LBLLM: Lightweight Binarization of Large Language Models via Three-Stage Distillation

Anonymous ACL submission

Abstract

Deploying large language models (LLMs) in resource-constrained environments is hindered by heavy computational and memory requirements. We present LBLLM, a lightweight binarization framework that achieves effective W(1+1)A4 quantization through a novel three-stage quantization strategy. The framework proceeds as follows: (1) initialize a high-quality quantized model via PTQ; (2) quantize binarized weights, group-wise bitmaps, and quantization parameters through layer-wise distillation while keeping activations in full precision; and (3) training learnable activation quantization factors to dynamically quantize activations to 4 bits. This decoupled design mitigates interference between weight and activation quantization, yielding greater training stability and better inference accuracy. LBLLM, trained only using 0.016B tokens with a single GPU, surpasses existing state-of-the-art binarization methods on W2A4 quantization settings across tasks of language modeling, commonsense QA, and language understanding. These results demonstrate that extreme low-bit quantization of LLMs can be both practical and highly effective without introducing any extra high-precision channels nor rotational matrices commonly used in recent PTQ-based works, offering a promising path toward efficient LLM deployment on resource-limited situations.

1 Introduction

Large language models (LLMs) have recently gained significant attention, particularly for long-context reasoning tasks where both weights and activations dominate memory and compute costs. Quantization, which reduces precision by mapping high-bit parameters to low-bit representations, offers an effective means to jointly compress weights and activations and accelerate inference (Ouyang et al., 2024; Kurtic et al., 2025). However, most existing approaches still operate at relatively high bit-widths (typically ≥ 4 bits), limiting their potential for extreme compression in resource-constrained environments.

In joint weight and activation quantization, outlier activations, characterized by extreme magnitudes and long-tailed distributions (see Figure 4), pose a major challenge to maintaining accuracy. Moreover, extreme low-bit settings further constrain representational capacity, making sub-4-bit quantization seldom explored. Existing approaches address these issues in two ways: (1) auxiliary-channel methods (e.g., (Zhao et al., 2024)) retain high-precision channels to stabilize quantization, but these extra bits reduce compression efficiency and hinder inference speed due to mixed-precision execution; and (2) rotation-based methods (e.g., (Ashkboos et al., 2024)) redistribute outliers via matrix transformations, which work well at W4A4 precision but degrade sharply at lower bit-widths and introduce additional computational and memory overhead.

Another line of research targets binarization, which significantly reduces bit-width but exacerbates quantization errors, often causing severe performance degradation. Existing binarization works typically quantize weights only, leaving activations in full precision (FP16). Depending on parameter optimization strategies, these methods fall into two categories: (1) Quantization-aware training (QAT) methods (Ma et al., 2024; Wang et al., 2024), recover representational capability through extensive training but demand substantial computational resources, which requires dozens of 80GB GPUs and thousands of GPU-hours on datasets exceeding 10 billion tokens, limiting their practical use. (2) Post-training quantization (PTQ) methods (Huang et al., 2024; Dong et al., 2024), rapidly generate binarized models from pretrained checkpoints by employing an additional bit for fine-grained grouping and extra bits to encode outliers. However, these approaches frequently suffer from large accuracy degradation (more than 15 perplexity points

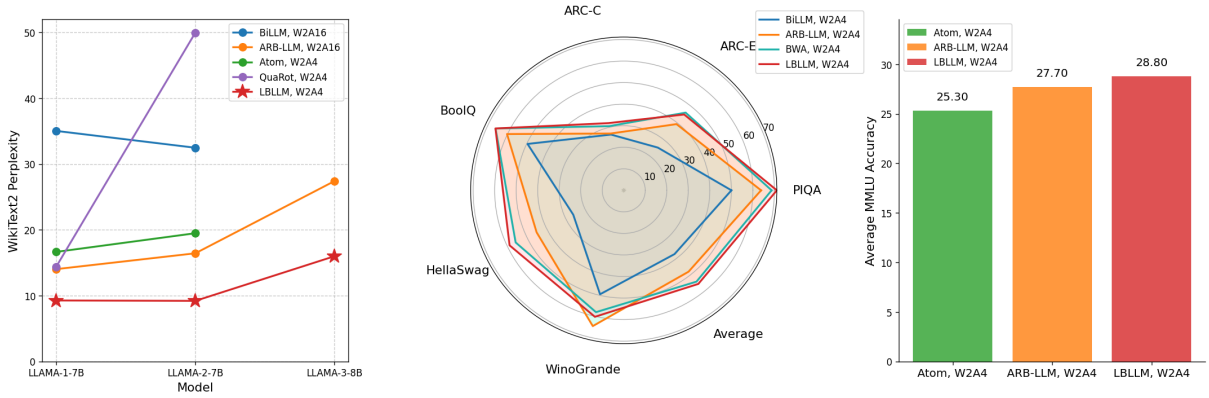

Figure 1: Left: Comparison of post-quantization perplexity between LBLLM and other methods across different LLAMA models on the W2A4 equivalent setting. Middle: Comparison of post-quantization commonsense QA performance on LLAMA-2-7B. Right: Comparison of average accuracy on the MMLU dataset after quantization across different LLAMA-1-7B models.

on Wiki2) compared to full-precision models.

To enable efficient and high-quality binarization, we propose LBLLM, a lightweight framework that combines the strengths of PTQ and QAT. The model is first initialized via PTQ and then fine-tuned by a two-stage lightweight QAT that quantizes model weights and activation separately. Compared to prior PTQ-based methods, LBLLM improves perplexity by over 10 points and achieves accuracy comparable to full QAT approaches, while requiring only 0.016B tokens and a few dozen GPU hours on a single GPU to binarize a 7B model. Crucially, LBLLM avoids auxiliary high-precision channels and rotation matrices, enabling both effective memory compression and real inference acceleration.

We observed that directly extending existing lightweight layer-wise training (Ding et al., 2024) to jointly quantize weights and activations to low bit-width is ineffective. Quantization errors from weights and activations interfere during straight-through gradient estimation, resulting in unstable optimization. To address this issue, we decouple the QAT process into two stages: (1) binarize weights and optimizing their quantization parameters while keeping activations in full precision; (2) subsequently quantize activations to 4 bits, introducing learnable clipping parameters to handle outliers and fine-tuning only quantization parameters. This staged approach achieves effective W(1+1)A4 quantization with minimal overhead.

The main contributions are summarized as follows.

- We introduce a three-stage quantization strategy combined PTQ initialization and lightweight QAT for joint weight-activation quantization, addressing the challenge of simultaneous low-bit quantization, which was not touched in previous layer-wise QAT training approaches.
- Our hierarchical distillation strategy eliminates the need for high-precision auxiliary channels and rotation matrices commonly used in PTQ-based binarization approaches.
- Extensive experiments demonstrate that our method surpasses state-of-the-art binarization on the W2A4 quantization level, while requiring only 0.016B tokens and a few dozen GPU hours on a single GPU.

2 Related Works

2.1 Binarization

Binarization compresses both weights and activations into 1-bit representations (Park et al., 2025b), achieving maximal storage savings and enabling highly efficient bitwise inference. However, applying binarization to LLMs is particularly challenging due to their structural complexity and the demanding nature of generative tasks.

Recent studies have proposed various strategies to address the challenges of LLM binarization. OneBit (Xu et al., 2024) achieves weight-only binarization by introducing auxiliary full-precision parameters, while BitNet (Wang et al., 2023, 2024) extends binary representations $(-1, +1)$ to ternary schemes $(-1, 0, +1)$, effectively achieving 1.58-bit weights and low-bit activations. Although these

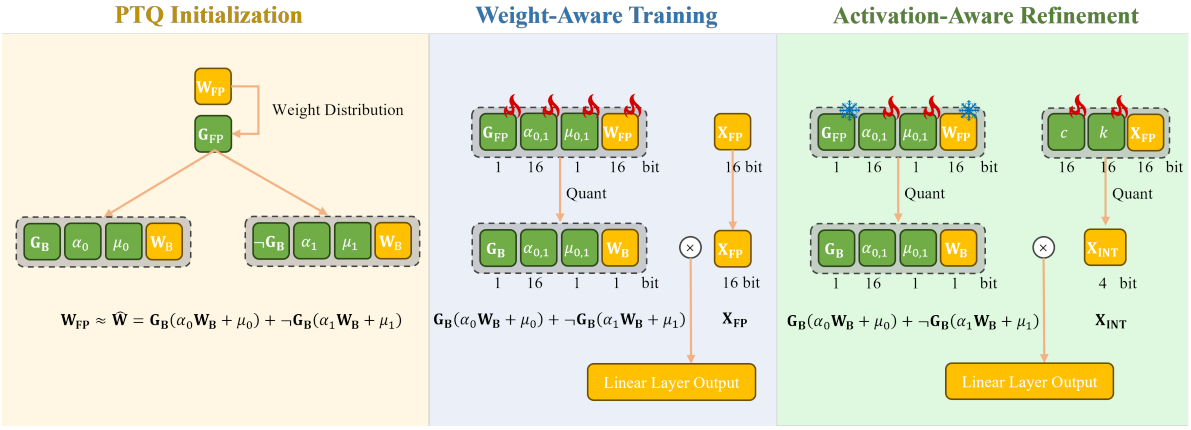

Figure 2: Illustration of the three-stage quantization strategy in LBLLM: Stage 1 uses a binarized PTQ method to obtain a high-quality initialized model; Stage 2 keeps activations in full precision and performs quantization-aware training on weights only; Stage 3 jointly quantizes activations and includes related parameters in training.

methods improve training stability, they demand extensive computational resources, often requiring hundreds of high-end GPUs, which limits their practicality for broader applications. Moreover, most works focus on weight binarization leaving the activations as full precision FP16 format.

To reduce computational cost, lighter PTQ-based methods such as BiLLM (Huang et al., 2024), ARB-LLM (Li et al., 2024), and BWA (Song et al., 2025) employ bitmaps for fine-grained weight grouping and retain a small fraction of salient values in higher precision. While effective, these methods rely on extra bits for grouping and auxiliary high-precision channels to stabilize quantization, which limits compression efficiency and slows inference. Our work builds on the PTQ paradigm but introduces a lightweight QAT stage to reach W(1+1)A4 quantization on both weights and activations.

2.2 Joint Weight and Activation Quantization

Quantizing only model weights reduces storage cost, but full acceleration with low-bit operators requires quantizing both weights and activations to lower bandwidth and computation, particularly for long-sequence inference (Jeon et al., 2025; Zhao et al., 2025; Tao et al., 2025). The key challenge is handling activation outliers with extremely large values that cannot be accurately represented in low-bit formats.

Several methods address this by introducing auxiliary channels (Park et al., 2025a; Su et al., 2025). LLM.int8 (Dettmers et al., 2022) preserves a small set of outlier channels in full precision, while Atom (Zhao et al., 2024) reorders channels based on Hessian values and quantizes outlier channels to 8 bits, using lower precision for the remaining values. Although effective, these approaches sacrifice compression efficiency and hardware friendliness due to their mixed-precision design.

An alternative line of work, including SmoothQuant (Xiao et al., 2023) and AWQ (Lin et al., 2024b), balances quantization difficulty by applying equivalent rescaling transformations to weights and activations. Extensions such as QuaRot (Ashkboos et al., 2024), SpinQuant (Liu et al., 2024), and DuQuant (Lin et al., 2024a) leverage Hadamard or learnable rotation matrices to achieve W4A4 quantization. However, these methods introduce additional storage and compute overhead, and their benefits diminish when targeting ultra-low-bit (2 bits) settings.

In contrast, our work removes the reliance on mixed-precision representations and rotation matrices, focusing on weight binarization with fine-grouping combined with 4-bit activation quantization, achieving higher compression W(1+1)A4 and hardware efficiency without sacrificing accuracy.

3 Preliminary

This section provides the background on LLM binarization, covering its scope, weight binarization, fine-grained grouping, and a PTQ-based binarization scheme.

Quantization in LLMs Our work targets Transformer-based LLMs, focusing on linear layers and the KV cache, which together account for over 90% of total computation. Less compute-intensive components, such as embeddings, normalization

layers, and activation functions, remain in high precision. In each linear layer, both weights and input activations are quantized to the target bit-width prior to matrix multiplication, and the KV cache is quantized to the same bit-width as the activations. For example, W2A4 denotes 2-bit quantized weights (W2) and 4-bit quantized activations (A4).

Weight Binarization Weight binarization represents the extreme form of quantization, using a single bit to encode each weight. In linear layers, weights are approximated as:

$$\mathbf{W} \approx \mathbf{W}_q = \alpha(\mathbf{W}_B - \mu), \tag{1}$$

where $\mathbf{W}_B \in 0, 1^{n \times m}$ is the binarized matrix, and α, μ are full-precision (FP16) scaling and offset parameters. Here, $\alpha = \max(\mathbf{W}) - \min(\mathbf{W})$ and $\mu = -\lfloor \min(\mathbf{W})/\alpha \rceil$; $\mathbf{B}$ is stored using 1 bit per entry. In the QAT setting, α and μ are treated as learnable parameters and optimized jointly with other network weights, allowing the binarization scheme to better adapt to the data distribution.

Fine-Grained Weight Grouping Prior PTQ-based binarization (Huang et al., 2024) studies show that a single bit is often insufficient to preserve representational capacity. One strategy is to add an auxiliary bit as a bitmap to enable fine-grained element-wise grouping. Weights in the same group share quantization parameters, while different groups are quantized independently. This scheme effectively encodes four distinct states (equivalent to 2 bits) by combining one bit for grouping and one bit for the binary weight. Compared to a conventional 2-bit weight representation, it enables faster inference through efficient bitwise operations.

Activation Quantization and Binarization In long-sequence autoregressive LLM inference, activations and KV caches dominate memory and bandwidth usage, making their quantization essential. Following prior work, we quantize activations to 4 bits, represented either directly or as four binary channels, and jointly optimize them with binary weights and bitmaps for improved representational capacity. Previous PTQ methods, such as BWA (Song et al., 2025), search quantization parameters via clustering to achieve higher-quality binarization without requiring large datasets or high computation.

PTQ approaches still exhibit performance gaps relative to QAT-based methods. Moreover, PTQ methods often rely on auxiliary channels to stabilize performance, leading to an average bit-width that falls short of the ideal compression target. In the subsequent section, we introduce LBLLM, which removes the additional channels using a lightweight training framework and achieves better performance.

4 Method

In this section, we introduce LBLLM, a lightweight binarization framework for LLMs based on QAT. We begin by outlining the overall three-stage quantization pipeline, followed by a description of the model adaptations used for quantization distillation, including a relaxed formulation of binary weights and a progressively staged training strategy tailored for efficient optimization using only a small fraction of the training data.

4.1 Three-Stage Training

As shown in Figure 2, LBLLM employs a three-stage training pipeline. The process initializes the quantized model via a PTQ approach. In the second stage, we binarize the weights and distill weight-related parameters while keeping activation parameters in full precision. Finally, activation quantization layers are introduced, and a subset of both weight and activation quantization parameters is jointly fine-tuned.

Stage 1: PTQ Initialization We adopt the binarization method proposed in (Song et al., 2025) to obtain an initial quantized model, but discard its mixed-precision configuration in order to achieve higher compression efficiency. Specifically, we employ a fine-grained bitmap grouping structure and optimize the quantization parameters using a Hessian-weighted EM algorithm to minimize the quantization error of individual weight matrices. In table 6, experiments demonstrate that a well-initialized PTQ model significantly improves the convergence of quantized models under lightweight training regimes.

Stage 2: Weight-Aware Training (WAT) We binarize the weights of all linear layers in the Transformer architecture, including those in both Attention and MLP modules, while keeping activations in full precision (FP16) during this stage. Training follows a layer-wise distillation strategy and employs the straight-through estimator (STE) to jointly update the quantization parameters μ, α, $\mathbf{G}$, and the original weight matrix $\mathbf{W}$ for each layer.

Stage 3: Activation-Aware Refinement (AAR)

In the final stage, we fine-tune the model while dynamically quantizing activations. To address the prominent long-tail distribution characteristics of LLM activations, we propose a differentiable distribution-aware quantizer. This mechanism aims to adaptively partition the core dense region from sparse outlier regions using learnable knee points. It assigns distinct quantization step sizes to each region while employing learnable clipping factors to mitigate the impact of extreme outliers

$$\begin{aligned}\hat{\mathbf{X}} = &\sum_{j=1}^{3} \mathbb{I}(k_{j-1} \le \mathbf{X} < k_j) \\ &\cdot \left(\left(\text{clamp} \left(\left\lfloor \frac{\mathbf{X}}{\alpha_j} + \mu_j \right\rceil, 0, b_j \right) - \mu_j \right) \cdot \alpha_j \right), \\ \alpha = &\frac{c_\alpha \max(\mathbf{X}) - c_\beta \min(\mathbf{X})}{2^k - 1}, \\ \mu = &-\lfloor \frac{c_\beta \min(\mathbf{X})}{\alpha} \rceil, \end{aligned} \tag{2}$$

where k_j denotes the trainable knee points. c_α and c_β are trainable clipping factors. $\mathbb{I}(\cdot)$ is the hard indicator function. For the j-th region, α_j, μ_j and b_j represent the quantization parameters and assigned bit-width, respectively. In this stage, we update only the quantization parameters (μ, α), while keeping the binarized weights $\mathbf{W}$ and group assignments $\mathbf{G}$ fixed. This refinement reduces activation quantization error and improves overall model accuracy as shown in Table 3.

4.2 Lighter QAT for Binarization

LBLLM retains the same bianrized model architecture as prior PTQ approach (Li et al., 2024; Song et al., 2025), but eliminates the auxiliary high-precision channels, resulting in a fully quantized W(1+1)A4 format. Unlike prior works relying on architecture-specific tricks, such as rotation matrices (Ashkboos et al., 2024) or extra high-precision channels, LBLLM operates without modifying the model structure.

Learning Group Bitmap The $W(1+1)$ weight parameterization consists of a binary weight matrix $\mathbf{W}_B$, a binary group-assignment matrix $\mathbf{G}_B$, and two pairs of shared scaling and offset parameters (α_g, μ_g) for $g \in 0, 1$ corresponding to the two groups.

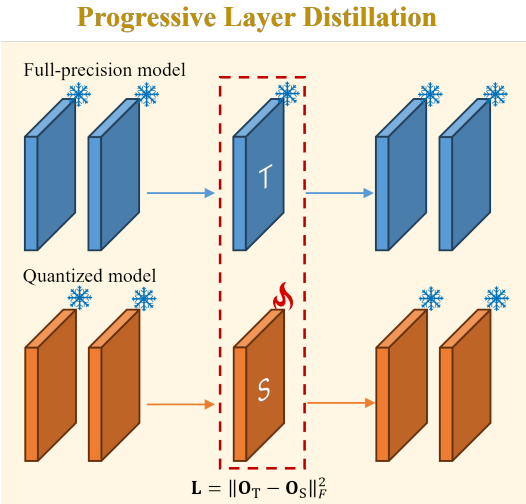

Figure 3: Illustration of the hierarchical distillation structure for LLMs, where each decoder layer is distilled progressively layer by layer.

To enable gradient-based optimization, the binary variables $\mathbf{W}_B$ and $\mathbf{G}_B$ are relaxed to full-precision counterparts $\mathbf{W}_{FP}$ and $\mathbf{G}_{FP}$ during training. A regularization term is applied to encourage these relaxed variables to gradually polarize toward binary values (0 or 1). In the forward pass, $\mathbf{W}_{FP}$ and $\mathbf{G}_{FP}$ are binarized via a clamp function to produce $\mathbf{W}_B$ and $\mathbf{G}_B$. For back-propagation, we employ the straight-through estimator (STE) to allow gradients to pass through the non-differentiable binarization step, ensuring stable and effective training of both weights and group assignments.

Specifically, the weight matrix is approximated as

$$\mathbf{W}_\text{q} = \mathbf{G}_B(\alpha_0 \mathbf{W}_B + \mu_0) + \neg\mathbf{G}_B(\alpha_1 \mathbf{W}_B + \mu_1), \tag{3}$$

where $\mathbf{G}_B$ denotes the group bitmap, and $\neg\mathbf{G}_B$ is its negation, ensuring only one group is chosen for each element. During training, the relaxed group identifier $\mathbf{G}_{FP}$ is encouraged to converge to either 0 or 1 via a regularization loss

$$\mathcal{L}_{reg} = \|\mathbf{I} - |2\mathbf{G}_{FP} - \mathbf{I}|^\beta\|_F^2, \mathbf{I} \in 1^{n \times m}, \tag{4}$$

where β is an annealing factor decreasing from 1 to 0. In the early training period, we encourage higher variability in the bitmap to explore a broader solution space. As training progresses, the value of β is gradually reduced, driving the bitmap values polarized to 0 or 1.

Learning Knee Point In AAR stage, we introduce a differentiable distribution-aware quantizer

to improve activation quantization with learnable knee points and clipping factors. However, direct region partitioning based on hard threshold blocks gradient flow during backpropagation, preventing the optimization of knee point positions via gradient descent. To overcome this non-differentiability bottleneck, we introduce a temperature-scaled soft masking mechanism as

$$\tilde{\mathbb{I}}_j(\mathbf{X}) = \text{sg}\left[\mathbb{I}(\mathbf{X} \in \mathcal{R}_j) - \pi_j(\mathbf{X};\tau)\right] + \pi_j(\mathbf{X};\tau). \tag{5}$$

During training, $\tilde{\mathbb{I}}_j(\mathbf{X})$ serves as the surrogate mask, where $\text{sg}[\cdot]$ denotes the stop-gradient operator, and $\pi_j(\mathbf{X};\tau)$ is the soft probability derived from the Sigmoid function with temperature τ. We retain discrete hard partitioning during forward propagation to ensure quantization efficiency, while utilizing soft masks for gradient computation during backpropagation.

Progressive Layer Distillation As shown in Figure 3, we formulate quantization as a layer-wise distillation process, treating the full-precision model as the teacher and the quantized model as the student (Shao et al., 2016; Chen et al., 2025). Transformer-based LLMs are composed of sequentially stacked decoder layers with identical structures; we regard each decoder layer as the minimal training unit and optimize them progressively from shallow to deep. The input to each layer is the output of the previously quantized layer, allowing downstream layers to perceive and compensate for quantization errors accumulated earlier. The reconstruction loss is defined as

$$\mathcal{L}_{rec} = \|\mathbf{O}_\text{T} - \mathbf{O}_\text{S}\|_F^2, \tag{6}$$

where $\mathbf{O}_\text{T}$ and $\mathbf{O}_\text{S}$ denote the outputs of Decoder-layer in the teacher and student models respectively. This progressive distillation strategy reduces peak memory usage and enables fine-grained optimization of quantization parameters and binarized weights within each layer.

The layer-wise distillation loss is given by $\mathcal{L} = \mathcal{L}_{rec} + \lambda\mathcal{L}_{reg}$, where λ is a hyperparameter used to balance the reconstruction loss and the regularization term.

4.3 Discusion

We compare LBLLM with earlier works on binarized PTQ and QAT approaches as well as the layer-wise training strategy.

Prior PTQ-based binarization methods (Li et al., 2024; Song et al., 2025) demonstrate efficient weight binarization using bitmap-based grouping but rely on auxiliary high-precision channels and do not compress activations. In contrast, LBLLM adopts a similar fine-grained grouping strategy yet removes auxiliary channels entirely, achieving higher compression and extending binarization to activations. Compared to QAT-based approaches (Wang et al., 2024; Ma et al., 2024), which achieve strong performance but require massive computational resources, LBLLM leverages PTQ-based initialization and a lightweight layer-wise QAT scheme, attaining comparable accuracy with drastically lower training cost.

Recent works have adopted layer-wise training for low-bit weight quantization (Chen et al., 2025; Ding et al., 2024) but did not realize the challenges of jointly quantizing activations, majorly focusing on the setting of W4A16 or W2A16. Unlike weight-only quantization, simultaneous weight–activation quantization is considerably harder as quantization errors from both sources interact and can destabilize optimization. Our experiments in Table 3 confirm that training a W(1+1)A4 model with a layer-wise strategy leads to unstable divergence. LBLLM decouples weight and activation quantization as separated two stages, which mitigates error interference and stabilizes training.

5 Experiments

5.1 Setup

In this work, we quantize all linear layer weights in the LLM into 1-bit Boolean matrices along with 1-bit fine-grained group bitmaps. Activations are dynamically and asymmetrically quantized to 4 bits. The group size is 128. For the KV cache, we ensure that it is quantized to the same bit-width as the activations. No additional high-precision parameters are retained in our quantized model. To ensure a consistent experimental setup, all training is conducted on a single 80GB GPU.

Models and Datasets We evaluate our method on the open-source LLaMA-1 (Touvron et al., 2023a), LLaMA-2 (Touvron et al., 2023b), and LLaMA-3 models. For training, we randomly sample 8,192 sequences (approximately 0.016B tokens) from the RedPajama corpus (Weber et al., 2024), which contains over 100B tokens. This small-scale subset highlights the data efficiency of

Method	Bits	Wiki	PTB	C4	PIQA	ARCe	ARCc	BoolQ	Hella.	Wino.	Avg
FP16	–	5.47	22.51	6.97	76.93	53.58	40.53	71.07	72.96	67.17	63.71
CBQ	W4A4	11.32	–	12.56	68.25	**46.23**	31.56	–	57.34	–	–
BiLLM	W(1+1)A4	32.48	3877.38	40.52	50.05	25.38	26.54	49.63	26.05	49.49	37.86
ARB-LLM	W(1+1)A4	19.14	165.46	20.88	63.82	39.31	27.05	60.21	44.94	54.54	48.31
BWA	W(1+1)A4	**8.89**	69.46	12.74	68.72	46.13	30.55	66.12	55.76	58.01	54.22
BWA w/o C	W(1+1)A4	243.73	5439.27	433.00	–	–	–	–	–	–	–
QuaRot	W2A4	49.98	571.22	80.14	54.41	28.45	23.21	57.89	28.57	48.15	40.11
LBLLM	W(1+1)A4	9.18	**36.79**	**10.91**	**70.97**	44.65	**32.51**	**64.50**	**58.54**	**58.27**	**54.91**

Table 1: Perplexity and zero-shot accuracy results of different quantization methods on LLAMA-2-7B. For CBQ, since the code is not publicly available, we directly adopt the quantization settings and results reported in the original paper, while all other methods use a consistent W2A4 configuration. "BWA w/o C" denotes the performance of the BWA method after removing the additional channels. The best results are highlighted in bold.

our approach while ensuring consistency across model sizes.

We evaluate performance across three categories: language generation, commonsense QA, and language understanding. For language generation, we report perplexity on WikiText-2 (Merity et al., 2016), PTB (Marcus et al., 1994), and C4 (Raffel et al., 2020); for C4, we randomly sample 256 sequences of length 2048 from the test set, while for other datasets we use the full test split. Commonsense QA is assessed via zero-shot accuracy on PIQA (Bisk et al., 2020), ARC (Clark et al., 2018), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), and WinoGrande (Sakaguchi et al., 2021). For language understanding, we evaluate on the MMLU benchmark (Hendrycks et al., 2020).

Baselines We compare LBLLM against existing weight-binarization methods such as BiLLM, ARB-LLM, and BWA (Huang et al., 2024; Li et al., 2024; Song et al., 2025); the weight and activation joint quantization method, QuaRot(Ashkboos et al., 2024), which employs rotation matrices; and CBQ(Ding et al., 2024), a lightweight training method based on progressive layer-wise distillation. As CBQ (Ding et al., 2024) does not provide open-source code, we used the results reported in the original paper under the W4A4 setting for comparison.

5.2 Main Results

Language Generation Tasks We evaluate the perplexity of the quantized models on multiple datasets. Our training set, RedPajama, includes portions of WikiText2 and C4 but excludes PTB. Therefore, perplexity evaluation on PTB serves as an indicator of the quantized model's overfitting tendency.

Table 1 presents the perplexity comparisons of various quantization methods on LLAMA-2-7B. It is evident that LBLLM achieves SOTA performance across all datasets and significantly outperforms a range of baseline methods. CBQ also employs layer-wise distillation training. Experiments demonstrate that our method achieves better performance in simultaneous weight and activation quantization. Specifically, LBLLM with W(1+1)A4 surpasses CBQ's results with W4A4.

The perplexity of LBLLM is slightly higher than that of BWA only on the WikiText2 dataset. However, BWA exhibits signs of overfitting on WikiText2 and employs a mixed-precision representation, which implies additional channel overhead. Compared to the setting of BWA without any additional channels, which is under the exactly same compression ratio and model structure, LBLLM demonstrates significantly superior performance.

Table 2 demonstrates the quantization performance of LBLLM on different versions of LLAMA models, further validating the generalization capability of our quantization approach.

Zero-Shot Common Sense QA Tasks We assess the effectiveness of our quantization method on several commonsense question answering tasks. Table 1 provides a comparison of accuracy across different quantization methods on LLAMA-2-7B. Table 2 presents the quantization performance of LBLLM across different models. The results show that LBLLM surpasses previous SOTA methods and further approaches the performance of the full-precision model.

Language Understanding Tasks We evaluate the language understanding ability of the LBLLM-quantized model on the MMLU benchmark. Fig-

Model	Bits	Wiki	PTB	C4	PIQA	ARCe	ARCc	BoolQ	Hella.	Wino.	Avg
LLAMA-1-7B	FP16	5.68	27.34	7.08	77.37	52.48	41.38	73.06	73.00	67.01	64.05
	W(1+1)A4	9.08	37.46	10.42	71.33	44.74	33.53	66.24	58.29	61.72	55.97
LLAMA-2-7B	FP16	5.47	22.51	6.97	76.93	53.58	40.53	71.07	72.96	67.17	63.71
	W(1+1)A4	9.18	36.79	10.91	70.97	44.65	32.51	64.50	58.54	58.27	54.91
LLAMA-3-8B	FP16	6.14	10.60	8.89	80.96	77.48	53.67	80.95	79.13	73.16	74.23
	W(1+1)A4	17.56	26.40	18.97	67.90	53.11	32.17	70.28	60.05	57.77	56.88

Table 2: Quantized perplexity and zero-shot accuracy results of LBLLM on different LLAMA models.

ure 1 presents the performance of the quantized model based on LLAMA-1-7B across various subdomains. Despite quantizing the full-precision model to an extremely low bit-width, it maintains a high level of language understanding performance.

5.3 Ablation Study

We conduct ablation studies on on LLAMA-2-7B with a unified quantization setting of W(1+1)A4 to test different training strategy, settings of trainable parameters, and initialization methods.

WAT	AAR	DT	Wiki↓	PTB↓	C4↓
✗	✗	✗	243.73	5439.27	433.00
✗	✗	✓	NAN	NAN	NAN
✓	✗	✗	9.40	42.86	11.08
✓	✓	✗	**9.18**	**36.79**	**10.91**

Table 3: Ablation study of different training stages. DT refers to jointly quantizing weights and activations followed by direct layer-wise distillation. NAN indicates that the PPL computed from the model output is excessively large.

Training Strategy To verify the effectiveness of our decoupled training strategy for weight binarization and synchronized activation quantization, Table 3 compares the average perplexity before and after each training stage. Experimental results show that the quantized model obtained after the fine-tuning on weights demonstrates a significant improvement over the initial quantized model. Although the activation-specific tuning requires low-bit quantization of activations, which leads to a performance drop compared to the first-stage result, the final performance after tuning confirms that our optimization mitigates the degradation caused by activation quantization errors.

Trainable Parameters To examine the influence of different parameter training strategies on the final model performance, we compare the impact of parameter inclusion/exclusion on the quantized model performance in Table 4 and Table 5. The experimental results confirm that global optimization across all parameters contributes to achieving a higher-quality quantized model.

α, μ	G	W	Wiki↓	PTB↓	C4↓	Avg.↓
✓	✗	✗	18.61	387.99	17.99	141.53
✗	✗	✓	15.16	1638.36	15.97	556.50
✓	✓	✓	**9.48**	**69.26**	**11.20**	**29.98**

Table 4: Ablation study on the impact of parameter participation in training each component of WAT on the final performance, all experimental results are conducted on LLAMA-2-7B under the W(1+1)A16 quantization setting.

α, μ	c, k	Wiki↓	PTB↓	C4↓	Avg. PPL↓
✗	✗	9.40	42.86	11.08	21.12
✓	✗	9.51	40.31	10.99	20.27
✗	✓	9.39	44.08	11.09	21.52
✓	✓	**9.18**	**36.79**	**10.91**	**18.96**

Table 5: Ablation study on the impact of parameter participation in training each component of AAR on the final performance.

Initialization	Wiki↓	PTB↓	C4↓	Avg. PPL↓
RTN	NAN	NAN	NAN	NAN
BWA	**9.48**	**69.26**	**11.20**	**29.98**

Table 6: Ablation study results using different initialization methods. NAN indicates that the PPL computed from the model output is excessively large.

Initialization Method Table 6 compares the effect of initializing the quantized model with or without PTQ-based methods. By contrasting the performance of models initialized via RTN quantization and subsequently fine-tuned, we demonstrate that better initialization schemes can improve model performance under our lightweight framework.

6 Limitations

Although LBLLM takes a significant step toward practical and efficient LLM deployment in resource-constrained environments, challenges remain in removing fine-grained grouping and further narrowing the gap with full-precision models. Empirically, the primary bottleneck in the W(1+1) configuration lies in the joint optimization of the 1-bit weight representation and the 1-bit grouping parameter. This process currently relies heavily on PTQ-derived priors for initialization, an area we intend to investigate further in future work. Moreover, current inference efficiency gains are predominantly software-based and yield marginal returns, often necessitating bespoke CUDA kernel implementations for different quantization schemes. True computational breakthroughs require dedicated hardware support, which remains largely in the research phase; we hope subsequent studies will bridge this gap.

References

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. 2024. Quarot: outlier-free 4-bit inference in rotated llms. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, pages 100213–100240.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Mengzhao Chen, Wenqi Shao, Peng Xu, Jiahao Wang, Peng Gao, Kaipeng Zhang, and Ping Luo. 2025. EfficientQAT: Efficient quantization-aware training for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pages 10081–10100.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of NAACL-HLT*, pages 2924–2936.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems*, 35:30318–30332.

Xin Ding, Xiaoyu Liu, Zhijun Tu, Yun Zhang, Wei Li, Jie Hu, Hanting Chen, Yehui Tang, Zhiwei Xiong, Baoqun Yin, and 1 others. 2024. Cbq: Cross-block quantization for large language models. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*.

Peijie Dong, Lujun Li, Yuedong Zhong, Dayou Du, Ruibo Fan, Yuhan Chen, Zhenheng Tang, Qiang Wang, Wei Xue, Yike Guo, and 1 others. 2024. Stbllm: Breaking the 1-bit barrier with structured binary llms. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Wei Huang, Yangdong Liu, Haotong Qin, Ying Li, Shiming Zhang, Xianglong Liu, Michele Magno, and Xiaojuan Qi. 2024. Billm: pushing the limit of post-training quantization for llms. In *Proceedings of the 41st International Conference on Machine Learning*, pages 20023–20042.

Hyesung Jeon, Yulhwa Kim, and Jae-Joon Kim. 2025. L4q: parameter efficient quantization-aware fine-tuning on large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2002–2024.

Eldar Kurtic, Alexandre Noll Marques, Shubhra Pandit, Mark Kurtz, and Dan Alistarh. 2025. "give me bf16 or give me death"? accuracy-performance trade-offs in llm quantization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 26872–26886.

Zhiteng Li, Xianglong Yan, Tianao Zhang, Haotong Qin, Dong Xie, Jiang Tian, Linghe Kong, Yulun Zhang, Xiaokang Yang, and 1 others. 2024. Arb-llm: Alternating refined binarizations for large language models. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*.

Haokun Lin, Haobo Xu, Yichen Wu, Jingzhi Cui, Yingtao Zhang, Linzhan Mou, Linqi Song, Zhenan Sun, and Ying Wei. 2024a. Duquant: Distributing outliers via dual transformation makes stronger quantized llms. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024b. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, pages 87–100.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. 2024. Spinquant–llm quantization with learned rotations. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*.

Liqun Ma, Mingjie Sun, and Zhiqiang Shen. 2024. Fbi-llm: Scaling up fully binarized llms from scratch via autoregressive distillation. *arXiv preprint arXiv:2407.07093*.

Mitch Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The penn treebank: Annotating predicate argument structure. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*.

Xu Ouyang, Tao Ge, Thomas Hartvigsen, Zhisong Zhang, Haitao Mi, and Dong Yu. 2024. Low-bit quantization favors undertrained llms: Scaling laws for quantized llms with 100t training tokens. *arXiv preprint arXiv:2411.17691*.

Jungwoo Park, Taewhoo Lee, Chanwoong Yoon, Hyeon Hwang, and Jaewoo Kang. 2025a. Outlier-safe pre-training for robust 4-bit quantization of large language models. *arXiv preprint arXiv:2506.19697*.

Seungcheol Park, Jeongin Bae, Beomseok Kwon, Minjun Kim, Byeongwook Kim, Se Jung Kwon, U Kang, and Dongsoo Lee. 2025b. Unifying uniform and binary-coding quantization for accurate compression of large language models. *arXiv preprint arXiv:2506.03781*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. 2016. Omniquant: Omnidirectionally calibrated quantization for large language models. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*.

Siqing Song, Chuang Wang, Rui-Qi Wang, Yi Yang, and Xu-Yao Zhang. 2025. Achieving binary weight and activation for llms using post-training quantization. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 8782–8795.

Yi Su, Yuechi Zhou, Quantong Qiu, Juntao Li, Qingrong Xia, Ping Li, Xinyu Duan, Zhefeng Wang, and Min Zhang. 2025. Accurate kv cache quantization with outlier tokens tracing. *arXiv preprint arXiv:2505.10938*.

Wei Tao, Haocheng Lu, Xiaoyang Qu, Bin Zhang, Kai Lu, Jiguang Wan, and Jianzong Wang. 2025. Moqae: Mixed-precision quantization for long-context llm inference via mixture of quantization-aware experts. *arXiv preprint arXiv:2506.07533*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Huaijie Wang, Lingxiao Ma, Fan Yang, Ruiping Wang, Yi Wu, and Furu Wei. 2023. Bitnet: Scaling 1-bit transformers for large language models. *arXiv preprint arXiv:2310.11453*.

Hongyu Wang, Shuming Ma, and Furu Wei. 2024. Bitnet a4. 8: 4-bit activations for 1-bit llms. *arXiv preprint arXiv:2411.04965*.

Maurice Weber, Daniel Y. Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, Ben Athiwaratkun, Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Ré, Irina Rish, and Ce Zhang. 2024. Redpajama: an open dataset for training large language models. *NeurIPS Datasets and Benchmarks Track*.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR.

Yuzhuang Xu, Xu Han, Zonghan Yang, Shuo Wang, Qingfu Zhu, Zhiyuan Liu, Weidong Liu, and Wanxiang Che. 2024. Onebit: Towards extremely low-bit large language models. *Advances in Neural Information Processing Systems*, 37:66357–66382.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Jiaqi Zhao, Miao Zhang, Ming Wang, Yuzhang Shang, Kaihao Zhang, Weili Guan, Yaowei Wang, and Min Zhang. 2025. Ptq1. 61: Push the real limit of extremely low-bit post-training quantization methods for large language models. *arXiv preprint arXiv:2502.13179*.

Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. 2024. Atom: Low-bit quantization for efficient and accurate llm serving. *Proceedings of Machine Learning and Systems*, 6:196–209.

A Difficulties in Joint Weight and Activation Quantization

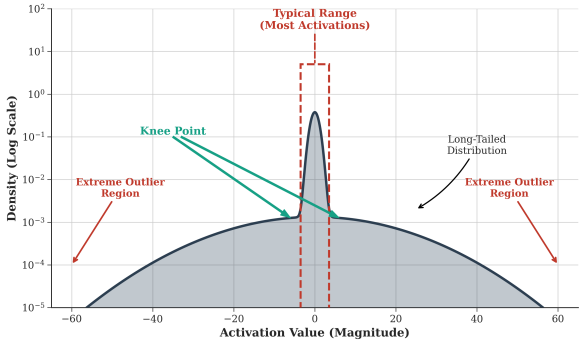

Figure 4: Illustration of activation distribution. The "Knee Point" marks the critical transition boundary where the density shifts significantly, separating the primary bell-shaped distribution from the sparse but magnitude-heavy outliers.

Activation Distribution As illustrated in Figure 4, activation values exhibit a pronounced long-tailed distribution. By employing a logarithmic scale, we visualize the distinct 'Knee Point,' which demarcates the typical range from the outlier regions. While the probability density of these outliers is minimal, their extreme magnitudes significantly expand the dynamic range, thereby degrading the quantization resolution for the majority of activations.

Error Coupling between Weights and Activations We observe that in joint weight and activation quantization, quantization errors are not merely additive. Instead, they are continuously coupled and accumulate during layer-wise propagation, ultimately precipitating rapid model collapse. As shown in Figure 5, there is a significant gap between quantization only quantization and joint weight & activation quantization. So we decouple the weight quantization and activation quantization to accomplish our goal.

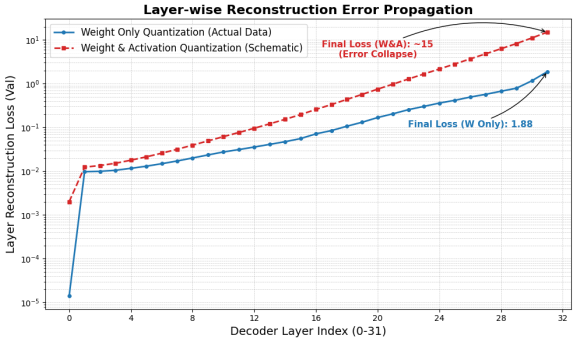

Figure 5: Layer-wise reconstruction error.

Method	Tokens	GPUs	GPU hours
FBI-LLM	108.5B	32	22599h
LBLLM	0.016B	1	20h

Table 7: Comparison of computational resource consumption between LBLLM and binarized QAT methods.

B Resource Requirements Analysis

Training Dataset Size QAT-based methods such as BitNet and FBI-LLM argue that binarization leads to significant loss of pretrained knowledge in LLMs. As a result, their approach is to train a binarized LLM from scratch using large-scale corpora. In contrast, LBLLM demonstrates that a combination of PTQ and lightweight QAT can produce a binarized LLM with comparable performance. Table 7 compares the data requirements of QAT and our method. As shown, LBLLM requires significantly fewer training resources than QAT methods. In the supplementary material, we compare the performance of LBLLM with several QAT-based quantization methods, showing that LBLLM achieves comparable results across various metrics.

Inference Memory Requirement LBLLM adopts a minimal quantization structure, without relying on additional channels to compensate for quantization errors. In Table 8, we report the inference-time memory usage of the quantized

Model	FP16	LBLLM
LLAMA-7B	13.5GB	1.8GB
LLAMA-13B	24.2GB	3.2GB
LLAMA-30B	60.5GB	8.0GB
LLAMA-65B	121.0GB	16.1GB

Table 8: Inference memory compression of LBLLM on models with different parameter sizes.

	LLAMA-1-7B	LLAMA-2-7B	LLAMA-3-8B
WAR Size	8192	8192	2048
WAR Epoch	2	2	5
AAR Size	8192	8192	8192
AAR Epoch	1	1	1
W Learning Rate	2e-5	2e-5	2e-5
G Learning Rate	1e-4	1e-4	1e-4
α, μ Learning Rate	1e-4/1e-5	1e-4/1e-5	1e-4/1e-5
Clipping Ratio Learning Rate	1e-4	1e-4	1e-4
Knee Point Learning Rate	5e-4	5e-4	5e-4
GPUs	1	1	1
GPU Hours	22	22	17

Table 9: The configuration and training details for LBLLM. Since the learning rates for α and μ differ between the WAR and AAR stages, we report both values in the table.

Weight Shape	LBLLM	INT8	INT4
4096×4096	14ms	43ms	80ms
11008×4096	27ms	83ms	158ms
4096×11008	28ms	101ms	197ms

Table 10: Martix multiplication Speedup between LBLLM (W2A4) and the INT4, INT8 kernels of CUTLASS.

models. Compared to full-precision models, LBLLM achieves an average compression ratio of over 7×, maintaining the lowest memory footprint among similar methods.

Inference Speedup LBLLM maintains a weight quantization configuration consistent with BWA and ARB-LLM, allowing it to leverage established CUDA kernels for inference acceleration. Although we introduce knee points for activations, this is implemented via vectorized masking with negligible computational overhead, ensuring the subsequent low-bit matrix multiplication remains identical to prior work. Table 10 demonstrates the matrix computation acceleration achieved using these CUDA kernels.

C More Details about Our Experiment Settings

This section provides additional experimental details required by the checklist but not covered in the main text. It includes information on datasets, evaluation metrics, hyperparameter settings, and computational setup used in our quantization experiments, aiming to support the reproducibility of this work.

C.1 Evaluation Metrics

Memory Usage of The Quantized Model This section explains how we compute the memory usage of the quantized model. Given the mainstream transformer architecture used in current LLMs, our work quantizes the linear layers—which account for over 90% of the total computation—while keeping components with relatively minor computational cost, such as embedding and activation layers, in full precision. This setting is consistent across all existing works on large language model quantization. In our memory usage calculation, we only consider the quantized linear layers. The final memory compression ratio is as follows

$$\mathrm{M}_q = \frac{\mathrm{Bits}_q + \mathrm{Bits}_g + \mathrm{Bits}_p}{\mathrm{Bits}_{fp}} \times \mathrm{M}_{fp} \qquad (7)$$

Here, m_q and m_{fp} denote the storage sizes of the quantized and full-precision models, respectively. Bits_q, Bits_g, Bits_p, and Bits_{fp} represent the effective bit-widths of quantized weights, group-wise bitmaps, quantization parameters, and full-precision weights. For example, the calculation of Bits_p includes the combined effective bit-width of the scaling factor α and the zero point μ, where each group shares one α in 16-bit format and one μ in 1-bit format. Therefore, while the group size is 128, $\mathrm{Bits}_p = \frac{16+1}{\text{group size}} \approx 0.148$.

Perplexity Perplexity measures how well a probability distribution or model predicts a sample. In

Bits	Method	STEM↑	Humanities↑	Social Science↑	Others↑	Average↑
FP16	-	36.1	43.3	51.6	51.8	45.5
W(1+1)A4	Atom	25.9	24.9	24.0	26.3	25.3
W(1+1)A4	ARB-LLM	30.2	25.4	30.5	26.0	27.7
W(1+1)A4	LBLLM	**26.4**	**28.8**	**27.7**	**32.0**	**28.8**

Table 11: MMLU results (%) under the W2A4 settings on LLAMA-1-7B. Our quantization performance highlighted in bold.

Method	Bits	Wiki	PTB	C4	PIQA	ARCe	ARCc	BoolQ	Hella.	Wino.	Avg.
FP16	–	5.47	22.51	6.97	76.93	53.58	40.53	71.07	72.96	67.17	63.71
Onebit	W1A16	10.19	–	11.40	68.01	42.47	30.20	57.28	51.54	58.48	51.33
FBI-LLM	W1A16	9.10	29.60	10.50	72.60	53.00	29.90	61.50	57.70	58.90	55.60
LBLLM	W(1+1)A4	**9.18**	**36.79**	**10.91**	**70.97**	**44.65**	**32.51**	**64.50**	**58.54**	**58.27**	**54.91**

Table 12: Perplexity and zero-shot accuracy results of different quantization methods on LLAMA-2-7B.

language modeling, it reflects how closely the predicted next token matches the ground truth—the more accurate the prediction, the lower the perplexity.

Commonsense QA CommonsenseQA is a multiple-choice question answering dataset that requires diverse types of commonsense knowledge to identify the correct answers. We evaluate on PIQA, ARC, BoolQ, HellaSwag, and WinoGrande, which cover physical commonsense reasoning, middle school-level science questions, natural language inference, and general commonsense reasoning. All evaluations are conducted in a zero-shot setting.

Massive Multitask Language Understanding MMLU is a challenging benchmark designed to evaluate the knowledge acquired during pretraining by testing models exclusively in zero-shot and few-shot settings. It covers 57 subjects across diverse domains, including STEM, humanities, and social sciences, with question difficulty ranging from high school to advanced professional levels. The benchmark assesses both factual knowledge and problem-solving ability, spanning topics from traditional areas like mathematics and history to specialized fields such as law and ethics. The breadth and granularity of the tasks make MMLU an ideal tool for identifying blind spots in language models.

C.2 Hyperparameters

In Table 9, we present the training hyperparameter settings and more training details. Due to time constraints, we used only 2,048 samples for training LLAMA-3-8B, and most hyperparameters were chosen based on empirical heuristics rather than thorough tuning. We leave comprehensive hyperparameter optimization to future work.

C.3 Computational Setup

Most of our experiments were conducted on NVIDIA H100 80GB with CUDA version 12.8; more detailed environment configurations will be provided in the upcoming open-source repository.

D Additional Experimental Results

D.1 Language Understading Tasks

Table 11 presents a comparison of different quantization methods on the MMLU dataset. The results show that our method achieves superior overall language understanding performance and yields results closer to the full-precision model.

D.2 Compared with QAT Method

We compare our results with two binarized QAT methods, Onebit and FBI-LLM; due to the high computational requirements of these approaches, we directly adopt the settings and results reported in their papers. Across various metrics, our method outperforms binarized QAT approaches in many aspects and achieves comparable performance on the remaining ones, demonstrating its strong potential.

D.3 Larger Model Results

We further evaluated LBLLM on LLaMA models with varying parameter sizes. The results in Table 13 demonstrate that even on relatively larger models (e.g., LLaMA-13B), LBLLM achieves robust quantization performance through lightweight training. Furthermore, LBLLM exhibits a narrower performance gap compared to the full-precision

Model	Bits	Wiki	PTB	C4	PIQA	ARCe	ARCc	BoolQ	Hella.	Wino.	Avg
LLAMA-1-7B	FP16	5.68	27.34	7.08	77.37	52.48	41.38	73.06	73.00	67.01	64.05
	W(1+1)A4	9.08	37.46	10.42	71.33	44.74	33.53	66.24	58.29	61.72	55.97
LLAMA-1-13B	FP16	5.09	19.23	6.61	79.05	59.89	44.71	68.47	76.23	70.24	66.43
	W(1+1)A4	7.90	38.19	9.23	74.05	50.21	34.90	65.81	64.49	63.85	58.88

Table 13: Quantized perplexity and zero-shot accuracy results of LBLLM on larger LLAMA models.

baseline on these larger models. Due to computational resource constraints, evaluations on larger-scale models are reserved for future work.