SLiM:
One-shot Quantization and Sparsity with Low-rank Approximation for LLM Weight Compression

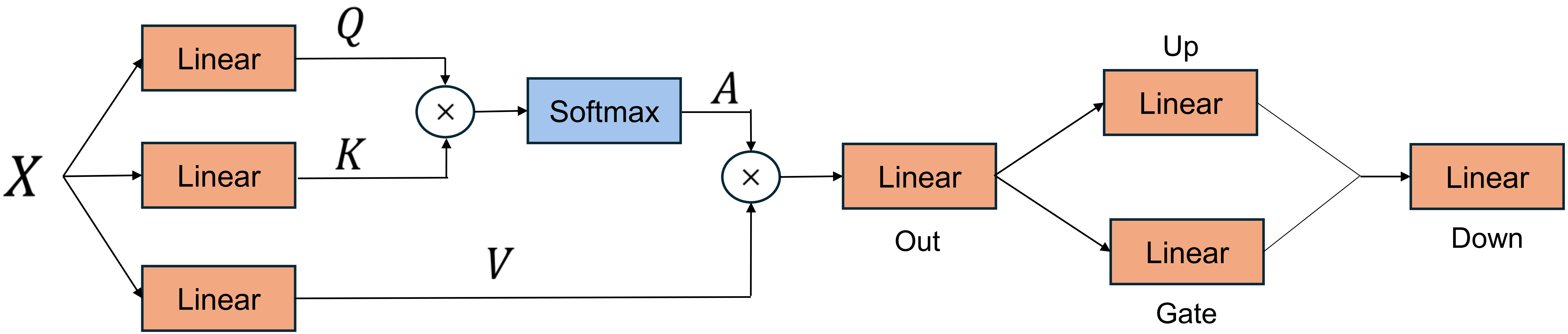
Mohammad Mozaffari[1], Amir Yazdanbakhsh[2], Maryam Mehri Dehnavi[1]

[1] University of Toronto, [2] Google DeepMind

Source Code Available Here!

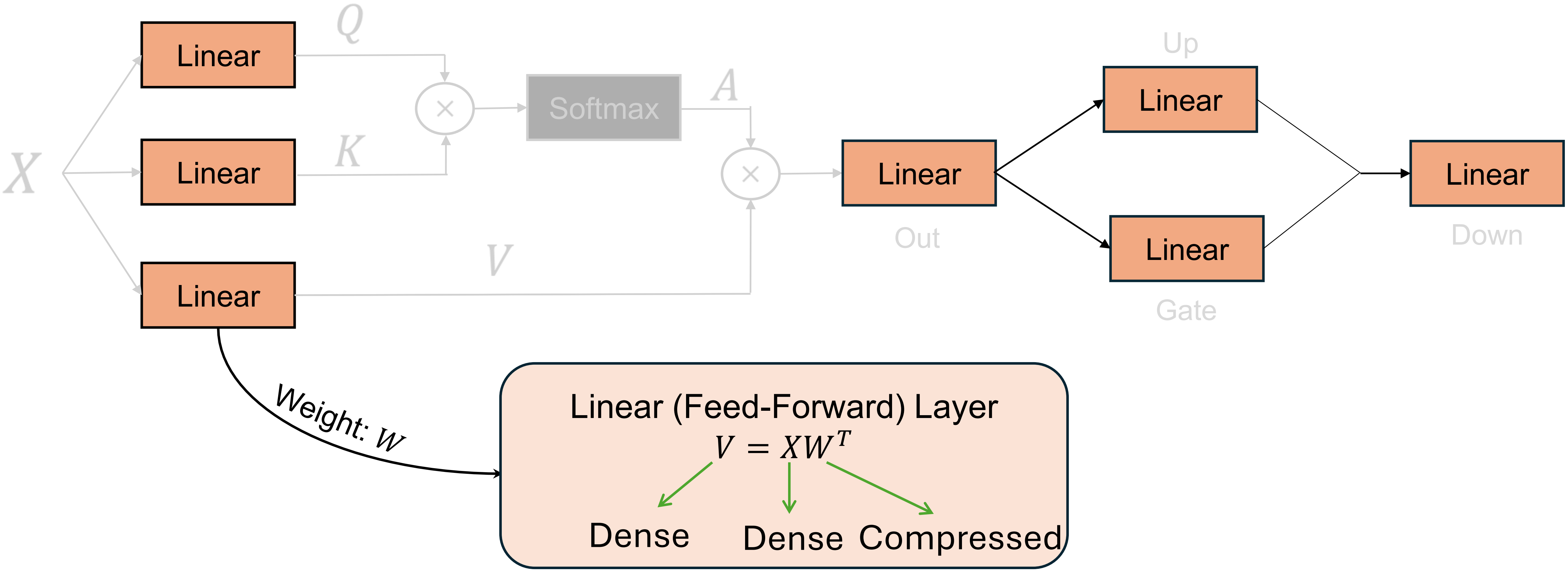
Compression Trinity Webpage

LLM Compute Graph



- Residual connections, layer norms, and other details of the compute graph are not illustrated.

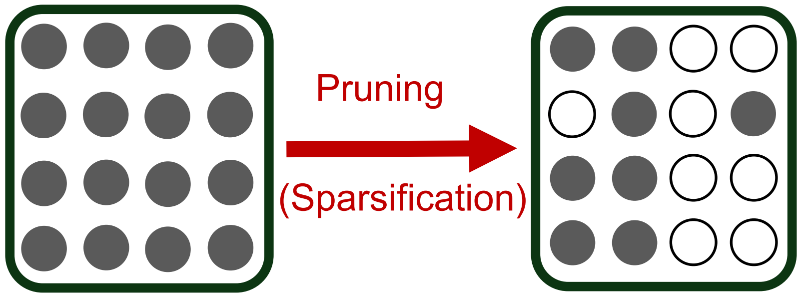
LLM Compute Graph | Weight Sparsity



Post-training Compression Methods

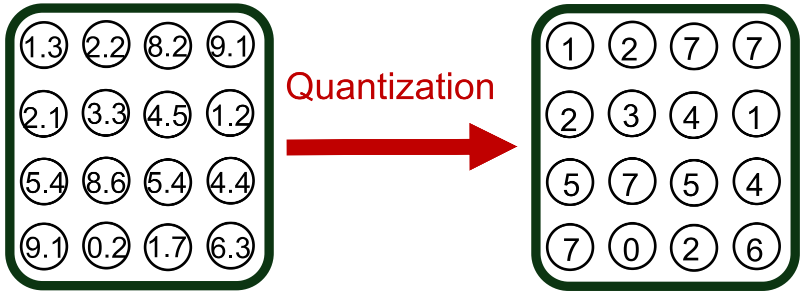
Sparsity

Set non-important weights to zero



Quantization

Reduce the precision of numbers



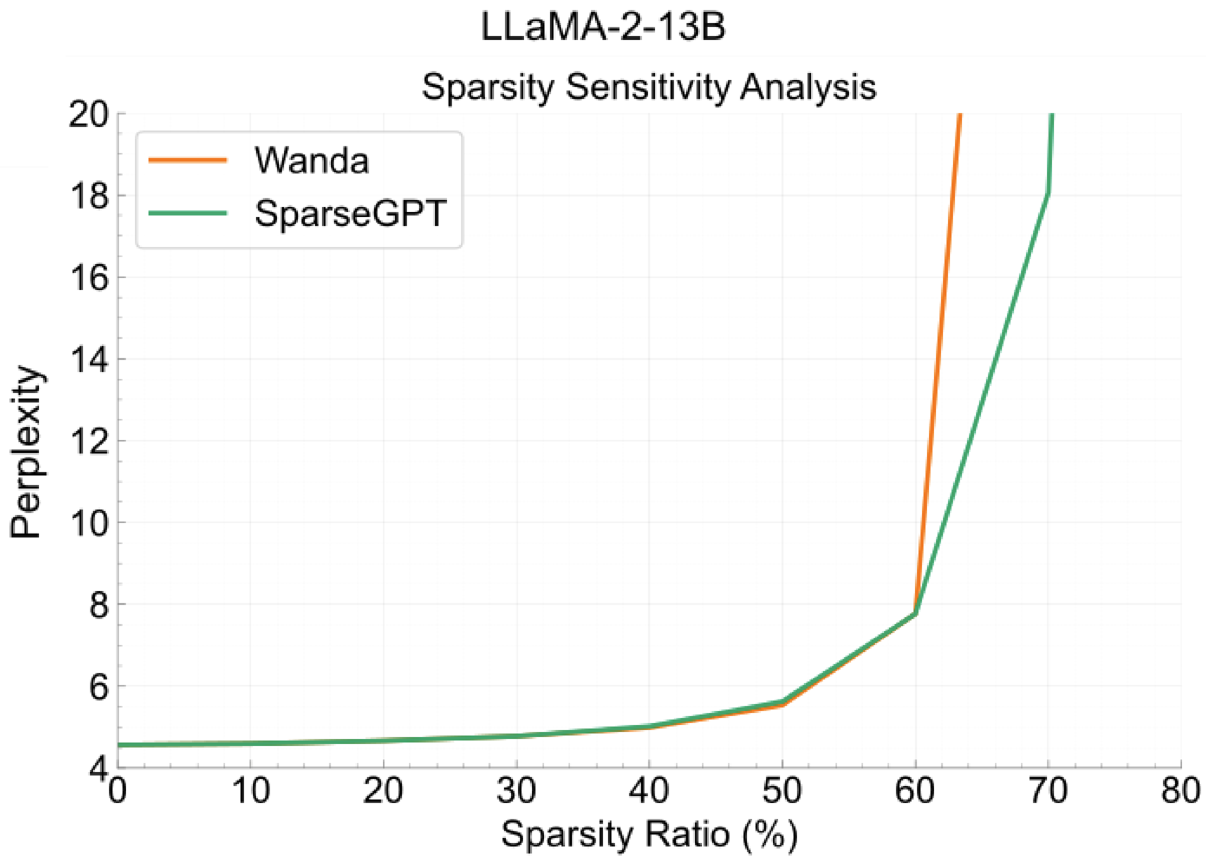
3-bit Quantization:

Round to the closest integer
Clip the data larger than 7

Sparsity Challenges

The perplexity of models become too big below 50% sparsity!

- Maximum 2 × reduction in model size

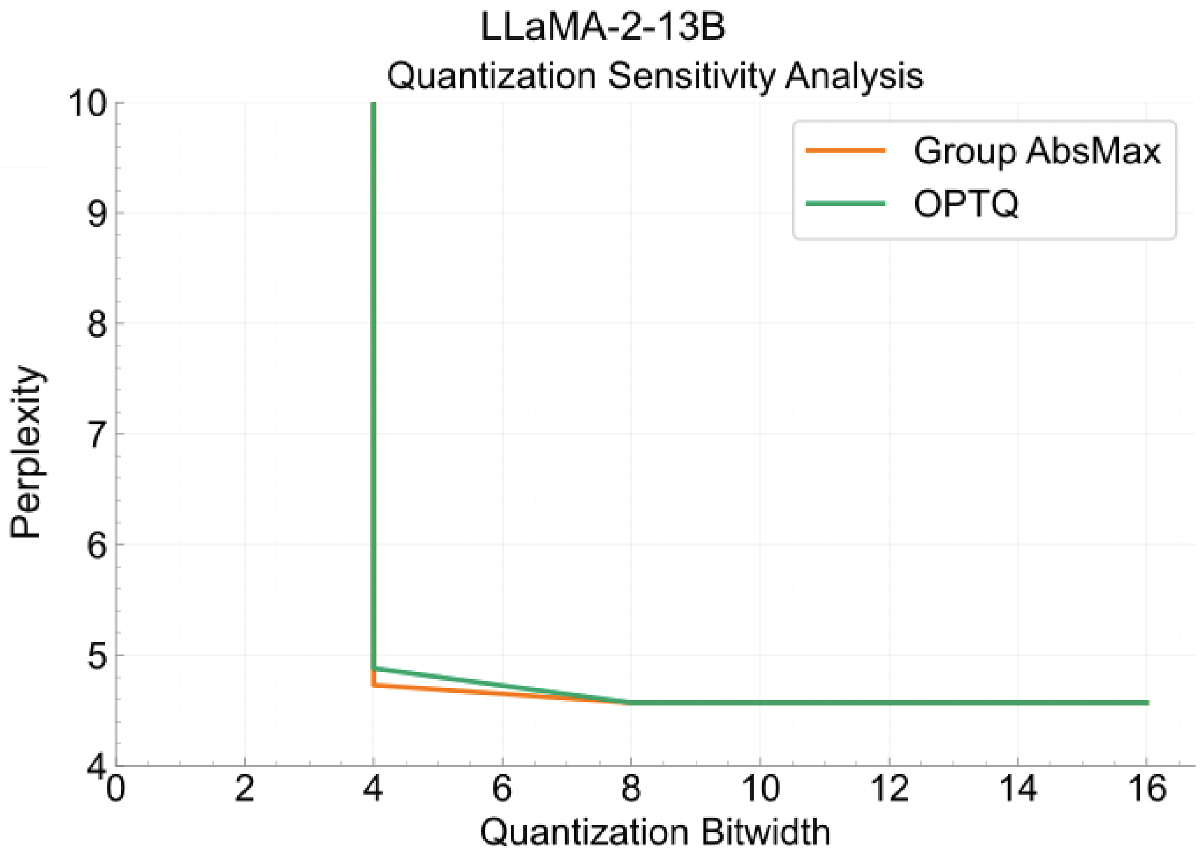


↓ indicates better performance.

Quantization Challenges

The perplexity of models become too big below 4-bit quantization!

- Maximum $4\times$ reduction in model size



↓ indicates better performance.

Higher Compression Ratios

$8 \times$ Compression ratio case study:

Average Accuracy on 6 LM Harness Tasks*

Method	LLaMA-2-7B	LLaMA-2-13B
Dense	56.6%	60.8%
87.5% Sparse**	31.06%	31.59%
2-bit Quantization***	31.81%	31.68%
4-bit Quantization + 50% Unstructured Sparsity	53.62%	57.00%
4-bit Quantization + 2:4 Sparsity	45.49%	51.05%

Combining sparsity and quantization gives better accuracy vs quantization or sparsity alone!

*The tasks include MMLU, PIQA, ARC-Easy, ARC-Challenge, WINOGRANDE, and OpenBookQA
**Best method among Wanda and SparseGPT
***Best method among AbsMax and OPTQ

Higher Compression Ratios

$8 \times$ Compression ratio case study:

Average Accuracy on 6 LM Harness Tasks*

Method	LLaMA-2-7B	LLaMA-2-13B
Dense	56.6%	60.8%
87.5% Sparse**	31.06%	31.59%
2-bit Quantization***	31.81%	31.68%
4-bit Quantization + 50% Unstructured Sparsity	53.62%	57.00%
4-bit Quantization + 2:4 Sparsity	45.49%	51.05%

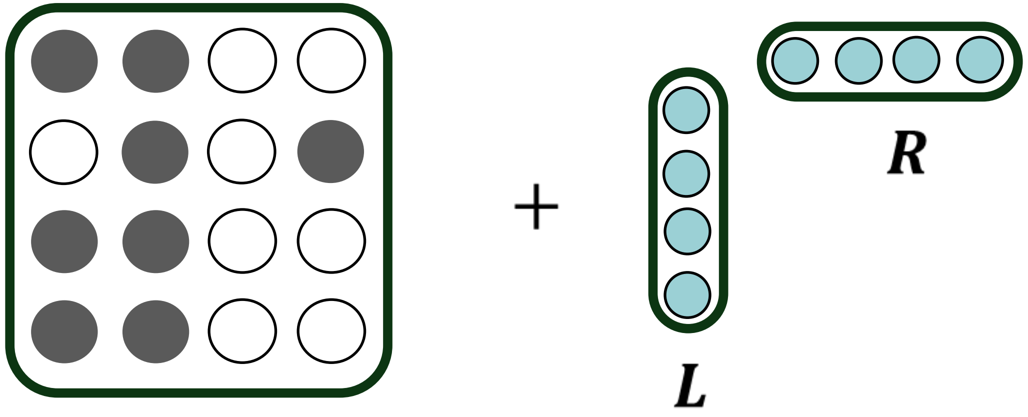
However, the accuracy gap between compressed and dense models is large

*The tasks include MMLU, PIQA, ARC-Easy, ARC-Challenge, WINOGRANDE, and OpenBookQA
**Best method among Wanda and SparseGPT
***Best method among AbsMax and OPTQ

Accuracy Recovery with Low-rank Adapters

Low-rank adapters can help recover the accuracy of the models[1,2]

- Challenge: They require millions of tokens to train
- Solution: One-shot Low-rank Adapters compute L and R mathematically (no training needed)

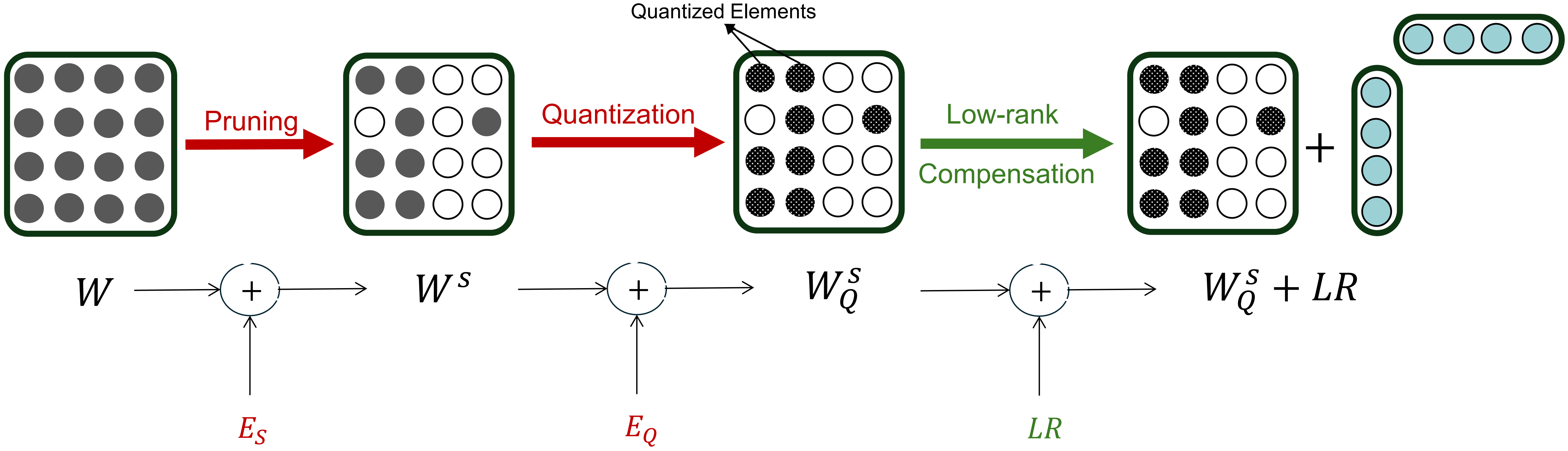


$$W = W^C + LR$$

[1] Dettmers et al., QLoRA: Efficient Finetuning of Quantized LLMs (NeurIPS23)
[2] Nikdan et al., RoSA: Accurate Parameter-Efficient Fine-Tuning via Robust Adaptation (ICML24)

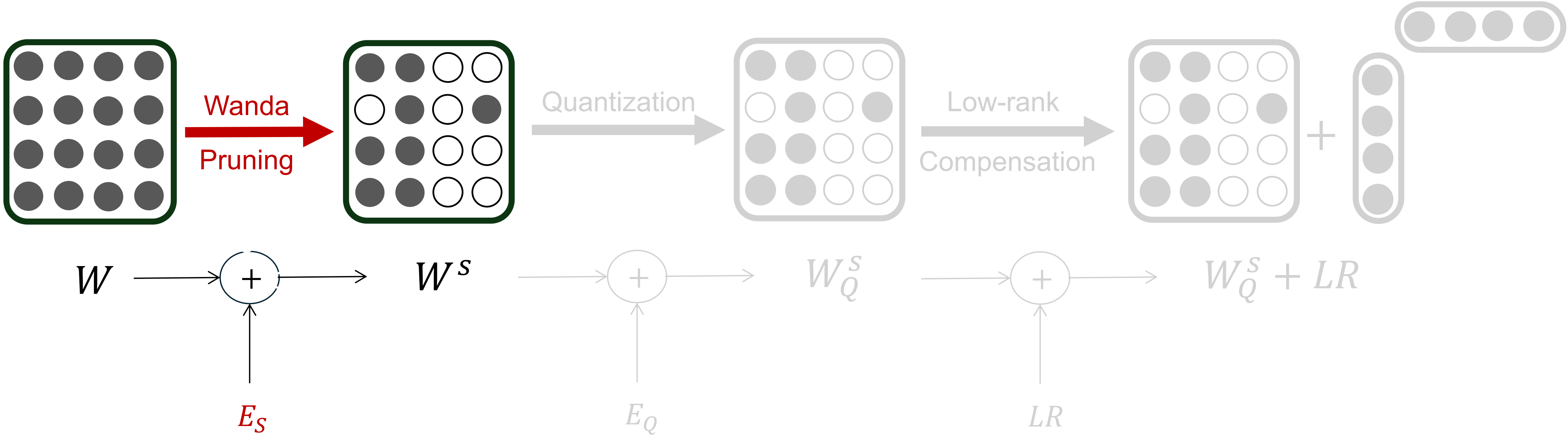
SLiM | Overview

E_S: Sparsity Error
E_Q: Quantization Error
L, R: Low-rank Adapters
W^S: Sparse Weight
W_Q^S: Sparse and Quantized Weight



SLiM | One-shot Pruning Method

E_S: Sparsity Error
E_Q: Quantization Error
L, R: Low-rank Adapters
W^S: Sparse Weight
W_Q^S: Sparse and Quantized Weight

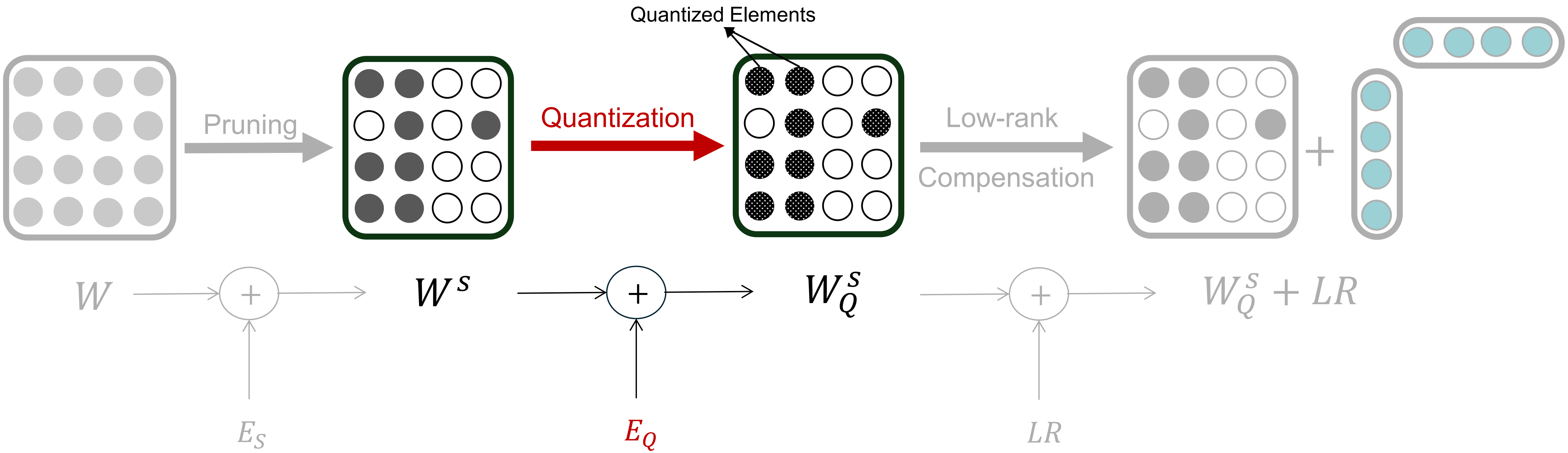


SLiM uses an off-the-shelf method (Wanda[1]) for one-shot pruning.

[1] Sun et al., A Simple and Effective Pruning Approach for Large Language Models

SLiM | Quantization

E_S: Sparsity Error
E_Q: Quantization Error
L, R: Low-rank Adapters
W^S: Sparse Weight
W_Q^S: Sparse and Quantized Weight



SLiM finds a tractable solution for minimizing the quantization error using novel a probabilistic approach.

Uniform Quantization

q: Quantization Bitwidth
Q: Quantization Function
$f(x)$: Weight PDF

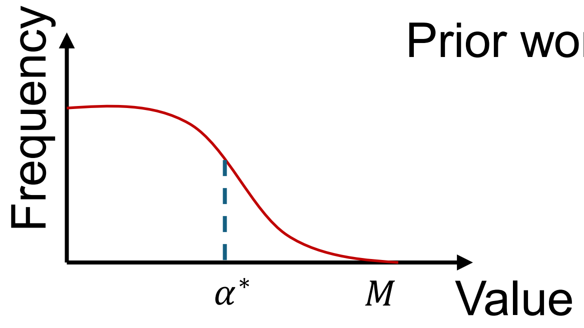
Uniform quantization uses a single parameter per tensor to quantize the weight.

- The values larger than α^* get clipped: $W_Q = clip(\frac{W}{\alpha^*}, \pm 1) \times 2^{q-1}$
- Tuning Parameter α^* $\rightarrow$ Minimize the MSE of the quantization.

$$\alpha^* = \arg\min_{\alpha} \left| W - W_Q \right|^2$$

Non-convex NP-Hard Problem!

Prior work[1] approximately solves it through exhaustive search.



[1] Zhao et al., Atom: Low-bit Quantization for Efficient and Accurate LLM Serving

Uniform Quantization | SLiM-Quant

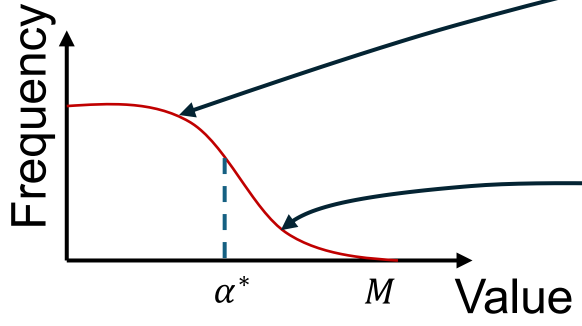
q: Quantization Bitwidth
Q: Quantization Function
$f(x)$: Weight PDF

SLiM-Quant uses a probabilistic approach to formulate the objective function in uniform quantization

$$\alpha^* = \arg\min_{\alpha} \left| W - W_Q \right|^2 \quad \Rightarrow \quad \alpha^* = \arg\min \int_0^{\alpha} Q(x) f(x) dx + \int_{\alpha}^{M} |x - \alpha|^2 f(x) dx$$

Rounding Error

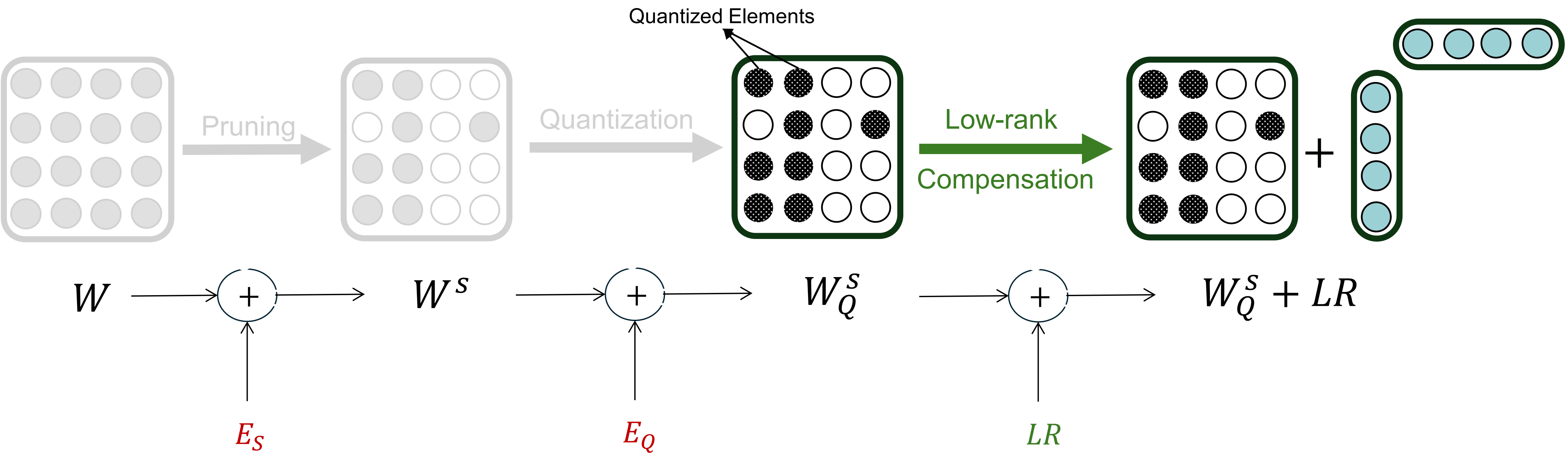
Clipping Error



Low-rank Adapters

E_S: Sparsity Error
E_Q: Quantization Error
L, R: Low-rank Adapters
W^S: Sparse Weight
W_Q^S: Sparse and Quantized Weight

Goal: Reduce the error added due to pruning and quantization.



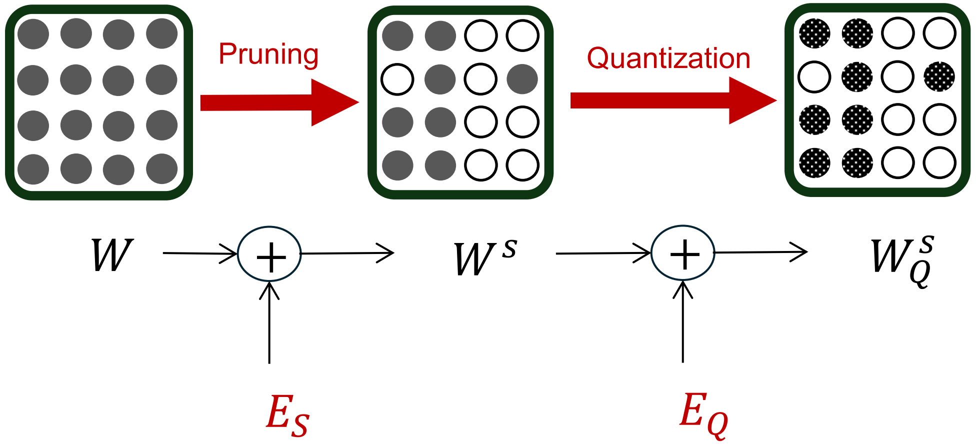
Low-rank Adapters | Naïve-LoRA

E_S: Pruning Error
E_Q: Quantization Error
F: Saliency Function

Error Norm Minimization

$$L^*, R^* = \arg\min \left| W - \left(W_Q^S + LR \right) \right|$$
$$L^*, R^* = \arg\min \left| E_S + E_Q - LR \right|$$

$$L^*, R^* = SVD\left(E_S + E_Q \right)$$



Error norm does not take the importance (saliency) of the weights into account.

Low-rank Adapters | SLiM-LoRA

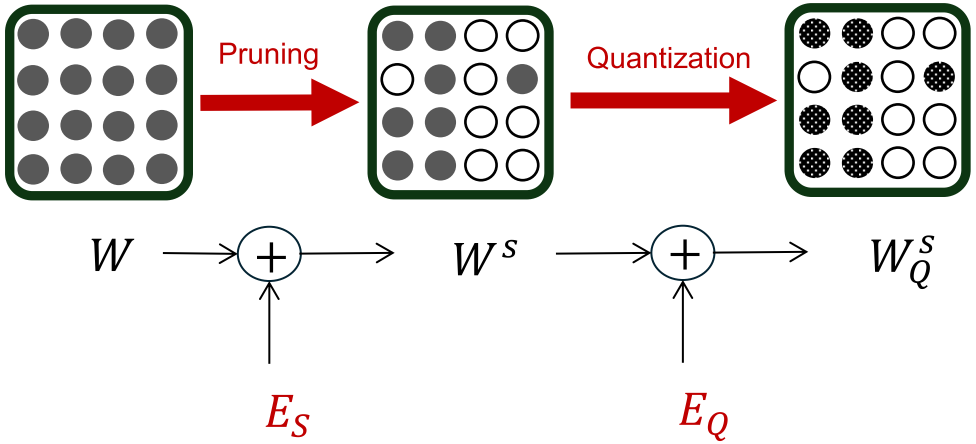
E_S: Pruning Error
E_Q: Quantization Error
F: Saliency Function
$\bar{x}$: Average Calibration Input

Error Saliency Minimization

$$L^*, R^* = \arg\min \left| F\left(W - \left(W_Q^S + LR\right)\right) \right|$$

$$\underbrace{L^*, R^* = \arg\min \left| F\left(E_S + E_Q - LR\right) \right|}$$

Minimizing the saliency of the reconstruction error!



Low-rank Adapters | SLiM-LoRA

E_S: Pruning Error
E_Q: Quantization Error
F: Saliency Function
$\bar{x}$: Average Calibration Input

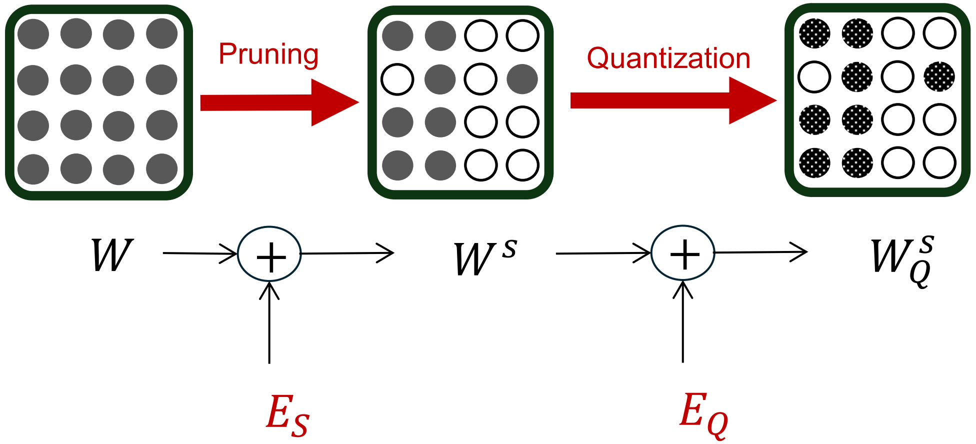
Error Saliency Minimization

$$L^*, R^* = \arg\min \left| F\left(W - \left(W_Q^S + LR\right)\right) \right|$$

$$\underbrace{L^*, R^* = \arg\min \left| F\left(E_S + E_Q - LR\right) \right|}_{\text{Minimizing the saliency of the reconstruction error!}}$$

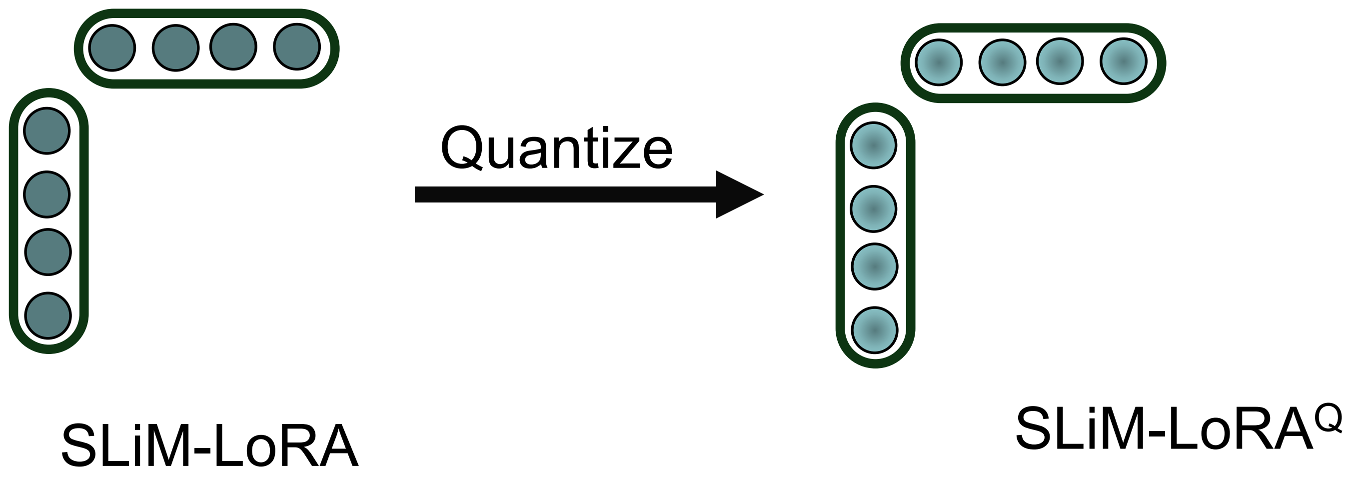
Saliency Function : $F(M) = diag(\bar{x})M$

$$L^*, R^* = diag\left(\frac{1}{\bar{x}}\right)\left(SVD\left(diag(\bar{x})(E_S + E_Q\right)\right)$$



Low-rank Adapters | Adapter Quantization

The low-rank adapters in SLiM are further **quantized to 4-bits**!



SLiM | Zero-shot Accuracy Results

Average Accuracy over 6 Zero-shot tasks

(up to) **5.7%**

over SOTA

2:4 Sparsity with 4-bit Weight Quantization

Method	OPT						LLaMA 2	
	125M	350M	1.3B	2.7B	6.7B	13B	7B	13B
SOTA*	33.70	33.38	38.75	40.15	44.32	45.64	45.49	51.05
Naïve-LoRA	34.28	33.38	38.36	41.21	44.91	45.25	48.45	51.94
SLiM-LoRA	**34.62**	**34.36**	**40.61**	**42.73**	**45.99**	**46.24**	**51.15**	**54.94**

Unstructured Sparsity with 4-bit Weight Quantization

Method	OPT						LLaMA 2	
	125M	350M	1.3B	2.7B	6.7B	13B	7B	13B
SOTA*	35.11	35.16	41.02	43.43	46.97	47.38	53.62	57.00
Naïve-LoRA	34.77	34.23	40.40	43.37	46.64	47.30	51.52	55.33
SLiM-LoRA	**35.20**	**35.32**	**41.85**	**43.63**	**47.16**	**47.96**	**54.26**	**57.85**

*SOTA refers to the best accuracy among SparseGPT and Wanda for pruning and OPTQ, AWQ, AbsMax, OmniQuant, and AffineQuant for quantization.

SLiM | Optional LoRA Fine-tuning

(up to) **1.7%**

Additional Improvement

Average Accuracy over 6 Zero-shot tasks

2:4 Sparsity with 4-bit Weight Quantization

Method	Fine-Tune	OPT						LLaMA 2	
		125M	350M	1.3B	2.7B	6.7B	13B	7B	13B
SLiM-LoRA	❌	34.62	34.36	40.61	42.73	45.99	46.24	51.15	54.94
SLiM-LoRA	✅	**35.03**	**34.58**	**41.11**	**43.35**	**46.71**	**47.25**	**52.12**	**56.60**

2:4 Sparsity with 4-bit Weight Quantization

Method	Fine-Tune	OPT						LLaMA 2	
		125M	350M	1.3B	2.7B	6.7B	13B	7B	13B
SLiM-LoRA	❌	35.20	35.32	41.85	43.63	47.16	47.96	54.26	57.85
SLiM-LoRA	✅	**35.59**	**35.71**	**42.37**	**44.58**	**47.69**	**48.26**	**54.69**	**57.96**

Only 300,000 tokens are used for finetuning!

SLiM | Speedup and Memory Reduction

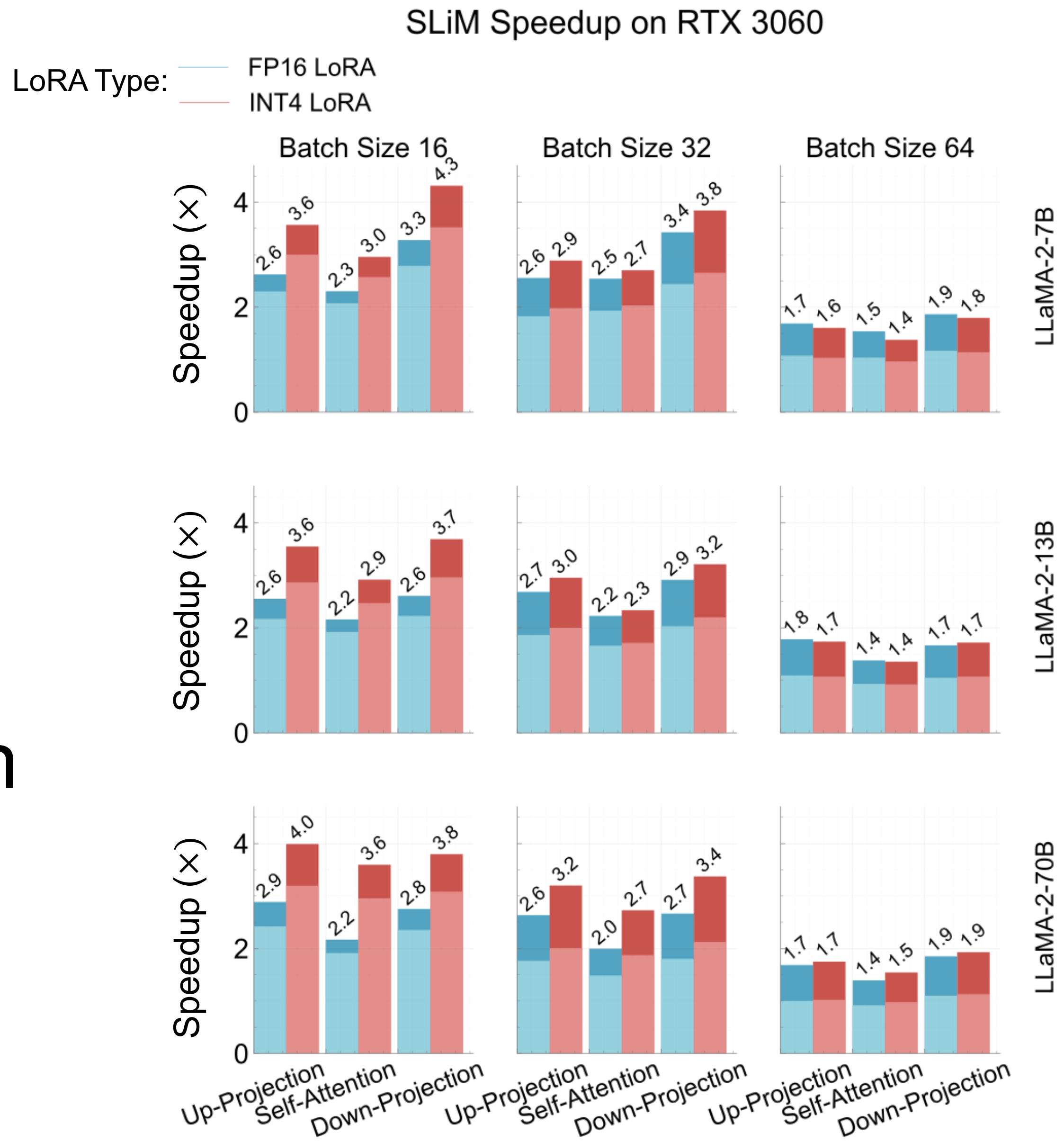
Speedup

(A100 GPU) **3.8×**

(RTX3060GPU) **4.3×**

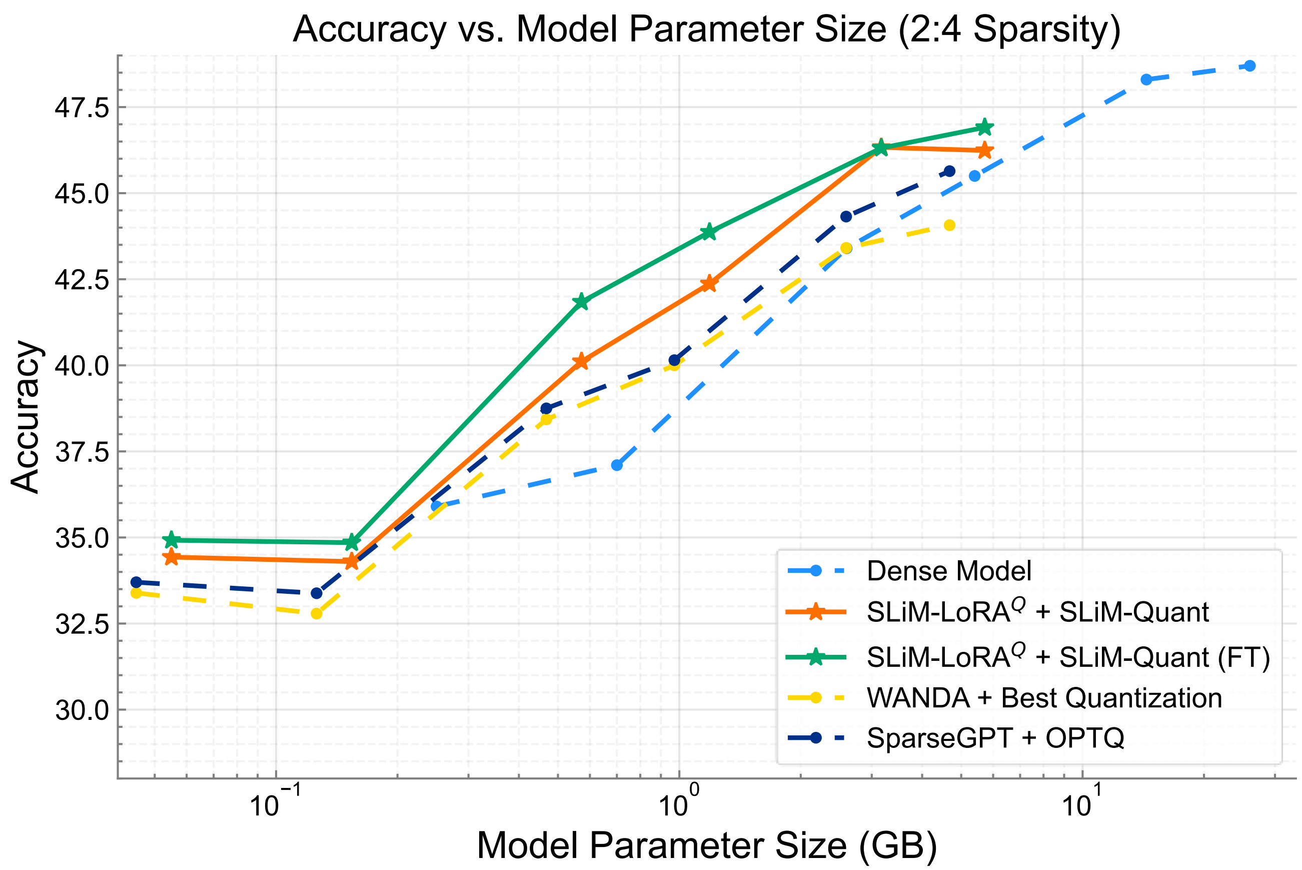
Memory Reduction

0.22×



SLiM | Larger Compressed vs. Smaller Dense

For a given parameter size budget, SLiM outperforms other methods! Even dense model!



The accuracy results are from OPT family of models.