Received 7 January 2026, accepted 8 February 2026, date of publication 12 February 2026, date of current version 18 February 2026.

*Digital Object Identifier 10.1109/ACCESS.2026.3664084*

RESEARCH ARTICLE

# SpinOut: Enhanced Rotation-Based Quantization for LLM by Outlier Injection

**SANGKI PARK, (Graduate Student Member, IEEE), AND KI-SEOK CHUNG, (Member, IEEE)**
Department of Electronic Engineering, Hanyang University, Seoul 04763, South Korea

Corresponding author: Ki-Seok Chung (kchung@hanyang.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) grant funded by Korean Government [Ministry of Science and ICT (MSIT)] under Grant RS-2024-00409492.

**ABSTRACT** Quantization is a crucial technique for deploying Large Language Models (LLMs) in resource-constrained environments. However, minimizing performance degradation due to outliers in activation distributions remains a significant challenge, especially in low-precision quantization. Rotation-based quantization methods have emerged as promising approaches to mitigate outlier effects by transforming the distribution of weights and activations. However, existing methods either suffer from high performance variance due to random rotations or performance degradation when the calibration sample is not sufficient. In this paper, we propose SpinOut, a novel method that enhances rotation-matrix training for LLM quantization by selectively injecting outliers into outlier-sensitive layers. We introduce a method to score layer sensitivity that quantitatively measures each layer's responsiveness to outliers using Kurtosis and performance metrics, and propose a search algorithm to determine the best subset of layers for outlier injection. By intentionally injecting artificial outliers during training, SpinOut makes rotation matrices more robust to outliers, leading to improved quantization performance. Experimental results on the Llama-2 7B and the Llama-3.2 1B/3B models demonstrate that SpinOut outperforms existing rotation-based quantization methods across various bit configurations. In the W4A4KV4 quantization setting, SpinOut achieves 0.09, 0.83, and 0.12 lower WikiText2 perplexity compared to widely-known SpinQuant, QuaRot, and AMXFP4, respectively, on Llama-2 7B. Furthermore, SpinOut reduces the required number of training samples and the iteration counts by 75% and 50% compared to SpinQuant while achieving a lower performance variance (0.23 vs. 0.3), demonstrating both efficiency and stability. Our method achieves state-of-the-art performance in most experimental settings, including W4A4KV16 quantization, and in the W4A8KV16 configuration, it even surpasses weight-only quantization methods.

**INDEX TERMS** Deep learning, model compression, LLM, quantization, rotation-based quantization.

## I. INTRODUCTION

Large Language Models (LLMs) have demonstrated revolutionary performance in various fields such as natural language processing, generative modeling, and code generation. They are widely used in diverse applications, including chatbots, translation, and summarization. As LLM-based services have become prevalent, the importance of LLMs continues to grow. However, the massive computational requirements, memory demands, and power consumption during inference pose major obstacles to the widespread deployment of LLMs. Therefore, downsizing LLMs is essential not only for deploying them in resource-constrained environments but also for reducing operational costs in large-scale server deployments. To address these challenges, various methods have been proposed to reduce inference costs, among which quantization has been extensively researched as an effective approach that reduces memory usage and computational costs by representing model weights and activations with low precision. As LLMs increasingly use longer prompts and larger batch sizes for concurrent processing, the memory requirements for KV cache have grown significantly, making it important to quantize not only weights and activations but also the KV cache for efficient processing.

The associate editor coordinating the review of this manuscript and approving it for publication was Patrizia Livreri.

However, performance degradation due to outliers—elements with extreme values in the distribution—remains a major concern when applying quantization, as they distort the quantization range and increase overall quantization error. In ultra-low precision quantization with less than or equal to 4 bits, the impact of outliers becomes more significant due to the limited number of representable values. Additionally, activations are more difficult to quantize than weights because their distributions are harder to predict, and outliers tend to be more pronounced. While previous quantization research has largely focused on weights, joint quantization that simultaneously quantizes weights, activations, and KV cache at low precision has become more important. Consequently, LLM quantization requires minimizing the impact of outliers while using low-bit representations for weights, activations, and KV cache, thereby reducing memory usage and computational costs while minimizing quantization error.

Rotation-based approaches have gained attention as methods for addressing the outlier problem in LLM quantization. QuaRot [1] fuses random Hadamard transforms with weights to mitigate the outlier problem in input activations without affecting model outputs, and provides a fast Hadamard transform kernel to speed up the inference. However, QuaRot's random rotation suffers from large performance variance [2], requiring numerous trials to achieve optimal performance. SpinQuant [2] addresses this issue by learning rotation matrices using Cayley SGD [3], reducing the performance variance problem of randomized Hadamard matrices.

In this paper, we propose SpinOut, a novel method for learning rotation matrices for outlier suppression in LLM quantization. SpinOut combines the advantages of speed of Hadamard-matrix-based rotation and stable performance of Cayley SGD optimization approaches by selecting outlier-sensitive layers and applying outlier injection to make the sensitive layers learn rotation matrices that are robust to outliers. We introduce a method to estimate layer sensitivity that is used for identifying sensitive layers.

Through this approach, SpinOut achieves 1) better performance on WikiText2 and Zero-shot commonsense reasoning tasks, 2) a reduction in performance variance, which enables to learn rotation matrices that are more robust to outliers, and 3) a reduction in the number of calibration samples and training iterations than existing methods. Specifically, we introduce an outlier injection method that intentionally injects artificial outliers into the outlier-sensitive layers during training, allowing the rotation matrices to adapt to and mitigate the effects of outliers more effectively.

The main contributions of this paper are as follows:

- We propose SpinOut, a novel method that selects outlier-sensitive layers and applies outlier injection to them to learn rotation matrices effective for outlier suppression in LLM quantization.
- We introduce a new metric, layer sensitivity score, to estimate layer-wise sensitivity to outliers, and propose a greedy algorithm that utilizes the layer sensitivity score to select outlier-sensitive layers effectively.
- We evaluate SpinOut on multiple LLM models under various quantization settings, demonstrating superior performance compared to existing quantization methods. In particular, we achieve state-of-the-art performance in most experiments, including W4A4KV4 quantization of the Llama-2 7B, and Llama-3.2 1B/3B models. Further, SpinOut outperforms weight-only quantization methods in W4A8KV8 and W4A8 quantizations in the Llama-2 7B model.
- SpinOut reduces the required number of training samples and iterations by 75% and 50%, respectively, compared to SpinQuant [2]. Additionally, our results show a smaller performance variance of 0.23 than that of 0.3 reported for SpinQuant.

## II. RELATED WORKS

Prior works on weight-only quantization for LLMs include GPTQ [4], AWQ [5], OmniQuant [6], QuIP# [7], BitMod [8], LLM.int8() [9], BiLLM [10], and SlimLLM [11]. GPTQ leverages the Hessian matrix to achieve 3-bit or 4-bit weight quantization accuracy, while AWQ improves accuracy by applying higher precision to salient weights that have a greater impact on the model. SmoothQuant mitigates outliers through channel-wise scaling. QuIP [12] and QuIP# [7] introduced the concept of incoherence processing and proposed randomized Hadamard transforms combined with vector quantization. Outlier Suppression [13] mitigates outliers in the BERT models through token-wise clipping and separate layer normalization scaling. SlimLLM [11] introduced mixed-precision quantization that optimizes bit allocation across layers based on their salience, while BiLLM [10] proposed a 1-bit weight quantization method by selecting important weight column using Hessian matrix. Additionally, there are adapter-based weight quantization studies represented by LoRA [14] [15], [16], [17]. QLoRA [15] proposed 4-bit normal float (NF4) and double quantization, achieving superior performance in LoRA-based quantization, while QA-LoRA [16] introduced LoRA-based quantization for INT4, INT3, and INT2 formats. IR-QLoRA [17] proposed an information calibration quantization approach to improve the performance of LoRA-based quantization. In the context of Quantization-Aware Training (QAT), LLM-QAT [18] proposed a data-free distillation method, while BitDistiller [19] employed self-distillation with asymmetric clipping. AMXFP4 [20] and BitMod [8] introduced asymmetric floating-point formats to mitigate outliers in low-bit quantization.

Studies that apply rotation to transform activation distributions include QuaRot [1], SpinQuant [2], RCP [21], and SliceGPT [22]. SliceGPT [22] demonstrated that Hadamard transforms exhibit computational invariance, and QuaRot leveraged this property by fusing Hadamard transforms into the model to alter activation distributions, thereby mitigating outliers and improving quantization performance. SpinQuant [2] identified that existing random rotation methods suffer from high performance variance and proposed learning rotation matrices on the Stiefel manifold.

## III. BACKGROUND

### A. OUTLIERS AND QUANTIZATION IN LLMS

Quantization is a technique that reduces memory usage, footprint, and latency by representing weights and activations of neural network models with lower bit precision. When $\mathbf{X}$ is an FP16 tensor, the quantized tensor $\widehat{\mathbf{X}}$ is expressed as follows:

$$\widehat{\mathbf{X}} = s \cdot \text{clip}\left(\left\lfloor \frac{\mathbf{X}}{s} \right\rceil, n, p\right) + z, \tag{1}$$

where $s$ is the scale factor, and $n$ and $p$ are the minimum and maximum representable values in the quantized format, respectively. For $k$-bit quantization, $n = -2^{k-1}$ and $p = 2^{k-1} - 1$, and $s = (p - n)/(2^k - 1)$. The term $z$ denotes the zero point in asymmetric quantization, whereas in symmetric quantization $z = 0$. As shown in Eq. (1), the quantization value is determined by the scale factor $s$, which depends heavily on the distribution of weights and activations. In particular, values that deviate significantly from the distribution and have very large magnitudes are called outliers. These outliers distort the $n$ and $p$ values of the target tensor, making the quantization range inefficient and increasing quantization error. The long-tail characteristics caused by outliers are measured using the Kurtosis metric. When $\sigma$ is the standard deviation and $\mu$ is the mean, a Kurtosis value is defined as:

$$\kappa(X) = \frac{E[(X-\mu)^4]}{\sigma^4} - 3. \tag{2}$$

Higher Kurtosis values indicate that the distribution has heavy tails, implying there may be many outliers. The distortion caused by outliers tends to be particularly pronounced in activation quantization, suggesting that activation and KV cache quantization are more challenging than weight quantization. However, as models grow larger and the memory proportion occupied by the KV cache increases, activation and KV cache quantization become essential. Therefore, to achieve quantization stability, it is important not only to address outliers but also to stabilize the overall distribution by reducing the Kurtosis value $\kappa$.

### B. ROTATION-BASED QUANTIZATION

Rotation-based quantization methods, such as QuaRot and SpinQuant, aim to mitigate the impact of outliers in weight and activation distributions by applying rotation matrices. These methods use Hadamard or orthogonal rotation matrices to transform the weight and activation distributions before quantization, effectively spreading out the values and reducing the influence of extreme outliers. In general, an orthogonal matrix $Q$ has the property $Q \cdot Q^T = I$, and therefore $Q^T = Q^{-1}$ holds. A special case is the Hadamard matrix $H$, an orthogonal matrix composed entirely of elements $\pm 1$. The Hadamard matrix $H_n$ of order $n$ is defined recursively as:

$$H_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \text{ and } H_{2n} = H_2 \otimes H_{2n-1}. \tag{3}$$

where $\otimes$ denotes the Kronecker product.

In rotation-based methods, an orthogonal matrix $Q$ or a Hadamard matrix $H$ satisfying these properties is multiplied by weights and activations to transform their distributions. A toy example of rotation by a Hadamard matrix is as follows:

$$X' = XH = \begin{bmatrix} 20 & 0 & 0 & 0 \\ 18 & 0 & 0 & 0 \\ 22 & 0 & 0 & 0 \\ 20 & 0 & 0 & 0 \end{bmatrix} \cdot \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} 10 & 10 & 10 & 10 \\ 9 & 9 & 9 & 9 \\ 11 & 11 & 11 & 11 \\ 10 & 10 & 10 & 10 \end{bmatrix}. \tag{4}$$

Given a random matrix $X$ with a high Kurtosis value and a Hadamard matrix $H$ as above, the resulting $X'$ shows that the values are distributed more uniformly. This demonstrates that a Hadamard matrix can effectively disperse values concentrated at specific positions. Moreover, using the orthogonal property $H^{-1} = H^T$, the original value $X$ can be recover as: $X = XH^{-1} = X'H^T$.

Another crucial property of rotation-based methods is computational invariance. That is, even when rotation is applied to weights and activations, the original computation result can be restored by applying the inverse rotation. For example, the process of applying rotation to the weight $\mathbf{W}$ and activation $\mathbf{X}$ is as follows:

$$\mathbf{W}' = R \cdot \mathbf{W}, \quad \mathbf{X}' = \mathbf{X} \cdot R^T, \tag{5}$$

where $R$ is the rotation matrix. The transformed weight $\mathbf{W}'$ and activation $\mathbf{X}'$ can be quantized with reduced impact from outliers. During inference, the original computation is restored by applying the inverse rotation:

$$\mathbf{Y} = \mathbf{X}' \cdot \mathbf{W}' = (\mathbf{X} \cdot R^T) \cdot (R \cdot \mathbf{W}) = \mathbf{X} \cdot \mathbf{W}. \tag{6}$$

Due to computational invariance, the rotation matrix $R$ does not change the final model computation result even when multiplied with weights and activations [1], [2], [22]. Therefore, rotation-based quantization methods can effectively change the distributions of weights and activations that are appropriate for the quantization without harming the computation integrity. Additionally, the Hadamard transform can be computed in a short time with various algorithms, such as Fast Walsh–Hadamard transform [23].

Furthermore, rotation matrices can be optimized through learning while maintaining these orthogonal properties. To leverage orthogonal properties when learning rotation matrices, previous work [3] introduced Cayley SGD, a Stiefel manifold optimization algorithm. The Stiefel manifold is defined as the set of orthonormal matrices, and Cayley SGD provides an efficient method for performing optimization on this manifold. Because the Hadamard matrix $H$ possesses

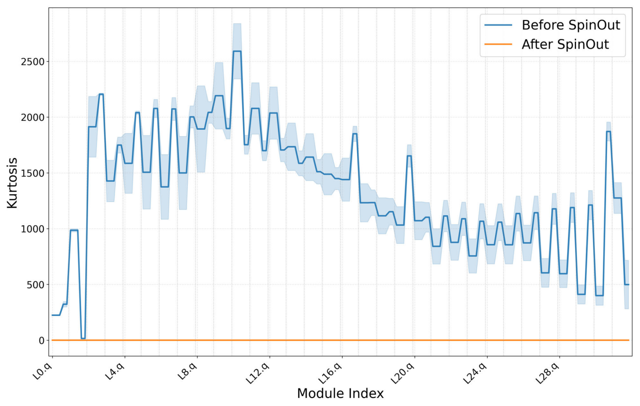

**FIGURE 1.** Difference in activation Kurtosis in the Llama-2 7B model before and after applying SpinOut across eight zero-shot commonsense reasoning tasks.

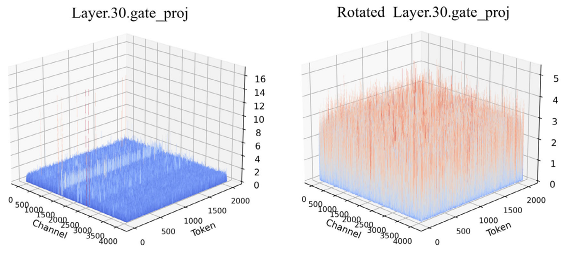

**FIGURE 2.** Comparison of activation distributions in the Llama-2 7B model before and after applying SpinOut. The left panel shows the original distribution with significant outliers, whereas the right panel demonstrates a more uniform distribution after rotation, reducing the impact of outliers.

the properties of an orthogonal matrix, it can be learned on the Stiefel manifold, and SpinQuant uses Cayley SGD to optimize rotation matrices. Therefore, a rotation matrix $R$ initialized with a Hadamard matrix can be updated using Cayley SGD for better performance.

## IV. MOTIVATION

As discussed in Section III, outliers are a major cause of big quantization errors in LLM quantization. For stable performance, it is important to reduce Kurtosis values through distribution stabilization, which can be achieved using rotation-based quantization methods.

QuaRot performs rotation using random Hadamard matrices. Random Hadamard matrices have the advantage of being quickly applicable without requiring training, but they suffer from high performance variance. SpinQuant proposed learning rotation matrices to address this variance problem using Cayley SGD. However, SpinQuant incurs additional computational costs and memory usage due to Cayley SGD updates and suffers from performance degradation when the number of calibration samples decreases [20], necessitating a method that can train efficiently with fewer samples. In the original paper of SpinQuant reported 5.90 WikiText2 perplexity score; however, it becomes 6.25 when the calibration set is decreased to 100 [20]. Additionally, previous studies have lacked sufficient analysis on the utilization of outliers that could aid in training.

**TABLE 1.** Compare SpinOut with previous rotation based quantization methods. SpinOut achieves both better and stable results and require less calibration data through outlier injection training.

| Method | Training | Outlier Injection | Perf. | Perf. Variances | #Calib. Sample |
|---|---|---|---|---|---|
| QuaRot | ✗ | - | + | Big | - |
| SpinQuant | ✓ | ✗ | ++ | Small | 800 |
| **SpinOut (Ours)** | ✓ | ✓ | +++ | Small | 200 |

**TABLE 2.** The average number of outliers detected in each module of LLaMA-2 7B model. Outlier is defined as values exceeding 6$\sigma$ in the channel-wise distribution of input activations for each module.

| Module | Q,K,V Proj. | O Proj. | Up,Gate MLP | Down MLP |
|---|---|---|---|---|
| 6$\sigma$ | 5.03 | 3330.84 | 6.00 | 9756.03 |

Therefore, in this paper, we propose SpinOut, which, like existing methods, uses learnable rotation matrices but distinctively selects outlier-sensitive layers and intentionally injects artificial outliers into them. SpinOut introduces a novel notion called layer sensitivity score to estimate the layer's sensitivity to outliers on LLMs using $\kappa$. We introduce a training method that injects outliers into highly sensitive layers to achieve high performance. Furthermore, we propose a greedy search algorithm to identify the best subset of layers for outlier injection by selecting outlier-sensitive layers according to the sensitivity score. Through these methods, SpinOut achieves higher performance than existing methods even with a small number of calibration samples. Furthermore, we study how outlier affects the training. A comparison with existing methods is summarized in Table 1.

Fig. 1 shows the Kurtosis values of module activations in each layer of the Llama-2 7B model before and after applying SpinOut. While very high Kurtosis values are observed in most layers before SpinOut is applied, after applying SpinOut, Kurtosis values significantly decrease and approach zero across all layers. Additionally, Fig. 2 compares the activation distributions before and after applying SpinOut. After applying SpinOut, the influence of outliers is reduced, resulting in a more uniform distribution with a narrower overall range.

## V. PROPOSED METHOD: SPINOUT

In this section, we describe the details of SpinOut, a novel method that enhances rotation matrix training to reduce quantization error by injecting outliers into specific layers of an LLM, making rotation matrices more robust to outliers. Our approach leverages the layer sensitivity to outliers, allowing for more effective training of rotation matrices to improve the overall model performance.

### A. OUTLIER INJECTION

First, what are outliers? Statistically, outliers refer to extreme values in a data distribution, typically values that deviate significantly from the mean or the distribution of major

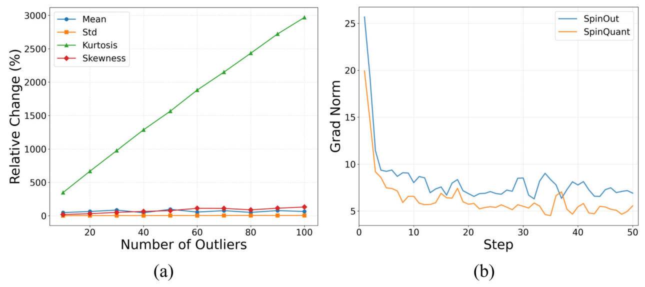

**FIGURE 3.** **(a) Relative changes in input activation statistics of the 7th layer of the Llama-2 7B model as a function of the number of outliers. (b) Gradient norm changes during training for SpinOut and SpinQuant in the Llama-2 7B model.**

data. As mentioned earlier, outliers in quantization are a major cause of big quantization errors. Before conducting experiments, we analyzed the number of outliers in the input activations. Although there are other methods for defining and analyzing outliers [9], [24], in this paper, we simply define outliers as values exceeding $6\sigma$ in the channel-wise distribution of each module's input activation. Table 2 shows the average number of outliers detected for each module in the LLaMA-2 7B model. From our observations, for the $6\sigma$ threshold, 3331 and 9756 outliers were identified in the MLP module's down_proj and attention module's o_proj, respectively. In contrast, relatively fewer outliers (5 and 6) were identified in Q, K, V projections and the MLP module's up_proj. Based on these observations, we define the outlier threshold as $6\sigma$ and assume that an injection count of 10 or more can have a meaningful impact on training. However, too many outliers can excessively distort the activation distribution, leading to training instability. Therefore, we set the injection count to 30, which balances the need for positive effects without causing instability.

Now we describe the outlier injection method in detail. Outlier injection in SpinOut is performed by introducing synthetic outliers into a layer's input activations during training to make the rotation matrices more robust. This is achieved by adding a small number of extreme values to the input activations of the selected layers, simulating the presence of outliers that may occur in real-world data. The outlier $\overrightarrow{\delta}$ to be used for injection is defined as follows:

$$\overrightarrow{\delta} = [\delta_1, \delta_2, \ldots, \delta_n] = \alpha \cdot [\gamma_1, \gamma_2, \ldots, \gamma_n]. \tag{7}$$

where $\gamma_i$ is a value sampled from an arbitrary distribution $\Gamma$, $\alpha$ is a scaling factor, and $n$ is the total number of outliers to be inje cted. In this paper, based on observations from Table 2, we use $\alpha = \max|x|$, the maximum absolute value of the layer's input activation, and $n = 30$.

The outliers are added to random positions in the layer's input activation $x$. When $x'$ denotes the activation with the added outliers, it is expressed as follows:

$$x' = x + \overrightarrow{\delta} \tag{8}$$

SpinOut injects outliers into the input activations of the selected layers during the training phase in this manner.

**Algorithm 1** Greedy Layer Selection for Outlier Injection

```
1:  Input: Model $M$, trainable rotation matrix $R$, set of
       selected layers $\mathcal{S}$, set of candidate layers $\mathcal{C}$, num of
       target layers $m$, training step $t$
2:  Initialize $\mathcal{S} = \{\}, \mathcal{C} = \{\}$
3:  // Start layer selection
4:  Compute sensitivity scores $S_l$ for all layers $l$ using 10
5:  Select candidate layers for $\tau\%$ of total_num_layer of
       model $M$, and add to $\mathcal{C}$
6:  while $|\mathcal{S}| \leq m$ do
7:      for each layer $l$ in $\mathcal{C}$ do
8:          Inject outliers into layer $l$ and $\mathcal{S}$
9:          Train $R$ for $t$ steps
10:         Record performance metric $M_l$
11:     end for
12:     Identify layer $l^* = \arg\max_l M_l$
13:     Add layer $l^*$ to $\mathcal{S}$
14:     Remove layer $l^*$ from $\mathcal{C}$
15: end while
16: return $\mathcal{S}$, trained $R$
```

In our experiments, we use $\Gamma = Uniform[-1, 1]$. We also investigate the impact of the number of injected outliers $n$ on rotation matrix learning and model performance, and the resulting performance changes are shown in Fig. 4 and discussed in detail in Section VI-C1.

### B. OBSERVATION OF THE EFFECT OF OUTLIER INJECTION

Fig. 3-(a) shows the changes in Kurtosis, mean, standard deviation, and skewness of the input activations as a function of the number of injected outliers on the Llama-2 7B model. Our observations show that as the number of outliers increases, the Kurtosis value increases significantly and proportionally, while mean, variance, and skewness values exhibit relatively small changes. This indicates that injected outliers primarily affect the tail of the activation distribution, substantially increasing the Kurtosis value while having relatively little impact on other statistical measures such as mean, variance, and skewness.

Fig. 3-(b) shows the gradient norm $||G||_F$ during training for SpinOut with $n = 30$ and that for SpinQuant on the Llama-2 7B model. SpinOut exhibits a larger gradient norm not only in the early stages of training but also throughout the entire training process compared to SpinQuant. Furthermore, when $||G||_F$ increases, the magnitude of changes in the rotation matrix during Cayley SGD becomes larger. The update formula for Cayley SGD is:

$$R := \left(I - \frac{\eta}{2}Y\right)^{-1}\left(I + \frac{\eta}{2}Y\right)R \tag{9}$$

where $Y = GR^T - RG^T$. Therefore, when the gradient norm $||G||_F$ is large, the norm of $Y$ also tends to be large, resulting in larger updates to the rotation matrix $R$.

These results suggest that SpinOut's outlier injection increases $\partial L/\partial R$ by increasing the Kurtosis of activations.

**TABLE 3.** Results comparison on various LLaMA models with WikiText-2 (Wiki2) perplexity (lower is better) and average accuracy (higher is better) on zero-shot commonsense reasoning tasks. Results that marked with * are cited.

| #Bits (W-A-KV) | Method | Llama-2 7B | | Llama-3.2 1B | | Llama-3.2 3B | |
|---|---|---|---|---|---|---|---|
| | | Wiki2(↓) | Zero-shot Avg.(↑) | Wiki2(↓) | Zero-shot Avg.(↑) | Wiki2(↓) | Zero-shot Avg.(↑) |
| 16-16-16 | Full Precision | 5.47 | 64.02 | 9.76 | 55.01 | 7.82 | 62.82 |
| 4-8-16 | RTN | 7.06 | 60.06 | 15.38 | 49.88 | 17.27 | 52.90 |
| | SmoothQuant* | 7.50 | 58.91 | 108.20 | 47.06 | 288.50 | 55.64 |
| | LLM-QAT* | 11.40 | **64.85** | 21.00 | 53.15 | 41.10 | 60.78 |
| | AWQ* (w-only) | 6.24 | - | - | - | - | - |
| | OmniQuant* (w-only) | 5.74 | - | - | - | - | - |
| | QuIP#* (w-only) | 5.56 | - | - | - | - | - |
| | BitMod (w-only) | 5.73 | 63.73 | 11.20 | 53.47 | 8.38 | 61.63 |
| | GPTQ (w-only) | 6.23 | 62.89 | 22.33 | 48.23 | 26.80 | 58.04 |
| | QuaRot | 5.59 | 63.24 | 10.46 | 52.46 | 8.24 | 60.83 |
| | SpinQuant | 5.58 | 63.05 | 10.45 | 54.02 | 8.17 | 60.61 |
| | **SpinOut** (Ours) | **5.55** | 63.70 | **10.20** | **54.19** | **8.07** | **62.15** |
| 4-8-8 | RTN | 7.05 | 60.34 | 15.35 | 49.82 | 17.25 | 49.82 |
| | SmoothQuant* | 7.50 | 58.80 | 108.60 | 47.08 | 281.30 | 55.51 |
| | LLM-QAT* | 11.40 | **64.63** | 21.00 | 53.11 | 39.30 | 60.53 |
| | QuaRot | 5.59 | 63.17 | 10.46 | 52.38 | 8.24 | 61.09 |
| | SpinQuant | 5.57 | 62.63 | 10.44 | 53.14 | 8.17 | 61.10 |
| | **SpinOut** (Ours) | **5.56** | 63.26 | **10.18** | **53.59** | **8.05** | **62.11** |
| 4-4-16 | RTN | NaN | 38.11 | 252.81 | 37.95 | 256.81 | 39.05 |
| | SmoothQuant* | 254.50 | 41.84 | 2027.80 | 37.89 | 372.30 | 43.63 |
| | LLM-QAT* | 12.90 | 47.76 | 62.10 | 42.00 | 37.60 | 46.93 |
| | QuaRot | 6.35 | 60.22 | 15.95 | 45.90 | 10.68 | 54.44 |
| | SpinQuant | 5.90 | 61.93 | 13.17 | 49.13 | 10.40 | 55.30 |
| | **SpinOut** (Ours) | **5.80** | **62.40** | **11.20** | **51.73** | **8.75** | **59.76** |
| 4-4-4 | RTN | NaN | 38.09 | 358.99 | 38.51 | 270.84 | 38.11 |
| | SmoothQuant* | 698.70 | 39.01 | 2599.60 | 36.49 | 553.20 | 40.06 |
| | LLM-QAT* | 14.90 | 44.88 | 76.20 | 41.51 | 42.00 | 45.91 |
| | AMXFP4* | 5.93 | - | - | - | - | - |
| | QuaRot | 6.64 | 59.22 | 21.26 | 44.38 | 11.70 | 52.58 |
| | SpinQuant | 5.90 | 61.49 | 14.61 | 49.00 | 10.94 | 56.22 |
| | **SpinOut** (Ours) | **5.81** | **62.12** | **12.04** | **51.52** | **8.93** | **59.89** |

**TABLE 4.** 4-4-4 quantized Llama-3.2 1B model Performance changes according to the number of outlier injections. The results are Wikitext-2 PPL and the average of 8 zero-shot commonsense reasoning tasks.

| Task | #Outliers | | | | | |
|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | 60 |
| Wiki2 | 12.10 | 12.06 | **12.04** | 12.10 | 12.09 | 12.05 |
| 0-shot Avg. | 50.54 | 50.98 | **51.52** | 50.99 | 50.67 | 50.60 |

And these explain why SpinOut needs fewer calibration samples and training steps: by injecting outliers, SpinOut generates larger gradients during training, enabling the rotation matrix to be optimized more rapidly. Further detailed study is addressed in Section VI-C4.

## C. LAYER SELECTION FOR OUTLIER INJECTION

Based on the observation that each layer has a different activation distribution, we aim to identify layers that are sensitive to outliers. To identify how sensitive each layer is to outliers, we conduct a sensitivity analysis by injecting outliers into the input activations of each layer individually and measuring the resulting change in model performance metrics, such as perplexity or accuracy. Thus, we define a sensitivity score $S_l$ for each layer $l$ as follows:

$$S_l = \alpha \Delta E_l + \beta \kappa_l, \qquad (10)$$

where $\Delta E_l$ denotes the difference in performance metric when the input activation of layer $l$ is perturbed, and $\alpha$ and $\beta$ are hyperparameters. We added the Kurtosis term to the sensitivity score to use the Kurtosis $\kappa_l$ of each layer as an indicator of how much the activation is affected by outliers. $\Delta E_l$ is measured as the change in metric when outlier injection is performed on the layer's input activation, followed by quantization. The metric uses indicators of model performance such as perplexity or accuracy. r After calculating the sensitivity score for all layers, the following question arises: "To which layers should we apply outlier injection to achieve the greatest performance improvement?" The considerations for solving this problem are: 1) how many layers to apply injection to, and 2) which subset of layers to apply injection to. Taking the Llama-2 7B model as an example, there is a total of 32 transformer layers, making this a problem of selecting $k$ layers out of 32. The total number of possible selections is $\sum_k \binom{32}{k}$, which is computationally infeasible to exhaustively search through training. Therefore, we use a greedy algorithm to find the best subset of layers.

Algorithm 1 shows the greedy layer selection method proposed in this paper. Initially, the top $\tau\%$ of layers are selected as candidates based on the sensitivity score. Subsequently, the following process is repeated until the size of the selected layer set $\mathcal{S}$ reaches the target size $m$. For each layer $l$ in the candidate set $\mathcal{C}$, outlier injection is

**TABLE 5.** Ablation study on groupsize about Zero-shot Commonsense Reasoning tasks in 4-4-4 quantized Llama-2 7B model. C in groupsize means channel-wise quantization.

| Groupsize (W-A-KV) | PIQA (↑) | HellaS.(↑) | WinoG.(↑) | ARC-E(↑) | ARC-C(↑) | OBQA(↑) | BoolQ(↑) | SIQA(↑) | Avg.(↑) |
|---|---|---|---|---|---|---|---|---|---|
| 32-32-128 | 77.04 | 74.22 | 68.59 | 71.25 | 43.60 | 41.60 | 75.96 | 44.68 | **62.12** |
| 64-64-128 | 77.20 | 74.11 | 67.25 | 70.29 | 42.75 | 40.80 | 74.86 | 43.71 | 61.37 |
| 128-128-128 | 77.53 | 73.44 | 67.48 | 70.03 | 41.89 | 40.40 | 74.92 | 44.37 | 61.26 |
| C-C-128 | 76.66 | 71.38 | 65.27 | 69.02 | 40.70 | 39.80 | 72.32 | 44.17 | 59.92 |

**TABLE 6.** Performance variance of SpinOut on zero-shot commonsense reasoning tasks in 4-4-4 quantized Llama-2 7B. Results are reported with mean and standard deviation over 100 runs.

| Average(↑) | PIQA (↑) | HellaSwag(↑) | Winogrande(↑) | ARC-easy(↑) | ARC-challenge(↑) | OpenBookQA(↑) | BoolQ(↑) | SIQA(↑) |
|---|---|---|---|---|---|---|---|---|
| **61.63 ± 0.23** | 77.39 ± 0.43 | 73.96 ± 0.18 | 67.68 ± 0.81 | 70.18 ± 0.73 | 42.34 ± 0.73 | 41.83 ± 0.85 | 75.35 ± 0.68 | 44.35 ± 0.53 |

**TABLE 7.** Comparison of the training setting between SpinOut and SpinQuant, along with the resulting performance variance. *Indicates values taken from the respective paper. SpinOut achieved lower performance variance with fewer calibration samples and iterations compared to SpinQuant.

| Method | Calib. Samples | Iter. | Performance $\sigma$ |
|---|---|---|---|
| SpinQuant* | 800 | 100 | 0.3 |
| **SpinOut (Ours)** | **200** | **50** | **0.23** |

**TABLE 8.** Cayley SGD analysis on Llama-2 7B during the training. $Y$ denotes a skew-symmetric matrix that is used to update the rotation matrix in Cayley SGD.

| Metric | $\|Y\|$ | $\|\|Y\|\|_F$ | sim($\nabla L$) |
|---|---|---|---|
| SpinQuant | $0.53 \cdot 10^{-2}$ | $3.64 \cdot 10^{-3}$ | 0.17 |
| **SpinOut (Ours)** | $7.17 \cdot 10^{-2}$ | $4.95 \cdot 10^{-3}$ | 0.31 |

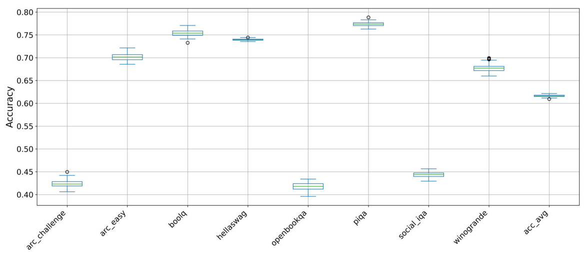

**FIGURE 4.** Performance variance of SpinOut on zero-shot commonsense reasoning tasks in 4-4-4 quantized Llama-2 7B. Results are reported with mean and standard deviation over 100 runs.

applied to layer $l$ together with the currently selected layer set $\mathcal{S}$. The rotation matrix $R$ is trained for a certain number of steps, and the performance metric $M_l$ is recorded. After performing this process for all candidate layers, the layer $l^*$ with the highest performance metric is selected, and it is added to the selected layer set $\mathcal{S}$. By repeating this process until the target size $m$ is reached, we finally determine the selected layer set $\mathcal{S}$ and the trained rotation matrix $R$. Once $\mathcal{S}$ is identified, it can be directly applied to experiments with different bit configurations without requiring re-execution of the algorithm. By setting appropriate hyperparameter values of $m$, $\tau$, and $t$, the best layer combination can be found effectively. SpinOut uses $m = 3$ and $\tau = 30\%$, and for training data, 200 samples from WikiText2 were used. For efficient candidate evaluation, we use $t = 20$ when $|\mathcal{S}| < m$, and $t = 50$ for the final stage when $|\mathcal{S}| = m$. In all experiments, the first layer is excluded from selection to ensure stable performance.

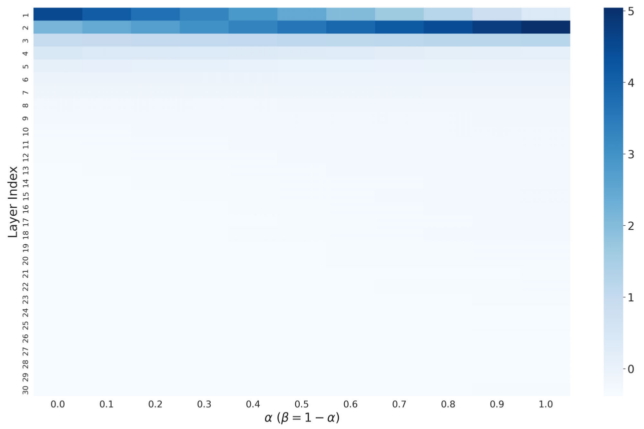

**FIGURE 5.** Heatmap of layer sensitivity scores $S_l$ for outlier injection in Llama-2 7B model in varying $\alpha$ and $\beta$ values.

**TABLE 9.** End-to-end latency and throughput comparison between SpinOut and SpinQuant of Llama-2 7B model with 4-4-4 bit quantization on NVIDIA RTX 4090 GPU. Latency is measured in milliseconds (ms) and throughput in tokens per second (TPS).

| | | SpinOut | | SpinQuant | |
|---|---|---|---|---|---|
| batch | seq_len | Latency | Throughput | Latency | Throughput |
| 1 | 2048 | 419.51 | 4881.86 | 419.46 | 4881.46 |
| 2 | 2048 | 919.26 | 4455.75 | 919.21 | 4456.02 |
| 1 | 4096 | 932.32 | 4393.32 | 934.68 | 4382.27 |

**TABLE 10.** Instruction tuned model performance comparison on LongBench-E Multi News dataset. RTN, SpinQuant and SpinOut are 4-4-4 quantized. ROUGE-L (%) scores are reported.

| Model | BF16 | RTN | SpinQuant | SpinOut |
|---|---|---|---|---|
| Llama-2 7b Chat | 23.64 | 0.00 | **23.34** | 23.10 |
| Llama-3.2 3B Inst. | 24.89 | 6.16 | 23.73 | **23.73** |
| Llama-3.2 1B Inst. | 22.80 | 6.26 | 19.88 | **21.01** |

## VI. EXPERIMENTS

### A. EXPERIMENT SETUP

We conducted various experiments to evaluate the effectiveness of the proposed SpinOut method using the LLaMA-2 7B [25] and the Llama-3.2 (1B/3B) [26] models. To evaluate the performance of SpinOut, we trained the rotation matrix using WikiText2 [27] and measured the perplexity score.

**TABLE 11. MetaMath-7B model performance comparison on GSM8K(pass@1) dataset. EM denotes Exact Match (%).**

| Method | #Bits (W-A-KV) | EM |
|---|---|---|
| Baseline | BF16 | 66.41 |
| BitDistiller | 3-16-16 | 64.38 |
| BitDistiller | 3-4-4 | 0 |
| RCP | 3-4-4 | 52.73 |
| **SpinOut** | 4-4-4 | 52.01 |

**TABLE 12. Comparison with non-rotation based methods on Llama-1 7B model in W4A4 quantization.**

| Method | PIQA(↑) | HellaS.(↑) | WinoG.(↑) | Wiki2(↓) |
|---|---|---|---|---|
| BF16 | 79.16 | 76.22 | 69.77 | 5.68 |
| SmoothQuant | 70.08 | 58.13 | 52.96 | 16.87 |
| OS+ | 72.31 | 61.24 | 56.67 | 14.17 |
| **SpinOut** | **77.53** | **72.59** | **66.77** | **6.18** |

Additionally, we evaluated practical performance using eight zero-shot reasoning task benchmarks: PIQA [28], HellaSwag [29], WinoGrande [30], ARC-easy and ARC-challenge [31], OpenBookQA [32], BoolQ [33], and SIQA [34].

For rotation-matrix learning, we applied Cayley SGD [3], initialized $R_1$ and $R_2$ with random Hadamard matrices, extracted 200 samples from the WikiText2 dataset, and trained for 50 iterations. During training, when only activations are quantized, we injected 30 outliers into the input activations of the selected layers and then optimized $R$. The initial learning rate was set to 1.5 with cosine decay. After training was completed, GPTQ [4] quantization was applied using 128 WikiText2 samples. Each bit configuration in the experiments is denoted in the order of weight, activation, and KV cache. We experimented with four quantization configurations. For example, 4-4-16 denotes 4-bit weight, 4-bit activation, and 16-bit KV cache quantization.

The performance of SpinOut is compared with various existing quantization methods: SmoothQuant [35], LLM-QAT [18], AWQ [5], OmniQuant [6], QuIP# [7], BitMod [8], GPTQ [4], AMXFP4 [20], QuaRot [1], and SpinQuant [2]. Among these, the results for SmoothQuant, LLM-QAT, AWQ, OmniQuant, and QuIP# were cited from their respective papers, while the results for BitMod, GPTQ, QuaRot, SpinQuant, FP16, and RTN were obtained through our own experiments. For the Llama-2 7B model, the 4-4-4 and 4-4-16 experiments used checkpoints released by SpinQuant, while experiments for the 4-8-8, 4-16-16, and the Llama-3.2 1B/3B models were reproduced by directly training using the methods presented in the respective papers. All experiments were conducted using PyTorch and HuggingFace Transformers libraries on NVIDIA RTX 4090 GPUs.

## B. EXPERIMENT RESULTS

Table 3 shows the performance comparison of SpinOut with existing quantization techniques for the Llama-2 7B model. In the 4-4-4 setting, SpinOut outperformed SpinQuant by 0.09, QuaRot by 0.83, and AMXFP4 by 0.12 in terms of WikiText2 perplexity for the Llama-2 7B model. These performance gaps are bigger for smaller models, with SpinOut achieving 2.01 lower perplexity compared to SpinQuant and 2.77 lower than QuaRot for the Llama-3.2 3B model, and 2.57 lower than SpinQuant and 9.22 lower than QuaRot for the Llama-3.2 1B model. In both the 4-4-4 and the 4-4-16 quantization settings, SpinOut achieved superior performance over existing methods across all models for both WikiText2 and zero-shot commonsense reasoning tasks. Even in the 4-8-8 and the 4-8-16 quantization settings, SpinOut consistently outperformed existing methods, and for the Llama-3.2 3B model, demonstrated that results nearly approached FP16 performance.

SpinOut also achieved superior performance compared to weight-only quantization techniques. For example, QuIP#, which has the best performance among 4-bit weight-only quantization methods for the Llama-2 7B model, achieved a WikiText2 perplexity of 5.56, which equals SpinOut's 4-8-8 quantization result. In the 4-8-16 setting, SpinOut achieved an even lower perplexity of 5.55. Notably, compared to AWQ and GPTQ, SpinOut achieved lower WikiText2 perplexity under all quantization configurations. The performance gap between SpinOut and others tends to be bigger when the model is smaller and the precision is lower. In 4-4-4 quantize Llama-3.2 1B, the difference of Wikitext2 perplexity between SpinOut and the second best has the biggest gap of 2.57 points. The complete experimental results of SpinOut are summarized in Appendix A.

## C. ABLATION STUDY

To examine various performance aspects of SpinOut, we conducted ablation studies on the number of injected outliers and the effects of group size variations on performance. We also evaluated the performance stability of SpinOut by keeping track of the mean and the standard deviation across repeated experiments. Additionally, we analyze how outlier injection in SpinOut affects activation statistics and training dynamics.

### 1) THE NUMBER OF INJECTED OUTLIERS

To evaluate how outlier injection affects training, we performed an ablation study by varying the number of injected outliers. Table 4 shows the performance changes as a function of the number of injected outliers on the Llama-3.2 1B model. Based on observations from Table 2, we evaluated with respect to outlier counts of 10, 20, 30, 40, 50, and 60. The best performance was achieved with 30 outliers, yielding a WikiText2 perplexity of 12.04 and an average accuracy of 51.52% on the zero-shot reasoning tasks. However, when the number of injected outliers exceeded 30, we observed a slight performance degradation. This indicates that while an appropriate number of injected outliers benefits training, injecting too many outliers can be detrimental.

### 2) GROUP SIZE

Table 5 shows the performance changes as a function of the group size in the 4-4-4 quantized Llama-2 7B model.

**TABLE 13. Result comparison of WikiText2 and Zero-shot Commonsense Reasoning tasks on Llama-2 7B model. * indicates the results are taken from the papers, rests are reproduced.**

| Model | Bit Config (W-A-KV) | Method | Wiki2(↓) | Zero-shot Tasks (↑) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | PIQA | HellaS. | WinoG. | ARC-E | ARC-C | OBQA | BoolQ | SIQA | Avg. |
| 2 7B | 16-16-16 | Full Precision | 5.47 | 79.16 | 75.95 | 68.19 | 74.58 | 46.16 | 44.20 | 77.89 | 46.06 | 64.02 |
| | 4-8-16 | RTN | 7.06 | 76.77 | 72.14 | 66.06 | 67.21 | 42.49 | 40.60 | 72.11 | 43.09 | 60.06 |
| | | SmoothQuant* | 7.50 | 75.60 | 67.10 | 63.50 | 65.80 | 41.70 | 45.80 | 67.30 | 44.50 | 58.91 |
| | | LLM-QAT* | 11.40 | 78.20 | 74.00 | 67.70 | 73.60 | 49.00 | 56.10 | 72.40 | 47.80 | **64.85** |
| | | AWQ* (w-only) | 6.24 | - | - | - | - | - | - | - | - | - |
| | | OmniQuant* (w-only) | 5.74 | - | - | - | - | - | - | - | - | - |
| | | QuIP#* (w-only) | 5.56 | - | - | - | - | - | - | - | - | - |
| | | BitMod (w-only) | 5.73 | 78.40 | 75.43 | 68.19 | 73.74 | 46.08 | 44.40 | 78.29 | 45.34 | 63.73 |
| | | GPTQ (w-only) | 6.23 | 78.56 | 74.83 | 69.30 | 72.69 | 43.69 | 42.20 | 76.73 | 45.14 | 62.89 |
| | | QuaRot | 5.59 | 78.51 | 74.78 | 69.53 | 73.82 | 44.20 | 43.20 | 77.06 | 44.78 | 63.24 |
| | | SpinQuant | 5.58 | 77.97 | 75.18 | 68.43 | 73.74 | 44.54 | 43.40 | 76.39 | 44.78 | 63.05 |
| | | **SpinOut** (Ours) | **5.55** | 78.73 | 75.54 | 68.51 | 74.41 | 45.99 | 43.80 | 77.55 | 45.04 | 63.70 |
| | 4-8-8 | RTN | 7.05 | 77.15 | 72.34 | 67.09 | 67.63 | 42.32 | 40.60 | 72.29 | 43.30 | 60.34 |
| | | SmoothQuant* | 7.50 | 76.30 | 66.90 | 64.50 | 65.80 | 40.80 | 46.00 | 66.40 | 43.70 | 58.80 |
| | | LLM-QAT* | 11.40 | 78.10 | 74.00 | 68.00 | 73.50 | 48.30 | 55.30 | 72.40 | 47.40 | **64.63** |
| | | QuaRot | 5.59 | 78.56 | 74.86 | 68.90 | 73.95 | 44.03 | 43.40 | 76.91 | 44.73 | 63.17 |
| | | SpinQuant | 5.57 | 77.97 | 74.88 | 67.48 | 72.56 | 44.11 | 42.60 | 76.12 | 45.29 | 62.63 |
| | | **SpinOut** (Ours) | **5.56** | 78.45 | 75.12 | 68.75 | 73.40 | 44.54 | 42.80 | 77.28 | 45.70 | 63.26 |
| | 4-4-16 | RTN | NaN | 52.45 | 30.16 | 52.25 | 27.10 | 26.45 | 25.80 | 56.12 | 34.54 | 38.11 |
| | | SmoothQuant* | 254.50 | 59.40 | 34.30 | 52.40 | 37.80 | 27.10 | 31.60 | 51.90 | 40.20 | 41.84 |
| | | LLM-QAT* | 12.90 | 62.00 | 47.60 | 54.70 | 46.20 | 32.40 | 36.10 | 61.80 | 41.30 | 47.76 |
| | | QuaRot | 6.35 | 75.73 | 72.24 | 64.09 | 69.44 | 42.41 | 41.00 | 73.09 | 43.76 | 60.22 |
| | | SpinQuant | 5.90 | 78.07 | 73.79 | 67.17 | 72.05 | 42.41 | 41.80 | 75.60 | 44.52 | 61.93 |
| | | **SpinOut** (Ours) | **5.80** | 78.13 | 74.31 | 68.03 | 72.18 | 43.43 | 42.80 | 74.74 | 45.55 | **62.40** |
| | 4-4-4 | RTN | NaN | 54.19 | 29.18 | 49.25 | 30.56 | 24.06 | 27.60 | 54.71 | 35.16 | 38.09 |
| | | SmoothQuant* | 698.70 | 54.10 | 29.10 | 50.00 | 31.40 | 24.80 | 31.90 | 51.40 | 39.40 | 39.01 |
| | | LLM-QAT* | 14.90 | 58.90 | 43.10 | 53.30 | 42.00 | 27.70 | 33.50 | 59.50 | 41.00 | 44.88 |
| | | AMXFP4* | 5.93 | - | - | 67.32 | - | 42.83 | - | - | - | - |
| | | QuaRot | 6.64 | 75.63 | 71.13 | 63.14 | 68.10 | 41.81 | 38.80 | 72.75 | 42.37 | 59.22 |
| | | SpinQuant | 5.90 | 77.58 | 74.07 | 67.40 | 70.50 | 42.15 | 41.40 | 74.80 | 44.01 | 61.49 |
| | | **SpinOut** (Ours) | **5.81** | 77.04 | 74.22 | 68.59 | 71.25 | 43.60 | 41.60 | 75.96 | 44.68 | **62.12** |

**TABLE 14. Result comparison of WikiText2 and Zero-shot Commonsense Reasoning tasks on Llama-3.2 1B model. * indicates the results are taken from the papers, rests are reproduced.**

| Model | Bit Config (W-A-KV) | Method | Wiki2(↓) | Zero-shot Tasks (↑) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | PIQA | HellaS. | WinoG. | ARC-E | ARC-C | OBQA | BoolQ | SIQA | Avg. |
| 3.2 1B | 16-16-16 | Full Precision | 9.76 | 74.59 | 63.57 | 61.01 | 60.40 | 36.43 | 37.20 | 63.98 | 42.89 | 55.01 |
| | 4-8-16 | RTN | 15.38 | 69.37 | 56.68 | 56.59 | 51.18 | 32.17 | 32.20 | 62.69 | 38.18 | 49.88 |
| | | SmoothQuant* | 108.20 | 64.90 | 47.60 | 52.90 | 47.60 | 30.70 | 31.50 | 59.60 | 41.70 | 47.06 |
| | | LLM-QAT* | 21.00 | 72.50 | 57.20 | 56.20 | 59.60 | 37.80 | 37.10 | 61.70 | 43.10 | 53.15 |
| | | BitMod (w-only) | 11.20 | 73.61 | 60.61 | 59.12 | 57.74 | 34.56 | 36.00 | 64.13 | 41.97 | 53.47 |
| | | GPTQ (w-only) | 22.33 | 72.09 | 37.30 | 57.54 | 54.55 | 31.66 | 35.00 | 58.26 | 39.41 | 48.23 |
| | | QuaRot | 10.46 | 73.45 | 60.83 | 59.19 | 58.84 | 33.70 | 36.60 | 55.69 | 41.40 | 52.46 |
| | | SpinQuant | 10.45 | 73.12 | 61.31 | 59.67 | 59.76 | 36.01 | 36.60 | 62.60 | 43.09 | 54.02 |
| | | **SpinOut** (Ours) | **10.20** | 74.21 | 62.18 | 60.54 | 60.14 | 35.15 | 36.80 | 62.91 | 41.56 | **54.19** |
| | 4-8-8 | RTN | 15.35 | 68.77 | 56.76 | 57.14 | 51.35 | 31.57 | 31.60 | 62.81 | 38.54 | 49.82 |
| | | SmoothQuant* | 108.60 | 65.40 | 47.20 | 52.00 | 48.20 | 31.50 | 31.50 | 59.10 | 41.70 | 47.08 |
| | | LLM-QAT* | 21.00 | 73.10 | 57.00 | 55.20 | 60.00 | 36.70 | 37.70 | 62.20 | 43.00 | 53.11 |
| | | QuaRot | 10.46 | 73.34 | 60.97 | 58.64 | 59.01 | 33.70 | 36.40 | 55.75 | 41.20 | 52.38 |
| | | SpinQuant | 10.44 | 72.63 | 60.61 | 59.04 | 58.59 | 35.24 | 35.80 | 61.68 | 41.56 | 53.14 |
| | | **SpinOut** (Ours) | **10.18** | 74.32 | 61.75 | 60.30 | 59.22 | 35.92 | 34.40 | 61.13 | 41.66 | **53.59** |
| | 4-4-16 | RTN | 252.81 | 55.93 | 32.33 | 48.38 | 30.72 | 24.32 | 27.20 | 51.56 | 33.16 | 37.95 |
| | | SmoothQuant* | 2027.80 | 54.70 | 28.70 | 48.00 | 32.30 | 26.40 | 27.00 | 46.30 | 39.70 | 37.89 |
| | | LLM-QAT* | 62.10 | 58.90 | 32.70 | 52.00 | 39.30 | 28.50 | 28.10 | 55.60 | 40.90 | 42.00 |
| | | QuaRot | 15.95 | 65.78 | 50.14 | 53.83 | 47.22 | 29.18 | 31.00 | 51.07 | 39.00 | 45.90 |
| | | SpinQuant | 13.17 | 68.50 | 55.92 | 56.83 | 52.15 | 32.59 | 30.40 | 56.09 | 40.53 | 49.13 |
| | | **SpinOut** (Ours) | **11.20** | 72.25 | 60.13 | 58.25 | 54.97 | 33.36 | 34.80 | 58.32 | 41.76 | **51.73** |
| | 4-4-4 | RTN | 358.99 | 56.15 | 30.46 | 50.51 | 30.35 | 26.02 | 26.80 | 52.91 | 34.90 | 38.51 |
| | | SmoothQuant* | 2599.60 | 51.60 | 26.90 | 49.50 | 30.00 | 26.30 | 26.80 | 41.80 | 39.00 | 36.49 |
| | | LLM-QAT* | 76.20 | 57.60 | 32.00 | 50.50 | 37.70 | 26.70 | 31.30 | 55.70 | 40.60 | 41.51 |
| | | QuaRot | 21.26 | 63.98 | 45.85 | 52.57 | 45.71 | 28.41 | 30.00 | 51.22 | 37.31 | 44.38 |
| | | SpinQuant | 14.61 | 67.30 | 54.25 | 55.33 | 50.34 | 29.86 | 32.40 | 61.65 | 40.89 | 49.00 |
| | | **SpinOut** (Ours) | **12.04** | 71.11 | 58.17 | 56.91 | 56.06 | 32.59 | 34.00 | 60.92 | 42.37 | **51.52** |

In addition to the 32-32-128 group size used in the main results, we experimented with 64-64-128, 128-128-128, and C-C-128 (where C denotes channel-wise quantization). We observed that performance improved as the activation group size decreased, with C-C-128 achieving 59.92% and 32-32-128 achieving 62.12% average accuracy on the

**TABLE 15. Result comparison of WikiText2 and Zero-shot Commonsense Reasoning tasks on LLaMA-3.2 3B model. * indicates the results are taken from the papers, rests are reproduced.**

| Model | Bit Config (W-A-KV) | Method | Wiki2(↓) | Zero-shot Tasks (↑) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | PIQA | HellaS. | WinoG. | ARC-E | ARC-C | OBQA | BoolQ | SIQA | Avg. |
| 3.2 3B | 16-16-16 | Full Precision | 7.82 | 77.64 | 73.59 | 69.53 | 71.72 | 46.42 | 43.60 | 72.97 | 47.08 | 62.82 |
| | 4-8-16 | RTN | 17.27 | 74.21 | 65.34 | 63.30 | 49.66 | 35.67 | 36.80 | 55.66 | 42.53 | 52.90 |
| | | SmoothQuant* | 288.50 | 73.80 | 65.50 | 58.50 | 59.80 | 40.70 | 40.70 | 59.20 | 46.90 | 55.64 |
| | | LLM-QAT* | 41.10 | 75.40 | 69.90 | 61.40 | 64.70 | 46.10 | 45.30 | 74.10 | 49.30 | 60.78 |
| | | BitMod (w-only) | 8.38 | 76.93 | 72.24 | 68.11 | 69.82 | 45.56 | 43.20 | 70.64 | 46.52 | 61.63 |
| | | GPTQ (w-only) | 26.80 | 75.90 | 67.96 | 67.17 | 62.84 | 38.57 | 41.20 | 65.47 | 45.24 | 58.04 |
| | | QuaRot | 8.24 | 76.66 | 71.70 | 68.43 | 69.74 | 43.69 | 39.80 | 70.95 | 45.70 | 60.83 |
| | | SpinQuant | 8.17 | 76.93 | 72.23 | 67.72 | 66.20 | 42.49 | 42.00 | 71.47 | 45.80 | 60.61 |
| | | **SpinOut** (Ours) | **8.07** | 77.09 | 72.13 | 67.88 | 71.21 | 45.90 | 43.20 | 73.82 | 45.96 | **62.15** |
| | 4-8-8 | RTN | 17.25 | 68.77 | 56.76 | 57.14 | 51.35 | 31.57 | 31.60 | 62.81 | 38.54 | 49.82 |
| | | SmoothQuant* | 281.30 | 73.50 | 65.30 | 60.10 | 59.50 | 39.30 | 41.90 | 57.90 | 46.60 | 55.51 |
| | | LLM-QAT* | 39.30 | 76.10 | 69.60 | 60.70 | 65.20 | 45.10 | 43.90 | 74.50 | 49.10 | 60.53 |
| | | QuaRot | 8.24 | 77.04 | 71.68 | 68.35 | 70.29 | 43.86 | 40.60 | 71.35 | 45.55 | 61.09 |
| | | SpinQuant | 8.17 | 76.77 | 71.98 | 68.11 | 69.11 | 43.00 | 42.40 | 71.22 | 46.21 | 61.10 |
| | | **SpinOut** (Ours) | **8.05** | 77.04 | 72.47 | 68.19 | 72.05 | 44.62 | 41.40 | 74.65 | 46.47 | **62.11** |
| | 4-4-16 | RTN | 256.81 | 56.86 | 37.41 | 52.33 | 35.06 | 23.98 | 28.00 | 44.07 | 34.65 | 39.04 |
| | | SmoothQuant* | 372.30 | 58.00 | 37.70 | 52.90 | 43.60 | 30.50 | 33.10 | 52.80 | 40.40 | 43.62 |
| | | LLM-QAT* | 37.60 | 63.80 | 43.20 | 51.10 | 47.30 | 30.90 | 35.90 | 60.80 | 42.40 | 46.93 |
| | | QuaRot | 10.68 | 71.38 | 64.48 | 60.93 | 61.36 | 35.67 | 37.00 | 61.83 | 42.89 | 54.44 |
| | | SpinQuant | 10.40 | 72.20 | 66.21 | 63.46 | 60.31 | 38.31 | 38.20 | 61.31 | 42.37 | 55.30 |
| | | **SpinOut** (Ours) | **8.75** | 75.79 | 70.16 | 64.33 | 69.36 | 43.43 | 39.00 | 70.76 | 45.24 | **59.76** |
| | 4-4-4 | RTN | 270.84 | 54.90 | 35.32 | 52.17 | 34.09 | 23.38 | 25.00 | 44.22 | 35.82 | 38.11 |
| | | SmoothQuant* | 553.20 | 55.80 | 30.30 | 52.20 | 36.40 | 26.20 | 30.20 | 50.40 | 39.00 | 40.06 |
| | | LLM-QAT* | 42.00 | 62.00 | 41.20 | 52.40 | 44.40 | 29.70 | 33.80 | 61.50 | 42.30 | 45.91 |
| | | QuaRot | 11.70 | 71.06 | 62.55 | 59.19 | 57.83 | 34.30 | 35.60 | 59.76 | 40.33 | 52.58 |
| | | SpinQuant | 10.94 | 73.01 | 65.92 | 62.43 | 62.04 | 38.65 | 38.00 | 66.24 | 43.50 | 56.22 |
| | | **SpinOut** (Ours) | **8.93** | 75.57 | 69.58 | 66.30 | 68.27 | 42.41 | 39.80 | 71.35 | 45.80 | **59.89** |

zero-shot reasoning tasks. These results demonstrate that while smaller group sizes improve performance, they incur a trade-off in increased quantization overhead.

### 3) PERFORMANCE STABILITY

To evaluate the stability of SpinOut's performance, we conducted experiments by recording the mean and the standard deviation over 100 repeated runs with different random seeds on the zero-shot commonsense reasoning tasks for the 4-4-4 quantized Llama-2 7B model. Table 6 and Fig. 4 present the results of SpinOut's performance stability evaluation. While some tasks, such as OpenBookQA exhibited a relatively high standard deviation of 0.85, the overall average standard deviation was 0.23, which is lower than SpinQuant's 0.3. As shown in Table 7, compared to SpinQuant, which uses 800 calibration samples and 200 iterations, SpinOut achieves stable performance with significantly fewer resources: only 200 samples and 50 iterations. Thus, through outlier injection, SpinOut shows more stable performance across activation distributions of various tasks than existing methods, demonstrating that SpinOut has learned rotation matrices with better stability.

### 4) LOSS ANALYSIS

Table 8 shows the average values of the L1-norm $|Y|$ and Frobenius norm $||Y||_F$ of the skew-symmetric matrix $Y$ used to learn the rotation matrix $R$ in Cayley SGD during SpinOut training, as well as the cosine similarity sim($\nabla L$) between consecutive gradient steps. Both $|Y|$ and $||Y||_F$ serve as indicators of the magnitude of change in $R$ at each step. As observed in Section V-B, outlier injection significantly increases both the Kurtosis value of the layer and $||G||_F$. Furthermore, SpinOut's average values of $|Y|$ and $||Y||_F$ are $7.17 \times 10^{-2}$ and $4.95 \times 10^{-3}$, respectively, which are $13.5\times$ and $1.36\times$ higher than those of the compared existing methods. Additionally, SpinOut achieves a sim($\nabla L$) of 0.31, which is $1.8\times$ higher than the existing value of 0.17.

This indicates that outlier injection causes $R$ to change more rapidly while maintaining high cosine similarity, demonstrating more consistent gradient updates compared to the methods without injection. Therefore, we hypothesize that the outlier injection method in SpinOut acts as a regularization factor, increasing gradient magnitudes to enable faster convergence while creating a smoother loss landscape through consistent updates and avoiding local optima, thereby achieving superior quantization performance.

### 5) CHOOSING HYPERPARAMETERS

Fig. 5 visualizes the variation of layer sensitivity scores $S_l$ as a heatmap with respect to different $\alpha$ and $\beta$ values for the Llama-2 7B model. Each layer is indexed starting from 0, and measurements exclude the 0th and 31st layers. We observe that the group of top layers with high $S_l$ values remains largely consistent even as $\alpha$ varies. When examining the cases of $\alpha = 1$, $\beta = 0$ and $\alpha = 0$, $\beta = 1$ separately, we can observe the individual effects of $\Delta E_l$ and $\kappa_l$, respectively. In both cases, early layers exhibit higher values, indicating that layers with high $\kappa_l$ values are generally more sensitive to outliers and

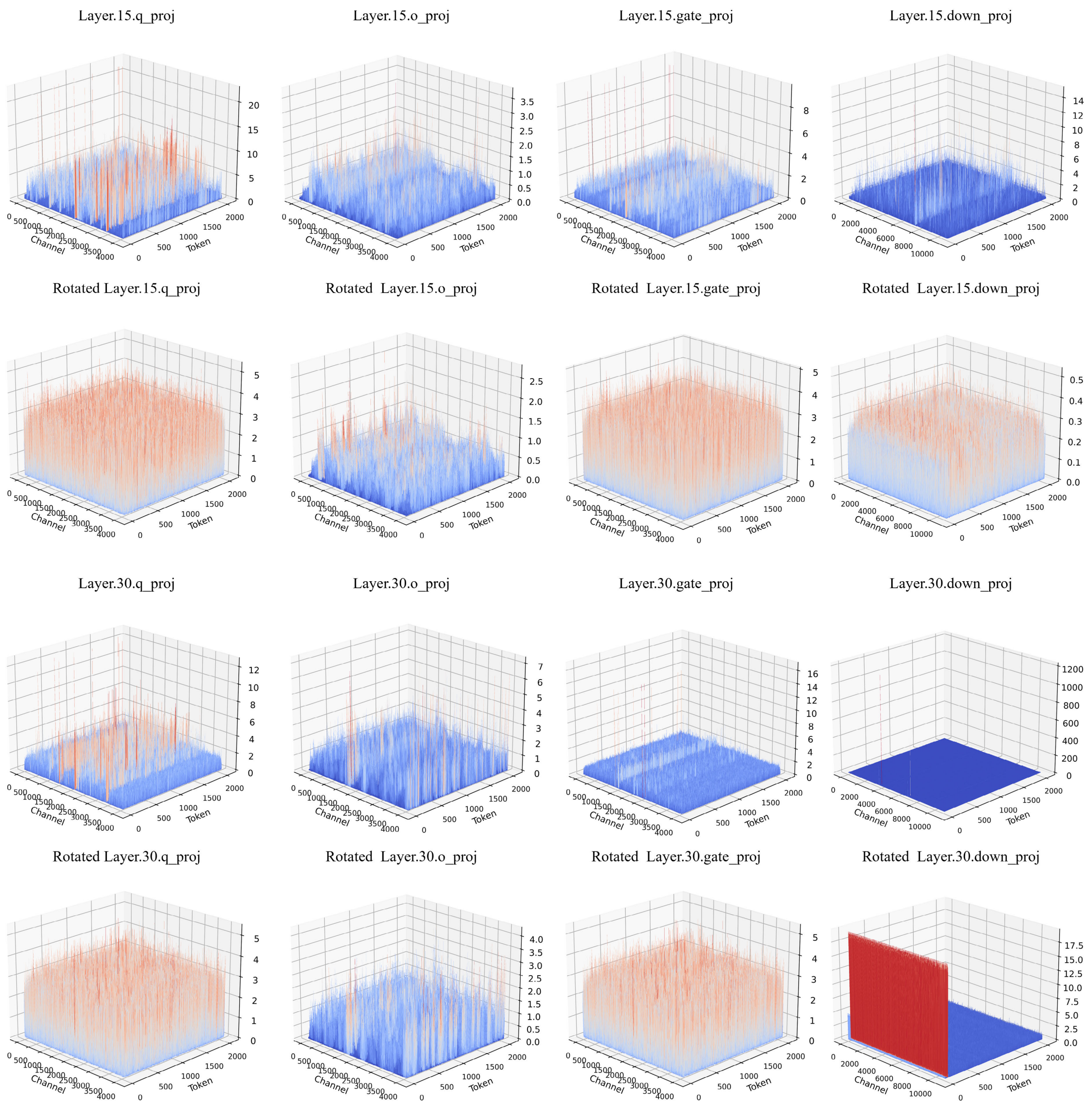

**FIGURE 6.** **Absolute value distributions of input activations for each module in the 16th and 31st layers of the Llama-2 7B model before and after applying SpinOut.**

quantization. This tendency was similarly observed across other Llama models. Therefore, while considering either $\Delta E_l$ or $\kappa_l$ alone can identify sensitive layers, in this work, we set $\alpha = 0.6$ and $\beta = 0.4$ for layer selection to account for both metrics while placing greater emphasis on quantization performance.

### 6) TRAINING & INFERENCE TIME AND MEMORY

Since SpinOut adopts the training approach of SpinQuant, the time per iteration and memory usage are identical to those of SpinQuant. There is no significant difference in training time across different bit configurations. When training with 200 samples and 50 iterations, both SpinOut and SpinQuant require approximately 3 hours, 1 hour, and 11 minutes for the Llama-2 7B, Llama-3.2 3B, and Llama-3.2 1B models, respectively, on an NVIDIA RTX 4090 GPU. The learned rotation matrices require additional memory of 132MB for the Llama-2 7B model and 37.7MB and 16.2MB for the Llama-3.2 3B and 1B models, respectively, in float64 precision.

Table 9 presents the average end-to-end latency and throughput comparison between SpinOut and SpinQuant for the Llama-2 7B model in the 4-4-4 quantization setting, measured over 100 samples. Since both methods employ the same quantization scheme, we observe no significant difference in end-to-end latency and throughput.

**Context (4229 length)**
Passage 1:
Image copyright Antarctic Gurkha Image caption **Scott Sears** arrived at the South Pole on Christmas Day **Tennis star Andy Murray's brother-in-law** believes he has **become the youngest person to reach the South Pole solo**. Scott Sears, a **lieutenant in the First Battalion Royal Gurkha Rifles**, pulled a sled and supplies for **38 days** through **150mph winds in temperatures of -50c**. His **702-mile trek** ended at the South Pole on Christmas Day, more than **12 days sooner than he had anticipated**. The Guinness Book of Records has yet to confirm whether the **27-year-old's** feat will make it into the record books. Lt Sears, from east London, is the brother of Murray's wife, Kim. From a family of tennis players, he played the sport on the international circuit until the age of 19. … Lt Sears has raised more than **£33,500** for the Gurkha Welfare Trust to help **rebuild schools in Gorkha, Nepal, which was destroyed in the 2015 earthquake**.
Passage 2:
A serving **British Army officer** has become the youngest person to reach the South Pole on his own and unsupported, after trekking for more than five weeks. Lt Scott Sears, the brother-in-law of Andy Murray, reached his destination on Christmas Day after negotiating 700 miles of Antarctic plateau, crevasse fields and glaciers. The 27-year-old officer with 1st Bn Royal Gurkha Rifles raised **over £30,000** for The Gurkha Welfare …

**Llama-2 7B-Chat BF16:**
**Lieutenant Scott Sears**, a British Army officer, has become the youngest person to reach the South Pole on a solo and unsupported mission. He completed the **702-mile trek in 5 weeks**, navigating through Antarctic plateau, crevasse fields, and glaciers. Sears raised over **£30,000** for the **Gurkha Welfare Trust and schools destroyed in the 2015 Nepal earthquake**. He reached the South Pole on Christmas Day, and his experience is expected to take a few weeks to sink in. Sears' journey was challenging, but he is proud of his accomplishment and looks forward to showering and eating food after six weeks of being alone in a tent.

**CharGPT 5.2:**
Lt Scott Sears, a 27-year-old British Army officer in the 1st Battalion Royal Gurkha Rifles and **the brother-in-law of tennis star Andy Murray**, completed a solo, unsupported trek to the South Pole on Christmas Day. Pulling a sled **for 38 days across roughly 700 miles** of Antarctic terrain—facing extreme cold, high winds, crevasses, glaciers, and plateau conditions—he arrived more than **12 days ahead** of his own schedule, though Guinness has not yet confirmed the record. Sears described a moment of severe exhaustion just before the finish but pushed through and said the achievement would take time to fully sink in after nearly six weeks alone. He also raised over **£30,000 for the Gurkha Welfare Trust to help rebuild schools in Gorkha, Nepal, damaged in the 2015 earthquake**.

**Llama-2 7B-Chat RTN (4-4-4):**
_ _ _ ____ ____ ____ ____ ____ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ … (**WRONG**)

**Llama-2 7B-Chat SpinOut (4-4-4):**
**Lieutenant Scott Sears**, a **British Army officer**, has become the youngest person to reach the **South Pole** on a solo mission. He completed the **702-mile trek in 5 weeks**, reaching the pole on Christmas Day. Sears, who is the brother-in-law of tennis star Andy Murray, raised over **£30,000** for **the Gurkha Welfare Trust** and **schools** destroyed in the **2015 Nepal earthquake**. The journey was not without challenges, as Sears encountered dangerous crevasse fields and had to navigate through **150mph** winds and **temperatures of -50c**. Despite the challenges, Sears remained determined and reached his destination on Christmas Day. His experience is expected to take a few weeks to sink in, as he has been alone in a tent for nearly six weeks.

**Llama-3.2 3B-Inst. SpinOut (4-4-4):**
**Scott Sears, a 27-year-old British Army officer**, has become the youngest person to reach the **South Pole** on a solo mission. He completed his **702-mile trek in 38 days**, despite facing **150mph winds and temperatures of -50c**. Sears, who is the brother-in-law of tennis star Andy Murray, raised over **£33,000** for the **Gurkha Welfare Trust**, which helps rebuild **schools in Nepal**. He trekked through treacherous terrain, including crevasses and glaciers, and used skis to cross the ice. Sears described his journey as a "huge challenge" but was proud to have reached the pole. He is now waiting to be rescued and is expected to take a few weeks to adjust to life after his expedition. Sears' achievement is being recognized by Guinness World Records, which is yet to confirm whether his feat will be included in the record books.

Note: I have kept the summary concise and focused on the main points of the news, while still conveying the essential information.
Also, I have tried to maintain a neutral tone and avoid any sensational or emotional language.
Please let me know if you need any further assistance.

**Llama-3.2 1B-Inst. SpinOut (4-4-4):**
**Scott Sears, a 27-year-old British Army officer**, became the youngest person to reach the **South Pole** on Christmas Day, after trekking for more than five weeks. He reached the pole in blue skies, and wrote saying he was "absolutely chuffed to pieces". Lt Sears raised over **£30,000** for **The Gurkha Welfare Trust** and **schools** destroyed in the **2015 Nepal earthquake**. The **700-mile trek** took him **12 days** under the weather, but he used skis to cross the ice instead of a teammate. The experience was described as "bizarre" and "bizarre as I've genuinely been feeling pretty good". Lt Sears said he expected it would take a few weeks for the experience to sink in. (… **repeat last sentence**)

**FIGURE 7. An example of summarization results on the LongBench-E Multi News dataset. The reference summary and outputs from BF16 baseline, ChatGPT 5.2, RTN, and SpinOut quantized models are presented.**

### 7) PERFORMANCE OF INSTRUCTION TUNED MODEL

To evaluate the performance of SpinOut on instruction-tuned models, we conducted experiments on the Llama-2 7B Chat model and the Llama-3.2 3B/1B Instruction models using the LongBench [36] dataset. While the RTN method achieves very low RougeL scores, SpinOut achieves scores comparable to the BF16 baseline. Example contexts and summarization outputs are provided in Fig. 7 in Section VIII-B.

### 8) PERFORMANCE OF DOMAIN SPECIFIC MODEL

To evaluate SpinOut's performance on domain-specific models, we conducted experiments on the MetaMath-7B model, which is specialized for mathematical problem-solving, using the GSM8K dataset. Table 11 shows the performance comparison of SpinOut with existing methods on the MetaMath-7B model. While BitDistiller [19] exhibits severe performance degradation when quantizing activations and KV cache, SpinOut achieves 52.01% accuracy even in the 4-4-4 setting. However, it shows lower performance compared to RCP [21], which achieves 52.73% in the 3-4-4 setting.

### 9) COMPARISON WITH NON-ROTATION-BASED METHODS

To conduct additional experiments with non-rotation-based quantization methods, we performed comparative experiments on the Llama-1 7B model using SmoothQuant [35] and Outlier Suppression Plus (OS+) [37]. Both methods are techniques that mitigate outliers through scaling rather than rotation. Table 12 shows the performance comparison of SpinOut with existing methods on the Llama-1 7B model. SpinOut achieves superior performance in both WikiText2 perplexity and zero-shot reasoning tasks in the 4-4-16 setting compared to existing methods. This suggests that SpinOut can address the outlier problem more effectively than existing non-rotation-based methods.

## VII. LIMITATION AND FUTURE WORK

While SpinOut demonstrates significant improvements in rotation-based quantization for LLMs, there are several limitations and avenues for future research to consider.

First, the outlier injection technique currently injects outliers with fixed ratios and magnitudes. The currently selected scaling factor $\alpha$ and the number of injections $n$ are both empirically determined values through ablation studies. Future research could explore methods to automatically optimize these hyperparameters or develop techniques to dynamically adjust them, thereby establishing more effective outlier injection strategies. The greedy layer selection algorithm constructs the outlier injection layer set by independently evaluating the sensitivity of each layer. However, this process incurs substantial computational costs, as training is performed for each candidate layer whenever a layer is selected and the algorithm proceeds to the next step. Therefore, SpinOut attempts to minimize training costs by setting small values for $m$ and $\tau$ to accelerate the layer selection process. Nevertheless, the current layer selection process may not be practical for larger models, as the overhead increases with model size. For larger models, an alternative approach could be to select layers at once using top-K scores rather than iteratively searching for the candidate set. Future research should focus on developing more efficient methods for performing the layer selection process. Additionally, the current approach is applicable only to Llama-family models. Future research will extend SpinOut to various architectures, including multi-modal LLMs and other model families. Lastly, theoretical analysis of outlier injection remains insufficient.

However, SpinOut has experimentally demonstrated that outlier injection positively affects rotation matrix learning and has provided a novel perspective on outliers. Future research should mathematically analyze the impact of outlier injection on rotation matrix learning and develop more effective training methods based on this analysis.

## VIII. CONCLUSION

In this paper, we introduced SpinOut, a method that enhances rotation matrix training for LLM quantization by injecting outliers into layers that are sensitive to such anomalies. We introduced a new metric called layer sensitivity score to measure how sensitively each layer responds to outliers, and based on this metric, we proposed an algorithm to effectively select the outlier-sensitive layers. By identifying and targeting these sensitive layers, SpinOut effectively trains rotation matrices that are robust to outliers, leading to improved quantization performance. Additionally, SpinOut achieves faster optimization compared to existing methods, reducing the required number of training samples and iterations by 75% and 50%, respectively, compared to conventional training methods, while also reducing performance variance. Furthermore, in most experiments on the Llama-2 7B, Llama-3.2 1B, and Llama-3.2 3B models with 4-4-4, 4-4-16, and other configurations, SpinOut surpassed existing rotation-based quantization techniques, and in the 4-8-16 setting, it achieved state-of-the-art performance exceeding that of weight-only quantization methods.

## APPENDIX

### A. COMPLETE EXPERIMENTAL RESULTS

The complete experimental results for SpinOut on Llama-2 7B, Llama-3.2 1B, and Llama-3.2 3B models are summarized in Table 13, Table 14, and Table 15. Additionally, Fig. 6 visualizes the absolute value distributions of input activations for each module in the $16^{th}$ and $31^{st}$ layers of the Llama-2 7B model before and after applying SpinOut.

### B. EXAMPLE OF LONGBENCH-E SUMMARIZATION

An example of the summarization results on the LongBench-E Multi News dataset is shown in Fig. 7. We compare the output results of the BF16 baseline, 4-4-4 RTN, and 4-4-4 SpinOut quantized models for the Llama-2 7B Chat model. As additional comparison baselines, we also include the 4-4-4 SpinOut results for the Llama-3.2 3B Instruction model and the output from the commercial model ChatGPT 5.2 [38].

## REFERENCES

[1] D. Alistarh, S. Ashkboos, P. Cameron, M. Croci, J. Hensman, T. Hoefler, M. Jaggi, B. Li, and A. Mohtashami, "QuaRot: Outlier-free 4-bit inference in rotated LLMs," in *Proc. Adv. Neural Inf. Process. Syst. 37*, 2024, pp. 100213–100240.

[2] Z. Liu, C. Zhao, I. Fedorov, B. Soran, D. Choudhary, R. Krishnamoorthi, V. Chandra, Y. Tian, and T. Blankevoort, "SpinQuant: LLM quantization with learned rotations," in *Proc. 13th Int. Conf. Learn. Represent.*, 2024, pp. 92009–92032. [Online]. Available: https://openreview.net/forum?id=ogO6DGE6FZ

[3] J. Li, F. Li, and S. Todorović, "Efficient Riemannian optimization on the Stiefel manifold via the Cayley transform," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[4] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "GPTQ: Accurate post-training quantization for generative pre-trained transformers," 2022, *arXiv:2210.17323*.

[5] J. Lin, J. Tang, H. Tang, S. Yang, G. Xiao, and S. Han, "AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration," in *Proc. Conf. Mach. Learn. Syst.*, vol. 28, 2025, pp. 12–17.

[6] W. Shao, M. Chen, Z. Zhang, P. Xu, L. Zhao, Z. Li, K. Zhang, P. Gao, Y. Qiao, and P. Luo, "OmniQuant: Omnidirectionally calibrated quantization for large language models," in *Proc. Int. Conf. Learn. Represent.*, 2023, pp. 45472–45496.

[7] A. Tseng, J. Chee, Q. Sun, V. Kuleshov, and C. De, "QuIP#: Even better LLM quantization with Hadamard incoherence and lattice codebooks," in *Proc. Int. Conf. Mach. Learn.*, 2024, pp. 48630–48656.

[8] Y. Chen, A. F. AbouElhamayed, X. Dai, Y. Wang, M. Andronic, G. A. Constantinides, and M. S. Abdelfattah, "BitMoD: Bit-serial mixture-of-datatype LLM acceleration," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Mar. 2025, pp. 1082–1097.

[9] T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer, "LLM.int8(): 8-bit matrix multiplication for transformers at scale," in *Proc. 36th Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 30318–30332.

[10] W. Huang, Y. Liu, H. Qin, Y. Li, S. Zhang, X. Liu, M. Magno, and X. Qi, "BiLLM: Pushing the limit of post-training quantization for LLMs," in *Proc. Int. Conf. Mach. Learn.*, 2024, pp. 20023–20042.

[11] W. Huang, H. Qin, Y. Liu, Y. Li, Q. Liu, X. Liu, L. Benini, M. Magno, S. Zhang, and X. Qi, "SliM-LLM: Salience-driven mixed-precision quantization for large language models," 2024, *arXiv:2405.14917*.

[12] J. Chee, Y. Cai, V. Kuleshov, and C. De, "QuIP: 2-bit quantization of large language models with guarantees," in *Proc. Neural Inf. Process. Syst.*, vol. 36, 2023, pp. 4396–4429.

[13] X. Wei, Y. Zhang, X. Zhang, R. Gong, S. Zhang, Q. Zhang, F. Yu, and X. Liu, "Outlier suppression: Pushing the limit of low-bit transformer language models," in *Proc. Neural Inf. Process. Syst.*, 2022, pp. 17402–17414.

[14] J. E. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, W. Chen, and C. Weizhu, "LoRA: Low-rank adaptation of large language models," in *Proc. Int. Conf. Learn. Represent.*, vol. 1, no. 2, 2021, p. 3.

[15] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient finetuning of quantized LLMs," in *Proc. Neural Inf. Process. Syst.*, 2023, pp. 10088–10115.

[16] Y. Xu, L. Xie, X. Gu, X. Chen, H. Chang, H. Zhang, Z. Chen, X. Zhang, and Q. Tian, "QA-LoRA: Quantization-aware low-rank adaptation of large language models," in *Proc. Int. Conf. Learn. Represent.*, 2023, pp. 52401–52418.

[17] H. Qin, X. Ma, X. Zheng, X. Li, Y. Zhang, S. Liu, J. Luo, X. Liu, and M. Magno, "Accurate LoRA-finetuning quantization of LLMs via information retention," in *Proc. Int. Conf. Mach. Learn.*, 2024, pp. 41498–41516.

[18] Z. Liu, B. Oguz, C. Zhao, E. Chang, P. Stock, Y. Mehdad, Y. Shi, R. Krishnamoorthi, and V. Chandra, "LLM-QAT: Data-free quantization aware training for large language models," in *Proc. Findings Assoc. Comput. Linguistics ACL*, 2024, pp. 467–484.

[19] D. Du, Y. Zhang, S. Cao, J. Guo, T. Cao, X. Chu, and N. Xu, "BitDistiller: Unleashing the potential of sub-4-bit LLMs via self-distillation," in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics*, 2024, pp. 102–116.

[20] J. Lee, J. Park, J. Kim, Y. Kim, J. Oh, J. Oh, and J. Choi, "AMXFP4: Taming activation outliers with asymmetric microscaling floating-point for 4-bit LLM inference," in *Proc. Findings Assoc. Comput. Linguistics, ACL*, Jul. 2025, pp. 14993–15013. [Online]. Available: https://aclanthology.org/2025.findings-acl.776/

[21] E. Choi, S. Song, W. Lim, and S. Yoo, "Rotate, clip, and partition: Towards W2A4KV4 quantization by integrating rotation and learnable non-uniform quantizer," 2025, *arXiv:2502.15779*.

[22] S. Ashkboos, M. L. Croci, M. G. Nascimento, T. Hoefler, and J. Hensman, "SliceGPT: Compress large language models by deleting rows and columns," in *Proc. Int. Conf. Learn. Represent.*, 2024, pp. 11682–11701.

[23] Fino and Algazi, "Unified matrix treatment of the fast Walsh–Hadamard transform," *IEEE Trans. Comput.*, vols. C–25, no. 11, pp. 1142–1146, Nov. 1976.

[24] B. He, T. Hofmann, L. Noci, D. Paliotta, and I. Schlag, "Understanding and minimising outlier features in transformer training," in *Proc. Adv. Neural Inf. Process. Syst. 37*, 2024, pp. 83786–83846. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2024/file/986292a930c3692168b177a770025ab3-Paper-Conference.pdf

[25] H. Touvron et al., "Llama 2: Open foundation and fine-tuned chat models," 2023, *arXiv:2307.09288*.

[26] (2024). *Llama 3 Model Card*. [Online]. Available: https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md

[27] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," in *Proc. Int. Conf. Learn. Represent.*, 2017. [Online]. Available: https://openreview.net/forum?id=Byj72udxe

[28] Y. Bisk, R. Zellers, R. L. Bras, J. Gao, and Y. Choi, "PIQA: Reasoning about physical commonsense in natural language," in *Proc. 34th AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 7432–7439.

[29] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, "HellaSwag: Can a machine really finish your sentence?" in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 4791–4800.

[30] K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi. (2019). *An Adversarial Winograd Schema Challenge At Scale*. [Online]. Available: https://api.semanticscholar.org/CorpusID:199370376

[31] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, "Think you have solved question answering? Try ARC, the AI2 reasoning challenge," 2018, *arXiv:1803.05457*.

[32] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal, "Can a suit of armor conduct electricity? A new dataset for open book question answering," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 2381–2391.

[33] C. Clark, K. Lee, M. Chang, T. Kwiatkowski, M. J. Collins, and K. Toutanova, "BoolQ: Exploring the surprising difficulty of natural yes/no questions," in *Proc. North Amer. Chapter Assoc. Comput. Linguistics*, 2019, pp. 2924–2936.

[34] M. Sap, H. Rashkin, D. Chen, R. LeBras, and Y. Choi, "SocialIQA: Commonsense reasoning about social interactions," 2019, *arXiv:1904.09728*.

[35] G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, and S. Han, "SmoothQuant: Accurate and efficient post-training quantization for large language models," 2022, *arXiv:2211.10438*.

[36] Y. Bai, X. Lv, J. Zhang, H. Lyu, J. Tang, Z. Huang, Z. Du, X. Liu, A. Zeng, L. Hou, Y. Dong, J. Tang, and J. Li, "LongBench: A bilingual, multitask benchmark for long context understanding," in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics*, Aug. 2024, pp. 3119–3137. [Online]. Available: https://aclanthology.org/2024.acl-long.172

[37] X. Wei, Y. Zhang, Y. Li, X. Zhang, R. Gong, J. Guo, and X. Liu, "Outlier suppression+: Accurate quantization of large language models by equivalent and effective shifting and scaling," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2023, pp. 1648–1665.

[38] OpenAI. (2025). *ChatGPT*. [Online]. Available: https://chatgpt.com

**SANGKI PARK** (Graduate Student Member, IEEE) received the B.S. degree in electronic engineering from Hanyang University, Seoul, South Korea, in 2017, where he is currently pursuing the Ph.D. degree in electronic engineering. His research interests include deep learning, model compression, and hardware acceleration.

**KI-SEOK CHUNG** (Member, IEEE) received the B.S. degree in computer engineering from Seoul National University, Seoul, South Korea, in 1989, and the Ph.D. degree in computer science from the University of Illinois at Urbana–Champaign, in 1998. He was a Senior Research and Development Engineer with Synopsys Inc., Mountain View, CA, USA, from 1998 to 2000; and a Staff Engineer with Intel Corporation, Santa Clara, CA, from 2000 to 2001. He was also an Assistant Professor with Hongik University, Seoul, from 2001 to 2004. Since 2004, he has been a Professor with Hanyang University, Seoul. His research interests include low-power embedded system design, multi-core architecture, image processing, reconfigurable processors and DSP design, SoC-platform-based verification, and system software for MPSoCs.