GWQ: Gradient-Aware Weight Quantization for Large Language Models

Anonymous ACL submission

Abstract

Large language models (LLMs) show impressive performance in solving complex language tasks. However, its large number of parameters present significant challenges for the deployment and application of the model on edge devices. Compressing large language models to low bits can enable them to run on resource-constrained devices, often leading to performance degradation. To address this problem, we propose gradient-aware weight quantization (GWQ), the first quantization approach for low-bit weight quantization that leverages gradients to localize outliers, requiring only a minimal amount of calibration data for outlier detection. GWQ retains the weights corresponding to the top 1% outliers preferentially at FP16 precision, while the remaining non-outlier weights are stored in a low-bit format. GWQ found experimentally that utilizing the sensitive weights in the gradient localization model is more scientific compared to utilizing the sensitive weights in the Hessian matrix localization model. Compared to current quantization methods, GWQ can be applied to multiple language models and achieves lower PPL on the WikiText2 and C4 dataset. In the zero-shot task, GWQ quantized models have higher accuracy compared to other quantization methods.GWQ is also suitable for multimodal model quantization, and the quantized Qwen-VL family model is more accurate than other methods. zero-shot target detection task dataset RefCOCO outperforms the current stat-of-the-arts method SPQR. GWQ achieves 1.2× inference speedup in comparison to the original model, and effectively reduces the inference memory.

1 Introduction

Large language models (LLMs) (Touvron et al., 2023a; Achiam et al., 2023; Almazrouei et al., 2023; Touvron et al., 2023b) based on Transformer (Vaswani, 2017) have demonstrated their outstanding capabilities in scenarios such as complex question and answer, language modeling (Brown, 2020), and so on. It can often generate more accurate answers with different inputs. Its ability to handle complex linguistic tasks is due to its huge pre-trained corpus (Kaplan et al., 2020) and an astronomical number of parameters (Chowdhery et al., 2023). Its outstanding performance is rapidly generating huge applications in people's lives. However, the huge memory consumption generated by its pre-training and inference phases leads to great difficulties in its application and deployment (Li et al., 2024a). Therefore, when deploying large language models, there are often huge GPU clusters to support the inference of the models (Xu et al., 2024).

In order to make large language models adaptable to resource-constrained edge devices (Li et al., 2024b; Huang et al., 2024), model compression (Ma et al., 2023; Gu et al., 2023) has become a common means to reduce the computational memory of models. Among them, post-training large language model quantization (Dettmers and Zettlemoyer, 2023) is a commonly used method to reduce the model inference overhead. Most current quantization methods compress the model to 3 or 4 bits (Dettmers et al., 2023; Frantar et al., 2022a), while trying to ensure that the model has as little loss of accuracy and performance as possible. Post-training quantization of pre-trained models reduces model inference overhead by allowing the number of parameter bits to decrease. This approach produces a loss of accuracy that affects the performance of the model in use. Therefore, to reduce the size of the model while ensuring its performance, it is necessary to design effective quantization algorithms (Lin et al., 2024). With effective quantization algorithms, researchers can make the performance of low-bit models infinitely close to that of 16-bit models.

OBQ (Frantar and Alistarh, 2022) indicates that when the pre-trained model has fully converged, the model should exhibit zero gradients. However,

our experiments reveal that the LLMs still generate gradients in response to different text inputs, indicating that the model has not completely converged and remains sensitive to the input. Inspired by this phenomenon, we propose a gradient-aware post-training weight-only quantization method called GWQ. GWQ is the first post-training quantization approach to utilize gradients to locate outliers in pre-trained models. It introduces two key innovations. First, GWQ selects 1% of the weights as outliers by looking at the magnitude of the gradient for the response to the calibration set, preserving these outliers in FP16 precision while quantizing the remaining 99% of the weights to 4 bits or 3 bits.

In summary, the primary contributions of this paper are as follows:

1) GWQ discovered that employing the first-order gradient to search for and locate sensitive weights is more rational than using the Hessian matrix for localization, and the model loss after quantization is also lower. Therefore, using a first-order gradient to locate outliers is more reasonable.

2) GWQ is the first accurate first-order gradient-aware post-training weight quantization method for pre-trained LLMs, requiring only a minimal quantity of calibration data to identify outliers efficiently.

3) GWQ outshines the current state-of-the-arts method SPQR on the wikitext and C4 datasets. The majority of zero-shot detection tasks in the multimodal dataset RefCOCO have reached the state-of-the-arts. Meanwhile, the quantified model has achieved a 1.2× acceleration compared to the original model and utilized less memory during inference.

2 Related Work

2.1 Post-training Quantization

Post-training quantization (PTQ) applies mainly to visual models (Gholami et al., 2022). For example, AdaRound (Nagel et al., 2020), AdaQuant (Hubara et al., 2020), BitSplit (Wang et al., 2024b) can be used for models with a small number of parameters. When models have a large number of parameters, they cause a significant loss of accuracy, and the reduction in memory overhead of the quantized model compared to the original model is not particularly significant. Most current quantization methods for models are improved by Optimal Brain Quantization (OBQ) (Frantar and Alistarh, 2022). OBQ is improved by the Optimal Brain Surgeon (OBS) (Hassibi et al., 1993; Singh and Alistarh, 2020; Frantar et al., 2021), both OBS and OBQ default to a model whose response to input is 0 at the end of training, but the OBQ method applies the idea of weight pruning to model quantization by quantifying the weights that have the least impact on the network and compensating for them with the remaining weights. Some current PTQ methods perform outlier search by reinforcement learning (Wang et al., 2024b) to retrieve the best outlier by an objective function on a calibration set with a very large sample size. However, the training process of reinforcement learning is not robust and is difficult to converge compared to other PTQ methods, making the calibration process very inefficient.

2.2 Large Language Model Quantization

Currently, large language model quantization is mainly divided into quantization-based training (Liu et al., 2023; Bondarenko et al., 2024; Zhu et al., 2023) and post-training quantization. Quantization during training is very difficult to apply due to the large amount of computational resources and overhead required, as well as the large amount of training data. The remaining methods such as ZeroQuant (Yao et al., 2022) utilize knowledge distillation to train one transformer layer at a time instead of training all transformer layers directly. However, poor quality of certain data can also affect the performance of the model after quantization (Wang et al., 2024a). Post-training quantization requires less data compared to quantization-aware training, and only a few thousand or even a few hundred calibration sets are needed to complete the retrieval of sensitive weights for the model. This greatly reduces the data cost and the computational cost of model quantization. Both GPTQ (Frantar et al., 2022a) and SPQR (Dettmers et al., 2023), quantization methods derived from the OBQ method, have shown superior performance in large language models. There are also methods such as AWQ (Lin et al., 2024), LLM.int8() (Dettmers et al., 2022) that determine the outlier of the model by searching for the activation of the model, and then quantify it with the corresponding method. However, the method of searching for outliers by activation has poor interpretability.

Table 1: **Gradient quantization of non-outliers with Hessian in comparison to the C4 dataset (Raffel et al., 2020).** We localize outliers by gradient and Hessian on the llama-2-hf series with the llama-3-hf model and quantify non-outliers with RTN (Nagel et al., 2020) to 4 bits.

		FP16(%) base on Gradients			FP16(%) base on Hessian		
PPL↓	FP16	0.1%	0.5%	1%	0.1%	0.5%	1%
Llama-2-7b-hf	6.97	13.4328	9.3408	7.1161	13.4358	9.3442	7.1187
Llama-2-13b-hf	6.47	11.5711	7.7271	6.5120	11.5755	7.7299	6.5142
Llama-3-8b-hf	8.99	18.4334	14.4928	11.5442	18.4390	14.4942	11.5483

3 The Proposed Method

The outlier search method based on Hessian matrix (Frantar et al., 2022a; Dettmers et al., 2023) is built upon the OBQ approach (Frantar et al., 2022b). The OBQ method retrieves the parameters that have the least impact on the model parameters by using the following equation:

$$\Delta E = \sum_i g_i \Delta w_i + \frac{1}{2} \sum_i h_{ii} \Delta w_i^2 + \frac{1}{2} \sum_{i \neq j} h_{ij} \Delta w_i \Delta w_j + O\left(\Delta w^3\right), \tag{1}$$

where $g_i = \frac{\partial E}{\partial w_i}$ is the first order gradient. In the OBQ framework, it is suggested that once a pre-trained model has fully converged, its gradients $g_i = \frac{\partial E}{\partial w_i}$ should ideally approach zero. However, our experiments demonstrate that LLMs continue to generate gradients in response to various text inputs. Bondarenko et al. (2023) denotes that the occurrence of outliers in large models arises from attention heads attempting to avoid updating hidden states. During this process, the softmax function magnifies the formation of strong outliers. Building on this observation, we hypothesize that when a well-trained LLM computes gradients for text input, it often focuses these gradients on irrelevant outliers as a mechanism to prevent hidden state updates. As shown in Fig. 1, compared to SPQR, GWQ demonstrates a sparser allocation of outliers. Specifically, the outliers identified are distributed either per-channel or per-row, which contrasts with the more dense and uniform distribution of outliers observed in SPQR. This sparsity in GWQ results in a more reasonable allocation, as it does not require to concentrate on the compensation of sensitive weight errors across every layer output. As shown in Tab. 1, with different proportions of outliers, the model using Hessian matrix search and quantized using RTN (Nagel et al., 2020) performs worse than the model using gradient search and quantized using RTN on the C4 dataset (Raffel et al., 2020), so it can be learned that it is more reasonable to use the gradient to do outliers localization.

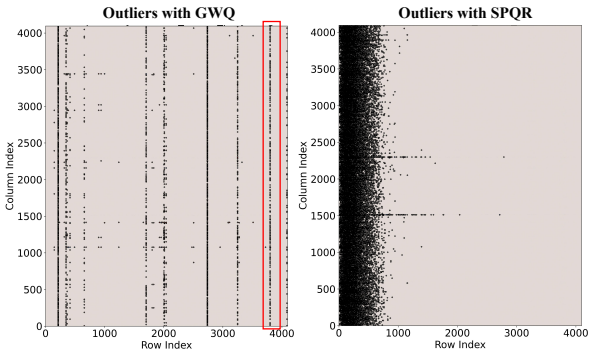

Figure 1: **Comparison of GWQ and SPQR outliers.** Taking query (Q) in Self-Attention as an example, the outliers localized by GWQ (left figure) are more sparse compared to SPQR (right figure), and the row distribution is more pronounced, which is more friendly to subsequent quantization.

3.1 Sensitive Weight Location

GWQ captures the absolute gradients with respect to LLM, acquired through back-propagation, the detail is shown in Alg. 1. We disabled bias in all selected linear layers, since the bias term does not get multiplied by inputs in the same way as weights do. This can cause the weight gradients to be unrepresentative of the actual importance of the weights, leading to a disproportionate influence on the MSE loss calculation. As it shown in Fig. 2, the calibration dataset is represented as D_c, the weights of LLM as W, the loss function of LLM as $\mathcal{L} = (W; D_c)$. The gradients g to the LLM is:

$$g = \bigtriangledown_W \mathcal{L} = \bigtriangledown_W \mathcal{L}(W; D_c). \tag{2}$$

In order to locate the sensitivities as outliers, we calculate loss $\mathcal{L}(W_Q)$ after model quantization based on Eq. (3).

$$\mathcal{L}(W_Q) \simeq \mathcal{L}(W) - g^\top (W - W_Q). \tag{3}$$

In order to search for quantitatively optimal outliers, we define the quantization process as an opti-

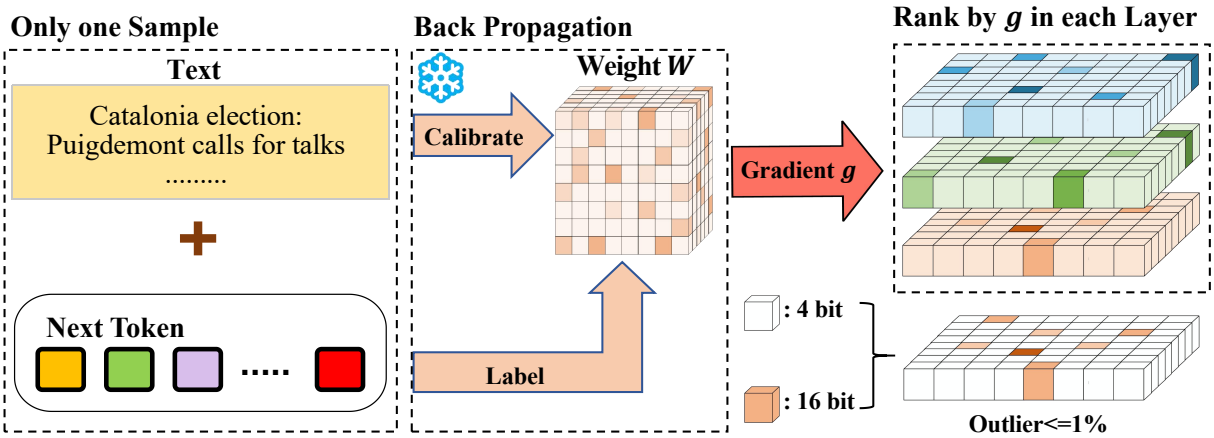

Figure 2: **Sensitive weight Location.** GWQ utilizes a single calibration sample to calibrate and backpropagates this sample with the subsequent token from the large model's output as the target label. By halting the backpropagation process, GWQ is able to capture the model gradient g prior to the update of the weights W. Within each layer, the gradients associated with each weight block are sorted in relation to their respective weights, and the top 1% with the largest $|g|$ are identified as the model's outliers, which are the weights that are particularly sensitive.

mization problem whose process can be expressed as follows:

$$Q\left(W\right) = arg\min_{Q} \|W - W_Q\|_2^2 . \tag{4}$$

Therefore the final weight optimization formula can be expressed as:

$$Q\left(W\right) = arg\min_{Q} \bigtriangledown_W \mathcal{L}(W; D_c)(W - W_Q). \tag{5}$$

Algorithm 1 GWQ Gradient Accumulation

func **get_gradient**(M, X)

$a := \emptyset$ ▷ Set of accumulated gradients
for $i = 1, \ldots,$ dim(X) **do**
  o := M($X_i[0]$) ▷ model output
  l := CrossEntropy(o,$X_i[1]$)
  l.backward()
  a := accum_grad(M)
  M.zero_grad()
for $j = 1, \ldots,$ dim(a) **do**
  $a_j := \frac{a_j}{\dim(X)}$
  $M_g := a_j$ ▷ update the model gradients
return M

func **accum_grad**(M, a)

q := [self_attn, mlp] //modules to be quantized
for $i = 1, \ldots,$ dim(a) **do**
  $a_i := a_i + M_{a_i}$ ▷ accumulate the gradients of each module in the model
return a

3.2 Gradient Aware Quantization

To quantize the LLM weight W, we first compute the scale s and the zero point z, which are calculated as depicted in Eq. (6) and (7), where β represents the group size (set to 16 in this case) and b is the bit-width for quantization. The detail is shown in Alg. 2.

$$s = \frac{\max(W) - \min(W)}{2^{b-1}}, \tag{6}$$

$$z = \frac{-\min W}{s}. \tag{7}$$

We select the weight by absolute values of the weight gradients $|W_{grad}|$, the largest 1% of them are filtered as outliers, and the rest are used as quantization weights. This process can be expressed as:

$$O = \mathbb{I}\left(|W_{grad}| \geq \text{quantile}(\alpha_1)\right), \tag{8}$$

where α_1 is the specific quantile value of the magnitude of weight gradients, which are used to identify outliers matrix O.

The quantization and dequantization operations are carried out per channel within each group to assess the quantization error. The formulas for quantization and dequantization are provided below (Eq. (9) and (10)), and any identified outliers are excluded from the error computation (Eq. (8)):

$$q = round(\frac{W}{s}) + z. \tag{9}$$

The quantized weight $Q(W)$ can be denoted as:

$$Q(W) = s \times (q - z). \tag{10}$$

Algorithm 2 GWQ Quantization Algorithm

func **GWQ_Quantize**(M, X, b, β, p)

Input: M – the model to be quantized
$X \in \mathbb{R}^{n \times d}$ – calibration data
b – base number of quantization bits
β – quantization group sizes
p – global outlier percentage

M :=get_gradient(M, X)
L :=M.layers
n := dim(layers)
l_{i_s} := L_i.sublayers
for $i = 1, \ldots, \dim(L)$ **do**
  for $j = 1, \ldots, \dim(l_{i_s})$ **do**
    $W :=$
GWQ_Sublayer_Quantize($l_{i_s}.W$,
$l_{i_s}.W_g, b, \beta, 1-p$)
    $l_{i_s}.W := W$
  return M

func **GWQ_Sublayer_Quantize**(W, W_g, b, β, p_s)

Input: $W \in \mathbb{R}^{m \times n}$ – weight matrix
$W_g \in \mathbb{R}^{m \times n}$ – gradient of weight matrix
p_s – sublayer outlier percentage

Q := int_matrix(m,n) //temporary quantized weight
$O := \mathbb{I}\left(|W_g| \geq \text{quantile}(\alpha_1)\right)$
for $i = 1, \beta 1, 2\beta 1, \ldots, n$ **do**
  $\vec{s}, \vec{z}$:= fit_quantizer($W_{:,i:i+\beta}$, $O_{:,i:i+\beta}$)
  for $j = i, \ldots, i+\beta$ **do**
    $Q_{:,j}$:= quantize($W_{:,j}, \vec{s}, \vec{z}$)
    $\vec{w_q}$:= dequantize($Q_{:,j}, \vec{s}, \vec{z}$)
    $W_{:,j} := \vec{w_q}(1 - O_{:,j}))$
return W

func **fit_quantizer**(M, β)

  m_i := flatten(M)
  s_i := vectors(O)
  for $i = 1, \beta_1, \beta_2, \ldots, \dim(m)$ **do**
    $s_i := \frac{\max(m) - \min(m)}{\dim(m)}$
    $z_i := \min(\frac{m - s_i}{s_i})$
  return s, z

func **quantize**(W, s, z)

  return $\lfloor \frac{W}{s} + z + 0.5 \rfloor$

func **dequantize**(Q, s, z)

  return $s \cdot (Q - z)$

4 Experiments

Overview. First, we validate the superiority of our method over other quantization approaches on the Llama-2 family (Touvron et al., 2023b), providing detailed compression metrics for comparison. Additionally, we present the results of our method compressed to 2 bits versus 3 bits, confirming its reliability at lower bit precision. Following this, we demonstrate the quantization performance of our method on other models, showcasing its generalizability. Finally, we present zero-shot quantization results on multimodal models to verify the multimodal context generalizability of our approach.

4.1 Experiments Setting

Setups. The language-only models we selected include Llama2-7B and 13B (Touvron et al., 2023b), Falcon-7B (Penedo et al., 2023), and Mistral-7B-v0.1 (Jiang et al., 2023). These models were quantized using the first sample of the RedPajama dataset. To evaluate Perplexity (PPL) and Accuracy (Acc.), we utilize the WikiText2 (Merity et al., 2016) and C4 (Raffel et al., 2020) validation sets. All experiments were conducted on an NVIDIA A100 (80G) GPU.

Baselines. Our primary baselines include AWQ (Lin et al., 2024) and GPTQ (Frantar et al., 2022a), while we benchmark our approach against the state-of-the-arts SPQR (Dettmers et al., 2023). To ensure that the average bit-width of the model remains below 4 bits—surpassing GPTQ and AWQ in compression—we quantize the non-outlier weights at 4-bit and 3-bit precision. These configurations are denoted GWQ-O for 4-bit quantization, GWQ-R for 3-bit quantization.

4.2 Main Result

We conduct separate evaluations for both pure large language models and large multimodal models. For the large language models, we assess the accuracy of the quantized model in terms of perplexity, zero-shot tasks, and quantized inference memory overhead and latency. In addition to evaluating performance on linguistic tasks for the large multimodal models, we also test the zero-shot target detection task across multiple datasets.

Language Model. As shown in Tab. 2 and Tab. 3, we first evaluate the perplexity of the quantized model on the WikiText2 (Merity et al., 2016) and C4 (Raffel et al., 2020) datasets on the Llama2-7B and 13B models. Additionally, we assess the

Table 2: **Perplexity of Llama-2-hf family and Llama-3-8B-hf on language task datasets.** GWQ outperforms other quantization methods, especially when the average bit-width of the model is significantly lower.

			Llama-2-7B-hf		**Llama2-13B-hf**		**Llama-3-8B-hf**	
PPL↓	Avg_bit	cab_num	WikiText2	C4	WikiText2	C4	WikiText2	C4
Original	16	-	5.47	6.97	4.88	6.47	6.23	8.99
+ GPTQ	4	1024	5.83	7.79	5.12	7.72	8.21	11.48
+ AWQ	4	512	5.60	7.70	4.97	7.10	6.64	10.69
+ SPQR	4.63	1024	5.53	7.02	4.93	6.50	6.42	9.23
+ SPQR	3.98	1024	5.55	7.06	4.95	6.53	6.45	9.26
+ GWQ-O (Ours)	4.63	**1**	**5.51**	**7.01**	**4.91**	**6.48**	**6.40**	**9.21**
+ GWQ-R (Ours)	3.98	**1**	**5.53**	**7.04**	**4.94**	**6.52**	**6.43**	**9.25**

Table 3: **Performance of the Llama-2-hf family and Llama-3-8B-hf on zero-shot tasks on language task datasets.** GWQ demonstrates a lower loss of accuracy compared to other quantization methods.

			Llama-2-7B-hf		**Llama-2-13B-hf**		**Llama-3-8B-hf**	
Acc.(%)↑	Avg_bit	cab_num	WikiText2	C4	WikiText2	C4	WikiText2	C4
Original	16	-	61.01	56.77	62.88	57.90	58.34	51.71
+ GPTQ	4	1024	59.88	54.74	61.91	56.18	43.78	47.75
+ AWQ	4	512	60.54	55.16	62.53	56.44	57.07	48.96
+ SPQR	4.63	1024	60.80	56.63	62.71	57.78	57.70	51.20
+ SPQR	3.98	1024	60.56	55.18	62.55	56.45	57.09	48.97
+ GWQ-O (Ours)	4.63	**1**	**60.83**	**56.65**	**62.73**	**57.80**	**57.71**	**51.22**
+ GWQ-R (Ours)	3.98	**1**	**60.58**	**55.20**	**62.56**	**56.47**	**57.11**	**48.99**

Table 4: **Perplexity of Qwen-VL family models on language tasks datasets.** GWQ quantized model has a lower PPL compared to other current quantization methods, and the PPL also remains essentially constant at lower bits.

		QWEN-VL		**QWEN-VL-chat**	
PPL↓	Avg_bit	WikiText2	C4	WikiText2	C4
Original	16	8.21	10.10	9.85	11.87
+ GPTQ	4	9.31	11.97	11.50	13.59
+ AWQ	4	8.69	10.98	11.02	12.88
+ SPQR	4.63	8.54	10.25	10.13	12.02
+ GWQ-O (Ours)	4.63	**8.24**	**10.10**	**9.86**	**11.88**
+ GWQ-R (Ours)	3.63	**8.53**	**10.42**	**10.94**	**12.62**

Table 5: **Performance of the Qwen-VL family on the zero-shot tasks on language tasks datasets.** GWQ has lower loss of accuracy compared to other quantization methods.

		QWEN-VL		**QWEN-VL-chat**	
Acc.(%)↑	Avg_bit	WikiText2	C4	WikiText2	C4
Original	16	54.18	49.51	52.06	48.07
+ GPTQ	4	52.07	47.08	49.83	46.09
+ AWQ	4	53.58	48.12	50.77	47.43
+ SPQR	4.63	54.05	49.26	51.99	47.35
+ GWQ-O (Ours)	4.63	**54.06**	**49.50**	**52.03**	**48.07**
+ GWQ-R (Ours)	3.63	**53.87**	**49.32**	**52.01**	**49.98**

Table 6: **Performance of the Qwen-VL family on the zero-shot tasks on target detection datasets.** In the three RefCOCO datasets, GWQ is more accurate compared to SPQR for the same bits, achieving less quantization loss.

		QWEN-VL			**QWEN-VL-chat**		
Acc.(%)↑	Avg_bit	test-A	test-B	val	test-A	test-B	val
Original	16	92.26	85.34	89.36	92.27	84.51	88.55
+ GPTQ	4	90.61	83.14	87.13	90.33	82.27	86.67
+ AWQ	4	91.33	84.51	88.01	91.14	83.61	87.77
+ SPQR	4.63	92.17	84.87	88.97	91.76	84.20	88.48
+ SPQR	3.98	91.33	84.51	88.01	91.15	83.61	87.77
+ GWQ-O (Ours)	4.63	**92.18**	**84.88**	**88.98**	**91.78**	**84.22**	**88.49**
+ GWQ-R (Ours)	3.98	**91.34**	**84.52**	**88.03**	**91.16**	**83.62**	**87.79**

Table 7: **Comparison of speedup and memory consumption of different quantization methods.** GWQ significantly improves the inference speed and reduces the memory required compared to the original model.

		Llama2-7B-hf		Llama2-13B-hf	
Methods	Avg_bit	Speedup(↑)	Memory(↓)	Speedup(↑)	Memory(↓)
Original	16	×1	12.8	×1	23.6
+ GPTQ	4	×2.0	3.1	×2.0	6.7
+ AWQ	4	×2.1	3.1	×2.1	6.7
+ SPQR	4.63	×1.2	N/A	×1.2	N/A
+ GWQ (Ours)	4.63	×1.2	4.1	×1.2	7.1

Figure 3: **Performance of Different Quantized Qwen-VL.** GWQ performs well compared to other quantization methods in the zero-shot grounding detection task.

models' ability to zero-shot on these three datasets and compare the results to several baseline methods. GWQ achieves a lower average bit-width than other quantization methods by applying more aggressive compression to the non-outlier weights. Compared to other quantization approaches, GWQ yields a lower PPL on all datasets under identical bit compression. For the zero-shot task, the Acc. of the quantized model of GWQ is significantly higher than that of other quantization methods. indicating that GWQ results in less performance degradation. Notably, under extreme compression, GWQ sustains comparable performance to 4-bit quantization, even at 3-bit.

Multimodal Models. For the multimodal model, we evaluate the performance of the compressed model in language tasks and multimodal tasks with the Qwen-VL family (Bai et al., 2023a,b). Similar to the language models, we evaluate the language generation capabilities of the quantized model on the WikiText2, C4 dataset. The multimodal tasks are mainly focused on zero-shot target detection. We choose the RefCOCO (Kazemzadeh et al., 2014; Yu et al., 2016) dataset to evaluate the zero-shot capability of the quantized model compared to other quantization methods. Intersection over Union IoU, which is denoted as:

$$IoU = \frac{Box_a \cap Box_b}{Box_a \cup Box_b}, \tag{11}$$

where Box_b denotes the human-labeled correct result in the dataset labeling (Ground-Truth Box) and Box_a denotes the result predicted by the algorithm (Predicted Box).

In the experiment setting, the threshold of IoU is 0.5, so the target detection accuracy Acc. can be defined as:

$$Acc. = \frac{correct_num}{total_num}, \tag{12}$$

where $correct_num$ is the number of samples detected correctly and $total_num$ is the total number of samples.

As it shown in Tab. 4 and Tab. 5, quantized qwen-vl family model of GWQ outperforms the current stat-of-the-arts in both the language task and the zero-shot test, showing better performance. The GWQ quantized MLLM performs better on the verbal task compared to other quantization methods, so we will use the quantized model to test it on the visual grounding task. As it shown in Tab. 6, in the zero-shot grounding detection task, both GWQ-quantified Qwen-VL family models with currently outperform the current stat-of-the-arts on most of the datasets, achieving a generalization of the multimodal model, The performance is shown in Fig. 3, for the task of visual grounding detection of long context, the GWQ post-quantization model is more advantageous than SPQR post-quantization.

Model Efficiency Comparison. For the model throughput (tokens/s), the speedup rate can be obtained by calculating the time required to generate 512 tokens after quantization and comparing it with the original model. For the model inference overhead (G), we use the peak value when the model generates 512 tokens as the maximum inference memory by recording it.

As illustrated in Tab. 7, the GWQ quantized model can achieve about 1.2× speedup relative to the original model, which is on par with the current mixed-accuracy method SPQR. Compared to the full-precision quantization method, GWQ results in higher inference latency due to floating-point conversion during inference. Compared to the original model, GWQ consumes less memory

Table 8: **Performance of GWQ at 4bit quantization for different number of calibration samples.** The quantization loss of the model essentially does not vary with the number of different calibration samples, allowing for low-loss quantization with fewer samples.

	num_sample=1		num_sample=4		num_sample=8		num_sample=12	
PPL↓	WikiText2	C4	WikiText2	C4	WikiText2	C4	WikiText2	C4
Llama2-7B-hf	5.51	7.01	5.50	7.01	5.51	7.00	5.50	6.99
Llama2-13B-hf	4.91	6.48	4.90	6.49	4.91	6.47	4.91	6.48
Llama-3-8B-hf	6.40	9.21	6.39	9.20	6.38	9.20	6.40	9.20

Table 9: **Performance of GWQ at 4bit quantization for different sample of calibration samples.** The model quantization loss is essentially invariant to changes in the quantization sample and is less dependent on sample type.

	sample one		sample two		sample three		sample four	
PPL↓	WikiText2	C4	WikiText2	C4	WikiText2	C4	WikiText2	C4
Llama2-7B-hf	5.51	7.01	4.91	6.97	5.47	6.97	5.47	6.97
Llama2-13B-hf	4.91	6.48	4.88	6.47	4.88	6.47	4.88	6.47
Llama-3-8B-hf	6.40	9.21	5.25	7.75	5.25	7.75	5.25	7.75

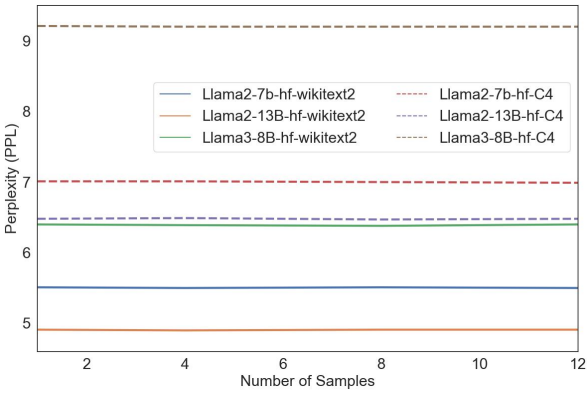

Figure 4: **Effect of calibration sample on quantification.** For different numbers of calibration sets, the PPL of the quantized model tends to be essentially constant on WikiText2 and C4. Therefore, the effect of the number of calibration set on the quantization effect is negligible.

during inference, which makes it easier to deploy on edge devices.

4.3 Ablation Study

In this section, we examine the impact of varying the number of calibration samples on the performance of our method. Tab. 8 presents the results for the calibration sample sizes 1, 4, 8 and 12, evaluated on the WikiText2 and C4 datasets. Tab. 8 and Fig. 4 indicate that increasing the number of calibration samples has minimal impact on the PPL of the quantized model. This suggests that the size of the calibration set has little influence on the quantization performance of GWQ. Compared to most of the current quantization methods, which require large datasets, GWQ has a lower calibration cost and can achieve calibration with fewer samples.

To further investigate the impact of different calibration samples, we selected four distinct samples, each of which was used as a calibration set to quantize the model. According to Tab. 9, when different calibration samples are used to quantize the same model, the resulting PPL values on the same dataset remain similar, indicating that the choice of calibration samples has a negligible effect on the quantization results. Unlike AWQ, GWQ can be well adapted to the calibration sets of different samples, and the quantized model performs well even if the calibration sets differ greatly.

5 Conclusion

This paper introduces GWQ, a weight-only post-training mixed-precision quantization approach based on first-order gradients. GWQ preserves the weights corresponding to the top 1% of outliers at FP16 precision, while storing the remaining non-outlier weights in a lower-bit format. GWQ found that it is easier to obtain more accurate outliers using the first-order gradient to locate the sensitive weights of the pre-trained model than the current search for model outliers via the Hessian matrix. GWQ achieves the lowest quantization loss compared to current state-of-the-arts methods by utilizing only a single calibration sample. Considerable results are also obtained by GWQ on multimodal models, which can achieve better results on most of the zero-shot target detection tasks compared to the current stat-of-the-art methods. GWQ achieves 1.2× inference acceleration compared to the original model internship and uses less memory in the inference process.

6 Limitation

GWQ gets gradients via back propagation and ranks them accordingly. However, in models such as OPT, the activation function is ReLU, and the correct gradients cannot be computed because of gradient vanishing. Additionally, backpropagation demands substantial memory resources, a single A100 80G GPU is insufficient to handle the large computational requirements. Furthermore, since GWQ employs mixed-precision quantization, it is less hardware-friendly compared to methods like AWQ, resulting in higher inference latency after quantization. We will continue to refine and optimize the GWQ methodology.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, et al. 2023. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023a. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023b. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *Preprint*, arXiv:2308.12966.

Yelysei Bondarenko, Riccardo Del Chiaro, and Markus Nagel. 2024. Low-rank quantization-aware training for llms. *arXiv preprint arXiv:2406.06385*.

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. 2023. Quantizable transformers: Removing outliers by helping attention heads do nothing. *Advances in Neural Information Processing Systems*, 36:75067–75096.

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

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

Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. 2023. Spqr: A sparse-quantized representation for near-lossless llm weight compression. *arXiv preprint arXiv:2306.03078*.

Tim Dettmers and Luke Zettlemoyer. 2023. The case for 4-bit precision: k-bit inference scaling laws. In *International Conference on Machine Learning*, pages 7750–7774. PMLR.

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

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022a. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022b. Optq: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*.

Elias Frantar, Eldar Kurtic, and Dan Alistarh. 2021. M-fac: Efficient matrix-free approximations of second-order information. *Advances in Neural Information Processing Systems*, 34:14873–14886.

Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. 2022. A survey of quantization methods for efficient neural network inference. In *Low-Power Computer Vision*, pages 291–326. Chapman and Hall/CRC.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2023. Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*.

Babak Hassibi, David G Stork, and Gregory J Wolff. 1993. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, pages 293–299. IEEE.

Wei Huang, Haotong Qin, Yangdong Liu, Yawei Li, Xianglong Liu, Luca Benini, Michele Magno, and Xiaojuan Qi. 2024. Slim-llm: Salience-driven mixed-precision quantization for large language models. *arXiv preprint arXiv:2405.14917*.

Itay Hubara, Yury Nahshan, Yair Hanani, Ron Banner, and Daniel Soudry. 2020. Improving post training neural quantization: Layer-wise calibration and integer programming. *arXiv preprint arXiv:2006.10518*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. 2014. Referitgame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yuanchun Li, Hao Wen, Weijun Wang, Xiangyu Li, Yizhen Yuan, Guohong Liu, Jiacheng Liu, Wenxing Xu, Xiang Wang, Yi Sun, et al. 2024a. Personal llm agents: Insights and survey about the capability, efficiency and security. *arXiv preprint arXiv:2401.05459*.

Zhengang Li, Alec Lu, Yanyue Xie, Zhenglun Kong, Mengshu Sun, Hao Tang, Zhong Jia Xue, Peiyan Dong, Caiwen Ding, Yanzhi Wang, et al. 2024b. Quasar-vit: Hardware-oriented quantization-aware architecture search for vision transformers. In *Proceedings of the 38th ACM International Conference on Supercomputing*, pages 324–337.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100.

Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. 2023. Llm-qat: Data-free quantization aware training for large language models. *arXiv preprint arXiv:2305.17888*.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. 2020. Up or down? adaptive rounding for post-training quantization. In *International Conference on Machine Learning*, pages 7197–7206. PMLR.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

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

Sidak Pal Singh and Dan Alistarh. 2020. Woodfisher: Efficient second-order approximation for neural network compression. *Advances in Neural Information Processing Systems*, 33:18098–18109.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Yifei Wang, Jizhe Zhang, and Yisen Wang. 2024a. Do generated data always help contrastive learning? *arXiv preprint arXiv:2403.12448*.

Yingchun Wang, Song Guo, Jingcai Guo, Yuanhong Zhang, Weizhan Zhang, Qinghua Zheng, and Jie Zhang. 2024b. Data quality-aware mixed-precision quantization via hybrid reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*.

Mengwei Xu, Wangsong Yin, Dongqi Cai, Rongjie Yi, Daliang Xu, Qipeng Wang, Bingyang Wu, Yihao Zhao, Chen Yang, Shihe Wang, et al. 2024. A survey of resource-efficient llm and multimodal foundation models. *arXiv preprint arXiv:2401.08092*.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. 2022. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35:27168–27183.

Licheng Yu, Patric Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. 2016. Modeling context in referring expressions. In *Springer International Publishing*.

Xuekai Zhu, Biqing Qi, Kaiyan Zhang, Xinwei Long, Zhouhan Lin, and Bowen Zhou. 2023. Pad: Program-aided distillation can teach small models reasoning better than chain-of-thought fine-tuning. *arXiv preprint arXiv:2305.13888*.