BiLLM: Pushing the Limit of Post-Training Quantization for LLMs

Wei Huang [1] **Yangdong Liu** [2] **Haotong Qin**✉ [3] **Ying Li** [2] **Shiming Zhang** [1]
Xianglong Liu [2] **Michele Magno** [3] **Xiaojuan Qi** [1]

Abstract

Pretrained large language models (LLMs) exhibit exceptional general language processing capabilities but come with significant demands on memory and computational resources. As a powerful compression technology, binarization can extremely reduce model weights to a mere 1 bit, lowering the expensive computation and memory requirements. However, existing quantization techniques fall short of maintaining LLM performance under ultra-low bit-widths. In response to this challenge, we present BiLLM, a groundbreaking 1-bit post-training quantization scheme tailored for pretrained LLMs. Based on the weight distribution of LLMs, BiLLM first identifies and structurally selects salient weights, and minimizes the compression loss through an effective *binary residual approximation* strategy. Moreover, considering the bell-shaped distribution of the non-salient weights, we propose an *optimal splitting search* to group and binarize them accurately. BiLLM, for the first time, achieves high-accuracy inference (e.g. 8.41 perplexity on LLaMA2-70B) with only **1.08-bit** weights across various LLM families and evaluation metrics, outperforms SOTA quantization methods of LLM by significant margins. Moreover, BiLLM enables the binarization process of a 7-billion LLM within 0.5 hours on a single GPU, demonstrating satisfactory time efficiency. Our code is available at https://github.com/Aaronhuang-778/BiLLM.

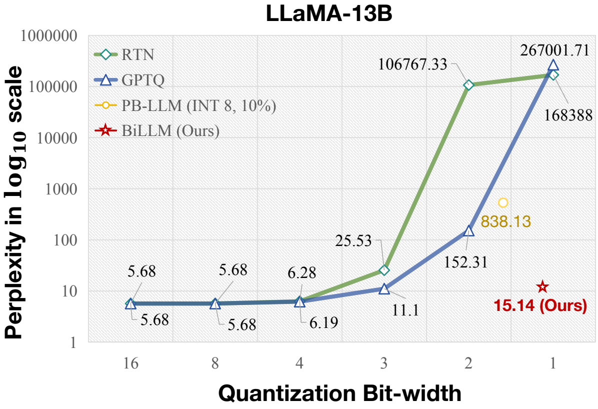

Figure 1: Perplexity of LLaMA-13B on WikiText2 under different bit-widths. Round-to-nearest (RTN), GPTQ, and PB-LLM (10% weight of INT8) suffer accuracy loss at ultra-low bits, facing the sharply increasing perplexity (↓). *BiLLM* demonstrates exceptional performance under binarization.

1. Introduction

Recently, large language models (LLMs) based on transformers (Vaswani et al., 2017) have garnered significant attention in natural language processing. Pre-trained LLMs such as OPT (Zhang et al., 2022) and LLaMA (Touvron et al., 2023a), have demonstrated excellent performance across a range of evaluation benchmarks. However, LLMs pose substantial challenges in deployment on memory-constrained devices due to their immense parameter size and computation requirements. For instance, the widely-used LLaMA2-70B (Touvron et al., 2023b) model, with its 70 billion parameters, requires 150 GB of storage in half-precision (FP16) format. This necessitates at least two A100 GPUs, each with 80 GB of storage space, for inference.

Model quantization has emerged as a highly effective technology for compressing neural networks, thereby reducing the model size of LLMs and substantially saving GPU memory consumption (Dettmers et al., 2022). Current quantization techniques primarily fall into Quantization-Aware Training (QAT) and Post-Training Quantization (PTQ). QAT involves fine-tuning and retraining during the quantization process, while PTQ significantly streamlines the computation by eliminating back-propagation, enabling a faster quantization process and promoting the practicality of quantization (Frantar et al., 2022; Shang et al., 2023; Lin et al., 2023). Given the deep structures and numerous parameters of LLMs, PTQ stands out for its ability to rapidly perform

[1] The University of Hong Kong [2] Beihang University [3] ETH Zürich. Correspondence to: Haotong Qin <haotong.qin@pbl.ee.ethz.ch>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

the quantization process, especially on time and resource-constrained scenarios (Zhu et al., 2023).

Despite the success of previous PTQ methods in 8-bit and 4-bit quantization (Dettmers et al., 2022; 2023b; Frantar et al., 2022; Xiao et al., 2023; Frantar & Alistarh, 2022), the expanding size of LLMs demands more aggressive quantization approaches (Shang et al., 2023). Neural network binarization, which reduces the weight bit-width to only 1 bit, is a promising approach (Helwegen et al., 2019; Qin et al., 2020; 2023). However, as depicted in Figure 1, current advanced PTQ methods for LLMs exhibit a performance collapse under ultra-low bit ($\leqslant$3 bits) quantization. This phenomenon can be attributed to the significant difference between quantized and original weights. Even the recent binary PTQ method for LLMs, PB-LLM (Shang et al., 2023), only maintains a perplexity metric of around 800 with an average weight of 1.7 bits. This observation underscores the challenges existing PTQ methods face in promoting the weight binarization of LLMs.

In pursuit of this goal, we conducted an empirical study to analyze the distribution of pre-trained weights in LLMs. The findings derived from this study are presented in Appendix G, revealing two key observations:

- The second-order Hessian matrix of weights demonstrates an **exceptionally long-tail distribution** and is often used to measure the importance of weight elements in neural networks (LeCun et al., 1989; Dong et al., 2019). As depicted in Figure 2, a small fraction of weights elements possesses significantly high Hessian values, substantially influencing the layer output. In contrast, most Hessian values cluster around 0.
- The density distribution of weight magnitudes in LLMs follows a **bell-shaped pattern**. This bell-shaped distribution exhibits a significant resemblance to both the Gaussian or Laplace distribution in terms of its characteristics (Blundell et al., 2015). Figure 2 illustrates that most weight values cluster around zero with a non-uniform bell-shaped distribution.

The above implies: a) A minority of weights play an important role in LLMs, whereas the majority of weights exhibit characteristics of redundancy (Shang et al., 2023; Dettmers et al., 2023b); b) With the most aggressive bit-width, binarization incurs most severe error among quantization under bell-shaped distributions in LLMs (Jacob et al., 2018).

Motivated by the above observation, we propose a novel 1-bit PTQ framework for LLMs, namely *BiLLM*, incorporating two core designs to achieve highly accurate weight binarization. First, guided by the Hessian-based metric, we select the salient weights structurally (Figure 3 upper-right) to achieve a trade-off between accuracy and storage savings and develop a residual approximation to maximize the restoration of salient weights with highly dynamic range. Second, for the remaining non-salient weights (Figure 3 lower-right), we design an optimal splitting binarization strategy, where a meticulous search process is applied to determine an optimal break-point for weight distribution and binarization of the segments is then processed separately to minimize binarization errors. Moreover, *BiLLM* incorporates error compensation on a block-wise basis by default following existing common practices (Frantar et al., 2022; Shang et al., 2023), which further reduces quantization error.

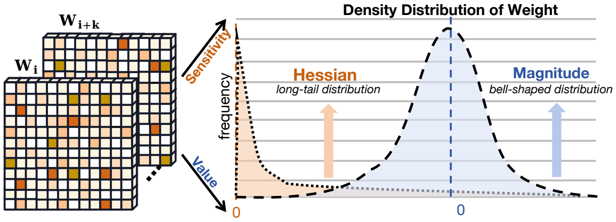

Figure 2: The Hessian metrics (sensitivity) and magnitude (value) of weights in LLMs. The weights of different layers in LLMs are characterized by bell-shaped distribution, accompanied by a few salient values.

Extensive experiments demonstrate that *BiLLM* achieve the state-of-the-art (SOTA) performance for LLMs across multiple LLM families on various evaluation metrics, and first achieves extremely compact 1.07$\sim$1.11 bit-width in average for the PTQ binarization. For example, on the Wikitext2(Merity et al., 2016) metric, *BiLLM* achieved perplexities of 8.49 and 8.41 with only 1.08-bit weights on LLaMA-65B (Touvron et al., 2023a)and LLaMA2-70B (Touvron et al., 2023b), respectively, even surpassing the 9.34 performance of the FP16 OPT-66B (Zhang et al., 2022).

2. Related Works

2.1. Large Language Model Quantization

Quantization maps high-precision parameters to a discrete range. This method, which compresses parameters without altering the model structure, effectively reduces the storage and computational overhead of deep neural networks. Recent work has successfully applied QAT and PTQ to LLMs. QAT, through a quantization-aware retraining strategy, better preserves the performance of quantized models. LLM-QAT (Liu et al., 2023) addressed data barrier issues in QAT training through data-free distillation. However, for LLMs with extremely large parameter sizes, the cost of retraining is prohibitively high and inefficient. Therefore, techniques such as QLoRA (Dettmers et al., 2023a) focus on parameter-efficient fine-tuning (PEFT) methods for quantizing LLMs, enhancing the efficiency of QAT. Nevertheless, even these

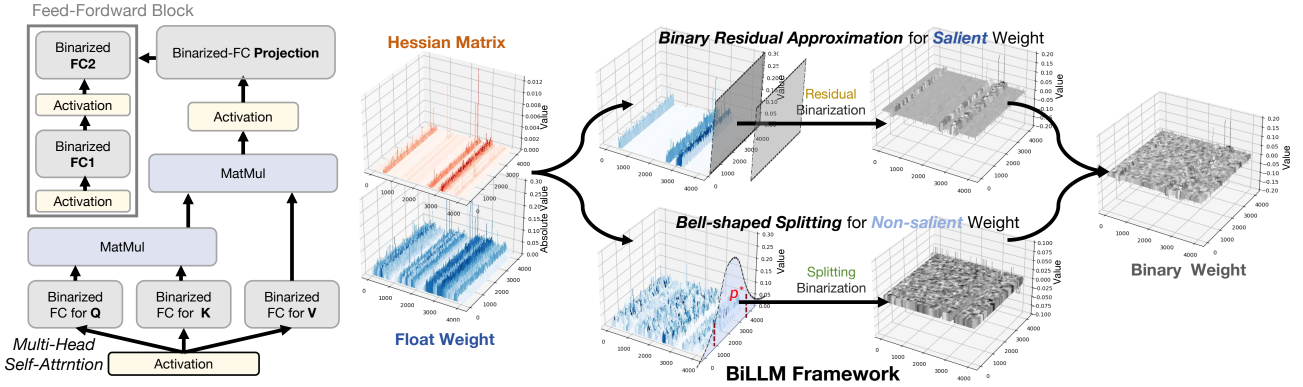

Figure 3: Schematic of the PTQ binarization framework for LLMs. The left side shows the structure of the Transformer block after binarization. The right side shows the binarization process of *BiLLM*, which consists of two parts, *Residual Approximation* for salient weights and *Bell-shaped Splitting* for non-salient weights.

efficient fine-tuning quantization strategies require over 24 hours of GPU time.

Therefore, the PTQ strategy has become a significant option for quantizing LLMs efficiently. Works like BRECQ (Li et al., 2021), ZerqQuant (Yao et al.) and LLM.int8() (Dettmers et al., 2022) enhance quantization accuracy by adding additional grouping labels for custom quantization blocks. Other studies adopt a feature segmentation strategy, such as PB-LLM (Shang et al., 2023) and SpQR (Dettmers et al., 2023b). They preserve the bit-width of outlier features or those with higher quantization errors to FP16 or INT8, mitigating the precision loss due to quantization. GPTQ (Frantar et al., 2022) employs a more precise quantization framework, reducing the block quantization errors of LLMs through Hessian-based second-order error compensation (Frantar & Alistarh, 2022), achieving commendable performance in low-bits (4 bits) quantization. Smoothquant (Xiao et al., 2023) introduces a strategy of scaling weight and activation outliers to simplify quantization. Subsequently, AWQ (Lin et al., 2023) and OWQ (Lee et al., 2023) also proposed scale transformations of more crucial weight channels for activation features, preserving their information representation capacity.

2.2. Network Binarization

Binarized compression can quantize parameters to only 1 bit, expressed as ±1. In forward propagation, the sign function is used to binarize the original parameter tensor:

$$\mathbf{W}_b = \alpha \cdot \mathrm{sign}(\mathbf{W}_f), \tag{1}$$

$$\mathrm{sign}(x) = \begin{cases} 1 & \text{if } x \geq 0, \\ -1 & \text{others.} \end{cases} \tag{2}$$

where $\mathbf{W}_f \in \mathbb{R}^{n \times m}$ is the full precision weight and $\mathbf{W}_b \in \mathbb{R}^{n \times m}$ is the binarized output. n and m represent the size of the weight matrix. α denotes the scaling factor (Courbariaux et al., 2016). Binarization usually uses the channel-wise scale (Rastegari et al., 2016; Qin et al., 2023), so $\alpha \in \mathbb{R}^n$.

Most previous binarization works adopt a framework based on QAT for quantization (Qin et al., 2023). Straight through estimator (STE) (Bengio et al., 2013) is deployed to address the issue of gradient vanishing caused by the $\mathrm{sign}(\cdot)$ function. Binary Weight Network (BWN) (Rastegari et al., 2016) was initially proposed for executing neural network computations by binarizing weights and using full-precision activations, while XNOR-Net (Rastegari et al., 2016) extends this approach by binarizing both weights and activations. Both methods minimize quantization errors through dynamic searching of α. DoReFa-Net (Zhou et al., 2016) further expands upon XNOR-Net, employing quantized gradients to accelerate network training. Group segmentation is also applied in binarization tasks, with Syq (Faraone et al., 2018) utilizing network weight to the small size of groups for minimizing binarization errors.

Based on the successful application of binarization in Transformers (Wang et al., 2023) and Bert (Qin et al., 2022), we believe that the binarization of LLMs is filled with potential. PB-LLM (Shang et al., 2023) investigates the impact of binarized QAT and PTQ strategies on LLMs, but it is necessary to retain a significant proportion (over 30%) of the weights at 8 bits to enable LLMs to produce reasonable answers. Due to the presence of a large amount of INT8, LLMs still have a relatively high average bit-width. To address this issue, we proposed *BiLLM*, which aims to push the limit of PTQ binarization for LLMs.

3. Method

To achieve accurate binarization of LLMs, our approach is designing distinct binarization strategies for salient and non-salient weights. We first introduce the selection rules for salient weights and their binarization strategies in Section 3.1. Then, we elaborate on the distribution-based binarization strategy for non-salient weights in Section 3.2.

3.1. Salient Weight Binarization for LLMs

In deep neural networks, not all parameters carry equal significance. Utilizing solely the magnitude of the weights can not fully capture the impact of each element on the model's performance. The Hessian metric serves as a common benchmark for detecting parameter sensitivity (Dong et al., 2019; Dettmers et al., 2023b; 2022). We thus leverage the Hessian matrix to assess the salience of parameters in each under-binarized layer. We implement an optimized computation process to derive weight sensitivity, which allows us to obtain the importance metric of parameters without compromising efficiency:

$$s_i = \frac{w_i^2}{[\mathbf{H}^{-1}]_{ii}^2}, \tag{3}$$

where $\mathbf{H}$ represents the Hessian matrix of each layer, and w_i represents the original value of each element. In the following section, s_i serves as a criterion for assessing the significance of weight elements and is used as a feature indicator for structured selection.

Structural Searching Selection. Utilizing an unstructured selection enables the coverage of all salient elements. However, it requires the implementation of an additional 1-bit bitmap index (Chan & Ioannidis, 1998), leading to increased average bit-width. This balance is inefficient, especially for Hessian outlier weights that constitute a mere 1-5% of the total (Yao et al., 2023). In our analysis of sensitivity distribution within LLMs, we discovered that the majority of the weights' sensitive Hessian values are predominantly concentrated in specific columns or rows (Appendix G). This pattern is attributed to the convergence effects inherent in the multi-head self-attention mechanism of these models and further motivates us to implement a structured approach for selecting salient weights, for reducing the additional bitmap. Given that *BiLLM* employs a per-channel (or per-row) type of binarization, we determine salience through a per-column segmentation on the whole weight matrix.

We organize the column salience in descending order and introduce an optimized search algorithm aimed at minimizing quantization error, which in turn determines the number of columns within the salient group. To elaborate on this methodology, we initially define the objective of binarization quantization, grounded on Equation (1):

$$\arg\min_{\alpha,\mathbf{B}} ||\mathbf{W} - \alpha\mathbf{B}||^2, \tag{4}$$

where $\mathbf{B} \in \{-1, +1\}^{n\times k}$, k is the number of selected columns. The problem (Rastegari et al., 2016) of optimal

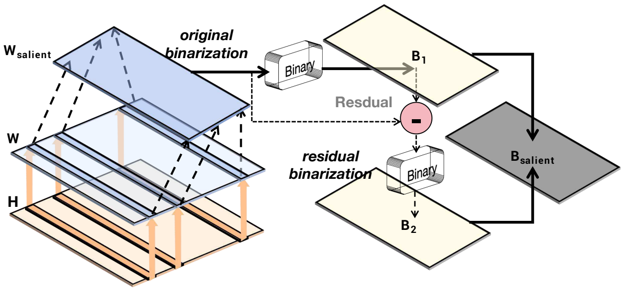

Figure 4: Illustration of salient weight binarization. The $\mathbf{B}_1$ binarized from salient weight is made into a residual with the original value and then binarized again to obtain $\mathbf{B}_2$.

α and $\mathbf{B}$ can simply be solved as $\alpha = \frac{||\mathbf{W}||_{\ell 1}}{n\times k}$ and $\mathbf{B} = \text{sign}(\mathbf{W})$. Then, the optimization function for selecting salient columns is defined as:

$$\arg\min_{\mathbf{W}_{\text{uns}}} ||\mathbf{W} - (\alpha_{\text{sal}}\,\text{sign}(\mathbf{W}_{\text{sal}}) \cup \alpha_{\text{uns}}\,\text{sign}(\mathbf{W}_{\text{uns}}))||^2, \tag{5}$$

where $\mathbf{W}_{\text{sal}}$ denotes the column-wise combination of original weight and $\mathbf{W}_{\text{uns}}$ is the left non-salient part. We can easily get that $\mathbf{W} = \mathbf{W}_{\text{sal}} \cup \mathbf{W}_{\text{uns}}$, so the only variable parameter is the number of rows in $\mathbf{W}_{\text{sal}}$.

Binary Residual Approximation. Salient weights are limited in quantity, yet exhibit significant variance when aggregated. Direct preservation of these weights in INT8 or FP16 formats leads to an increase in the average weight bits, undermining the compressive benefits of binarization. Traditional binarization methods for salient weights, however, result in substantial quantization errors. To that end, we develop a residual approximation approach for binarizing salient weights. Contrary to the comprehensive high-order quantization (Li et al., 2017) applied to the entire weight matrix, our technique minimizes binarization error through a second-order approximation of merely a select subset of salient weights. This method guarantees the precision of salient weights while simultaneously decreasing bit-width overhead. As illustrated in Figure 4, this approach incorporates a recursive computation strategy for weight binarization compensation, applying a subsequent binarization process to the residuals remaining after the initial binary process. Building upon Equation (4), we propose a redesigned residual approximation optimization specifically for salient weights, which is defined as follows:

$$\begin{cases} \alpha_o^*, \mathbf{B}_o^* = \arg\min\limits_{\alpha_o,\mathbf{B}_o} ||\mathbf{W} - \alpha_o\mathbf{B}_o||^2), \\ \alpha_r^*, \mathbf{B}_r^* = \arg\min\limits_{\alpha_r,\mathbf{B}_r} ||(\mathbf{W} - \alpha_o^*\mathbf{B}_o^*) - \alpha_r\mathbf{B}_r||^2), \end{cases} \tag{6}$$

where $\mathbf{B}_o$ represents the original binary tensor, while $\mathbf{B}_r$ denotes the residual binarized matrix with the same size as $\mathbf{B}_o$. We efficiently solve for the two binarized optimization objectives using the same solution method as in Equation (4).

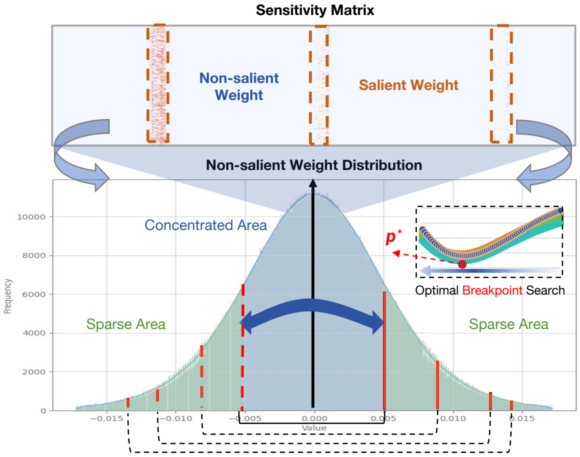

Figure 5: Distribution and splitting schematic of the 4^{th} projection layer in LLaMA2-7B. The top 5% of the Hessian elements are orange, and the optimal break-point divides the non-salient weights into sparse and concentrated areas.

Ultimately, we arrive at the following approximation:

$$\mathbf{W} \approx \alpha_o^* \mathbf{B}_o^* + \alpha_r^* \mathbf{B}_r^*. \tag{7}$$

It can be easily proven that the residual approach of Equation (7) has a lower quantization error than the direct one of Equation (4). We define the residual binarization error $\mathcal{E}$:

$$\mathcal{E}_{rb} = ||\mathbf{W} - \alpha_o^* \mathbf{B}_o^* - \alpha_r^* \mathbf{B}_r^*||^2. \tag{8}$$

The original binarized quantization error is calculatde as $||\mathbf{W} - \alpha_o^* \mathbf{B}_o^*||^2$ by Equation (4), and from the second sub-equation of Equation (6) we can get that loss $\mathcal{E}_{rb} \leq ||\mathbf{W} - \alpha_o^* \mathbf{B}_o^*||^2$. Therefore, through the method of residual approximation, we are able to further reduce the binary quantization error of salient weights with ultra-low bit-width storage compared to retaining salient weights at 8 or 16 bits.

3.2. Bell-shaped Distribution Splitting for Binarization

Following the removal of salient weights, the remaining weights maintain a bell-shaped distribution, which becomes closer to symmetric with the exclusion of salient weights' impact, as depicted in Figure 5. Binary quantization, representing an extreme form of uniform quantization, encounters more loss in the presence of non-uniform distributions. A practical approach involves the group-wise quantization (Park et al., 2018; Fang et al., 2020; Jain et al., 2019) of weights according to their distribution. Balancing between quantization accuracy and compression efficiency, we identify a single break-point within the distribution. As shown in Figure 5, this partition divides the non-salient bell-shaped distribution into two categories: the sparse area and the concentrated area.

The segmentation process identifies a break-point that categorizes non-salient weights into two groups: $A_c[-p, p]$ for concentrated weights and $A_s[-m, -p] \cup [p, m]$ for sparse weights, where signifies the maximum extent of non-salient weights. We then apply binarization to both A_c (concentrated) and A_s (sparse). To determine the optimal break-point p^*, we assume that the non-salient weights possess a symmetrical probability density function (PDF)-$g(x)$ over the bounded domain $[-m, m]$, with the properties $g(x) = g(-x)$. Then the mean squared quantization error of binarization is defined as:

$$\theta_q^2 = \int_{-m}^{0} (-\alpha - x)^2 g(x) dx + \int_{0}^{m} (\alpha - x)^2 g(x) dx. \tag{9}$$

Since $g(x)$ is a symmetric function, the above formula is simplified to:

$$\theta_q^2 = 2 \int_{0}^{m} (\alpha - x)^2 g(x) dx. \tag{10}$$

Then, the break-point p divides the non-salient weights into two parts. According to the Equation (10), under the discontinuous weight distribution, we get a new binary quantization error:

$$\theta_{q,p}^2 = ||\mathbf{W}_s - \alpha_s \mathbf{B}_s||^2 + ||\mathbf{W}_c - \alpha_c \mathbf{B}_c||^2, \tag{11}$$

where $\mathbf{W}_s$ and $\mathbf{W}_c$ denote the weights of the sparse and concentrated area, respectively. $\mathbf{B}_s$ and $\mathbf{B}_c$ were calculated from Equation (2), α_s and α_c are the binarization scales, determined by Equation (4):

$$\alpha_s = \frac{1}{n_s} ||\mathbf{W}_s||_{\ell 1}, \alpha_c = \frac{1}{n_c} ||\mathbf{W}_c||_{\ell 1}, \tag{12}$$

where n represents the number of weight elements in each area. Therefore, the problem function is only related to p, and our target to find the optimal p^* can be defined as:

$$p^* = \arg\min_{p} (\theta_{q,p}^2). \tag{13}$$

When the remaining weights follow an ideal Gaussian distribution, Equation (11) is demonstrated to be a convex function with a global minimum, as evidenced in prior studies (Fang et al., 2020; You, 2010). Nonetheless, the actual distribution of non-salient weights, while bell-shaped, diverges from the ideal Gaussian model. Simultaneously, we retain the block-wise compensation strategies of GPTQ (Frantar et al., 2022) and OBC (Frantar & Alistarh, 2022) to offset quantization errors, which could change the distribution of weights. In response, we employ a percentile search method to identify the optimal break-point based on the objective function outlined in Equation (13). This percentile search strategy is efficient and straightforward, completing the binarization process for a 7B LLM within merely 30 minutes. Furthermore, our findings indicate that despite the deviation of non-salient weights from the ideal Gaussian distribution, the error curve associated with the search process still exhibits convex properties (as detailed in Appendix C), confirming the feasibility of pinpointing the optimal break-point.

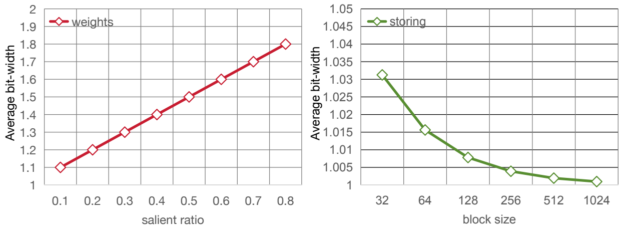

Figure 6: Weights and hardware overhead changes on Llama-7B. The left shows the calculation parameters as a function of the significant weight ratio; the right shows the hardware overhead as a function of the block.

Table 1: Average bit results from structural searching and residual binarization of OPT, LLaMA, and LLaMA2.

Model	7B	13B	30B	66B/65B/70B*
OPT	1.10	1.12	1.12	1.13
LLaMA	1.09	1.09	1.10	1.10
LLaMA2	1.07	1.08	N/A	1.09

*: OPT-66B, LLaMA-65B and LLaMA2-70B.

3.3. Pipeline of *BiLLM*

As depicted in Figure 3 left, *BiLLM* primarily performs binary quantization on all Linear weights within the Transformer blocks. This section introduces the detailed pipeline of *BiLLM*.

Binarization Workflow. We first deploy the structural search of salient columns and a residual approximation binarization for salient columns. The process of salient columns incurs additional weight bits due to the search proportion and residual mechanism. Table 1 presents the extra bits generated in some LLMs (Zhang et al., 2022; Touvron et al., 2023a;b). It can be observed that the searching and residuals bring only about **0.1** additional weight bits. Then, for these non-uniformly distributed weights, we use a split binarization strategy searching optimal p^*. The concentrated area and the sparse area are binarized separately. This part incurs the cost of an additional 1 bit for hardware group identification, but the computing parameters are still compressed to 1 bit. By retaining only block-wise compensation(Frantar et al., 2022; Frantar & Alistarh, 2022) and eliminating column-wise quantization error compensation, we further enhance the efficiency of PTQ and ensure the effectiveness of distribution exploration. Algorithm 1 illustrates the complete process of *BiLLM*, and detailed implementation of *BiLLM* is shown in Appendix A.

Extra Storing Bits. The extra bits is acceptable under the binary weight quantization of *BiLLM*. The weight parameters and additional hardware overhead are as follows:

$$\begin{cases} N_{\text{param}} = 2 \times r_{\text{salient}} + 1 \times (1 - r_{\text{salient}}), \\ N_{\text{storing}} = 1 + \dfrac{1}{b_{\text{size}}}, \end{cases} \tag{14}$$

where $r_{salient}$ signifies the proportion of salient weights and b_{size} denotes the block size in OBC compensation, with 1 bit allocated for marking the division of non-salient weights. $\frac{1}{b_{size}}$ represents the identifier for the structured column of salient weights. For example, a 10% structural selection along with an OBC compensation of size 128 was employed. This results in a weight parameter bit-width of **1.1** bits and a hardware flag bit-width of 1.008 bits. Figure 6 illustrates the weight overhead for different proportions and block sizes. It is important to note that flag weights do not participate in the computation; actual calculations are executed solely with parameter weights. Therefore, additional hardware identification bits do not affect the acceleration effect of binary quantization.

Algorithm 1 Main Framework of BiLLM: Inner details of each function are shown in Algorithm 2

```
func BinaryLLM($\mathbf{W}$, $\mathbf{X}$, $\beta$, $\lambda$)
Input: $\mathbf{W} \in \mathbb{R}^{n \times m}$ - weight matrix
       $\mathbf{X} \in \mathbb{R}^{r \times d}$ - calibration data
       $\beta$ - block size
       $\lambda$ - hessian regularizer
Output: $\mathbf{B}$ - binarized weights
$\mathbf{H} := 2\mathbf{X}\mathbf{X}^\top$ // $\ell^2$ error hessian matrix
$\mathbf{H}^c :=$ Cholesky$((\mathbf{H} + \lambda\mathbf{I})^{-1})$
$\mathbf{B} := 0_{n \times m}$
for $b = 0, \beta, 2\beta, ..., N$ do
    $\mathbf{W}^b := \mathbf{W}_{:,b:b+\beta}$
    $row_s\{\cdot\} :=$ salient$(\mathbf{W}_{:,b:b+\beta}, \mathbf{H}^c)$
    $\tilde{\mathbf{B}}_1 :=$ res_approximation$(\mathbf{W}^b_{:,j \in \{row_s\}})$
    $p^* :=$ seg_search$(\mathbf{W}^b_{i,j \notin \{row_s\}})$
    $\tilde{\mathbf{B}}_2 :=$ binary$(\mathbf{W}^b_{|w_{i,j}| \leq p^*, j \notin \{row_s\}})$
    $\tilde{\mathbf{B}}_3 :=$ binary$(\mathbf{W}^b_{|w_{i,j}| > p^*, j \notin \{row_s\}})$
    $\mathbf{B}_{:,b:b+\beta} := \tilde{\mathbf{B}}_1 + \tilde{\mathbf{B}}_2 + \tilde{\mathbf{B}}_3$
    $\mathbf{E} := (\mathbf{W}_{:,b:b+\beta} - \mathbf{B}_{:,b:b+\beta}) / \mathbf{H}^c_{bb:b+\beta b+\beta}$
    $\mathbf{W}_{:,b+\beta:} := \mathbf{W}_{:,b+\beta:} - \mathbf{E} \cdot \mathbf{H}^c_{b:b+\beta,b+\beta:}$ // block-wise OBC
end for
return B
```

4. Experiments

4.1. Setup

We deploy *BiLLM* within the Pytorch (Paszke et al., 2019)-Huggingface libraries (Wolf et al., 2019). All the binarization processes and experiments are conducted on a single 80

Table 2: Perplexity of RTN, GPTQ, PB-LLM, and BiLLM on OPT Family. The columns represent the perplexity results on Wikitext2 datasets with different model sizes.

Method	**Block Size**	**Weight Bits**	**1.3B**	**2.7B**	**6.7B**	**13B**	**30B**	**66B**
Full Precision	-	16.00	14.62	12.47	10.86	10.13	9.56	9.34
RTN	-	3.00	13337.38	15594.72	5797.32	3357.01	1566.00	6126.09
GPTQ	128	3.00	20.97	16.88	14.86	11.61	10.27	10.51
RTN	-	2.00	11272.65	9505.76	28363.14	194086.78	169616.47	1165864.25
GPTQ	128	2.00	115.17	61.59	50.19	21.36	15.71	82.10
RTN	-	1.00	17165.72	36516.69	11550.91	6986.35	6485.99	184796.30
GPTQ	128	1.00	14884.73	14144.58	10622.81	15196.96	12478.37	13106.45
PB-LLM †	128	1.70	265.52	124.35	105.16	81.92	25.14	29.09
BiLLM ‡	128	**1.11**	**69.97**	**49.55**	**35.36**	**18.82**	**12.71**	**12.06**

-: Vanilla RTN conducts layer-wise quantization. †: PB-LLM selects 10% elements in the original tensor as salient weights based on Hessian. ‡: BiLLM uses structural searching for salient weights. The table gives the average bit-width of the OPT family.

GB NVIDIA A100. Given that *BiLLM* is an efficient PTQ framework, it eliminates the need for any fine-tuning, allowing for completion through a single quantization process.

Models and Datasets. We facilitate our method on the OPT (Zhang et al., 2022) and LLaMA (Touvron et al., 2023a;b) families. Additionally, considering the customary need for instruction-based fine-tuning of LLMs to adapt to varying contexts, we also conducted experiments on Vicuna (Chiang et al., 2023). In terms of evaluation metrics, we mainly focused on the perplexity of LLMs' outputs, which is widely acknowledged in prior studies as a challenging yet stable indicator of LLM capabilities, particularly apt for network compression (Yao et al.; Frantar et al., 2022; Frantar & Alistarh, 2023; Xiao et al., 2023). We consider the test of WikiText2 (Merity et al., 2016), PTB (Marcus et al., 1994), as well as a part of the C4 (Raffel et al., 2020) data. Then, we further conduct the experiments on seven zero-shot evaluation tasks (PIQA (Bisk et al., 2020), BoolQ (Clark et al., 2019), OBQA (Mihaylov et al., 2018), Winogrande (Sakaguchi et al., 2021), ARC-e (Clark et al., 2018), ARC-c (Clark et al., 2018) Hellaswag (Zellers et al., 2019)) in the Appendix D, further verifying the robustness of our proposed *BiLLM* to the binarization of LLMs.

Baseline. Our primary baseline is PB-LLM (Shang et al., 2023), the most recent PTQ approach on binary LLMs. GPTQ (Frantar et al., 2022) and vanilla RTN are also selected. GPTQ is currently the advanced technology in PTQ, and many works(Lin et al., 2023; Dettmers et al., 2023b; Shang et al., 2023) choose it as the baseline. Other methods oriented towards 8-bit and 4-bit quantization are deemed unsuitable for binarization and were thus not considered.

4.2. Results

Comparison results. We conduct a meticulous comparison of the binary performance of different LLMs across various model sizes. We deploy the *BiLLM* on the OPT models (Zhang et al., 2022) under the condition of a block size equal to 128. As seen in Table 2, the model outputs under the RTN and GPTQ methods have already collapsed at 1-bit weights, whereas *BiLLM* still maintains reasonable linguistic output capabilities with an average weight of **1.1** bits. In comparison with PB-LLM at 1.7 bits, our method achieves a 35% reduction in weight bit-width while enhancing the performance of different sizes of the OPT model by 49.4% to 77.0%. It is noteworthy that when the parameter size exceeds 30B, *BiLLM* can achieve performance nearly equivalent to that of GPTQ with 3-bit quantization.

Due to the exceptional performance of the LLaMA (Touvron et al., 2023a;b) series, they have become the foundation for many open-source models (Chiang et al., 2023). Then, in Table 3, we evaluate the perplexity of outputs from the LLaMA series models using different methods. It can be observed that, even at ultra-low weight bit-width, *BiLLM* consistently outperforms the 2-bit RTN and GPTQ methods. And **1.08** bits *BiLLM* for LLaMA-65B and LLaMA2-70B even surpasses the output of the full-precision OPT-66B model, which demonstrates the further binary potential of the LLaMA family. We extend perplexity evaluation to the PTB and C4 datasets. Figure 7 illustrates the performance of the 7B parameter LLaMA series as well as the 6.7B OPT models. *BiLLM* continues to achieve a leading edge in performance compared to other methods (more additional comparisons are discussed in Appendix D).

Experiments of instruction-tuned models. Instruction fine-tuning can significantly improve the application capabilities of the model and has become a necessary process for LLMs deployment in different scenarios (Wei et al., 2021; Sanh et al., 2021; Chiang et al., 2023). We also deployed *BiLLM* on the recently popular fine-tuning instruction model Vicuna for benchmark testing. As shown in Table 4, the

Table 3: Perplexity of RTN, GPTQ, PB-LLM, BiLLM on LLaMA Family. The columns represent the perplexity results on Wikitext2 datasets with different model sizes.

Model	Method	Block Size	Weight Bits	7B	13B	30B	65B/70B*
LLaMA	Full Precision	-	16.00	5.68	5.09	4.10	3.53
	RTN	-	2.00	106767.34	57409.93	26704.36	19832.87
	GPTQ	128	2.00	152.31	20.44	13.01	8.78
	RTN	-	1.00	168388.00	1412020.25	14681.76	65253.24
	GPTQ	128	1.00	267001.72	113894.12	67093.73	25082.88
	PB-LLM †	128	1.70	102.36	36.60	33.67	12.53
	BiLLM ‡	128	**1.09**	**35.04**	**15.14**	**10.52**	**8.49**
LLaMA2	Full Precision	-	16.00	5.47	4.88	N/A	3.32
	RTN	-	2.00	17788.93	51145.61	N/A	26066.13
	GPTQ	128	2.00	60.45	19.70	N/A	9.12
	RTN	-	1.00	157058.34	47902.32	N/A	160389.91
	GPTQ	128	1.00	115905.67	9387.80	N/A	74395.42
	PB-LLM †	128	1.70	69.20	151.09	N/A	28.37
	BiLLM ‡	128	**1.08**	**32.48**	**16.77**	N/A	**8.41**

The table gives the average bit-width of the LLaMA family. N/A: LLaMA2 do not have 30B version. *: LLaMA has 65B version and LLaMA2 has 70B version.

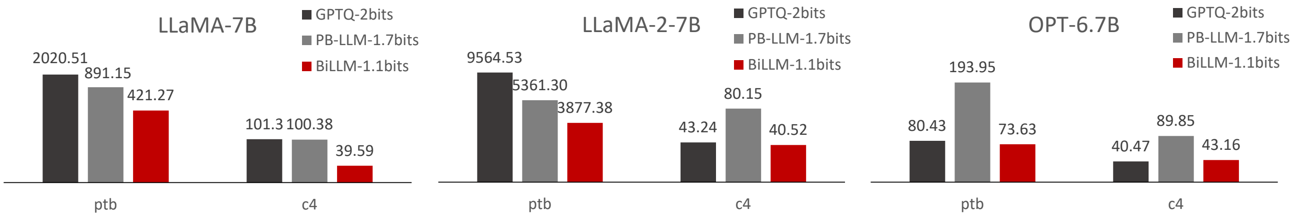

Figure 7: GTPQ, PB-LLM, BiLLM performed on the PTB and c4 datasets, mainly on LLaMA-7B, LLaMA2-7B, and OPT-6.7B, and we found that BiLLM performed relatively well.

Table 4: Perplexity of *BiLLM* on Vicuna-7B and Vicuna-13B. The columns of different models represent the perplexity results on Wikitext2, PTB, and C4 datasets. The block size is set to 128.

Model	Method	Weight Bits	Wiki -text2 ↓	PTB ↓	C4 ↓
Vicuna-7B	GPTQ	2.00	109.56	6227.73	64.28
	PB-LLM	1.70	68.01	477.52	67.23
	BiLLM	**1.08**	**33.00**	**332.17**	**36.24**
Vicuna-13B	GPTQ	2.00	41.75	465.94	40.57
	PB-LLM	1.70	362.17	772.44	346.16
	BiLLM	**1.08**	**36.57**	**300.31**	**28.76**

perplexity performance of GPTQ and PB-LLM are compared on Vicuna-7B and Vicuna-13B with three evaluations. *BiLLM* can achieve better performance at an average weight bit of **1.08**, which further proves that *BiLLM*'s universal LLMs binarization potential. We also provide dialogue examples of binary models in Appeandix F.

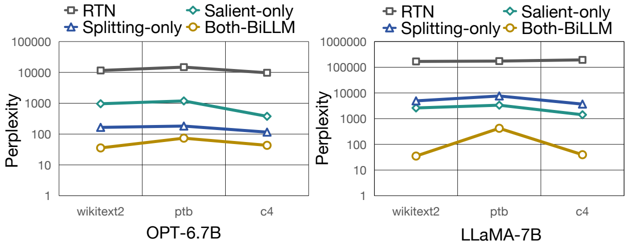

Figure 8: Ablation results of salient-only and splitting-only methods on OPT and LLaMA.

Zero-Shot results. To conduct a more comprehensive evaluation of binary LLMs, we extend our experiments to 7 zero-shot datasets. Appendix D provides detailed results of our approach compared to previous methods in ultra-low bit quantization, further showing the outlier of *BiLLM*.

Ablation results. *BiLLM* enhances binarization precision through two primary methods: structured salient binarization via residual approximation, and non-salient weight binarization via optimal splitting. To examine the effects of these

Table 5: Model size comparison of LLaMA family.

Method	**LLaMA-7B**	**LLaMA2-7B**	**LLaMA-13B**	**LLaMA2-13B**	**LLaMA-30B**	**LLaMA-65B**	**LLaMA-70B**
FP16	13.5GB	13.5 GB	24.2 GB	25.0 GB	60.5 GB	121.0 GB	129.3 GB
BiLLM	1.5 GB	1.6 GB	2.7 GB	2.8 GB	6.1 GB	14.8 GB	15.4 GB

Table 6: The memory occupancy rate compared with FP16 and the corresponding accuracy on OPT-30B.

Configuration	**BiLLM**	**PB-LLM (10%)**	**GPTQ**
Average bit-width	1.11	1.7	2
Memory Occupancy*	9.70%	16.50%	13.30%
PPL on WikiText-2	12.71	25.14	15.71

*: Equation of memory occupancy [R6]: memory_occupancy_compared_w_FP16 = (binary_unsalint_weight_size + residual_binary_salint_weight_size + CSR_compressed_bitmap_size + scaling_factor_size) / floating_point_weight_size.

Table 7: Memory occupancy rate compared with FP16 OPT.

Model	**BiLLM**	**GPTQ-2bit**
OPT-1.3B	9.40%	13.30%
OPT-2.7B	9.50%	13.30%
OPT-6.7B	9.30%	13.30%
OPT-13B	9.70%	13.30%
OPT-30B	9.70%	13.30%

strategies, we conducted decomposition experiments. As shown in Figure 8, both approaches significantly improve binary performance. Notably, we found that OPT-6.7B exhibits greater sensitivity to the splitting of non-salient weights (the blue line is lower than the green line), whereas LLaMA-7B is more responsive to salient weights' residual approximation (the green line is lower than the blue line). This further indicates that different LLMs exhibit varying responses to distinct binarization optimization strategies, showing that the two binarization strategies proposed by *BiLLM* are efficient to various LLMs. We further discuss details on the block-size ablation results in Appendix E.

Model size. In Table 5, we present the FP16 of models ranging from LLaMA-7B to 65B and LLaMA2-7B to 70B, as well as the sizes of binarized models after compression by BiLLM. Notably, BiLLM achieved close to a tenfold compression of weights across LLMs of different sizes.

GPU memory. The motivation of our BiLLM is to push the bit-width compression limit of LLM weights under post-training conditions, which reduces both the storage and GPU memory footprint of LLMs and retains their accuracy to the greatest extent for being practical. Although binarized GEMM is hard to implement directly due to fine-grained grouping, the extreme bit-width compression of our BiLLM brings significant savings in GPU memory requirements (size and bandwidth), which is considered to be one of the most significant efficiency bottlenecks of LLM inference (Gholami et al., 2024; Dettmers et al.; 2023a; Xiao et al., 2023; Chee et al., 2024; Shang et al., 2023). Here, we provide detailed memory and performance comparisons to demonstrate the advantages of BiLLM (as shown in Table A.1): for the OPT-30B model, BiLLM (1.1-bit) achieves a 41.57% and 27.07% memory compression improvement compared to PB-LLM (1.7-bit) and GPTQ (2-bit), respectively, while enhancing accuracy by 49.44% and 19.10%. We further provide a detailed comparison of memory usage with the 2-bit GPTQ method under different sizes of LLM in Table 7. The memory occupancy of our BiLLM is only about 69.9% of 2-bit quantization, which shows the great memory-saving benefit of our BiLLM from the extreme bit-width reduction, and we also achieve higher accuracy with the significantly saved memory.

5. Conclusions

This work proposed a novel post-training binary quantization method named *BiLLM*, specifically tailored for compressing pre-trained LLMs. Inspired by the characteristics of weight's value and Hessian distributions, we adopted a binary residual approximation for structurally salient weights to preserve their capabilities at ultra-low bits. For non-salient weights, we employed optimal segmentation for grouped binarization. Our results demonstrate that LLMs can undergo a one-time weight quantization at ultra-low bits without substantial loss of precision. *BiLLM* has pioneered the achievement of LLM performance guarantees at an average bit rate close to 1 bit. We validated the binarization performance of *BiLLM* across multiple open-source LLM families and conducted generalization tests on a fine-tuned instruction model. *BiLLM* advances the bit-width quantization frontier of LLMs, promising to facilitate the deployment of LLMs in edge scenarios and resource-constrained devices, and encourages further exploration in LLMs compression.

Acknowledgement. This work was supported by the National Science and Technology Major Project (2021ZD0110503), the Swiss National Science Foundation (SNSF) project 200021E_219943 Neuromorphic Attention Models for Event Data (NAMED), the Baidu Scholarship, and the National Natural Science Foundation of China (No. 62306025, No. 92367204).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Bisk, Y., Zellers, R., Gao, J., Choi, Y., et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural network. In *International conference on machine learning*, pp. 1613–1622. PMLR, 2015.

Chan, C.-Y. and Ioannidis, Y. E. Bitmap index design and evaluation. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pp. 355–366, 1998.

Chee, J., Cai, Y., Kuleshov, V., and De Sa, C. M. Quip: 2-bit quantization of large language models with guarantees. *Advances in Neural Information Processing Systems*, 36, 2024.

Chiang, W.-L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J. E., et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2023.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R., and Bengio, Y. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*, 2016.

Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. Llm. int8 (): 8-bit matrix multiplication for transformers at scale, 2022. *CoRR abs/2208.07339*.

Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023a.

Dettmers, T., Svirschevski, R., Egiazarian, V., Kuznedelev, D., Frantar, E., Ashkboos, S., Borzunov, A., Hoefler, T., and Alistarh, D. Spqr: A sparse-quantized representation for near-lossless llm weight compression. *arXiv preprint arXiv:2306.03078*, 2023b.

Dong, Z., Yao, Z., Gholami, A., Mahoney, M. W., and Keutzer, K. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 293–302, 2019.

Fang, J., Shafiee, A., Abdel-Aziz, H., Thorsley, D., Georgiadis, G., and Hassoun, J. H. Post-training piecewise linear quantization for deep neural networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pp. 69–86. Springer, 2020.

Faraone, J., Fraser, N., Blott, M., and Leong, P. H. Syq: Learning symmetric quantization for efficient deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4300–4309, 2018.

Frantar, E. and Alistarh, D. Optimal brain compression: A framework for accurate post-training quantization and pruning. *Advances in Neural Information Processing Systems*, 35:4475–4488, 2022.

Frantar, E. and Alistarh, D. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pp. 10323–10337. PMLR, 2023.

Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

Gholami, A., Yao, Z., Kim, S., Hooper, C., Mahoney, M. W., and Keutzer, K. Ai and memory wall. *IEEE Micro*, 2024.

Helwegen, K., Widdicombe, J., Geiger, L., Liu, Z., Cheng, K.-T., and Nusselder, R. Latent weights do not exist: Rethinking binarized neural network optimization. *Advances in neural information processing systems*, 32, 2019.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., and Kalenichenko, D. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2704–2713, 2018.

Jain, S., Venkataramani, S., Srinivasan, V., Choi, J., Gopalakrishnan, K., and Chang, L. Biscaled-dnn: Quantizing long-tailed datastructures with two scale factors for deep neural networks. In *Proceedings of the 56th Annual Design Automation Conference 2019*, pp. 1–6, 2019.

LeCun, Y., Denker, J., and Solla, S. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

Lee, C., Jin, J., Kim, T., Kim, H., and Park, E. Owq: Lessons learned from activation outliers for weight quantization in large language models. *arXiv preprint arXiv:2306.02272*, 2023.

Li, Y., Gong, R., Tan, X., Yang, Y., Hu, P., Zhang, Q., Yu, F., Wang, W., and Gu, S. Brecq: Pushing the limit of post-training quantization by block reconstruction. *arXiv preprint arXiv:2102.05426*, 2021.

Li, Z., Ni, B., Zhang, W., Yang, X., and Gao, W. Performance guaranteed network acceleration via high-order residual quantization. In *Proceedings of the IEEE international conference on computer vision*, pp. 2584–2592, 2017.

Lin, J., Tang, J., Tang, H., Yang, S., Dang, X., and Han, S. Awq: Activation-aware weight quantization for llm compression and acceleration. *arXiv preprint arXiv:2306.00978*, 2023.

Liu, Z., Oguz, B., Zhao, C., Chang, E., Stock, P., Mehdad, Y., Shi, Y., Krishnamoorthi, R., and Chandra, V. Llm-qat: Data-free quantization aware training for large language models. *arXiv preprint arXiv:2305.17888*, 2023.

Marcus, M., Kim, G., Marcinkiewicz, M. A., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., and Schasberger, B. The penn treebank: Annotating predicate argument structure. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*, 1994.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Mihaylov, T., Clark, P., Khot, T., and Sabharwal, A. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

Park, E., Yoo, S., and Vajda, P. Value-aware quantization for training and inference of neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 580–595, 2018.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Qin, H., Gong, R., Liu, X., Shen, M., Wei, Z., Yu, F., and Song, J. Forward and backward information retention for accurate binary neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2250–2259, 2020.

Qin, H., Ding, Y., Zhang, M., Yan, Q., Liu, A., Dang, Q., Liu, Z., and Liu, X. Bibert: Accurate fully binarized bert. *arXiv preprint arXiv:2203.06390*, 2022.

Qin, H., Zhang, M., Ding, Y., Li, A., Cai, Z., Liu, Z., Yu, F., and Liu, X. Bibench: Benchmarking and analyzing network binarization. *arXiv preprint arXiv:2301.11233*, 2023.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Rastegari, M., Ordonez, V., Redmon, J., and Farhadi, A. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, pp. 525–542. Springer, 2016.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Sanh, V., Webson, A., Raffel, C., Bach, S. H., Sutawika, L., Alyafeai, Z., Chaffin, A., Stiegler, A., Scao, T. L., Raja, A., et al. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*, 2021.

Shang, Y., Yuan, Z., Wu, Q., and Dong, Z. Pb-llm: Partially binarized large language models. *arXiv preprint arXiv:2310.00034*, 2023.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, H., Ma, S., Dong, L., Huang, S., Wang, H., Ma, L., Yang, F., Wang, R., Wu, Y., and Wei, F. Bitnet: Scaling 1-bit transformers for large language models. *arXiv preprint arXiv:2310.11453*, 2023.

Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., and Han, S. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pp. 38087–38099. PMLR, 2023.

Yao, Z., Aminabadi, R., Zhang, M., Wu, X., Li, C., and He, Y. Z. Efficient and affordable post-training quantization for large-scale transformers, 2022. *URL https://arxiv.org/abs/2206.01861*.

Yao, Z., Li, C., Wu, X., Youn, S., and He, Y. A comprehensive study on post-training quantization for large language models. *arXiv preprint arXiv:2303.08302*, 2023.

You, Y. *Audio coding: theory and applications*. Springer Science & Business Media, 2010.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Zhou, S., Wu, Y., Ni, Z., Zhou, X., Wen, H., and Zou, Y. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.

Zhu, X., Li, J., Liu, Y., Ma, C., and Wang, W. A survey on model compression for large language models. *arXiv preprint arXiv:2308.07633*, 2023.

A. *BiLLM* Implementation

Algorithm 2 BiLLM: Detailed functions process

```
func salient $(\mathbf{W}, \mathbf{H^c})$
  $\mathbf{S} := \mathbf{W}^2/[\mathbf{H}^c_{b:b+\beta b:b+\beta}]^2$ // salient matrix
  $rows_s\{\cdot\} :=$ topk(sum(abs($\mathbf{S}$)).(dim $= 0$))
  $e = \inf$ // searching error
  $n^* = 0$ // optimal number of salient columns
  for $i = 1, 2, ..., \mathrm{len}(rows_s)$ do
    $\mathbf{B}_1 :=$ binary($\mathbf{W}_{:,j,\ j\in rows_s[:i]}$)
    $\mathbf{B}_2 :=$ binary($\mathbf{W}_{:,j,\ j\notin rows_s[:i]}$)
    if $||\mathbf{W} - (\mathbf{B}_1 \cup \mathbf{B}_2)||^2 < e$ then
      $e := ||\mathbf{W} - (\mathbf{B}_1 \cup \mathbf{B}_2)||^2$
      $n^* := i$
    end if
  end for
  return $rows_s\{: n^*\}$
func binary $(\mathbf{W})$
  $\alpha := \dfrac{||\mathbf{W}||_{\ell 1}}{m}$
  $\mathbf{B} := \alpha \cdot \mathrm{sign}(\mathbf{W})$
  return $\mathbf{B}$
func res_approximation $(\mathbf{W})$
  $\mathbf{B}_1 := \mathrm{binary}(\mathbf{W})$
  $\mathbf{R} := \mathbf{W} - \mathbf{B}_1$
  $\mathbf{B}_2 := \mathrm{binary}(\mathbf{R})$
  $\mathbf{B} := \mathbf{B}_1 + \mathbf{B}_2$
  return $\mathbf{B}$
func seg_search $(\mathbf{W})$
  $e = \inf$ // searching error
  $p^* = 0$ // optimal break-point
  for $i = 0.1, 0.2, 0.3, ..., 9$ do
    $p := i \cdot \max(\mathrm{abs}(\mathbf{W}))$
    $\mathbf{B}_1 :=$ binary($\mathbf{W}_{|w_{i,j}|\le p}$)
    $\mathbf{B}_2 :=$ binary($\mathbf{W}_{|w_{i,j}|> p}$)
    if $||\mathbf{W} - (\mathbf{B}_1 + \mathbf{B}_2)||^2 < e$ then
      $e := ||\mathbf{W} - (\mathbf{B}_1 + \mathbf{B}_2)||^2$
      $p^* := p$
    end if
  end for
  return $p^*$
```

BiLLM necessitates the structured selection of salient rows and their subsequent quantization through residual approximation binarization. This is followed by dividing the non-salient weights, which exhibit a bell-shaped distribution, into a sparse area and a concentrated area. The division requires the optimization of the segmentation point p^* by minimizing quantization loss. Ultimately, the two regions of non-salient weights are binarized separately to derive the final binary weights for LLMs. The implementation details of the aforementioned function are enumerated in Algorithm 2.

B. Quantization Error

Quantization error definition for weight distribution The numerical range covered by the uniform quantizer spans from $[X_{min}, X_{max}]$. The number of intervals post-quantization, denoted as M, typically equals 2^b, where b represents the target bit-width of quantization. So the quantization step size is:

$$\Delta = \frac{X_{\text{max}} - X_{\text{min}}}{M} \tag{15}$$

The boundaries can be calculated as:

$$b_q = X_{\text{min}} + \Delta \cdot l \tag{16}$$

where $l \in 0, 1, ..., M$, and we have $b_q \in \{-\alpha, 0, \alpha\}$ under binarization. Then we give the mean of each interval:

$$x_q = X_{\text{min}} + \Delta \cdot l - 0.5\Delta \tag{17}$$

where $l \in 1, ..., M$. In this quantization scheme, we can get the MSQE from (You, 2010):

$$\theta^2 = \sum_{l=1}^{M} \int_{X_{\text{min}}+\Delta\cdot(l-1)}^{X_{\text{min}}+\Delta\cdot l} (X_{\text{min}} + \Delta \cdot l - 0.5\Delta - x)^2 g(x) dx \tag{18}$$

then we let the y to replace the $X_{\text{min}} + \Delta \cdot l - 0.5\Delta - x$ part, so the Equation (18) becomes:

$$\theta^2 = \sum_{l=1}^{M} \int_{-0.5\Delta}^{0.5\Delta} y^2 f[X_{\text{min}} + \Delta \cdot l - (y + 0.5\Delta)]^2 dx \tag{19}$$

consider the Equation (16) and Equation (17), the above equation becomes:

$$\theta^2 = \sum_{l=1}^{M} \int_{-0.5\Delta}^{0.5\Delta} x^2 f(x_p - x) dx \tag{20}$$

The aforementioned reasoning indicates that the MSQE of a uniform quantizer depends on the PDF and the quantization bit-width. Due to previous observations of the weights in pretrained LLMs, we have eliminated the salient weights. The remaining distribution of non-salient weights' $g(x)$, is not uniform and resembles a Gaussian distribution. In binarization, therefore, we substitute α into Equation (18), resulting in:

$$\begin{aligned} \theta^2 &= \sum_{l=1}^{M} \int_{(l-1-0.5M)\Delta}^{(l-0.5M)\Delta} [(l - 0.5 - 0.5M)\Delta - x]^2 g(x) dx \\ &= \int_{X_{\text{min}}}^{0} (-\alpha - x)^2 g(x) dx + \int_{0}^{X_{\text{max}}} (\alpha - x)^2 g(x) dx \end{aligned} \tag{21}$$

C. Searching Curve of Salient Column and Non-salient Distribution

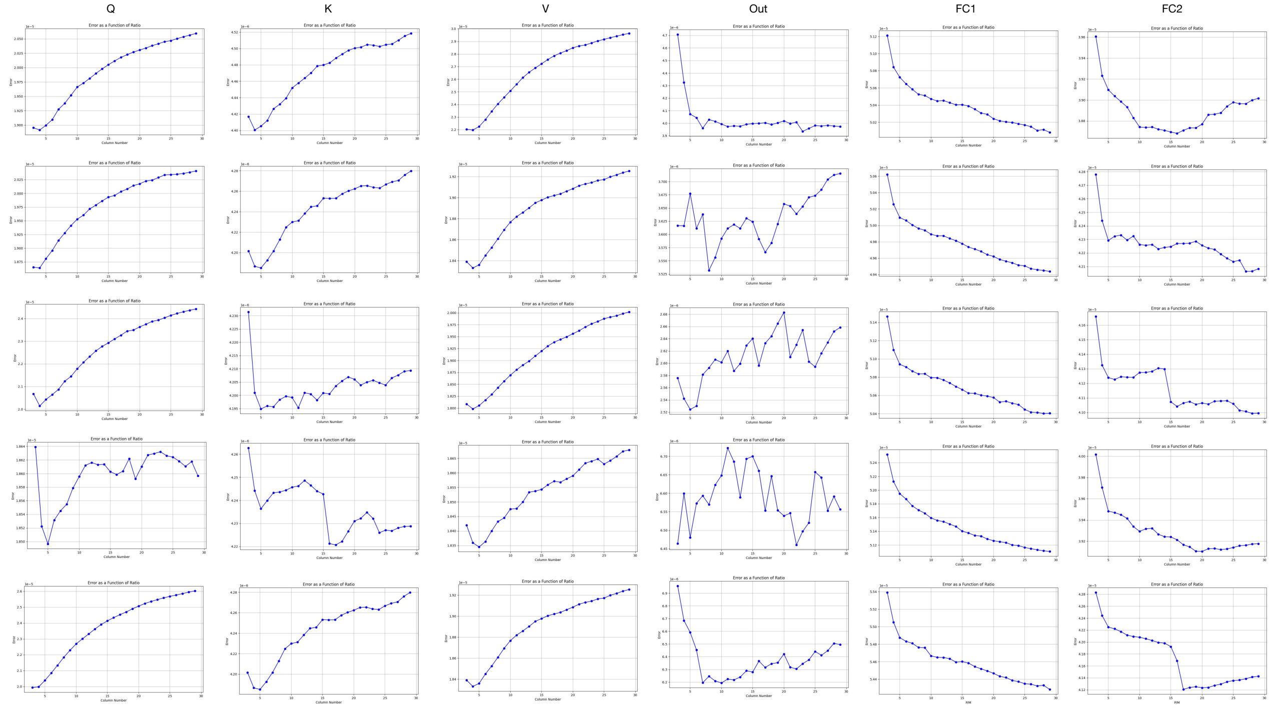

Figure 9: Block-wise searching curve of salient columns in OPT-6.7B. The majority of the curves indicate that the minimal quantization error can be achieved at the block level by considering only a few columns as salient. The *Out Projection* layer has a larger number of salient columns, hence varying coverage for each block. The distribution in the *FC* layer is more dispersed. After optimal searching, the overall average weight bit is merely **1.1** bits.

We implemented a column-level segmentation and formulated a minimal-error column number search, as delineated in Equation (5). The identification of the optimal count of salient column groups commences with the column exhibiting the highest salience. To mitigate the increase in bit-width resulting from residual approximation, we confined the search range to between 3 to 30 columns. Figure 9 illustrates the search curve pertinent to the inaugural Transformer block within the OPT6.7B model. It includes six layers of operators (*Q*, *K*, *V*, *Out Projection*, *FC1*, and *FC2*), with each layer showing the search curves for the first five blocks. Figure 15 elucidates the clustering of salient weights, suggesting that a majority of the layers and blocks are capable of attaining minimal quantization errors with a limited number of salient columns.

The block-wise changes in weight distribution brought about by OBC (Frantar & Alistarh, 2022) introduce fluctuations in the search curve; however, the structured selection still manages to encompass the majority of salient weights. In the *Feedforward* layer, where salient weight distribution is more scattered, the search curve leans towards employing residual approximation across an increased number of columns. Nonetheless, Table 1, displaying the average weight bit numbers across various LLMs, confirms that this search strategy effectively maintains weight compression at approximately **1.1** bits.

Figure 10 shows the unstructured search curve for the non-salient weights in the OPT6.7B model, with the same composition as that in Figure 9. The horizontal axis represents the ratio between p and the maximum weight value. Despite searching on a block-wise basis, the search curve still exhibits convex properties, indicating the presence of an optimal $p*$. This phenomenon demonstrates that the non-salient weights exhibit characteristics closely resembling an ideal Gaussian or Laplacian distribution (You, 2010; Fang et al., 2020).

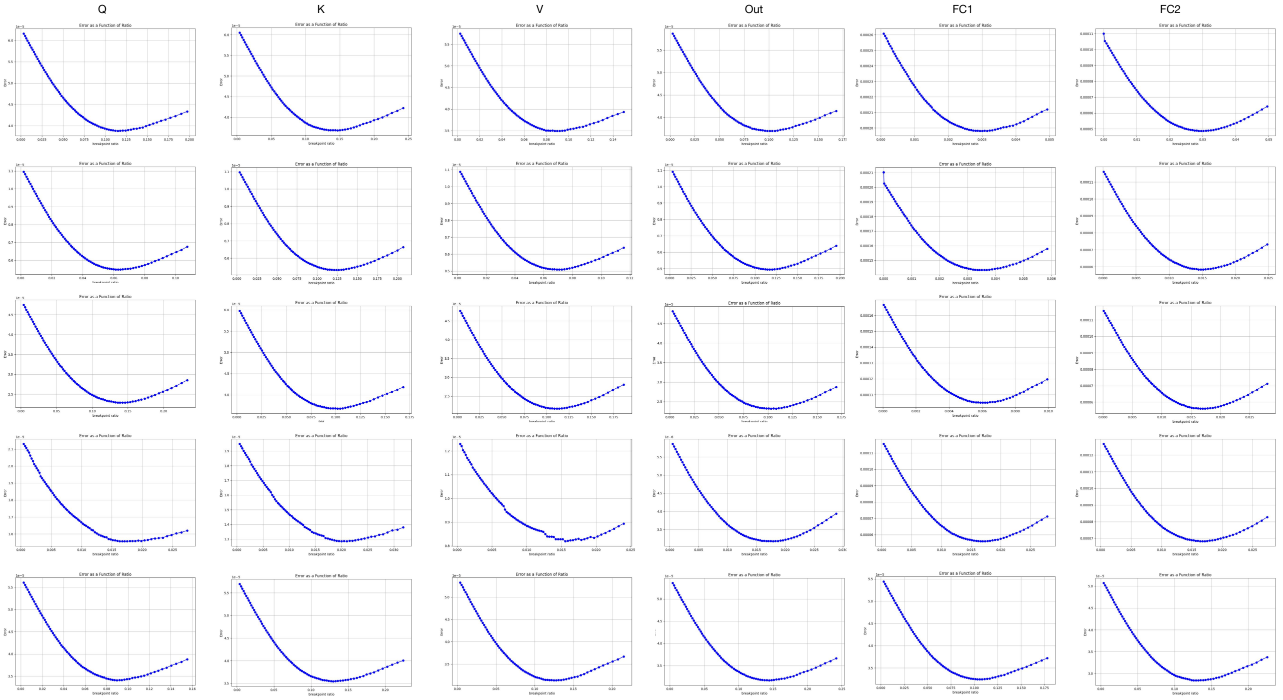

Figure 10: Block-wise splitting curve of bell-shaped distribution in OPT6.7B. The overall presentation exhibits the characteristics of a convex function, fundamentally aligning with the theoretical optimal point in terms of theoretical basis.

D. Multi-evaluation Comparisons

Perplexity results on PTB and C4.

We use tables in the main text to show the perplexity of the three methods GPTQ, PB-LLM, and BiLLM on the Wikitext2 dataset, and bar charts to show the perplexity results for LLaMA-7B, LLaMA2-7B, and OPT-6.7B on the PTB and C4 datasets. In the appendix, we show the quantitative comparison results for models of other sizes on the PTB and C4 datasets with more images.

In Figure 11, we find that although different models have different perplexity results, they still roughly follow the law that the larger the model, the lower the perplexity. BiLLM is generally still relatively better than the GPTQ and PB-LLM results in terms of perplexity with a lower bit-width configuration, while PB-LLM and GPTQ are higher or lower than each other, with slightly inferior results at very low bits.

Zero-shot results

For completeness of testing, we have also tested and compared metrics such as the accuracy of GPTQ, PB-LLM, and BiLLM on datasets such as PIQA and BoolQ, all using Zero Shot's experimental setup. From Table 8, We find that despite the loss

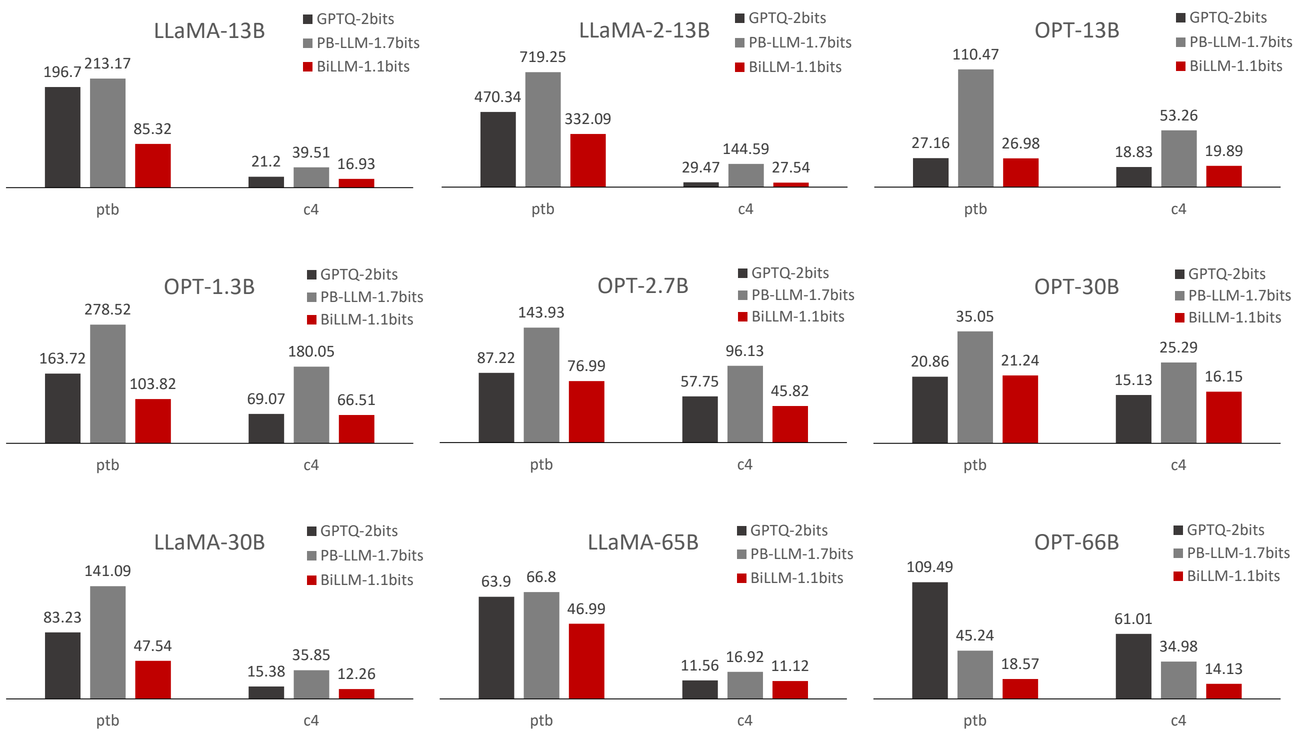

Figure 11: GPTQ, PB-LLM, BiLLM performed on the PTB and C4 datasets, mainly on LLaMA-13B, LLaMA2-13B, OPT-13B, and so on. The results showed that BiLLM performed relatively well.

in quantification, a side-by-side comparison between the three methods still shows BiLLM to be superior overall, testing one level higher on some datasets, while the effect of some random perturbations, although present, does not pull down BiLLM's performance across the board. This suggests that BiLLM's quantization results have significantly improved performance at very low bits, and further validates the conclusions.

Table 8: Accuracy on 7 data sets, from binarization LLaMA, LLaMA2, and OPT, and we also compare the results among GPTQ, PB-LLM, and BiLLM to validate the quantization effect.

Model	**Method**	**Weight Bits**	**Block Size**	**PIQA ↑**	**BoolQ ↑**	**OBQA ↑**	**Winogrande ↑**	**ARC-e ↑**	**ARC-c ↑**	**Hellaswag ↑**
LLaMA-7B	GPTQ	2.00	128	52.8	50.0	28.2	49.3	26.6	29.5	26.3
	PB-LLM	1.70	128	54.6	59.7	30.4	50.6	28.2	24.6	28.7
	BiLLM	**1.09**	128	**61.2**	**62.7**	**31.8**	**51.1**	**36.0**	**25.7**	**36.8**
LLaMA2-7B	GPTQ	2.00	128	51.1	43.9	29.0	50.8	26.6	28.5	26.3
	PB-LLM	1.70	128	53.8	62.3	30.2	49.3	28.0	25.0	27.7
	BiLLM	**1.08**	128	**60.6**	**61.8**	**33.2**	**52.4**	**36.2**	**24.4**	**34.8**
OPT-6.7B	GPTQ	2.00	128	56.6	51.1	25.6	51.2	31.3	22.9	30.4
	PB-LLM	1.70	128	57.6	55.5	24.2	47.7	33.2	21.0	31.0
	BiLLM	**1.11**	128	**58.6**	**62.2**	**29.0**	**51.5**	**34.1**	**23.9**	**31.9**

E. Ablation of *BiLLM* with different block size

To explore the effect of different chunk sizes on the quantization effect of BiLLM, we set up block size settings including 32 columns and 64 columns up to 512 columns and performed quantization experiments on them. The results show that the overall perplexity is lower as the chunk granularity becomes finer and the number of bits used becomes relatively smaller.

We believe this is because the smaller the chunks, the finer the data representation, and the more scale is used, but increasing the diversity of quantization results also increases the weighting overhead. A block size of 128 can better balance the bit-width and quantization effect.

Table 9: Perplexity on Wikitext2, PTB, and C4 with different block size settings on *BiLLM*.

Model	Block Size	Wikitext2	PTB	C4
LLaMA-7B	512	74.14	1078.90	81.76
	256	48.91	574.34	57.60
	128	**35.04**	**421.27**	**39.59**
	64	27.23	399.81	27.74
	32	17.56	263.39	19.85
LLaMA2-7B	512	52.90	267.82	43.86
	256	43.69	232.34	43.21
	128	**32.48**	**3877.38**	**40.52**
	64	20.12	830.36	24.46
	32	13.58	440.40	17.34
OPT-6.7B	512	151.81	257.22	101.96
	256	84.42	116.44	77.25
	128	**35.36**	**73.63**	**43.16**
	64	33.36	48.16	31.94
	32	20.48	31.02	21.47

F. Dialog Examples

In this section, we show some dialogue examples of binarized LLaMA-13B and Vicuna-13B.

G. Magnitude and Hessian Distribution of LLMs

Figure 2 displays the distribution characteristics of weights and Hessian in LLMs. In this section, we provide additional examples to illustrate the bell-shaped distribution of weight values and the long-tailed distribution of Hessian weights. Figure 13 depicts the distributions of four linear layers in the first Transformer block of the OPT-1.3B model, while Figure 14 shows the distributions of seven linear layers in the sixth block of the LLaMA-7B model. The selection of these specific block positions is intended to demonstrate the universality of these distribution characteristics in LLMs.

Figure 15 displays the distribution of sensitive weights across 5 Transformer blocks within the OPT-1.3B model. We present the Hessian distribution results for both the attention and feedforward blocks, with the red portion indicating the top 10% of the most significant weight distribution. We observed that the salient weights of Q, K, and V in the OPT family tend to concentrate in some columns or rows. Moreover, we noticed that salient weights in the *Out Projection* layer of multi-head self-attention blocks are distinctly concentrated in specific columns, supporting our structured selection approach discussed in the main text. In contrast, the distribution of salient weights in the feedforward layers is more dispersed. Based on these observations, we adopt a sensitivity-based structured search method to identify salient columns.

LLaMA-13B

Prompt:	Alaska is a state of ___
PB-LLM-1.7bit:	mind & body aided with other person.
BiLLM-1.1bit:	the US, it is situated in North America.
Prompt:	Yangtze and Yellow River locate in ___
PB-LLM-1.7bit:	the lower land.
BiLLM-1.1bit:	the country which is called as the country of the China.
Prompt:	After climbing thirty flights of stairs, he was drenched in sweat and didn't feel much like climbing. I advised him ___
PB-LLM-1.7bit:	he is really not very health.
BiLLM-1.1bit:	to get used now and the next day. After 2 days, he'd do a lot better and his sweat had lower.

Vicuna-13B

Prompt:	What are the capitals of China, US and Russia?
PB-LLM-1.7bit:	A’s, U’sa and U's were. 2, U’s with, A’s, U’s, U's: W, S’A and U’ with, U’s, or US's, or U's: U's S – with
BiLLM-1.1bit:	The capital of China is Beijing, the capital of US is Washington, DC, and the capital of Russia is Moscow.
Prompt:	I have a friend who lives in New York and I would like to visit him. How would I go there from Paris?
PB-LLM-1.7bit:	I would like it to go to Paris.
BiLLM-1.1bit:	You can fly from Paris to New York, and there are many airlines that can take you there. You can also take a flight to the New York, but you will have to check the schedule, as it will depend.
Prompt:	What should I do after failed in a math exam?
PB-LLM-1.7bit:	I failed to fail.
BiLLM-1.1bit:	Retake math exams if you don’t pass this year. Take more time to study and work on your weak areas. Or you need more support for the final, You can also seek outside help in the same subject.

Figure 12: Some examples of conversations. LLaMA-13B and Vicuna-13B are chosen to show the case of language supplementary and Q&A ability. And PB-LLM (int 8, 10%) is selected as the comparison. We color the text to show the reasonable or inappropriate responses.

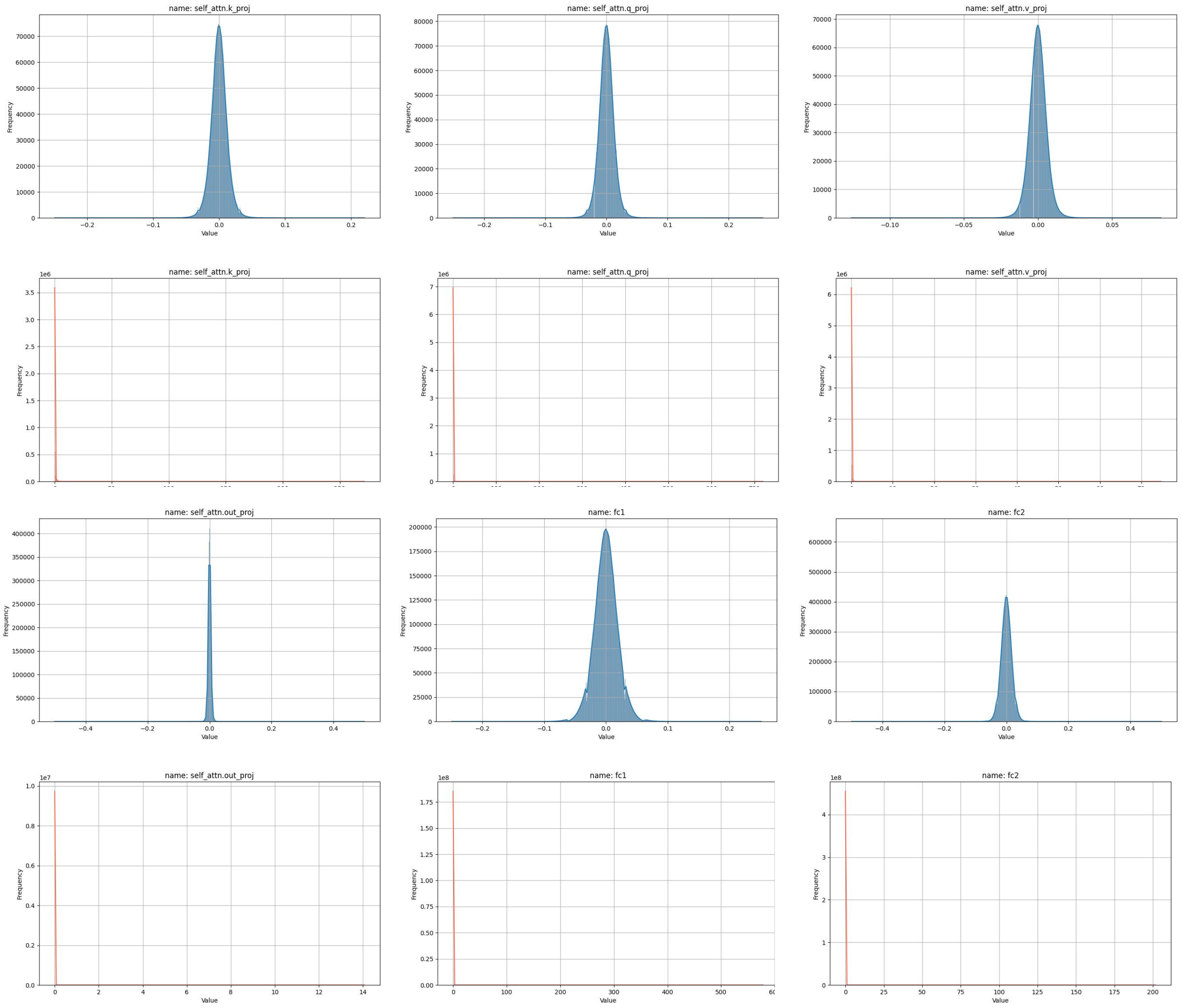

Figure 13: Different layers weight density distribution (blue) and hessian density distribution (orange) of the 1^{st} Transformer block of the OPT-1.3B model

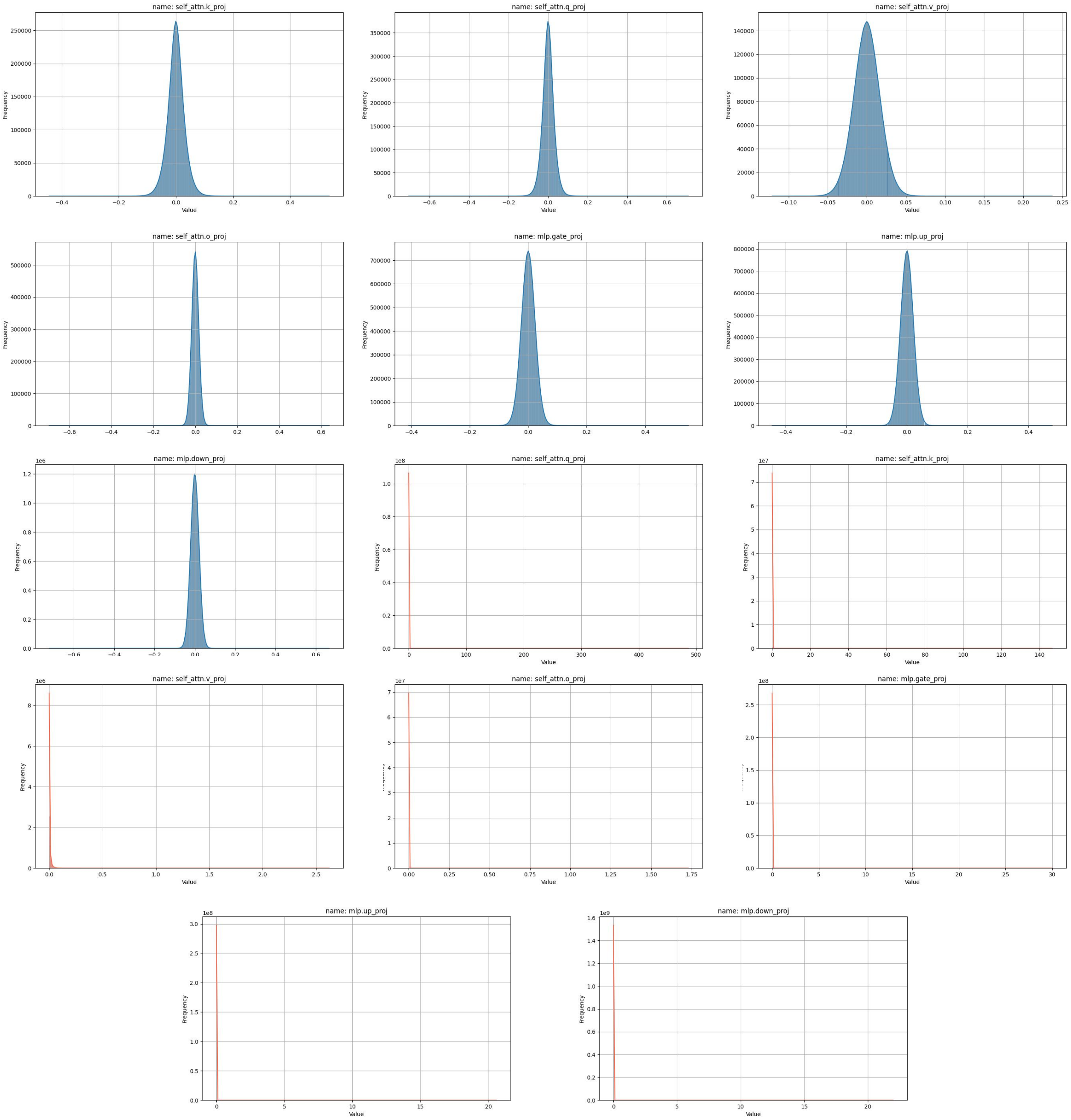

Figure 14: Different layers weight density distribution (blue) and hessian density distribution (orange) of the 6^{th} Transformer block of the LLaMA-7B model

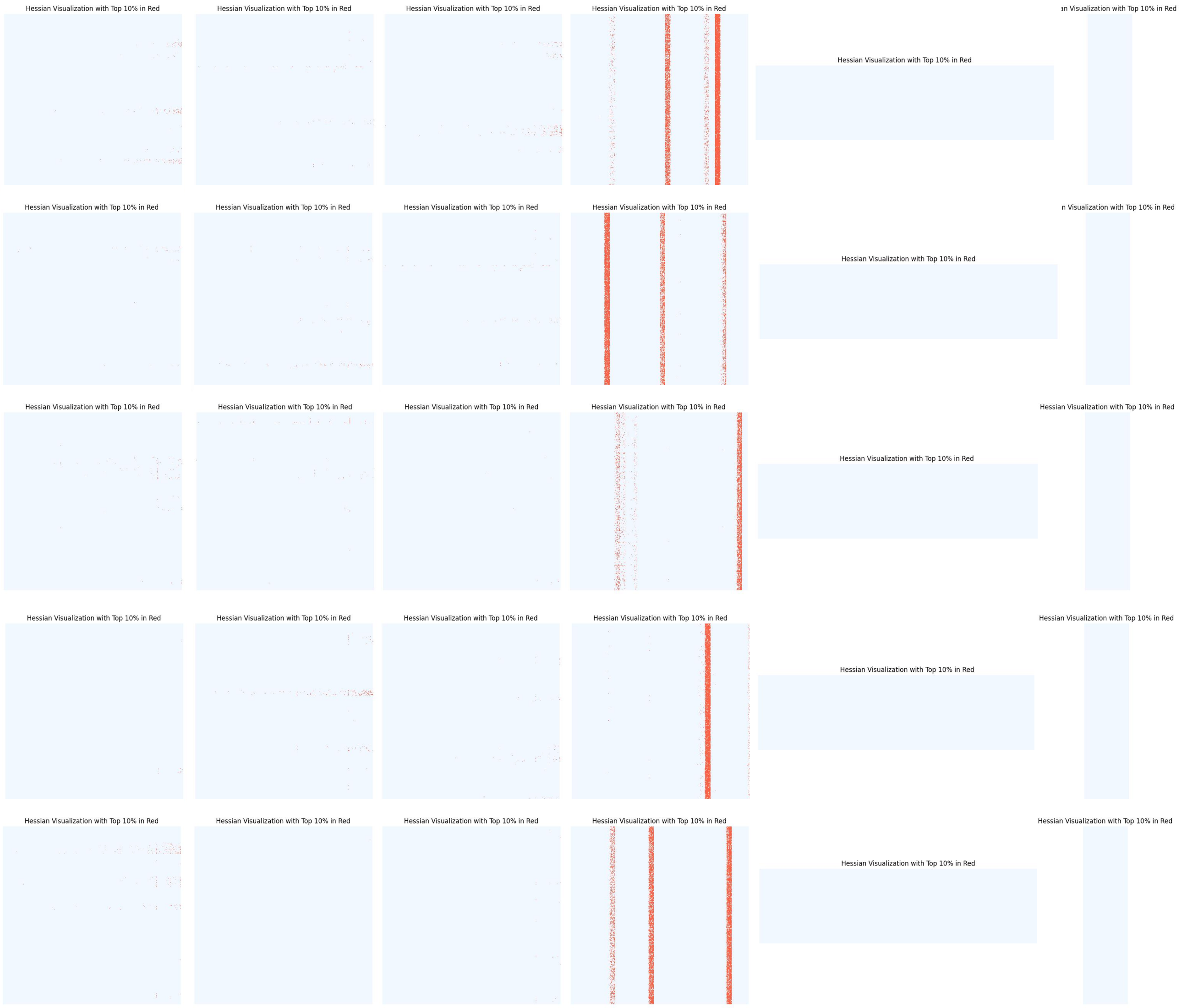

Figure 15: Distribution of top 10% salient elements in Hessian matrix. The distribution of $1^{st} - 5^{th}$ Transformer blocks in OPT-1.3B