BiE: Bi-Exponent Block Floating-Point for Large Language Models Quantization

Lancheng Zou[1], Wenqian Zhao[1], Shuo Yin[1], Chen Bai[1], Qi Sun[2], Bei Yu[1]

[1]The Chinese University of Hong Kong, [2]Zhejiang University

Highlights

- We propose **Bi-Exponent Block Floating-Point (BiE)**, a novel numerical representation, tackling the drawbacks of current quantization methods on LLMs.
- We theoretically investigate the quantization error of the naive Block Floating-Point and point out the rationale of the quantization error reduction of BiE.
- An offline thresholding optimization strategy is proposed to enhance the BiE encoding flow with Bayesian Optimization.

Background

Challenges in LLM Quantization

- Quantization-aware Training (QAT) is not a practical choice for LLM quantization due to its significant hardware cost. In general, Post-training Quantization (PTQ) shows less flexibility and obtain worse performance without retraining and finetuning compared with QAT, but it's the feasible solution for LLM quantization due to its affordable overhead.
- When scaling up LLMs beyond 6.7B, systematic outliers with large magnitude will emerge in activations, leading to large quantization errors and accuracy degradation.

Motivation

Observations For the distribution of activation outliers, it can be seen as two distribution superposition. When applying the integer quantization, it will waste lots of bits to keep the precision. Otherwise, they need to conduct some complex transformation to fuse the two distribution.

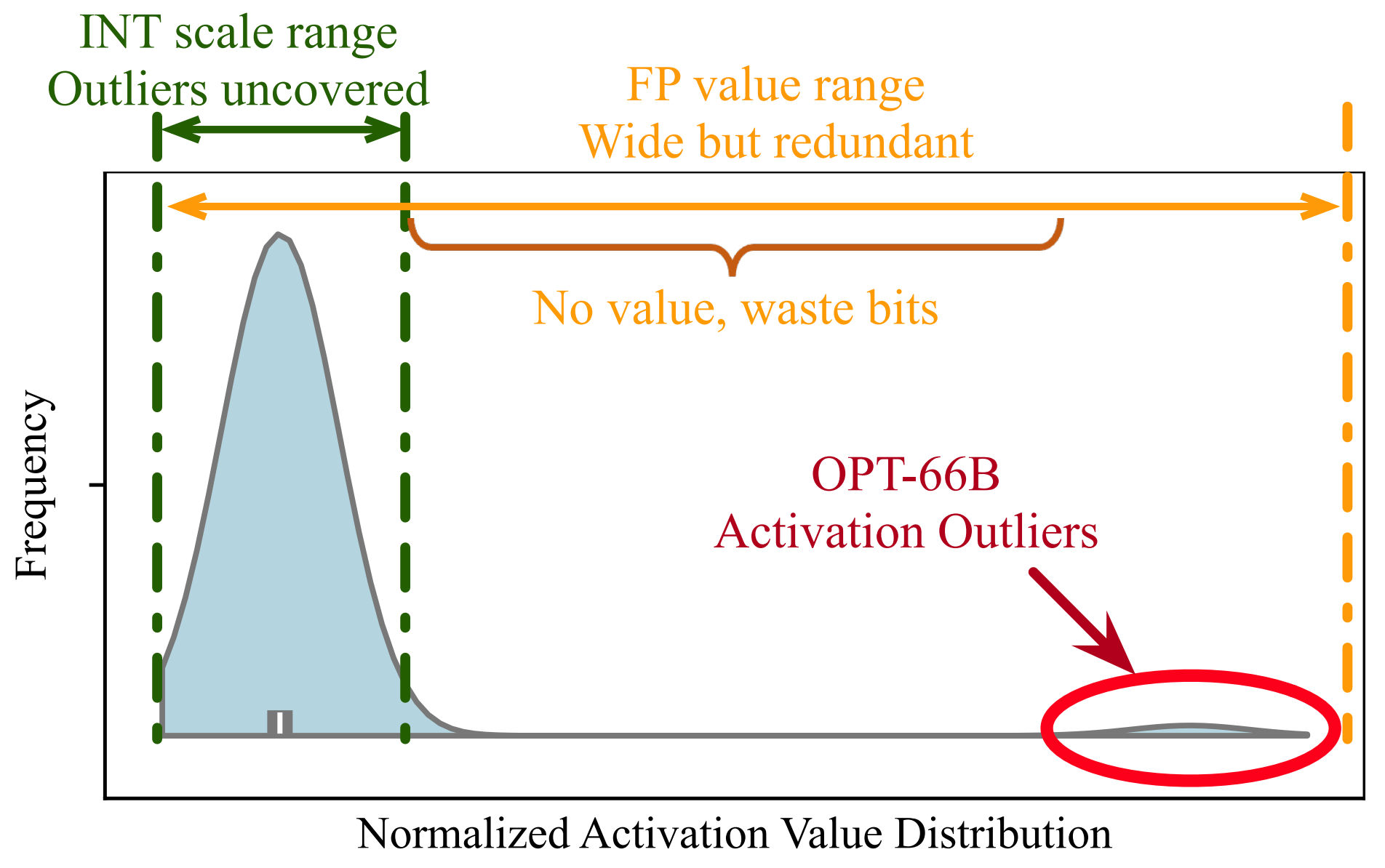

Figure 1. Distribution of input activation in one Linear Layer of OPT-66B model.

The quantization error of BFP has zero mean and variance σ^2 which is defined as

$$\sigma^2 = \frac{2^{-2L_m}}{12}\sum_{i=1}^{N_\gamma} p_{\gamma_i} 2^{2\gamma_i}, \tag{1}$$

It is found that the quantization errors mainly depend on L_m and p_{γ_i}. When L_m is increasing, the quantization error will be reduced. Moreover, if the probabilities of taking a larger exponent as the shared exponent are smaller, then the quantization errors of BFP will be reduced. Since the bit length is fixed in the typical quantization flow, we can only decrease the probabilities of larger shared exponents for quantization error reduction.

Methodology

Bi-Exponent Block Floating-Point Different from the vanilla BFP, the significant modification is that the BiE format has a bi-shared exponent for each block, e_o for the outlier part and e_n for the normal part, and private 1-bit type t_i that indicates this component belongs to outlier part or normal part. That means the normal part of the block will use e_n as their shared exponent, and the outlier part will use e_o as the corresponding shared exponent.

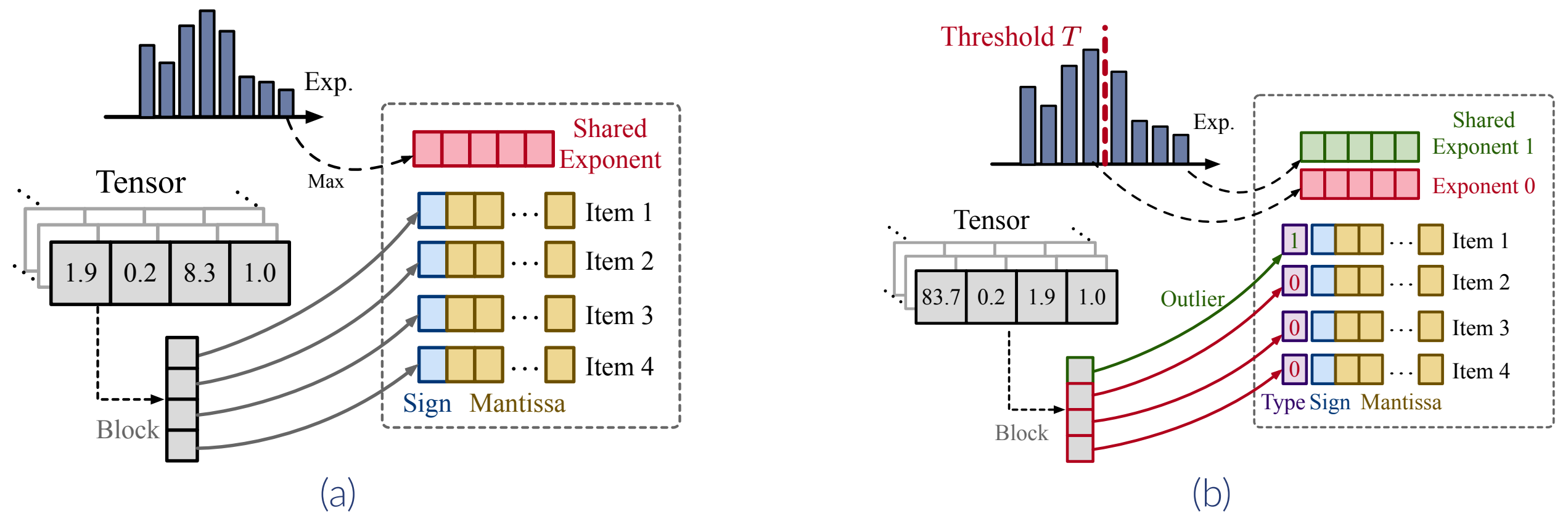

Figure 2. Data format of (a) Block Floating-Point, (b) Bi-Exponent Block Floating-Point

Given a vector **X** with N elements in FP16, we can obtain its BiE representation **X′** as

$$2^{e_n|e_o}[(-1)^{s_0}m'_0, (-1)^{s_1}m'_1, ..., (-1)^{s_{N-1}}m'_{N-1}], \tag{2}$$

$$e_n = \max\{e_i \mid |x_i| \le T\}, e_o = \max\{e_i \mid |x_i| > T\}, \tag{3}$$

$$t_i = \begin{cases} 0 & |x_i| \le T, \\ 1 & |x_i| > T, \end{cases} \tag{4}$$

$$m'_i = m_i >> (e_n \cdot (1 - t_i) + e_o \cdot t_i - e_i) \tag{5}$$

where T is the threshold value in FP16 for distinguishing the normal part and the outlier part. m'_i denotes the private mantissa of BiE. $e_n|e_o$ means that if $t_i = 0$, shared exponent of x'_i is e_n, otherwise it is e_o.

Offline Threshold Searching Strategy In order to select the optimal threshold value for each tensor during inference, we build an efficient offline threshold searching strategy based on Bayesian Optimization (BO). Firstly, we define the threshold value search space for all tensors of LLM inference stage (including weights and activations) and perform search space pruning to reduce the size of the search space and speed up the convergence.
We adopt the Gaussian Process (GP) as the surrogate model with matérn kernel function for its robust ability to capture uncertainty quantification during modeling. For the quantization performance indicator, we simply use the mean square error (MSE) between the output of the full-precision model and the quantized model.

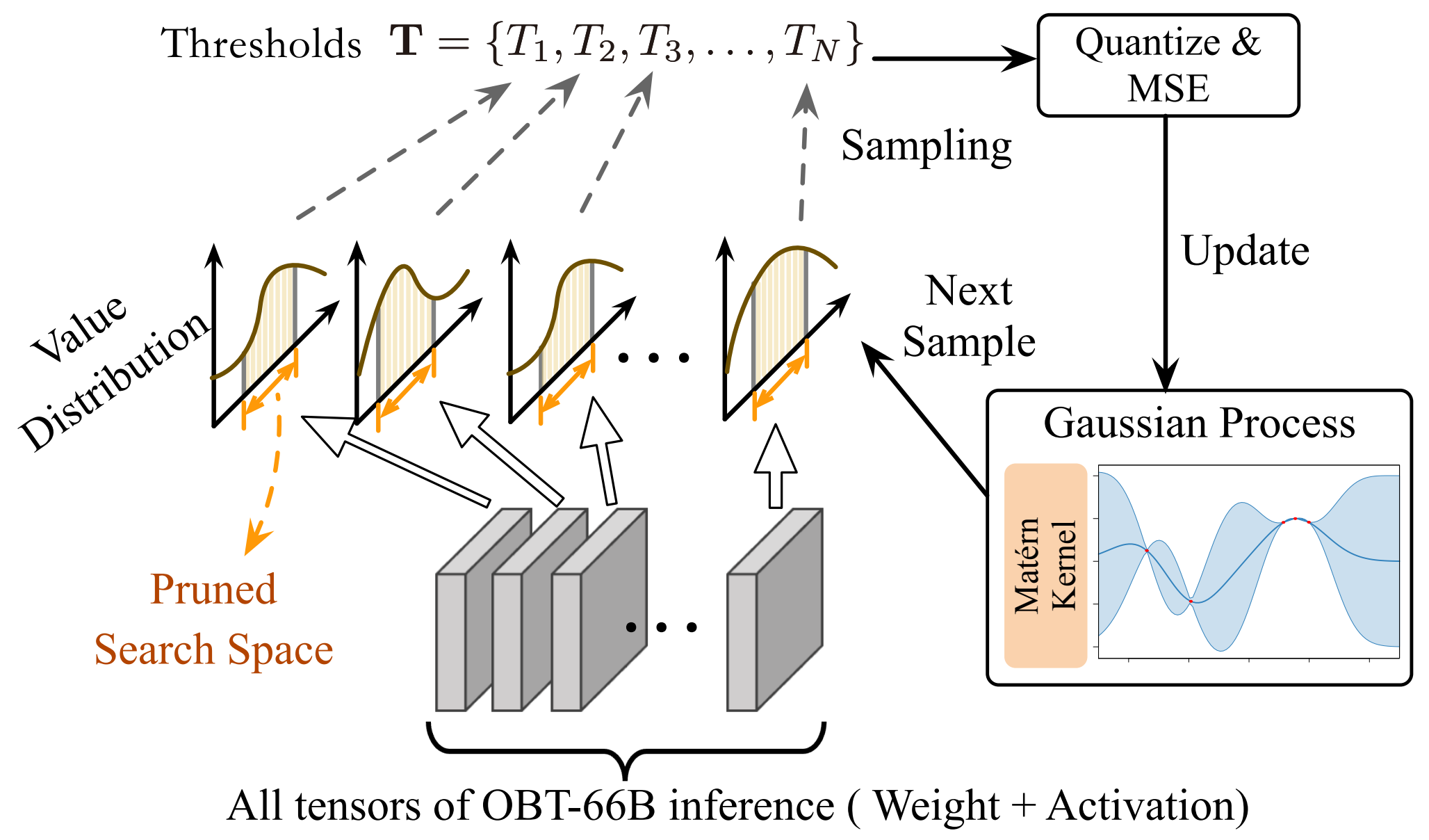
Thresholds $\mathbf{T} = \{T_1, T_2, T_3, \ldots, T_N\}$
Quantize &
MSE
Sampling
Update
Next
Sample
Value
Distribution
Gaussian Process
Matérn
Kernel
Pruned
Search Space
All tensors of OBT-66B inference (Weight + Activation)

Figure 3. Threshold searching with Bayesian optimization.

Evaluation Results

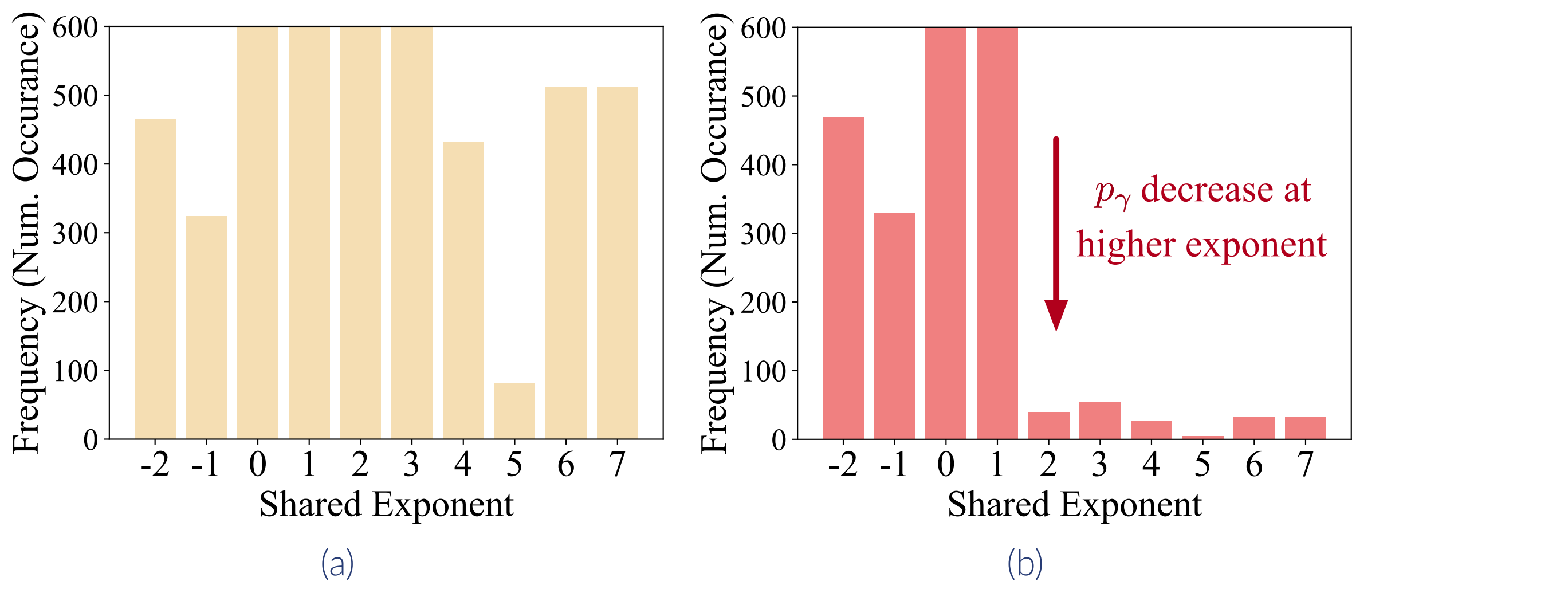

Figure 4. p_γ distribution of (a) BFP and (b) BiE on the same activation in OPT-66B model.

Table 1. Comparison with different methods and different quantization configurations for OPT-models on Wikitext2 (**Perplexity↓**). We **highlight** our 4-bit BiE results which are comparable with SmoothQuant W8A8.

Method	Config	6.7B	13B	30B	66B
FP16	/	10.64	9.91	9.33	9.12
SmoothQuant	W8A8	11.33	12.79	9.35	9.62
SmoothQuant	W6A6	13.16	13.75	82.54	3383.21
BFP	W4A4	11.22	11.15	9.90	14.16
BiE (Ours)	W4A4	**10.93**	**10.39**	**9.37**	**9.82**
BFP	W3A3	14.61	13.85	13.83	137.72
BiE (Ours)	W3A3	12.10	11.13	10.01	32.41

Table 2. PTQ performances using different methods on LLaMA-2 models for various tasks. **Average↑** is the average accuracy among various tasks. **Perplexity↓** is for WikiText2. SQ represents SmoothQuant. We **highlight** our 4-bit BiE results.

	Config	Average↑			Perplexity↓		
		7B	13B	70B	7B	13B	70B
FP16	/	76.16	77.50	81.62	6.73	5.95	4.51
SQ	W8A8	75.12	77.47	80.35	6.93	5.94	4.56
SQ	W6A6	66.07	67.51	76.96	10.38	8.28	5.73
BFP	W4A4	68.65	71.20	80.24	7.69	6.74	4.69
BiE	W4A4	**72.83**	**75.90**	**80.21**	**7.00**	**6.16**	**4.61**
BFP	W3A3	42.28	43.52	73.05	20.86	14.70	5.78
BiE	W3A3	50.00	63.25	77.24	8.92	7.86	5.21

Conclusion

- BiE can be naturally adapted to the numerical distribution characteristics of the LLMs and achieve negligible loss in 4-bit activations and weights quantization.
- BiE can balance precision and hardware efficiency.
- BiE is not limited to LLM quantization, it can be used in any model with any distribution.

References

[1] Zhourui Song, Zhenyu Liu, and Dongsheng Wang. Computation error analysis of block floating point arithmetic oriented convolution neural network accelerator design. In *Proc. AAAI*, volume 32, 2018.

[2] Cheng Zhang, Jianyi Cheng, Ilia Shumailov, George Constantinides, and Yiren Zhao. Revisiting Block-based Quantisation: What is Important for Sub-8-bit LLM Inference? In *Proc. EMNLP*, pages 9988–10006, 2023.

International Conference on Machine Learning 2024, Vienna