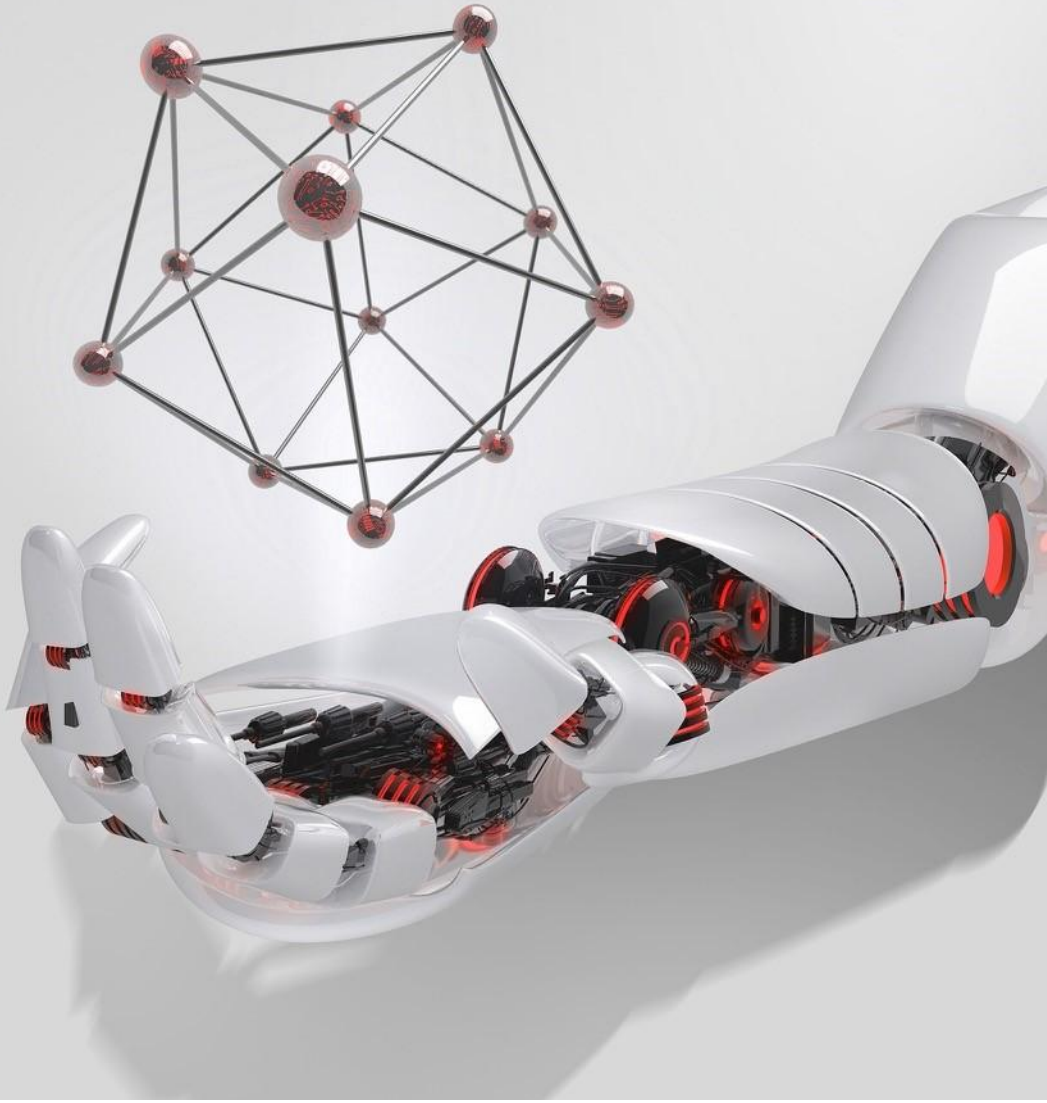
Highly Efficient and Effective LLMs with Multi-Boolean Architectures



Ba-Hien Tran

ba.hien.tran@huawei.com

Joint work with Van Minh Nguyen

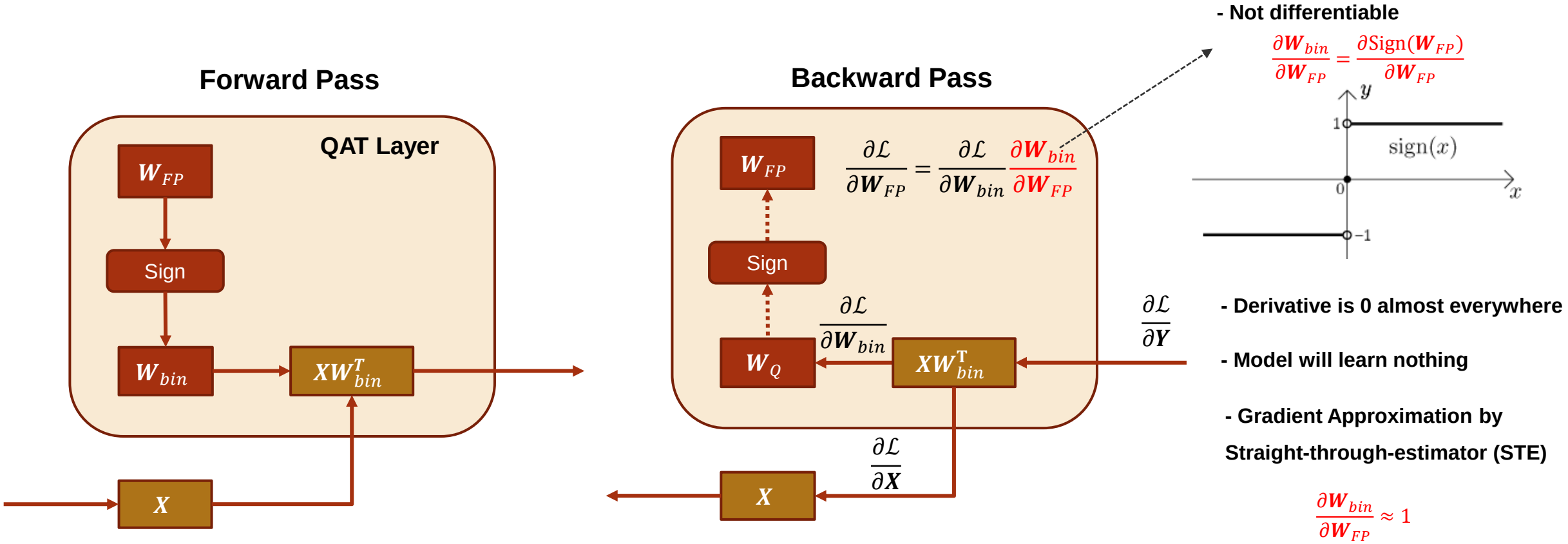
Huawei Paris Research Center

HUAWEI

Pitfalls of Binarized Neural Networks

- **Binarized linear layer:** $\boldsymbol{Y} = \alpha \cdot \boldsymbol{X}\boldsymbol{W}_{bin}^{T} + \boldsymbol{b}$, **with** $\boldsymbol{W}_{bin} = Sign(\boldsymbol{W}_{FP})$
- **Binarized weights are updated as :** $\boldsymbol{W}_{bin} = Sign(\boldsymbol{W}_{FP} - \eta \cdot \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_{FP}})$

→ **We train the weights in full-precision not binary domain**



Native Boolean Neural Networks

- **Bool linear layer:** $\boldsymbol{Y}_{[k,j]}^{(l)} = \sum_{i=1}^{n} L\left(\boldsymbol{X}_{[k,i]}^{(l)}, \boldsymbol{W}_{[i,j]}^{(l)}\right) + \boldsymbol{b}_{[j]}^{(l)}$, where L is a logic gate (**xnor**), and $\boldsymbol{W}_{[i,j]}^{(l)}$ are Boolean
- Logic gate can be extended to handle mixed-type data, such as real input and Boolean weights
- **Backward**. We can compute the loss signal for the weights as

$$\frac{\delta \mathcal{L}}{\delta \boldsymbol{W}_{[i,j]}^{(l)}} =: \boldsymbol{Q}_{[i,j]}^{(l)} = \sum_{k=1}^{b} \mathbf{1}\left(\boldsymbol{Q}_{[k,i,j]}^{(l)} = \text{True}\right)\left|\boldsymbol{Q}_{[k,i,j]}^{(l)}\right| - \sum_{k=1}^{b} \mathbf{1}\left(\boldsymbol{Q}_{[k,i,j]}^{(l)} = \text{False}\right)\left|\boldsymbol{Q}_{[k,i,j]}^{(l)}\right|$$

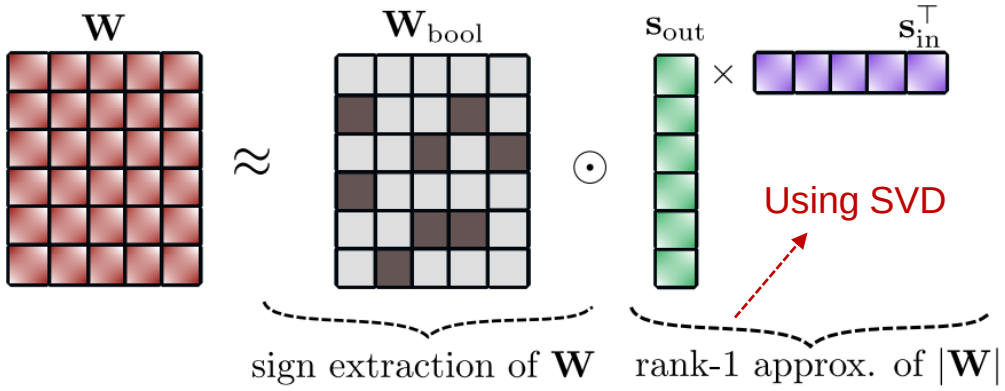
where $\boldsymbol{Q}_{[k,i,j]}^{(l)} = \mathbf{xnor}(\boldsymbol{Z}_{[k,j]}^{(l)}, \mathbf{X}_{[k,i]}^{(l)})$

- **Update rule**. $\boldsymbol{W}_{[i,j]}^{(l)} = \neg \boldsymbol{W}_{[i,j]}^{(l)}$ if $\mathbf{xnor}\left(\boldsymbol{Q}_{[k,j]}^{(l)}, \boldsymbol{W}_{[i,j]}^{(l)}\right) = \text{True}$
- **Accumulator**. $\boldsymbol{M}_{[i,j]}^{(l),(t+1)} \leftarrow \beta^{t} \boldsymbol{M}_{[i,j]}^{(l),t} + \eta \boldsymbol{Q}_{[i,j]}^{(l),t}$
- We train the weights directly in Boolean domain
- $\mathbf{xnor}(s, w) = s \times w$, enabling direct use of existing linear algebra operations

BOLD: Boolean Logic Deep Learning [Nguyen et al., NeurIPS 2024]

Boolean Reformulation for FP Linear Layers

Sign-value-independent Decomposition (SVID)



We can prove that this initialization is optimal for preserving information from FP weights

Approximate the full-precision weights

$$\mathbf{W} = \mathbf{W}_{\text{bool}} \odot |\mathbf{W}| \approx \mathbf{W}_{\text{bool}} \odot \left(\mathbf{s}_{\text{out}}\mathbf{s}_{\text{in}}^{\top}\right)$$

The linear layer becomes

$$\mathbf{X}\mathbf{W}_{\text{FP}}^{\top} \approx \left[\left(\mathbf{X} \odot \mathbf{s}_{\text{in}}^{\top}\right)\mathbf{W}_{\text{bool}}\right] \odot \mathbf{s}_{\text{out}}^{\top}$$

Proposition 4.1 (Xu et al. [59]). *For* $\mathbf{W} \in \mathbb{R}^{m\times n}$, *write* $\mathbf{W} = \widetilde{\mathbf{U}}\widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{V}}^{\top}$ *its* SVD. *Let* $\mathbf{a} = \sqrt{\tilde{\sigma}_1}\widetilde{\mathbf{U}}_{[:,1]}$, *and* $\mathbf{b} = \sqrt{\tilde{\sigma}_1}\widetilde{\mathbf{V}}_{[:,1]}$. *With the notations as described above, we have:*

$$\left\|\mathbf{W} - \mathbf{W}_{\text{bool}} \odot \mathbf{s}_{\text{out}}\mathbf{s}_{\text{in}}^{\top}\right\|_F^2 \le \left\|\mathbf{W} - \mathbf{a}\mathbf{b}^{\top}\right\|_F^2. \tag{4}$$

Proposition 4.3. *For* $\mathbf{W} \in \mathbb{R}^{m\times n}$ *and the notations as described above, we have:*

$$\left\|\mathbf{W} - \mathbf{W}_{\text{bool}} \odot \mathbf{s}_{\text{out}}\mathbf{s}_{\text{in}}^{\top}\right\|_F^2 \le \left\|\mathbf{W} - \mathbf{W}_{\text{bool}} \odot \mathbf{c}\mathbf{d}^{\top}\right\|_F^2, \quad \forall \mathbf{c} \in \mathbb{R}^{m\times 1}, \forall \mathbf{d} \in \mathbb{R}^{n\times 1}. \tag{5}$$

OneBit: Towards Extremely Low-bit Large Language Models [Xu et al., NeurIPS 2024]

Enhanced Expressivity with Multiple Boolean Kernels

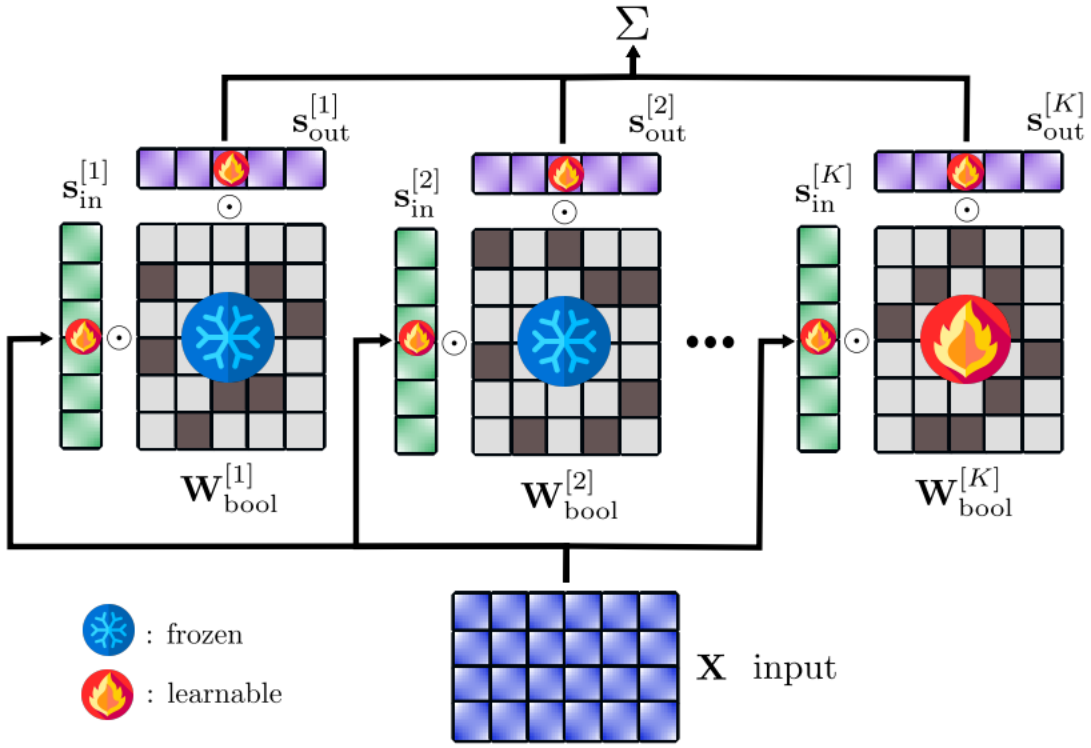
SVID is good but still not expressive enough to fully capture FP weights

- We employ multiple kernels, where each kernel utilizes distinct Boolean weights and scaling factors

$$\mathbf{W}_{\text{FP}} \approx \mathbf{W}_{\text{approx}} \triangleq \sum_{k=1}^{K} \mathbf{W}_{\text{bool}}^{[k]} \odot (\mathbf{s}_{\text{out}}^{[k]} \mathbf{s}_{\text{in}}^{[k]\top})$$

- The linear layers become

$$\mathbf{X}\mathbf{W}_{\text{FP}}^{\top} \approx \sum_{k=1}^{K} \left[\left(\mathbf{X} \odot \mathbf{s}_{\text{in}}^{[k]\top} \right) \mathbf{W}_{\text{bool}}^{[k]} \right] \odot \mathbf{s}_{\text{out}}^{[k]\top}$$



- The dominant computational cost is from the matrix multiplication between the Boolean weights and scaled inputs

- We showed that training the last kernel is enough!

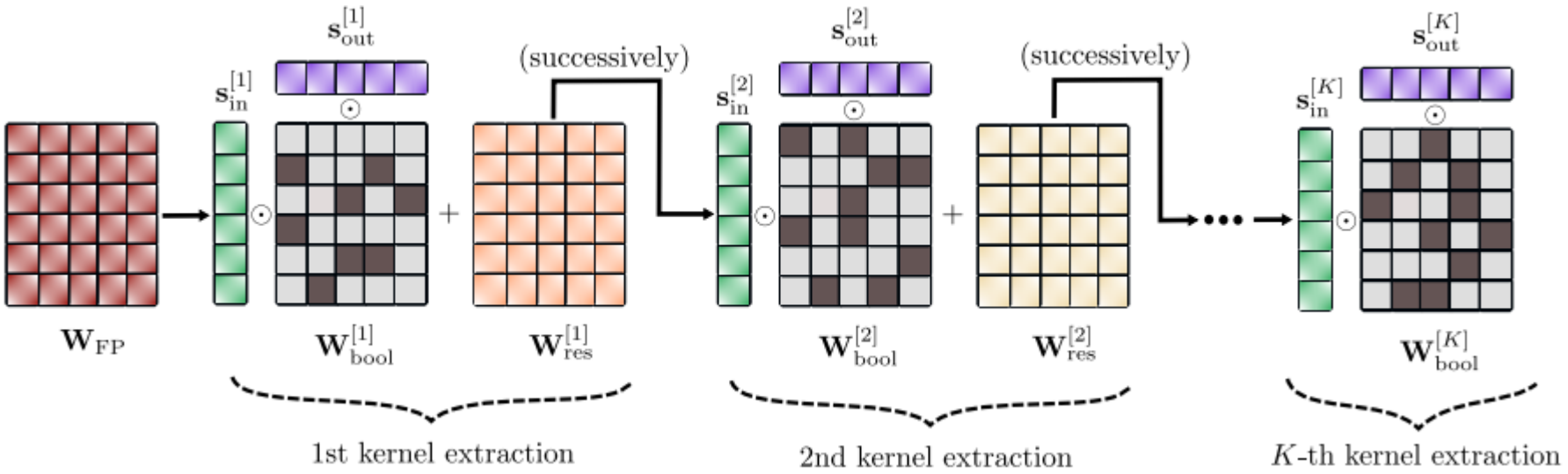
Effective Knowledge Transfer into Boolean Models

Successive Extraction using SVID

- Further proceed to SVID process to approximate **residual error** introduced by the previous step

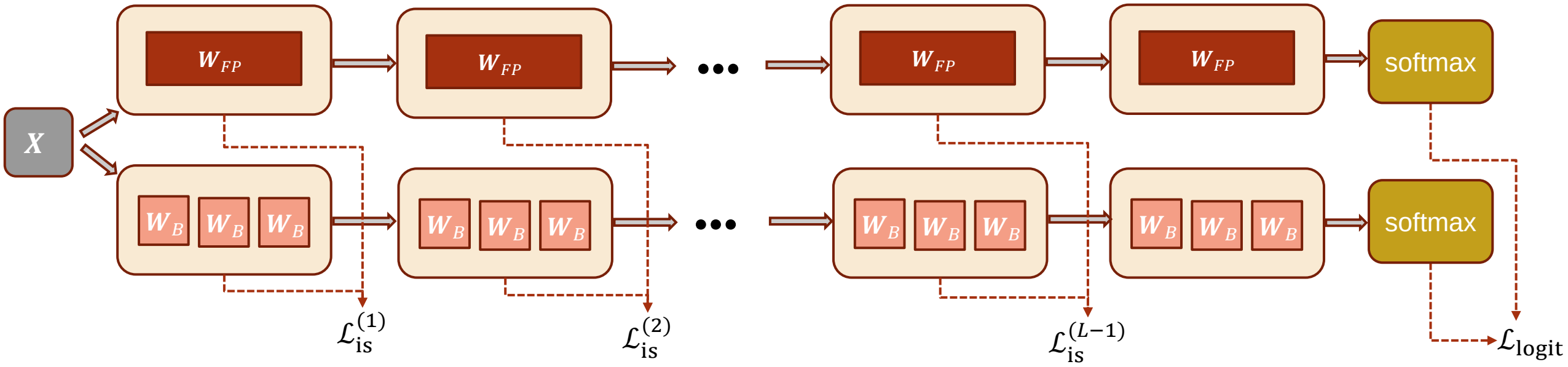
$$\mathbf{W}_{\text{res}}^{[k]} = \mathbf{W}_{\text{input}}^{[k]} - \mathbf{W}_{\text{bool}}^{[k]} \odot \left(\mathbf{s}_{\text{out}}^{[k]} \mathbf{s}_{\text{in}}^{[k]\top} \right)$$

Residual matrix



Effective Knowledge Transfer into Boolean Models

Finetuning with Knowledge Distillation

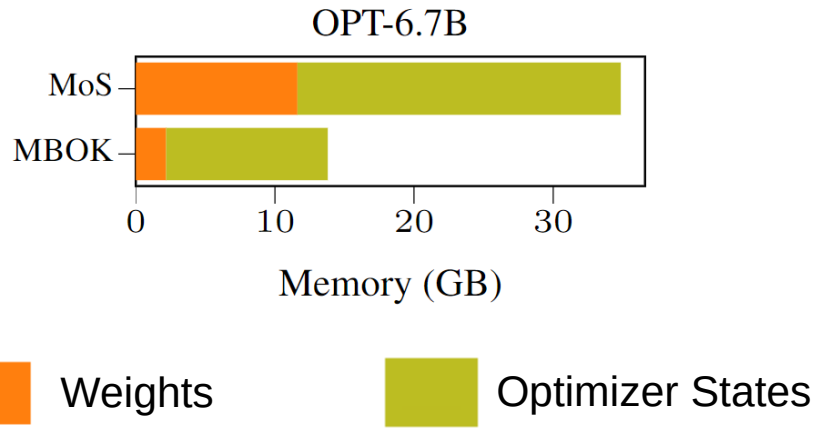


- Using multiple kernels effectively captures the original weight matrix, very small residual errors may still remain
- These errors can accumulate as they propagates through the layers
- Knowledge distillation for calibrating the whole model using very efficient Boolean optimizer

$$\mathcal{L} = \mathcal{L}_{\text{logits}} + \gamma \mathcal{L}_{\text{is}} \qquad \mathcal{L}_{\text{logits}} = \frac{1}{L}\sum_{j=1}^{L} \mathrm{D}_{\text{logits}}\left(p_{\text{FP}}(\mathbf{X}_{[j]};\tau), p_{\text{bool}}(\mathbf{X}_{[j]};\tau)\right) \qquad \mathcal{L}_{\text{is}} = \frac{1}{L}\sum_{h\in H}\sum_{j=1}^{L}\left\|\mathbf{Q}_{\text{FP}}^{j,h} - \mathbf{Q}_{\text{bool}}^{j,h}\right\|_2^2$$

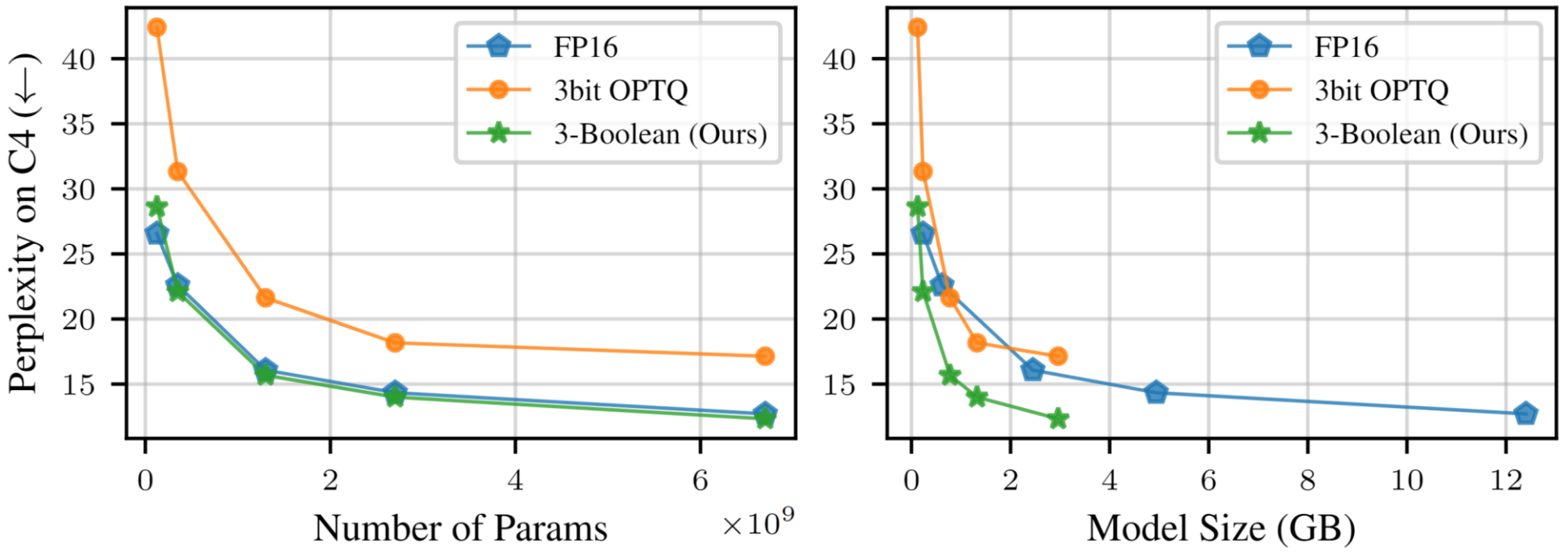
State-of-the-art Ultra Low-Bit Performance

- We evaluate our method on LLMs
- Ultra low-bit compression is very challenging
- Significantly outperforms SOTA quantization and binarization methods in terms of performance
- Closely matches the performance of FP model
- Efficient in both training and inference



Model	Method	Wbits	Perplexity (↓) Wiki2	C4	Zero-shot Accuracy (↑) BoolQ	PIQA	Hella.	WinoG.	ARC-e	ARC-c	Average
OPT-1.3B	FP-16	16	14.62	14.72	57.82	72.42	53.70	59.51	50.97	29.52	53.99
	PB-LLM	1.7	272.83	175.42	62.17	54.24	27.25	50.27	27.98	23.72	40.94
	BiLLM	1.11	69.45	63.92	61.92	59.52	33.81	49.32	34.38	22.35	43.55
	OneBit	1	20.36	20.76	57.85	66.53	39.21	54.61	42.80	23.97	47.50
	MoS	1	18.45	18.83	60.34	68.66	41.99	53.99	44.87	26.19	49.34
	GPTQ	2	9.5e3	3.8e3	39.60	52.07	25.57	49.33	26.68	23.63	35.15
	LLM-QAT	2	4.9e3	2.1e3	37.83	50.05	25.72	49.72	25.76	25.09	34.07
	OmniQuant	2	42.43	55.64	56.45	60.94	33.39	51.85	38.76	23.38	44.13
	MBOK [Ours]	2×1	**16.13**	**16.61**	58.53	70.67	48.11	56.75	48.19	27.90	**51.69**
	MBOK [Ours]	3×1	**15.30**	**15.68**	60.64	70.78	50.71	56.83	48.82	28.49	**52.71**
	MBOK [Ours]	4×1	**14.83**	**14.92**	60.95	70.85	51.02	56.85	49.13	29.24	**53.01**
LLaMA-7B	FP-16	16	5.68	7.08	73.21	77.42	72.99	66.85	52.53	41.38	64.06
	PB-LLM	1.11	41.66	48.15	62.23	58.65	34.64	51.14	33.08	25.68	44.24
	OneBit	1	8.48	10.49	62.50	70.40	54.03	55.32	41.07	30.88	52.36
	MoS	1	7.97	9.72	64.59	71.82	58.18	58.88	42.09	31.31	54.48
	GPTQ	2	1.9e3	7.8e2	43.79	49.95	25.63	49.41	25.84	27.47	37.02
	LLM-QAT	2	7.1e2	3.0e2	37.83	50.87	24.76	51.78	26.26	25.51	36.17
	OmniQuant	2	15.34	26.21	58.69	62.79	43.68	52.96	41.54	29.35	48.17
	MBOK [Ours]	2×1	**6.83**	**8.53**	69.20	74.32	64.80	60.30	49.05	34.90	**58.76**
	MBOK [Ours]	3×1	**6.20**	**7.76**	67.89	76.15	68.91	63.30	48.94	37.62	**60.47**
	MBOK [Ours]	4×1	**6.01**	**7.53**	68.16	76.71	69.85	62.09	49.24	38.14	**60.70**
LLaMA-13B	FP-16	16	5.09	6.61	68.47	79.05	76.24	70.17	59.85	44.54	66.39
	PB-LLM	1.7	35.83	39.79	62.17	58.70	33.97	52.17	31.86	23.63	43.75
	BiLLM	1.11	14.56	16.67	62.53	68.17	52.24	59.43	41.91	29.94	52.37
	OneBit	1	7.65	9.56	63.30	71.98	60.61	59.43	42.85	32.42	55.10
	MoS	1	7.16	8.81	63.82	73.88	64.05	60.93	44.28	33.11	56.68
	GPTQ	2	3.2e3	9.9e2	42.39	50.00	25.27	50.67	26.14	27.39	36.98
	LLM-QAT	2	1.8e3	1.2e3	37.83	50.33	25.40	51.62	27.02	26.87	36.51
	OmniQuant	2	13.43	19.33	62.20	68.99	54.16	53.83	45.50	30.38	52.51
	MBOK [Ours]	2×1	**6.17**	**7.88**	68.10	76.33	69.88	64.17	52.34	37.88	**61.45**
	MBOK [Ours]	3×1	**5.58**	**7.15**	67.39	77.74	73.37	66.61	54.04	41.21	**63.39**
	MBOK [Ours]	4×1	**5.38**	**6.91**	68.69	77.63	74.23	66.53	56.14	41.38	**64.10**

Compression-performance Tradeoff



OPT Model	WBits	Wiki2					C4				
		125M	350M	1.3B	2.7B	6.7B	125M	350M	1.3B	2.7B	6.7B
FULL-PRECISION	16	27.65	22.00	14.63	12.47	10.86	26.56	22.59	16.07	14.34	12.71
RTN [61, 12]	3	37.28	25.94	48.17	16.92	12.10	33.91	26.21	24.51	18.43	14.36
QPTQ [18]	3	31.12	24.24	15.47	12.87	11.39	29.22	24.63	16.97	15.00	13.18
MBOK [Ours]	3×1	**29.10**	**23.12**	**15.30**	**13.09**	**11.03**	**28.62**	**22.10**	**15.68**	**14.00**	**12.33**

Low Latency

- Latency (ms) of Linear Layers in LLaMA-13B

- Our approach (MBOK) dramatically speedups the inference

Weight Size	Full-precision 16	MBOK (Ours)	QUIP#	QTIP
5120 x 5120	0.16540	0.05074	0.62260	1.96368
5120 x 13824	0.42830	0.05098	0.62836	5.23681
13824 x 5120	0.43411	0.04987	0.62840	5.21193

- It is even much more significant with a native Boolean accelerator

Thank you.