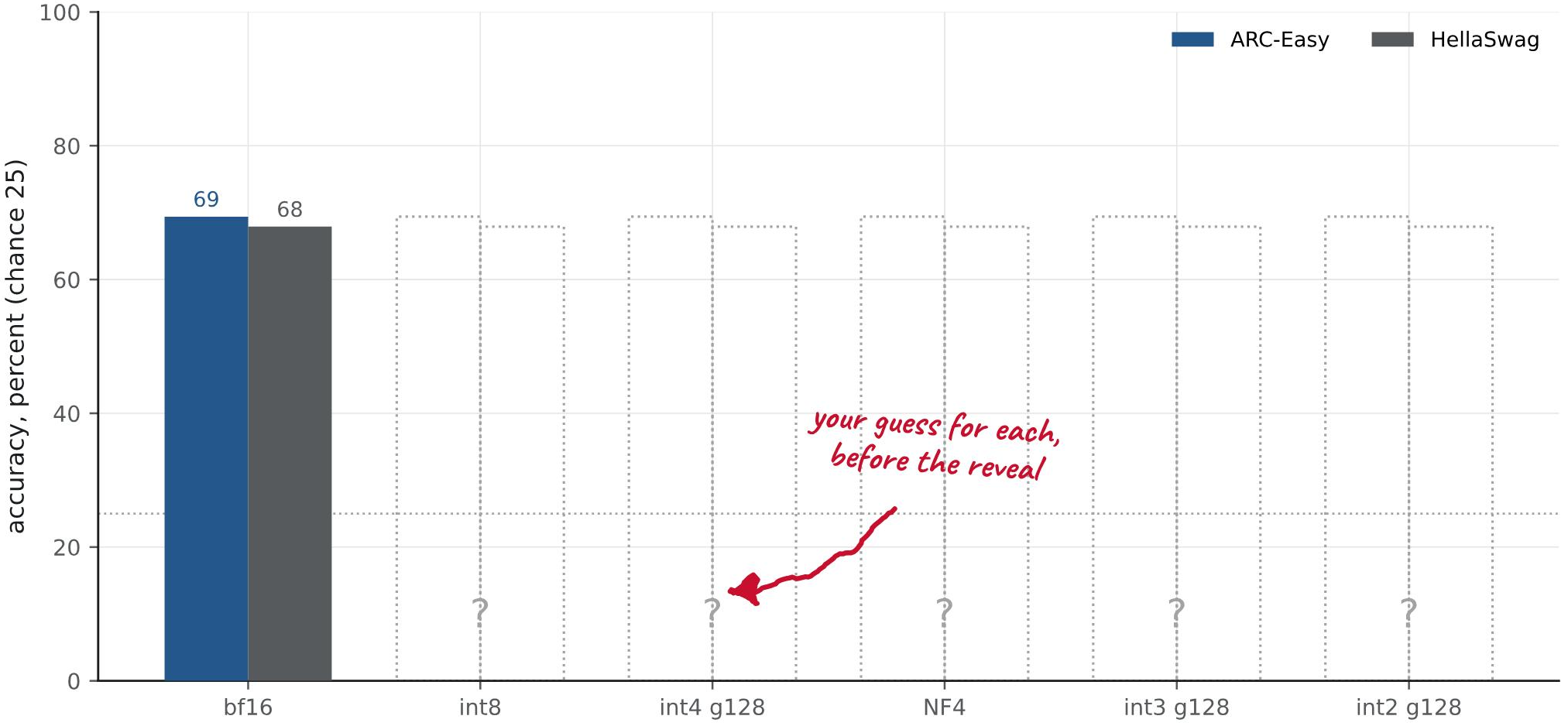
`the forward pass sees the shipped weights; the gradient trains the fp32 copy`

Straight-through estimator after Bengio, Léonard and Courville, "Estimating or Propagating Gradients Through Stochastic Neurons," 2013, arXiv:1308.3432. For language models: Liu et al., "LLM-QAT," 2023, arXiv:2305.17888; Liu et al., "ParetoQ," 2025, arXiv:2502.02631.

6 WHAT QUANTIZATION LOSES

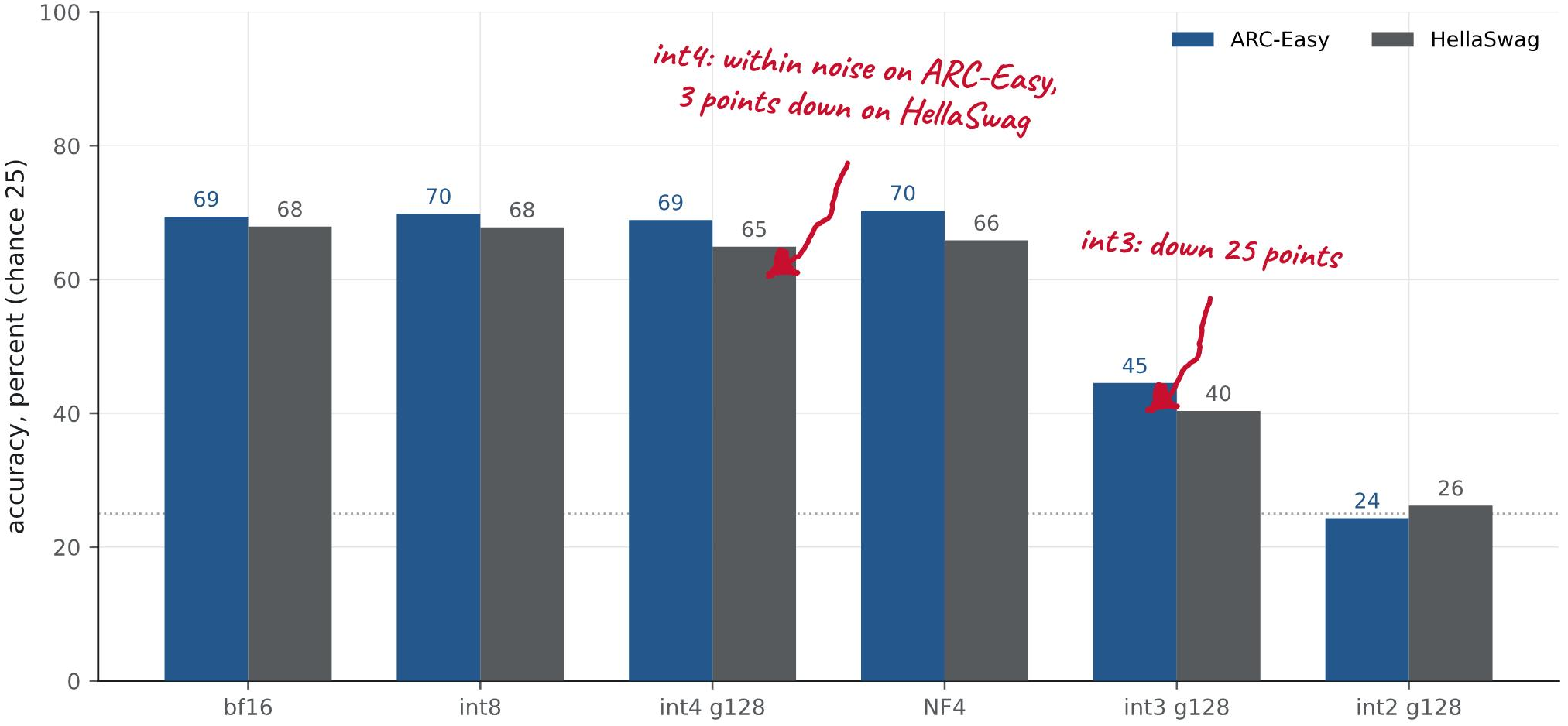
Perplexity and Tasks

Predict the Drop



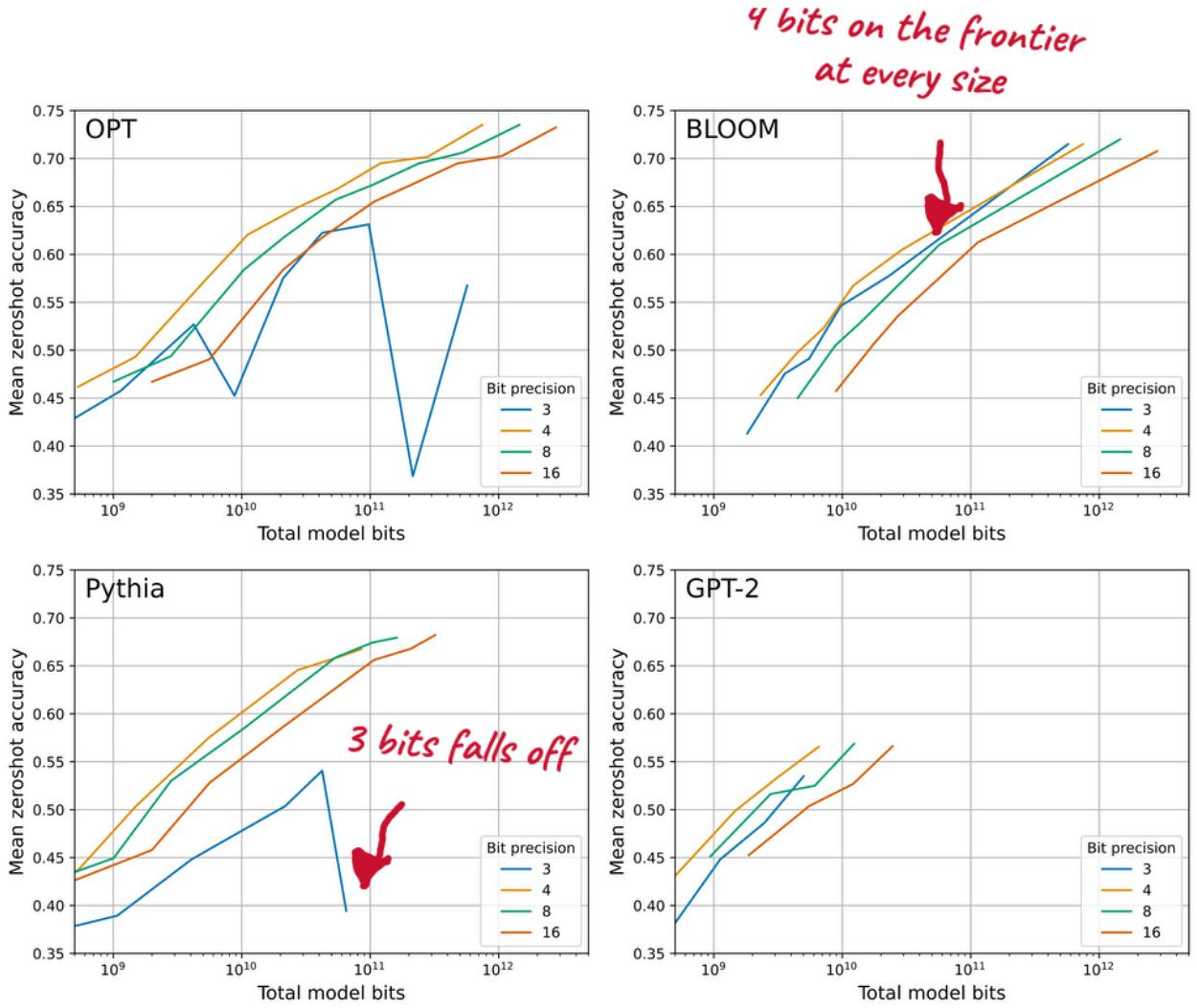
My runs on Qwen3-1.7B-Base on my RTX 4090: ARC-Easy and HellaSwag, zero-shot.

The Measured Drop



My runs on Qwen3-1.7B-Base on my RTX 4090: ARC-Easy and HellaSwag, zero-shot.

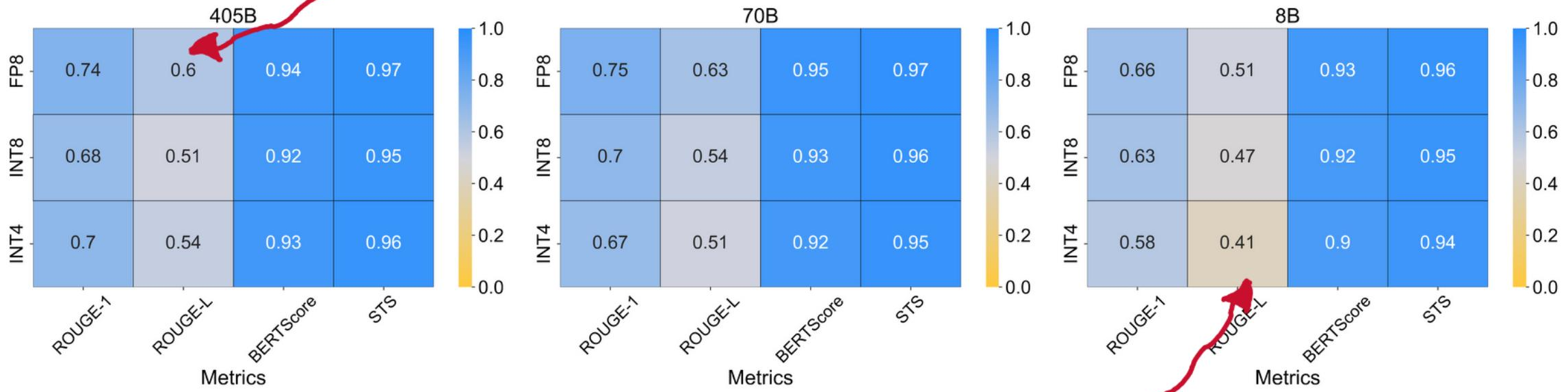
k-bit Scaling



Dettmers and Zettlemoyer, "The case for 4-bit precision: k-bit Inference Scaling Laws," 2023, arXiv:2212.09720.

Accuracy and Output Text

even FP8 at 405B: 0.6



same score, different words:
ROUGE-L 0.41 for the 8B at int4

Kurtic et al., "Give Me BF16 or Give Me Death? Accuracy-Performance Trade-Offs in LLM Quantization," 2024, arXiv:2411.02355.

Reading

Required

Lin et al., AWQ: Activation-aware Weight Quantization · arXiv:2306.00978

Optional

Frantar et al., GPTQ · arXiv:2210.17323 Dettmers et al., LLM.int8() · arXiv:2208.07339

Xiao et al., SmoothQuant · arXiv:2211.10438

DeepSeek-AI, DeepSeek-V3 Technical Report, the FP8 sections · arXiv:2412.19437

Assignment 5 is due October 20 with its quiz in class. The project concept document is due tonight.