decoupleQ: Towards 2-bit Post-Training Uniform Quantization via decoupling Parameters into Integer and Floating Points

Anonymous Author(s)
Affiliation
Address
email

Abstract

Quantization emerges as one of the most promising compression technologies for deploying efficient large models in recent years. However, existing quantization schemes suffer from significant accuracy degradation at very low bits, or require some additional computational overhead when deployed, making it difficult to be applied to large-scale applications in industry. In this paper, we propose decoupleQ, achieving a substantial increase in model accuracy, especially at very low bits.

decoupleQ abandons the traditional heuristic quantization paradigm and decouples the model parameters into integer and floating-point parts, then transforming the quantization problem into a mathematical constrained optimization problem, which is then solved alternatively by off-the-shelf solution methods. decoupleQ gets rid of any tricks for dealing with outliers, sensitive channels, etc., and focuses only on the basic optimization objective to achieve high model accuracy on extreme low bit quantization. Quantization via decoupleQ is linear and uniform, making it hardware-friendlier than non-uniform counterpart, and enabling the idea to be migrated to high-bit quantization to enhance its robustness.

decoupleQ has achieved comparable accuracy as fp16/bf16 for 2-bit quantization of large speech models in our company. The code (including the W2 CUDA kernels) is attached and will be made public.

1 Introduction

Serving large models (1; 2; 37; 38) in industry is budget-consuming because of the huge computational, IO and storage cost. Model compression (10; 11; 16) has therefore become a necessity to alleviate this pain. Among which, Post-Training Quantization (PTQ) (9; 26) has gained more and more popularity among researchers and engineers because it does not require heavy GPU-hours training with labeled datasets.

However, previous quantization schemes remain confined within the traditional heuristic quantization paradigm, e.g., how to deal with outliers (33; 35), how to deal with sensitive channels (6), how to determine the clipping range (29), and so on. These methods have achieved some success, but the quantization at extreme low bit often suffers from significant accuracy degradation, thus failing to meet the launching requirements of industrial practice. There are also some other options to mitigate the accuracy loss. QuIP (4) pushes the accuracy limits of 2-bit quantization and can achieve performance close to fp16/bf16. However, compared to traditional quantization schemes, its inference imposes an additional burden due to the need to multiply two random orthogonal matrices to dequant the weights. N2UQ (20) fit the real-value distribution with non-uniform grids then quantize them into equidistant output levels. But it need to train to get the input thresholds. SpQR (7)

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

and SqueezeLLM (14) use mixed-precision quantization or non-uniform scheme to safeguard the important channels, but they need customized hardware support.

In order to alleviate the above pains in industry, we proposed decoupleQ, which completely abandons the traditional heuristic quantization paradigm and instead decouples the model parameters into integer and floating point parts, then transforming the quantization problem into a mathematical constrained optimization problem, which is then solved alternatively by off-the-shelf solution methods. The integer part contains the main weights of the model, and the floating-point part contains scales and zero points induced via quantization. decoulpeQ starts from an abstract objective function and thus does not need any tricks to deal with the minutiae of traditional quantization paradigm, such as outlier, salient weights (19), and so on. Quantization via decoupleQ is linear and uniform, making it hardware-friendlier than non-uniform counterpart, and enabling the idea to be migrated to high-bit quantization to enhance its robustness.

decoupleQ contains two stages: 1. layer-wise minimization, defined in Eq. 1, is used to optimize the integer part and the floating-point part; 2. block-wise minimization, defined in Eq. 2, is used to further optimize the floating-point part while freezing the integer part[1].

Layer-wise minimization is to minimize the ℓ^2 loss of the outputs between pre- and post-quantization for a linear layer:

$$\min_{\widetilde{W}} \|X\widetilde{W} - XW_0\|_2^2 \tag{1}$$

where $X \in \mathbb{R}^{batch \times d_{in}}$ is the input of this layer, $W_0 \in \mathbb{R}^{d_{in} \times d_{out}}$ is the pre-trained full precision weight, d_{in} and d_{out} are the input and output dimensions respectively. The objective is to find a matrix $\widetilde{W}$ with quantized-then-dequantized elements to minimize Eq. 1.

Some works (4; 8; 9; 13; 25) started from Eq. 1 and achieved some success, but they still haven't thought outside the box of traditional quantization. GPTQ series (8; 9) fake-quantize the first element of W_0 and then update the the remaining elements so as to keep Eq. 1 minimized. This process is then continued element by element until all elements are fake-quantized. However, on the one hand, they do not give any indication of how scale and zero point should be calculated, and on the other hand, the optimization problem formulated for updating the remaining elements is unconstrained (explained in detail later). decoupleQ models Eq. 1 as a constrained optimization problem, as shown in Eq. 6. It no longer needs to pay attention to some of the minutiae unique to quantization, such as outliers, clipping threshold, etc., but abstracts the essence of the problem from a higher level.

In the second stage, block-wise minimization is used to further improve the model accuracy:

$$\min \|\widetilde{\text{Block}}(X) - \text{Block}(X)\|_2^2 \tag{2}$$

where $\widetilde{\text{Block}}(\cdot)$ is a common transformer block (32) with quantized weights. In this stage, we freeze the integer part of the weights, and train the scales, zero points and norm layers.

decoupleQ implements 2-bit uniform quantization and achieves state-of-the-art accuracy in Llama-1/2 (30; 31). Like traditional uniform quantization, decoupleQ does not incur additional inference burden and only requires a linear transformation to convert the quantized weights into floating point ones.

Our main highlights are summarized as follows:

- **New insight:** We abandoned the traditional quantization paradigm, and no longer need to focus on some of the minutiae unique to quantization, but abstracts the essence of the problem from a higher level and transforms it into a constrained optimization problem.
- **Extreme low-bit:** decoupleQ achieves 2-bit uniform quantization with performance matching fp16/bf16 for industrial applications in the ASR model in our company, and we will also release the W2A16 CUDA kernel as one of our core contribution.
- **Extensibility:** As a bonus, if labeled datasets are available, the idea of decoupleQ can be easily extended to supervised fune-tuning (sft) to further improve model accuracy, or the adaptation to the downstream sub-tasks.

[1]We define the term "layer" as a linear transformation, "block" as a common transformer block containing the multi-head attention, feed forward, and some layer norm.

2 Related Works

Quantization can be roughly divided into Quantization Aware Training (QAT) (21; 33) and Post-Training Quantization (PTQ) (4; 35). In this paper, we focus on weight-only quantization in PTQ, and we will only summarize a few works that are closely related to our work.

PTQ is commonly used for LLM quantization because it does not require a lot of GPU hours of training with labeled datasets. AdaRound (25) and BRECQ (18) start from the rounding operation and explore whether to round up or down is better. SqQR (7) and OWQ (17) use mixed-precision quantization strategy to protect sensitive parameters, while AWQ (19) opts for scaling up the weights of sensitive channels to reduce the loss of quantization of sensitive channels. OmniQuant (29) use gradient decent to optimize for the weight clipping threshold and the rescale factors. In decoupleQ, we abandon patchwork solutions and transform the quantization into a principled traditional optimization problem by decoupling the model parameters into integer and floating-point parts.

GPTQ (9) is an influential work, and it quantizes the current weights and then updates the remaining weights to minimize the ℓ^2 loss of the output of the layer between pre- and post-quantization. As we will see later, this update actually approximates much, and GPTQ does not optimize for the scale and zero point reduced by quantization.

QALora (36) also decouples model parameters at a certain level and uses labeled datasets to fine-tune the zero points. decoupleQ takes this idea a step further, optimizing the scales, zero points and norm layers with supervised fine-tuning, while freezing the integer weights.

3 Methods

3.1 Preliminaries

For a linear layer with input dimension d_{in} and output dimension d_{out}, quantization maps the weights with high-precision into discrete level, and the previous scheme can be described as follows:

$$\widehat{W} = \text{clip}(\lfloor \frac{W_0 - z}{s} \rceil, \alpha, \beta) \tag{3}$$

$$\widetilde{W} = \widehat{W} * s + z \tag{4}$$

where $W_0 \in \mathbb{R}^{d_{in} \times d_{out}}$ is the pre-trained full precision weights, s and z are the scale and zero point (what we call floating-point part above), $\lfloor \cdot \rceil$ is the round-to-nearest function, $\widehat{W} \in \mathbb{R}^{d_{in} \times d_{out}}$ is the quantized integer-point matrix (what we call integer part above), $\widetilde{W}$ is the de-quantized floating-point matrix, α and β are the lower and upper bounds of the range of integer representations, respectively. For example, in 2-bit weight only linear quantization scheme, the value of each entry of $\widehat{W}$ is limited to one of $\{-2, -1, 0, 1\}$, and $\alpha = -2$, $\beta = 1$ in this case. To get the values of $\widetilde{W}$, previous methods (8; 9) show that layer-wise ℓ^2 loss between the outputs pre- and post-quantization is well related to the model accuracy, i.e., to optimize the following objective function,

$$\arg\min_{\widetilde{W}} \|X\widetilde{W} - XW_0\|_2^2 = \text{tr}\{(\widetilde{W} - W_0)^T H (\widetilde{W} - W_0)\} \tag{5}$$

where $X \in \mathbb{R}^{batch \times d_{in}}$ is the input of this linear layer, generated by a small set of calibration dataset, and $H = X^T X$.

In the very low-bit quantization regime, the model accuracy can be further improved via finer-grained grouping. This would impose additional overhead on inference. For example, when groupsize $= 64$, it imposes an average overhead of 0.5 bit per element (FP16/BF16 for scale s and zero point z). The extra overhead is acceptable compared to the model accuracy gain.

3.2 decoupleQ

When a model is quantized, only the integer part $\widehat{W}$ and the floating-point part (s, z) in Eq. 4 are delivered to the downstream inference engine, and the inference process does not need to know how $\widehat{W}$ and (s, z) are obtained at all. That is, if we can find the values of $\widehat{W}$ and (s, z) to minimize Eq. 5 by other methods, then we don't need to use Eq. 3. So, we can decouple the model parameters into integer part $\widehat{W}$ and floating point part (s, z), which are then optimized alternatively via off-the-shelf

solution methods. decoupleQ views the process of solving for $\widehat{W}$ and (s, z) in Eq. 4 as a constrained optimization problem independent of the previous quantization paradigm! We only need to regard Eq. 4 as an ordinary affine transformation, in which the value of s can be 0 or even negative.

In per-channel quantization, each column of the weight matrix is optimized independently of each other. For simplicity of notation, we only focus on one column in $\widehat{W}$ later and re-define the notations. Based on Eq. 5, the optimization problem of decoupleQ in the first stage, layer-wise minimization, can then be formulated as:

$$\begin{aligned} &\min_{w;s,z} g(w; s, z) \\ &\text{s.t. } \forall i = 1, 2, ..., d_{in} \\ &\quad w_i - \beta \leq 0 \\ &\quad -w_i + \alpha \leq 0 \\ &\quad w_i \in \mathbb{Z} \end{aligned} \tag{6}$$

where the objective function is:

$$g(w; s, z) = \frac{1}{2}(w * s + z - b)^T H (w * s + z - b) \tag{7}$$

$w \in \mathbb{R}^{d_{in}}$ is one column of $\widehat{W}$, $b \in \mathbb{R}^{d_{in}}$ is the corresponding column of W_0, $s \in \mathbb{R}^{ng}$ is the scale and $z \in \mathbb{R}^{ng}$ is the zero point, ng is the number of groups when grouping-quantization. The operations w.r.t (s, z), i.e., $*s$ and $+z$, need to be broadcasted to each group. In this paradigm, we have completely abandoned the traditional framework of quantization and instead transformed quantization into a constrained optimization problem 6, which is then solved to achieve the purpose of quantization. (s, z) in problem 6 have lost the traditional meaning of scale and zero point, and are just two optimization variables.

Transforming the traditional quantization problem into problem 6 is the soul of decoupleQ! Problem 6 is a quadratic programming problem with an additional non-convex constraints $w_i \in \mathbb{Z}$. Quadratic programming has been studied for many years and there are now many well-established solution (24; 34). We provide one solution in the next subsection, which may not be efficient or optimal.

The core idea of decoupleQ is to decouple the model weights into the integer part w and the floating-point part (s, z), with the integer part occupying most of the model's expressive power. The extensibility of the idea of decoupleQ is that we can freeze the integer part of the entire model, and use labeled data to train the (s, z) as well as other floating point parameters. The advantage of this is that on the one hand, it can further improve the accuracy of the model, on the other hand, it can fit specific downstream sub-tasks while maintaining the generalization ability of the model.

3.3 Optimization via Alternative Iteration

The problem 6 is not easy to solve because of the non-convex constraint $w_i \in \mathbb{Z}$. After obtaining a good initialization (explained in detail later), we solve for w and (s, z) alternately and iteratively. In each round of alternation, the objective function 7 w.r.t (s, z) is an unconstrained quadratic function, thus (s, z) can be readily determined *analytically*: by differentiating the objective function and equating the derivative to zero, followed by solving the resultant linear system of equations. While for w, the problem become problem 8:

$$\begin{aligned} &\min_{w} g(w; s, z) \\ &\text{s.t. } \forall i = 1, 2, ..., d_{in} \\ &\quad w_i - \beta \leq 0 \\ &\quad -w_i + \alpha \leq 0 \\ &\quad w_i \in \mathbb{Z} \end{aligned} \tag{8}$$

$$\begin{aligned} &\min_{w_i; i>j} g(w; s, z) \\ &\text{s.t. } \forall i = j+1, ..., d_{in} \\ &\quad w_i - \beta \leq 0 \\ &\quad -w_i + \alpha \leq 0 \\ &\quad w_i \in \mathbb{Z} \end{aligned} \tag{9}$$

For problem 8, one solution is to round-and-clip one element of w to be integer in $[\alpha, \beta]$ and then update the remaining. And then this process is then performed sequentially for all elements. After the j-th element has been rounded-and-clipped, the objective for the updating then becomes problem 9.

problem 9 is also intractable, and we can make two levels of approximation:

$$\min_{w_i;i>j} g(w;s,z) \quad \text{s.t. } \forall i = j+1, ..., d_{in} \quad w_i - \beta \leq 0 \quad -w_i + \alpha \leq 0 \tag{10}$$

$$\min_{w_i;i>j} g(w;s,z) \tag{11}$$

In the first-level approximation 10, only the non-convex constraint $w_i \in \mathbb{Z}$ is discarded, while in the second-level approximation 11, both the non-convex constraint $w_i \in \mathbb{Z}$ and the convex constraint $w_i \in [\alpha, \beta]$ are discarded. Intuitively, problem 11 is much simpler to solve than problem 10, but solving problem 10 will lead to a better convergence of the primary objective(6) than solving problem 11. GPTQ (9) provides an efficient analytical solution for problem 11, which we will directly utilize in our experiments. (GPTQ updates the remaining elements by considering only the second-level approximation 11 and ignoring the constrain $w_i \in [\alpha, \beta]$ in the first (10), which is what we mentioned in the introduction, that the update of GPTQ is unconstrained.) As for problem 10, there are many mature solutions in the field of convex optimization, such as active-set method, projected gradient descent (PGD), projected coordinate descent and so on (3). We choose PGD because its parallelization is much better than the other two methods. In the experimental part, we will compare the final accuracy of the model via between solving the first level (10) and the second level 11 approximation on small models, while on large models (e.g. lager than 7 billion parameters), we have to choose the second level 11 approximation because the intolerable runtime of solving the first (10). The algorithm is shown in Alg. 1 and Alg. 2.

Algorithm 1: Alternative Iteration to solve problem 6.

Input: predefined iteration number N.
Result: w^*, s^*, z^*

Initialize $t = 1, w_0, s_0, z_0$;
while $t \leq N$ **do**
  Freeze (s_{t-1}, z_{t-1}), and optimize $g(w; s_{t-1}, z_{t-1})$ to obtain an approximate solution w_t via solving 8 via 2;
  Freeze w_t, and solve the unconstraint quadratic equation $g(w_t; s, z)$ to obtain an *analytic* solution for (s_t, z_t);
  $t = t + 1$
end
$w^* = w_N; s^* = s_N; z^* = z_N$

Algorithm 2: Approximate solution of 8

Input: predefined iteration number K, M, and the froozen (s, z).
Result: w^*

if *Approximaton (10) is used* **then**
  Ignoring the constraint $w_i \in \mathbb{Z}$ in Eq. 8, and train Eq. 8 with M iterations via PGD;
Initialize $j = 1$;
for $j = 1 \rightarrow d_{in}$ **do**
  round and clip the j-th element of w, then keep the first j elements frozen, and update the remainings via PGD to optimize 10 with K iterations or until converged, or via the method in GPTQ to optimize 11.
end
$w^* = w$

3.4 Initialization of w and (s, z)

Since the values of w are discrete, a good initialization is very important in order to obtain a more accurate solution to the original problem 6 with a faster convergence. Intuitively, the function $g(w; s, z)$ contains the term $w * s$, which means that the scales of the initial values of w and s have to be reasonably distributed. For example, in the extreme case when the initial value of (s, z) have a very large scale, the first iteration will make most of the entries of w strictly 0, which will make the iteration crash. We start by initializing (s, z). We can use grid search to solve the Eq. 12 for the initial value of (s, z). In Eq. 12, p is a single number, may be different for different columns of W_0, b_{min} and b_{max} are the minimum and maximum value of b respectively. This step is the same as the previous post-training quantization (19) process. Once the grid search is

$$\begin{aligned} &\min_p \frac{1}{2}(w * s + z - b)^T H (w * s + z - b) \\ &\text{s.t.} \\ &w = \text{clip}(\lfloor \frac{b - z}{s} \rceil, \alpha, \beta) \\ &s = \frac{p * (b_{max} - b_{bmin})}{\beta - \alpha} \\ &z = p * b_{min} - s * \alpha \end{aligned} \tag{12}$$

finished, we no longer need to concern ourselves with the (s, z) inside the $\lfloor \cdot \rceil$ function. The point of this step is simply to find an initial value for (s, z) for the optimization problem 6.

When solving problem 8 via the first-level approximation (10), before entering the for-loop in Alg. 2, we ignore the constraint $w_i \in \mathbb{Z}$ in problem 8 and optimize it via projected gradient decent with M iterations. The purpose of this is to allow the first-level approximation to converge in a small number of iterations, i.e., a small K.

3.5 Block-wise minimization

After solving problem 6, we obtain a solution for the layer-wise minimization stage and a reasonable model accuracy. But minimizing the ℓ^2 loss at the layer level does not necessarily lead to the minimizing the ℓ^2 loss at the block level. We found that the model accuracy can be further improved via optimization 2. BRECQ (18) also shows that block-reconstruction results in a better model accuracy than layer-reconstruction. In this stage, we freeze the integer part $\widehat{W}$ in the whole block and fine-tuning (s, z) and the parameters in norm layer with J epochs.

4 Experiments

In this section, we describe in detail the experimental results of our method in comparison with other methods. Unless otherwise stated, all the experiments are conducted on a single A100-SXM-80GB, and the default experimental setting is as follows:

ResNet: 10240 images in the training dateloader are used as calibration data, with the standard augmentation in Pytorch official code (27), and the pretrained full precision checkpoints are from Torchvision (22). $N = 4, M = 50$ (N and M is defined in refalg1 and refalg2). All the convolution layers and fully-connected layers are quantized into W2 without groups.

Llama-1/2: 128 2048-token segments from C4 (28) are used as calibration data. We choose C4 as calibration dataset instead of WikiText2 (23) to be consistent with GPTQ. If the block-wise minimization is used, we use Adam optimizer (15) to finetune the (s, z) and the parameters in norm layer with $J = 4$ epochs. The learning rate is $1e$-5, weight decay is $1e$-6.

4.1 Private Experiments

We applied decoupleQ to our company's two Automatic Speech Recognition models(ASR) (corresponding to task A and task B). Each of the models contain an encoder and an LLM decoder. The input of the models is a speech sequence and some prompt, and the output is the corresponding text. We quantize the LLM decoder to W2A16g64. The decoders of the two models contain 40 transformer blocks with 13 billion parameters and 32 transformer blocks with 7 billion parameters, respectively. Word Error Rate (WER) is used as metric to measure the accuracy of the models (less is better). In this experiments, we use about 8 millions of speech tokens as calibration dataset, and train 3 epoch in each block-wise minimization process. When an input batch contains sequences of varying lengths, we use a mask to make sure that the padding part is not involved in the computation of H and the loss of Eq. 2. In task B, once the whole model is quantized, we also fine-tune all the (s, z) and layer norm in the LLM with labeled dataset, while freezing all the integer part $\widehat{W}$, with 8 A100-SXM-80GB GPUs. The accuracy is shown in Tab. 1, and the CUDA kernel latency is shown in Fig. 1. The W2A16 CUDA kernel is attached and will be merged into the NVIDIA repo as one of our core contribution.

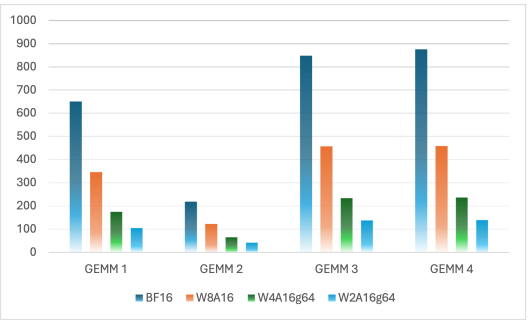

Figure 1: The latency (in $1e$-6 seconds) of the four GEMMs in transformer block on L4 GPU, (The three GEMMs for query, key and value are concatenated into GEMM 1), with $hidden\_dim = 5120$, $batch\_size = 4$.

Table 1: The results of our two ASR models. The models are quantized into W2A16g64. runtime for the quantization process is measured in hours. There are two sub-domains in task B, and we report the WER of both.

	Task A		Task B		
	BF16	decoupleQ	BF16	decoupleQ	decoupleQ+sft
WER	6.68	6.70	(5.86, 11.43)	(5.87, 11.56)	(5.77, 11.43)
runtime	-	25	-	32	32+5

Table 2: Comparison of decoupleQ with other methods. In decoupleQ, we only use the first stage, layer-wise minimization. All the models are quantized into W2A16 without groups. In decoupleQ+sft, we train the (s, z) and norm layers for one epoch, using the regular labeled dataset containing 1.2 million images.

method	res18-69.76%			res50-76.13%		
	2bit	3bit	4bit	2bit	3bit	4bit
GPTQ	-	67.88	69.37	-	74.87	75.71
OBQ	64.04	68.69	69.56	70.71	75.24	75.72
BRECQ	64.70	68.47	69.37	72.41	75.32	75.88
decoupleQ	64.15	68.65	69.58	71.34	75.24	76.00
decoupleQ+sft	65.45	68.94	69.71	72.65	75.61	75.97

4.2 Public Comparison

As a first comparison, we compare decoupleQ with other methods on ImageNet (5) with ResNet (12), which are standard benchmarks and are efficient to implement. Most importantly, its Top-1 is a strong indicator of model accuracy. Tab. 2 shows the results of decoupleQ and others. The results other than decoupleQ are copied from GPTQ (9) and OBQ (8).

Tab. 3 shows the results on Llama. In this experiment, we have to choose the second level approximation(11) because the intolerable runtime of solving the first(10). For a fair comparison, the calibration dataset contains 128 samples, although a larger calibration dataset will result in stronger results. we can see that decoupleQ outperforms others almost in all settings, although we use a weaker approximation(11) to save time. As for the hype-parameters, we choose $\{N = 4, J = 4\}$.

4.3 Ablation studies

4.3.1 the two approximations

The soul of decoupleQ is problem 6, but when solving problem 6, we have to take some approximations(10 or 11). Obviously, solving approximation 10 will be much more time consuming than solving approximation 11. But if solving approximation 10 yields better results, the time cost may be worth it. We first evaluate these two approximations from the perspective of model accuracy. In practice, we don't have to wait for approximation 10 to fully converge when we solve it via projected gradient decent, and only need to iterate some steps to get a sub-optimal solution. In Alg. 2, the for-loop takes up the majority of the runtime. So, we first study the influence of the number of iterations K (defined in the for-loop) on the final accuracy of the model. Fig. 2 shows the Top-1 accuracy of ResNet-18 on ImageNet w.r.t the number of iterations K. First of all, in the blue line, we use only the layer-wise minimization of decoopleQ to quantize the model. After the quantization is finished, in the red line, we use the labeled dataset with the common 1.2 millions images to fine-tune all the (s, z) and norm layers for one epoch, with the integer part being frozen. In this step, we use SGD optimizer with learning rate $1e$-6, weight decaying rate $1e$-4 to train for only one epoch. Fig. 2 clearly indicates the following conclusions: 1. As the number of iterations K increases, the model accuracy increases almost monotonically; 2. When $K > 4$, model accuracy via the first approximation(10) is better than via the second(11). This is to be expected, since the second approximation(11) drops the constraint $\alpha \leq w_i \leq \beta$, leading to a looser approximation; 3. By the supervised fine-tuning (sft), the model accuracy is further improved. The same experimental phenomenon also occurs on the ResNet-50 model, which we do not show here.

Table 3: The results of PPL of wikitext-2 on Llama-1/2. We also report the runtime (measured in hours) for the W2 quantization via decoupleQ in the gray background row. The results other than decoupleQ are copied from OmniQuant (29). All the results of decoupleQ use the approximation 11.

Llama		1-7B	1-13B	1-30B	1-65B	2-7B	2-13B	2-70B
FP16		5.68	5.09	4.10	3.53	5.47	4.88	3.31
W2A16	GPTQ	2.1e3	5.5e3	499.75	55.91	7.7e3	2.1e3	77.95
	OmniQuant	15.47	13.21	8.71	7.58	37.37	17.21	7.81
	decoupleQ	**9.49**	**7.86**	**6.37**	**5.59**	**9.74**	**13.03**	**5.23**
	runtime	2.5	4.8	12.7	27.6	2.5	4.5	33.4
W2A16g128	GPTQ	44.01	15.60	10.92	9.51	36.77	28.14	-
	OmniQuant	8.90	7.34	6.59	5.65	9.62	7.56	6.11
	decoupleQ	**8.65**	**7.25**	**6.04**	**5.19**	**8.79**	**7.44**	**4.96**
	runtime	3.7	7.7	24.3	55.0	3.7	7.9	70.6
W2A16g64	GPTQ	22.10	10.06	8.54	8.31	20.85	22.44	-
	OmniQuant	8.90	7.34	6.59	5.65	9.62	7.56	6.11
	decoupleQ	**8.18**	**6.96**	**5.81**	**5.07**	**8.41**	**6.98**	**5.34**
	runtime	4.3	8.9	27.9	64.5	4.4	9.0	98.2
W3A16	GPTQ	8.06	6.76	5.84	5.06	8.37	6.44	4.82
	AWQ	11.88	7.45	10.07	5.21	24.00	10.45	-
	OmniQuant	6.49	5.68	4.74	**4.04**	6.58	**5.58**	3.92
	decoupleQ	**6.38**	**5.60**	**4.67**	6.05	**6.22**	5.72	**3.84**
W4A16	GPTQ	6.13	5.40	4.48	3.83	5.83	5.13	3.58
	AWQ	6.08	5.34	4.39	3.76	6.15	5.12	-
	OmniQuant	5.86	5.21	4.25	3.71	5.74	**5.02**	3.47
	decoupleQ	**5.85**	**5.21**	**4.24**	**3.67**	**5.70**	5.06	**3.45**

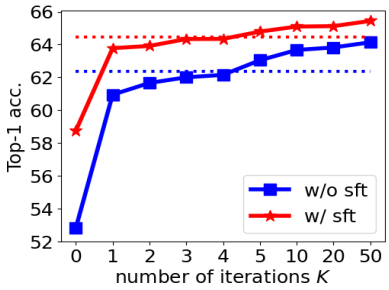

Figure 2: The Top-1 accuracy of ResNet-18 on ImageNet. Solid and dashed lines are for approximation 10 and 11 respectively.

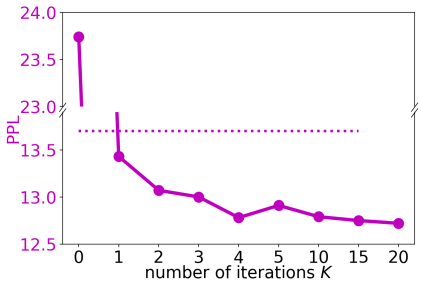

Figure 3: The PPL of Llama-7B on WikiText2. Solid and dashed lines are for approximation 10 and 11 respectively.

In the experiment shown in 3, we randomly select 512 2048-token segments from C4 (28). We chose 512 segments here instead of the common 128 in order to reduce the effect of overfitting and thus compare the two approximations more objectively. In this experiment, we take $N = 2$, and quantize Llama-7B into W2A16 without groups, and only the layer-wise minimization is used to exclude the interference of other factors. The PPL decrease almost monotonically as the number of iterations K increases. It shows that, when $K > 1$, solving approximation 10 yields better model accuracy than approximation 11.

However, when block-wise minimization is introduced in addition to the experiment in 3, the situation becomes a little more elusive. The results are shown in 4. The model's best PPL is where $K = 1$, and then fluctuates within a range as K continues to increase. But all PPLs are inferior to when the second-level approximation (11) is used. We also plot the loss, defined in 2, of the first block between pre-and post quantization on the right vertical axis. As K increases, the loss decreases strictly monotonically, and when $K > 2$, the loss falls below the case when the approximation 11 is used. This suggests that the correlation between PPL and loss is perhaps weak, and we will investigate this in the future.

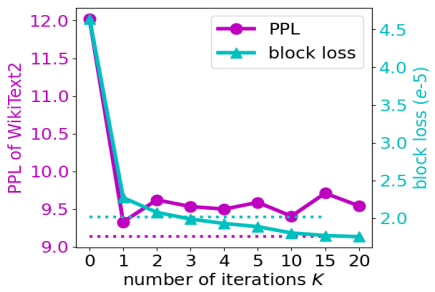

Figure 4: The PPL of Llama-7B on WikiText2 and the loss of the first block between pre-and post-quantization. Solid and dashed lines are for approximation 10 and 11 respectively.

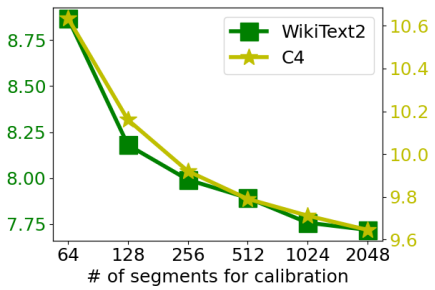

Figure 5: The perplexity of Llama-7B on WikiText2 and C4 dataset w.r.t the number of segments as calibration datasets. The model is quantized into W2A16g64.

4.3.2 the size of calibration dataset

The solution of problem 6 is dependent on H and thus on the the calibration dataset, as does Eq. 2. Fig. 5 shows the relationship between dataset size and PPL. In this experiment, Llama-7B is quantized into W2A16g64. We use the second-level approximation (11) to save time, and $\{N = 4, J = 4\}$. For runtime reference, when the number of segments is 128/2048, the experiment took 4.3/19.5 hours.

4.3.3 the necessity of block-wise minimization

Tab. 4 shows that block-wise minimization(2) can further improve the model accuracy. In this experiment, we choose $N = 4$ and the approximation 11 for the layer-wise minimization, and $J = 4$ if block-wise minimization is used.

Table 4: The perplexity of Llama on WikiText2 with and without the block-wise minimization. All the models are quantized into W2A16.

Llama	1-7B	1-13B	1-30B	2-7B	2-13B
w/o	13.66	9.68	7.35	14.66	12.93
w	9.49	7.86	6.37	9.74	13.03

5 Conclusion and Discussion

deocupleQ decouples the model parameters into the integer part and a floating point part, and then optimizes them alternately. This optimization process contains two stages. In the layer-wise minimization, we transform the quantization problem into the purely mathematical constrained optimization problem refdecoupleQ; while in the block-wise minimization, we freeze the integer part and then finetune the floating point part.

The risks of decoupleQ include the following: 1. How much the minimization of the ℓ^2 loss of the layer's or block's output correlates with the accuracy of the model; 2. decoupleQ is prone to overfitting the calibration dataset; 3. The runtime of the quantization process is longer than others.

For the first risk, we find experimentally that the correlation between Top-1 and the loss is strong in the Imagenet classification task; however, the correlation between PPL and the loss is slightly weaker in LLM. This could be mainly because of an inherent bias between the loss and the accuracy of the model, or because PPL is not a good indicator of the accuracy of LLM, or for other reasons. For the second risk, when H in Eq. 7 is an underdetermined matrix, the risk of overfitting rises sharply. In this case, the possibility of H being underdetermined can be reduced either by enhancing the diagonal element values of H or by increasing the amount of calibration data. In our practice, we found that the accuracy of quantization models can rise monotonically with the increase of the size of the calibration dataset, especially in W2 quantization, but the runtime of quantization rise as well. In addition, due to time constraints, we do not provide a wealth of public comparisons. However, we believe that the novelty of a method may outweigh the number of experiments.

The idea of decoupleQ is helpful for the adaptation of large model to downstream sub-task. We can quantize a large foundation model via decoupleQ, then freeze the integer part of the model, and finetune the floating-point part with labeled dataset from downstream sub-task. Tab. 1 and Tab. 2 show that the model accuracy can be further improved by end-to-end supervised learning.

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[2] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

[3] Sébastien Bubeck et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015.

[4] Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher De Sa. Quip: 2-bit quantization of large language models with guarantees. *arXiv preprint arXiv:2307.13304*, 2023.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[6] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022.

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

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

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

[10] Yi Guo, Yiqian He, Xiaoyang Li, Haotong Qin, Van Tung Pham, Yang Zhang, and Shouda Liu. Rdimkd: Generic distillation paradigm by dimensionality reduction. *arXiv preprint arXiv:2312.08700*, 2023.

[11] Yi Guo, Huan Yuan, Jianchao Tan, Zhangyang Wang, Sen Yang, and Ji Liu. Gdp: Stabilized neural network pruning via gates with differentiable polarization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5239–5250, 2021.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Itay Hubara, Yury Nahshan, Yair Hanani, Ron Banner, and Daniel Soudry. Accurate post training quantization with small calibration sets. In *International Conference on Machine Learning*, pages 4466–4475. PMLR, 2021.

[14] Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W Mahoney, and Kurt Keutzer. Squeezellm: Dense-and-sparse quantization. *arXiv preprint arXiv:2306.07629*, 2023.

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] Raghuraman Krishnamoorthi. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv preprint arXiv:1806.08342*, 2018.

[17] Changhun Lee, Jungyu Jin, Taesu Kim, Hyungjun Kim, and Eunhyeok Park. Owq: Lessons learned from activation outliers for weight quantization in large language models. *arXiv preprint arXiv:2306.02272*, 2023.

[18] Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. Brecq: Pushing the limit of post-training quantization by block reconstruction. *arXiv preprint arXiv:2102.05426*, 2021.

[19] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, and Song Han. Awq: Activation-aware weight quantization for llm compression and acceleration. *arXiv preprint arXiv:2306.00978*, 2023.

[20] Zechun Liu, Kwang-Ting Cheng, Dong Huang, Eric P Xing, and Zhiqiang Shen. Nonuniform-to-uniform quantization: Towards accurate quantization via generalized straight-through estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4942–4952, 2022.

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

[22] Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1485–1488, 2010.

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

[24] Katta G Murty and Feng-Tien Yu. *Linear complementarity, linear and nonlinear programming*, volume 3. Heldermann Berlin, 1988.

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

[26] Yury Nahshan, Brian Chmiel, Chaim Baskin, Evgenii Zheltonozhskii, Ron Banner, Alex M Bronstein, and Avi Mendelson. Loss aware post-training quantization. *Machine Learning*, 110(11-12):3245–3262, 2021.

[27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep

learning library. *Advances in neural information processing systems*, 32, 2019.

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

[29] Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. Omniquant: Omnidirectionally calibrated quantization for large language models. *arXiv preprint arXiv:2308.13137*, 2023.

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

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

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[33] Xiuying Wei, Yunchen Zhang, Yuhang Li, Xiangguo Zhang, Ruihao Gong, Jinyang Guo, and Xianglong Liu. Outlier suppression+: Accurate quantization of large language models by equivalent and optimal shifting and scaling. *arXiv preprint arXiv:2304.09145*, 2023.

[34] Stephen J Wright. *Numerical optimization*. 2006.

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

[36] Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhensu Chen, Xiaopeng Zhang, and Qi Tian. Qa-lora: Quantization-aware low-rank adaptation of large language models. *arXiv preprint arXiv:2309.14717*, 2023.

[37] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

[38] Yu Zhang, Wei Han, James Qin, Yongqiang Wang, Ankur Bapna, Zhehuai Chen, Nanxin Chen, Bo Li, Vera Axelrod, Gary Wang, et al. Google usm: Scaling automatic speech recognition beyond 100 languages. *arXiv preprint arXiv:2303.01037*, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our claims and justification include:

   a. Our results are higher than others in very low bit (2-bit) quantization.(This is justified in Tab. 3.);
   b. decoupleQ has achieved comparable accuracy as fp16/bf16 for 2-bit quantization of large speech models in our company. (This is justified in Tab. 1, and the W2 CUDA kernel used in our company are attached.);
   c. decoupleQ gets rid of any tricks for dealing with outliers, sensitive channels, etc. (This is justified in the Problem 6, we do not use any tricks, such as scaling factor (19; 29), mixed-precision quantization (6), etc., to deal with outliers and sensitive channels.)

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper has discussed the three limitations of decoupleQ in the last section, **Conclusion and Discussion**, and the risk overall that we did not provide as many public comparison experiments as other work due to time constraints.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer:[NA]

   Justification: This paper does not include theoretical results.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: At the beginning of the section **Experiments**, we provide details of the experimental parameters; specifically for each experiment, we also provide the key experimental parameters.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code (including W2 CUDA kernels) is attached in supplementary material, and can reproduce the results in the public experiments.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: At the beginning of the section **Experiments**, we provide details of the experimental parameters; specifically for each experiment, we also provide the key experimental parameters. The code is attached in supplementary material and will be made public.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The cost of the experiment is high.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We have reported that most of the experiments are conducted in one single A100-SXM-80GB, except for the sft process. And we also reported the time of execution.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This is a work for accelerating the inference of deep models, where the social impact is determined by the function of the model, not by how the inference is accelerated.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This work does not release models or datasets.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper has cited all related works, and included the relevant license.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide the source code, and a readme and license file are alongside.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: the paper does not involve crowdsourcing nor research with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

Justification: the paper does not involve crowdsourcing nor research with human subjects

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.