동계 세미나 발표

발표자: 이재욱

Quantization

: 모델의 크기와 연산 속도를 최적화하기 위한 기술

weight, activation을 더 작은 data 형식으로 변환하여 메모리 사용량을 줄이고, 연산 속도를 향상시킴

➔ 모델 크기를 줄이면서 성능을 유지하기(또는 살짝 손해)가 목표!

양자화를 적용하면 메모리 절약, 연산 효율 증가, 에너지 효율 증가 등 효율 면에서 장점

➔ on-device LLM 과 같이 capacity가 적은 머신에서 돌아가야 할 때 필수적임

The case for 4-bit precision: k-bit Inference Scaling Laws

Tim Dettmers [1] **Luke Zettlemoyer** [1]

LLM-FP4: 4-Bit Floating-Point Quantized Transformers

Shih-yang Liu[*1], **Zechun Liu**[*2], **Xijie Huang**[1], **Pingcheng Dong**[1], **Kwang-Ting Cheng**[1]
[1]Hong Kong University of Science and Technology, [2]Meta Reality Labs
{sliuau, xhuangbs, pingcheng.dong}@connect.ust.hk
zechunliu@meta.com
timcheng@ust.hk

The case for 4-bit precision: k-bit Inference Scaling Laws

Tim Dettmers [1] **Luke Zettlemoyer** [1]

Introduction

Large Language Model은 zero-shot & few-shot 추론을 위해 널리 쓰임

LLM의 파라미터 수가 클수록 당연히 성능은 좋아지지만,

당연하게도 파라미터가 클수록 더 많은 메모리 공간이 필요하고, 추론 시간이 더 길어짐

메모리와 추론 시간은 parameter의 총 비트 수로 결정됨

-> Quantization을 통해서 model의 bit를 줄이면, 메모리랑 추론 시간을 덜 먹겠네?

Introduction

Quantization은 모델의 parameter를 표현하는데 필요한 bit의 수를 줄이고,
사용하는 memory를 효율적으로 줄이는 방법임

그러나 효율을 얻는 대신 정확도를 희생하는데...

그럼 모델의 parameter를 표현하는 bit-precision을 얼마나 줄여야
정확도를 최대한 덜 희생하면서, 모델의 parameter를 최대한 작게 표현할 수 있을까?

뭐가 더 좋을까?
4-bit precision의 60B LLM v.s. 8-bit precision의 30B LLM

이런 트레이드 오프를 연구해서 scaling law를 발견하는 것은 매우 중요함

Introduction

Contributions

+ 그래서 OPT, Pythia/NeoX, GPT-2, BLOOM & BLOOMZ와 같은 LLM의 # parameter와 bit-precision을 조절하면서 zero-shot inference에서 각 변인들이 미치는 영향을 분석

>>> Quantized model bits와 zero-shot precision의 trade-off를 분석

+ 논문에서 진행한 분석을 통해, 독자에게 zero-shot quantization model을 사용하는데 대한 간단한 권장 옵션을 제안함!

+ bit-level scaling trade-off를 개선하기는 어렵지만, 양자화 정밀도를 향상시키는 blocking 기법이 개선을 위한 가장 중요한 조치임을 발견

Backgrounds

Data Types

1. Integer: IEEE 표준을 사용, signed integer와 unsigned integer로 나뉨. 2진수 체계로 표현하며 특정 비트 수에 따라 값의 범위가 결정

   ex) 11 (4 bit, unsigned integer) = 1011(2)

2. Floating-Point: IEEE 표준을 사용. Precision(비트 수)에 따라 여러 형식을 제공하며,
   각 형식은 부호 비트, 지수 비트, 가수 비트로 구성.

   ex) 1.5 (4 bit, FP) = **0110**

Backgrounds

3. Quantile Quantization

양자화 과정에서 사용되는 데이터의 범위인 Quantization bin에 동일한 수의 값을 할당하는 방법

예를 들어,

1비트 양자화의 경우, Quantile Quantization은 0과 1의 두개의 bin을 가짐

두개의 bin에 전체 값의 하위 사분위수, 상위 사분위수가 할당됨

데이터에 0~10까지의 값이 있다고 하면, 0~5는 0으로, 1~10은 1로 양자화

데이터의 분포를 고려하여 Quantization bin을 설정하여, 오류를 최소화

Backgrounds

4. dynamic exponent quantization(Dettemers, 2016)

- 지수 비트와 linear quantization 영역을 분리하는 indicator bit를 사용.
- indicator bit를 이동하여 값마다 지수를 다르게 조정할 수 있음.
- 순서가 여러 단계로 변하는 tensor에 대한 Quantization error가 적음

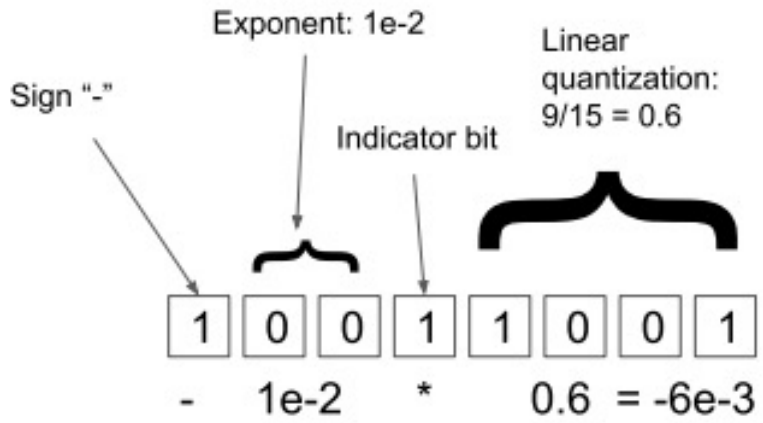


Figure 6. Schematic of dynamic exponent quantization.

Backgrounds

Blocking은 실험에서 양자화 정밀도를 향상시키기 위해 포괄적으로 사용

Blocking: tensor를 일련의 value들의 1-dim sequence로 보고, sequence를 block size n으로 나눔

이 방법을 사용하면 tensor를 더 작은 블록으로 나누고, 각 블록을 독립적으로 Quantization하여, 정밀도를 높이고, 양자화 과정에서 발생하는 Outlier를 한정된 범위로 제한하는데 유용함

Block-wise Quantization에서는 k-bit, block size B, input tensor T를 사용함
각 블록의 index를 0../B라고 정의하고, 정규화 상수를 $m_b = \max(|\mathbf{T}_b|)$로 계산하면,
Block-wise Quantization을 다음과 같이 정의

$$\mathbf{T}_{bi}^{Q_k^{\text{map}}} = \underset{j=0}{\overset{n=2^k}{\arg\min}} \left| \mathbf{Q}_k^{\text{map}}(j) - \frac{\mathbf{T}_{bi}}{m_b} \right| \Big|_{0<i<B},$$

Backgrounds

Outlier-dependent Quantization Through Proxy Quantization

논문에서 실험에 사용하는 LLM 중, 3비트로 양자화된 OPT 및 Pythia 모델이 불안정하여 완화하기 위해 사용

이는 hidden state에서의 표준 편차가 later layer에서 증가하면서 너무 많은 outlier가 감지되기 때문
(이상치를 생성하는 hidden unit weight의 표준편차가 다른 dimension보다 최대 20배 더 큼)

이를 해결하기 위해, hidden unit weight의 표준편차를 outlier feature가 있는 차원을 결정하는 proxy로 사용

input dim h, output dim o를 갖는 n개의 linear layer가 있는 transformer가 주어진 경우, 더 높은 정밀도로 양자화 할 index 집합 J를 아래와 같이 정의할 수 있음

$$J_{i+1} = \arg\max_{j=0}^{k} \text{std}(\mathbf{W}_i)\big|_{i..n}$$

Experimental Setup

실험에서는 16-bit input, quantized k-bit precision parameter를 사용 ($3 \geq k \geq 8$)
(attention matrix는 parameter 포함하지 않으므로 양자화되지 않음)

baseline은 16-bit 부동소수점 모델

k-bit quantization 방법의 추론 성능 측정을 위해 아래와 같은 지표를 사용

- CommonCrawl의 하위 데이터 집합에서의 **perplexity**
- EleutherAI LM evaluation harness에서의 **mean zero-shot performance**

Task로는

- LAMBADA, Winogrande, HellaSwag, PiQA를 사용

Results & Analysis

Bit-level Inference Scaling Laws

4개 모델에 대한 3~16-bit parameter 매개변수의 총 bit 수를 고려한 mean zero-shot performance

다음과 같은 관찰을 얻음

1. 주어진 제로샷 성능에 대해 4-bit precision이 거의 모든 model family와 model size에 대해 최적의 scaling을 제공

2. scaling curve는 거의 평행하며, bit-level scaling이 대부분 scale과 독립적임

3. 3-bit 추론에서 Pythia와 OPT는 불안정, 가장 큰 모델의 경우 랜덤에 가까운 성능(35%)

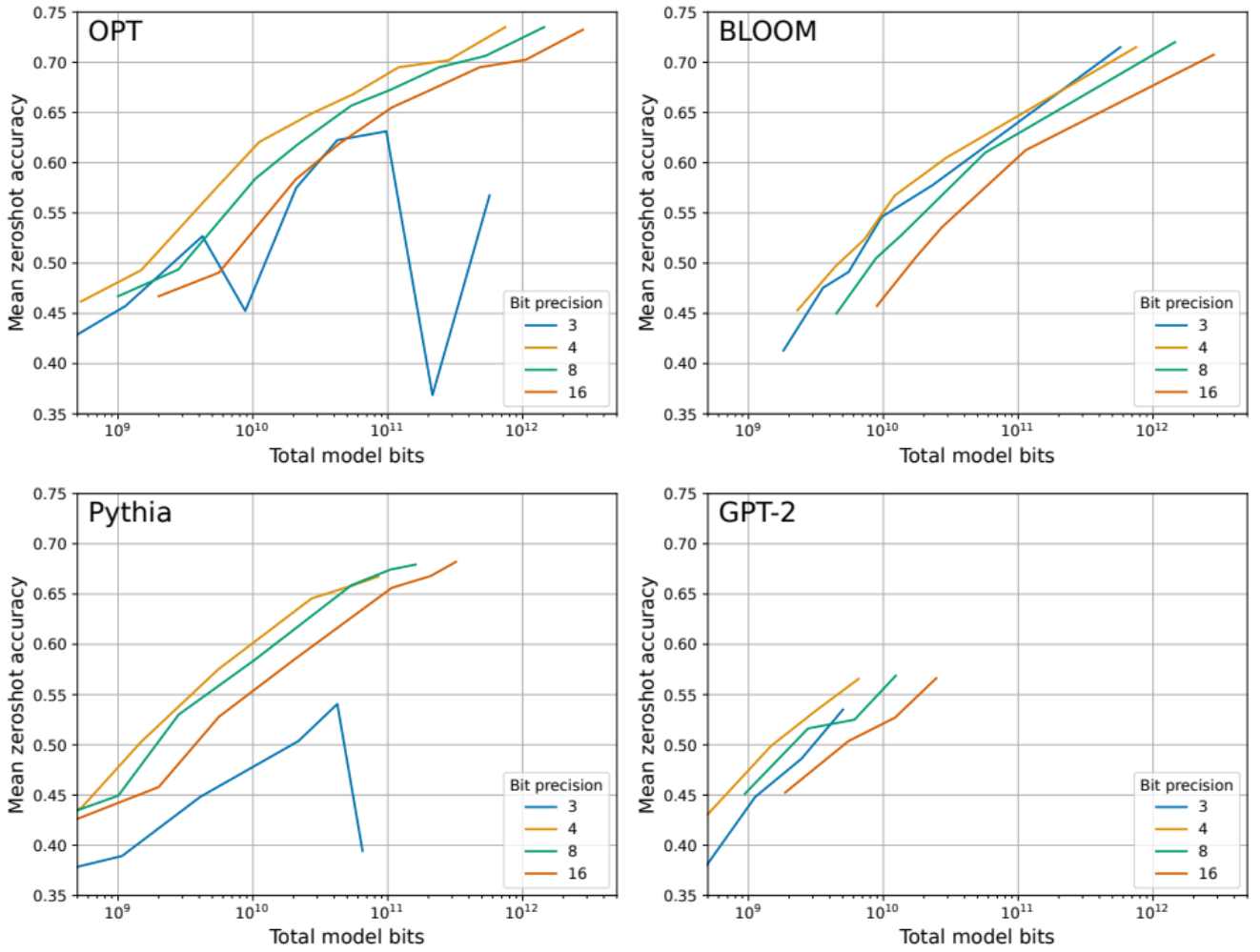


Figure 2. Bit-level scaling laws for mean zero-shot performance across Lambada, PiQA, Winogrande, and Hellaswag. 4-bit precision is optimal for almost all models at all scales, with few exceptions. While lowering the bit precision generally improves scaling, this trend stops across all models at 3-bit precision, where performance degrades. OPT and Pythia are unstable at 3-bit precision, while GPT-2 and BLOOM remain stable. Plots for all intermediate bit precisions can be found in the Appendix C.1.

Results & Analysis

Improving Scaling Laws

k-bit precision에 대한 Scaling 결과를 확인했으니, 이걸 어떻게 개선하면 좋을까? 에 대한 분석 실험을 진행

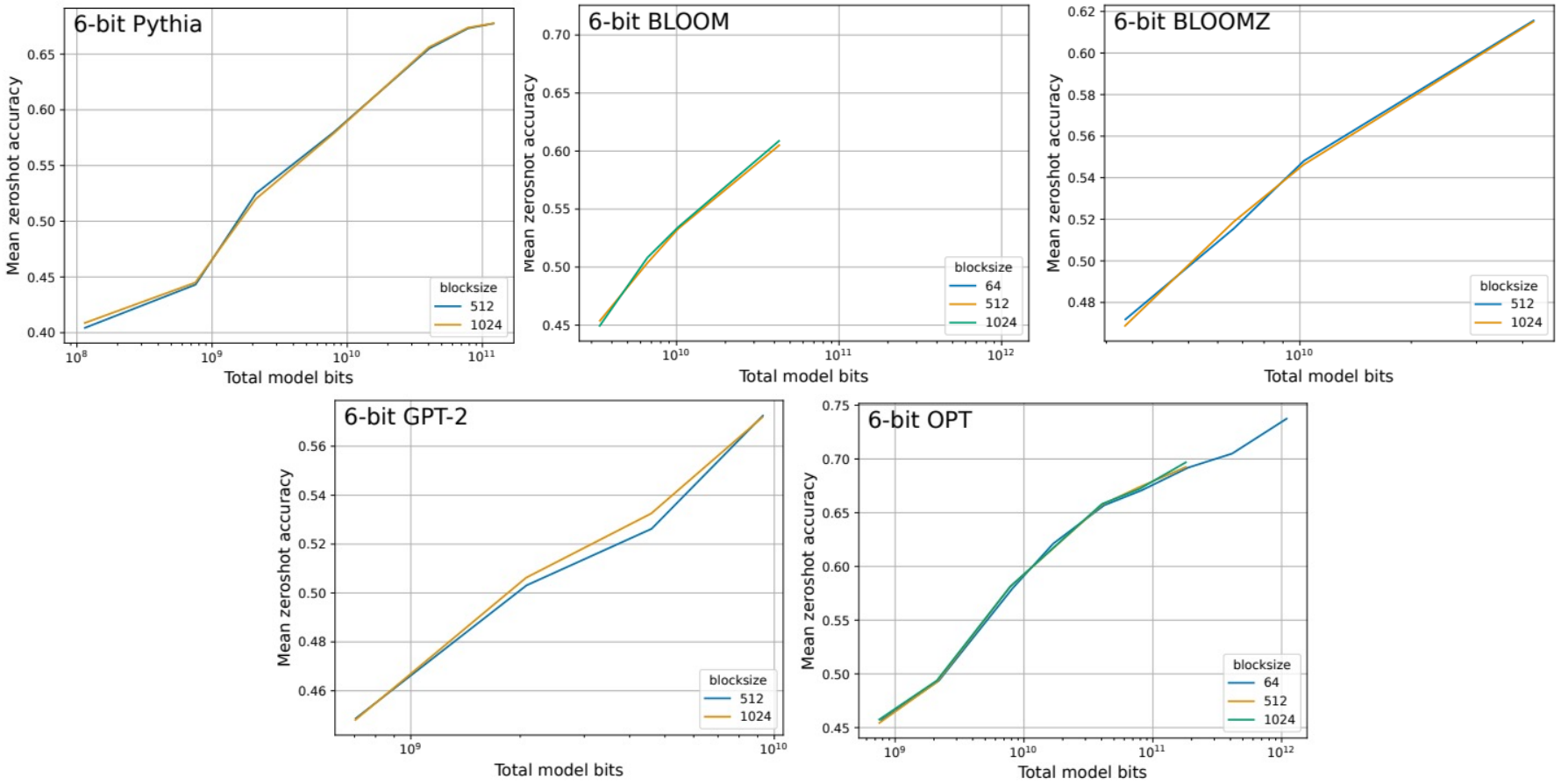
이를 위해서 Quantization precision에 대한 최근의 다양한 개선 방법을 적용 (앞 슬라이드에서 설명한...)

실험을 통해서 아래와 같은 결론을 냄

1. **No scaling improvements for 6 to 8-bit precision**
2. **Small block size improves scaling.**
3. **Data types improve scaling.**
4. Outlier-dependent quantization improves stability, but not scaling (OPT, Pythia)

Results & Analysis

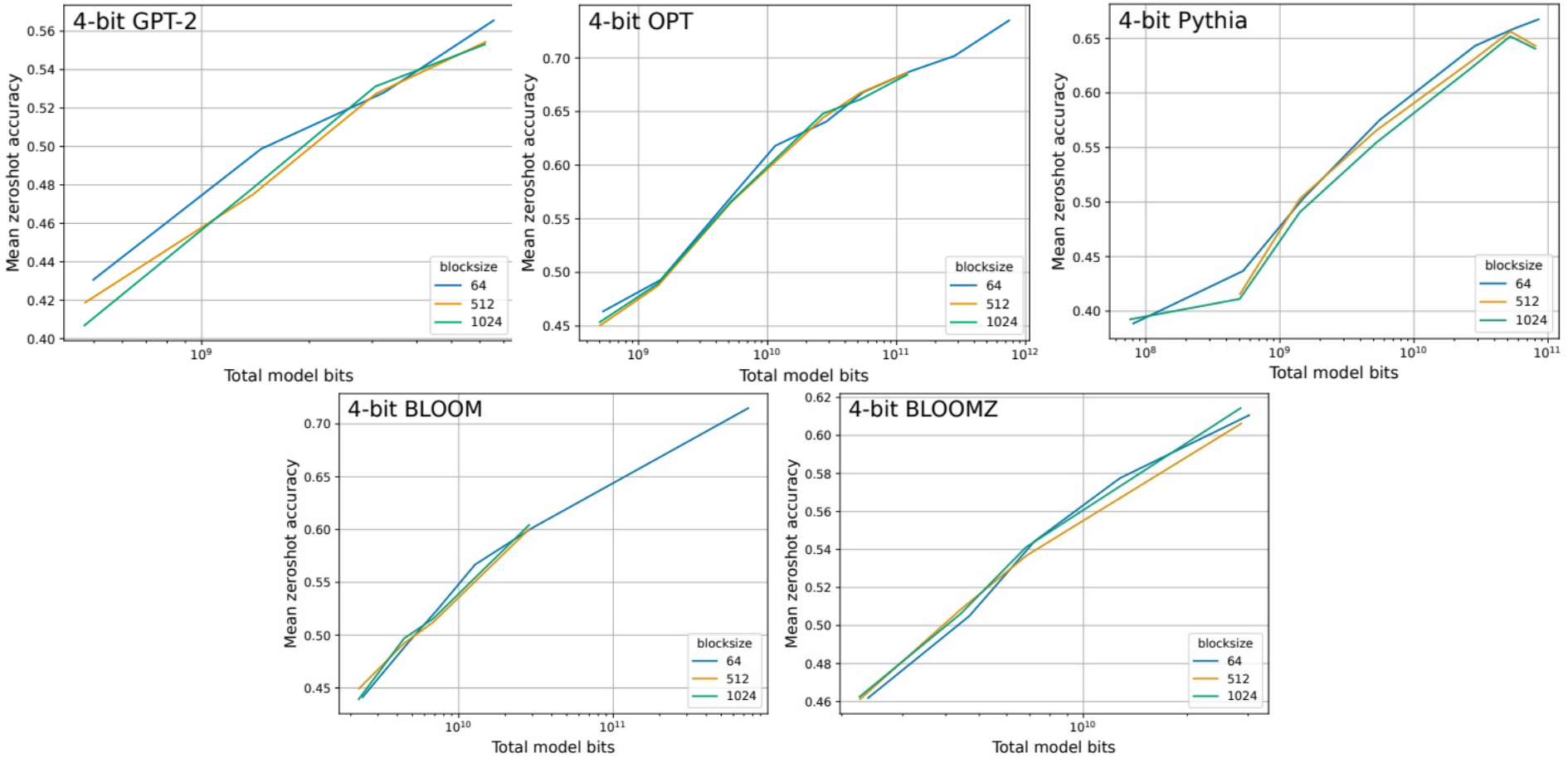
No scaling improvements for 6 to 8-bit precision



6~8-bit 양자화와 함께, Data type 변경, Blocking 적용 등 가능한 모든 조합을 결합했지만,
어떤 방법도 비트 스케일링을 개선하지 못함

Results & Analysis

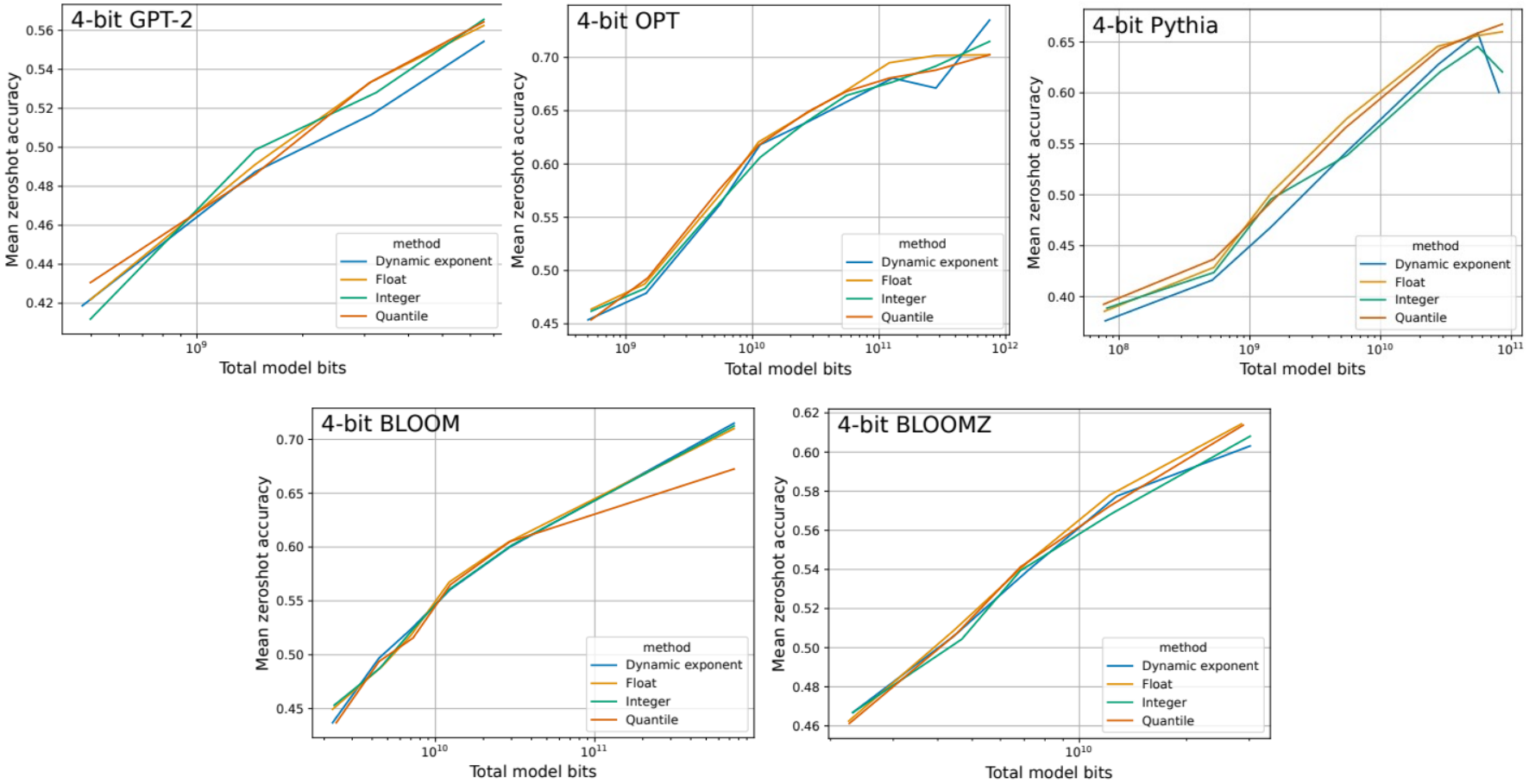
Small block size improves scaling



3~5-bit precision의 경우 다양한 양자화 방법을 적용하여 Scaling 개선이 관찰됨
Blocking Quantization을 적용했을 때, 블록이 작을수록 개선
(예를 들어 블록 크기가 1024에서 64로 바뀌면 4-bit에서 5-bit로 변경하는 것만큼의 개선)

Results & Analysis

Small block size improves scaling & Data types improve scaling



3~5-bit precision의 경우 다양한 양자화 방법을 적용하여 Scaling 개선이 관찰됨
Data type이 Scaling 트렌드를 개선함
특히 Quantile Quantization, Float가 Integer, Dynamic Exponent Quantization보다 더 나은 스케일링

Results & Analysis

Outlier-dependent quantization improves stability, but not scaling (OPT, Pythia)

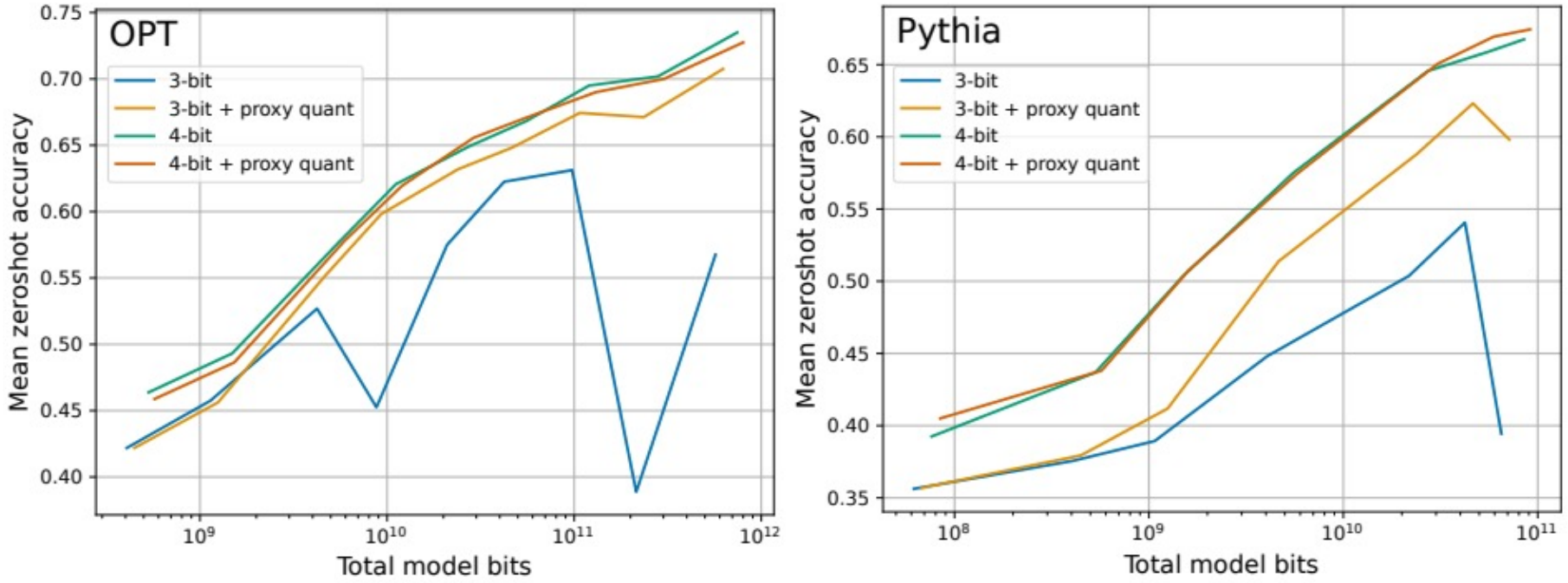


Figure 4. Bit-level scaling laws for outlier dependent quantization for OPT and Pythia. While proxy quantization removes instabilities and improves the 3-bit precision scaling of OPT and Pythia, it still scales worse than 4-bit precision. Proxy quantization is only useful for 3-bit precision weights. Proxy quantization is not useful for models that are relatively stable such as 3-bit BLOOM and GPT-2.

OPT, Pythia의 경우, 3비트 안정성이 매우 떨어지는데,
이상치에 대해서만 더 높은 비트로 양자화하는 방법을 적용하면 3-bit OPT, Pythia의 안정성과 전반적인 스케일링을 높일 수 있음.
(하지만 그래도 4비트 모델보다 성능 낮음)

Recommendation

논문에서 분석을 통해 얻은 통찰을 기반으로 권장 사항을 언급

1. **LLM 추론에는 4비트 양자화가 좋다!**
   -> 총 모델 bit 수와 zero-shot performance의 균형이 가장 좋음

2. **4-bit Quantization의 안정화와 zero-shot 성능 향상을 위해 128-Block Quantization을 적용해라**

3. **Float 또는 Quantile Quantization 데이터 타입을 사용해라**
   -> but, GPU 메모리에 따라 다름
   -> 예를 들어 48GB GPU는 5-bit precision으로 66B 모델 가능하지만,
   4-bit precision으로 175B는 못 씀
   -> 하드웨어에 따라 최적의 조합을 찾아서 써봐!

LLM-FP4: 4-Bit Floating-Point Quantized Transformers

Shih-yang Liu[*1], **Zechun Liu**[*2], **Xijie Huang**[1], **Pingcheng Dong**[1], **Kwang-Ting Cheng**[1]
[1]Hong Kong University of Science and Technology, [2]Meta Reality Labs
{sliuau, xhuangbs, pingcheng.dong}@connect.ust.hk
zechunliu@meta.com
timcheng@ust.hk

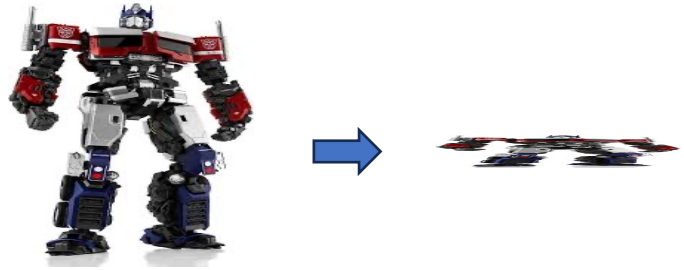
Introduction

Transformer 구조의 등장 이후,

Transformer의 성능 개선은 모델 크기와 계산 비용의 증가와 함께 이루어지고 있음

하지만, 광범위한 Transformere 관련 연구와 채용 정도에 비해, Transformer의 압축은 상대적으로 미개척

본 연구는 Floating-point Post-Training Quantization(PTQ) 기술을 통한 Transformer의 압축에 중점



Introduction

4-bit FP-PTQ

- 기존 PTQ는 주로 Integer Quantization에 중점을 두고, 8-bit 미만의 precision에서 종종 실패

- 반면 FP-PTQ는 다양한 활성화 및 weight 분포를 잘 수용하는 더 유연한 대안으로 주목

- FP-PTQ의 관건은 적절한 exponent(지수) bit와 scale parameter를 선택하는 것
  -> 잘못 선택하면 부적절하거나 발산하는 양자화 결과를 초래

Post-Training Quantization?
: 이미 훈련된 모델의 weight와 activation을 lower-bit precision으로 변환하는 것
: 모델의 크기를 줄이고 추론 시간을 개선
: 모델 재훈련 X

Introduction

Contribution

- FTQ를 위한 최적의 exponent bias와, 최대 양자화 값을 결정하기 위한 검색 기반 프레임워크를 제안

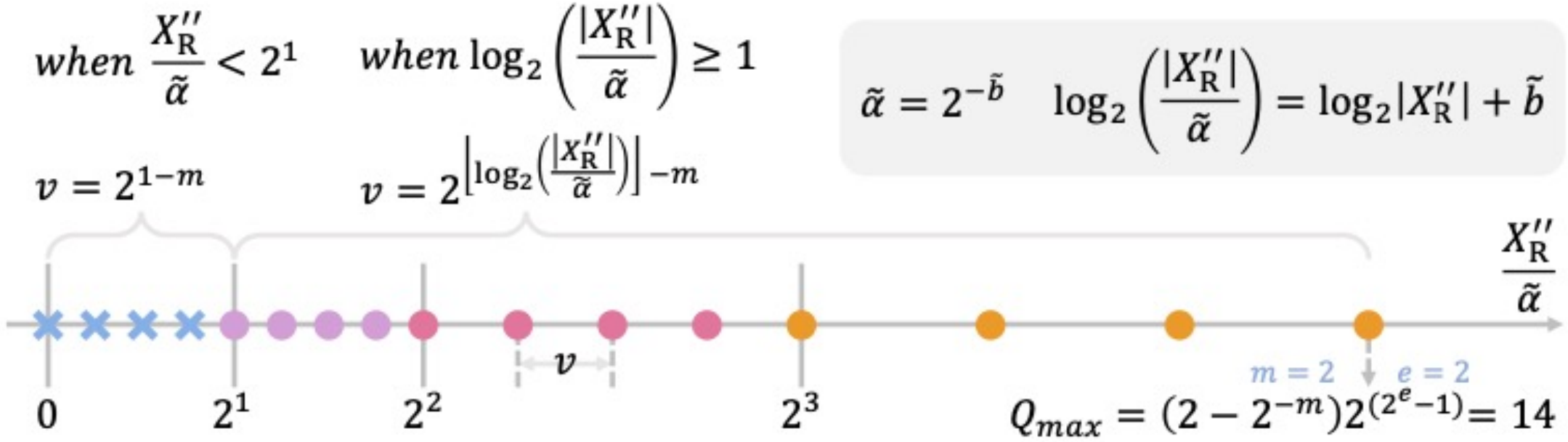
- **pre-shift exponent bias**라는 새로운 시프트 방법을 제안
  -> transformer의 high inter-channel variance 문제를 해결하면서,
  계산 비용은 무시할 수 있는 수준에서만 증가

- zero-shot inference에서 5.8p의 성능 저하만으로, 4-bit **LLaMA-13B** 모델을 최초로 사용 가능하게 구현

- 제안하는 방법을 BERT와 Vision-Transformer로 확장하여,
  이전 4-bit BERT보다 7.8p, 이전 SoTA ViT Quantization 방법보다 31.4p 높은 precision 달성

Preliminaries

Floating-Point Quantization Process – Figure (FP5, E2M2)
부동소수점 양자화 과정을 그림으로 나타내면…



Preliminaries

간단한 예시:

16-bit 부동소수점 값 1.5

2진수로 표현하면 1.1 $(1\times2^{0}+1\times2^{-1})$

4-bit 부동소수점 형식을 아래와 같이 정함

부호 비트: 1 bit
지수 비트: 1 bit
가수 비트: 2 bit

Preliminaries

간단한 예시:

16-bit 부동소수점 값 1.5

2진수로 표현하면 1.1 ($1 \times 2^{0} + 1 \times 2^{-1}$)

지수에 대한 bias = 2^(지수 비트 수 - 1) - 1

4-bit 부동소수점 형식을 아래와 같이 정함

부호 비트: 1 bit = 2^(1 - 1) – 1
지수 비트: 1 bit
가수 비트: 2 bit = 0

Preliminaries

간단한 예시:

1. 부호 비트 결정: 1.5는 양수이므로 부호 비트는 0
2. 1.5를 표현하기 위해서는 지수가 0이어야 함. 지수의 바이어스가 0이므로 지수 비트도 0
3. 가수 결정: 1.5의 이진 표현은 1.1이고, 정규화된 표현 형태는 1.1 여기서 1은 생략되고, 가수는 0.1(이진수로는 10)이 됨

이제 4-bit FP로 1.5를 양자화하면, '0010'이 됨 (실제로 1.0) -> (0.5만큼의 손실)

실제로는 Clipping을 위한 최소-최대 범위, 절삭 규칙 등을 정해야 함

Preliminaries

Formulation of Floating-Point Variables

일반적으로 FP는 아래와 같이 표현 (Eq.1)

$$X_{\mathrm{FP}} = (-1)^s 2^{p-b}(1 + \frac{d_1}{2} + \frac{d_2}{2^2} + \ldots + \frac{d_m}{2^m})$$

위 식에서 **m**은 가수 비트의 개수, p는 [0, 2^e-1] 범위의 정수인데 **e**는 지수 비트의 개수를 나타냄

그렇다면, **j** 개의 지수 비트와 **k** 개의 가수 비트를 가진 부동소수점은 FP형식 **EjMk**로 나타낼 수 있음.

Preliminaries

Floating-Point Quantization Process

정수 양자화에서 실수 값 변수 X_R은 아래 공식을 통해 정수 X_INT로 양자화 (Eq.2)

$$X_{\mathrm{INT}} = \alpha \left\lfloor \mathrm{Clip}\left(\frac{X_{\mathrm{R}}}{\alpha}, Q_{min}, Q_{max} \right) \right\rceil$$

양자화 범위의 최소-최대값으로 clipping 한 후,
반올림하여 full-precision scaling factor α를 곱하면 됨

Preliminaries

Floating-Point Quantization Process – Scale and clip.

FP Quantization에서는 양자화 이전에도 실수 값 변수를 Scaling & Clipping (Eq.3)

$$X'_{\mathrm{R}} = \mathrm{Clip}(X_{\mathrm{R}}, Q_{min}, Q_{max})$$

부호가 있는 FP 양자화의 최소-최대 값 범위는 아래 식에서 계산 (Eq.4)

$$Q_{max} = -Q_{min} = (2 - 2^{-m})2^{2^e - b - 1}$$

Preliminaries

Floating-Point Quantization Process – Scale and clip.

여기서 Integer exponent bias **b** 는 Q_max와 Q_min을 제어하는 hyperparam 중 하나로,
정수 양자화에서의 α와 비슷한 역할을 함

따라서 간단화를 위해 Eq.3을 아래와 같이 재정의함 (Eq.5)

$$X''_{\mathrm{R}} = \mathrm{Clip}\left(X_{\mathrm{R}}, \tilde{Q}_{min}, \tilde{Q}_{max}\right),$$

Preliminaries

Floating-Point Quantization Process – Scale and clip.

FP Quantization의 양자화 최대값 $\tilde{Q}$_max이 아래와 같을 때, (Eq.6)

$$\begin{aligned}\tilde{Q}_{max} = \alpha Q_{max} &= \alpha \cdot (2 - 2^{-m}) 2^{2^e - b - 1} \\ &= \alpha \cdot 2^{-b} \cdot (2 - 2^{-m}) 2^{2^e - 0 - 1} \\ &= 2^{-\tilde{b}} \cdot (2 - 2^{-m}) 2^{2^e - 0 - 1}\end{aligned}$$

또, 텐서별 실수 값 scaling factor α와 Integer exponent bias b 를 결합하여
새로운 scaling factor $\tilde{\alpha} = 2^{-\tilde{b}} = 2^{-b} \cdot \alpha$ 를 정의함

여기서 $\tilde{b}$는 완화된 텐서별 실수 값 지수(relaxed tensor-wise real-valued exponent)를 나타내며,
Eq.6의 원하는 clipping 값 $\tilde{Q}$_max에서 $\tilde{b}$를 아래와 같이 유도할 수 있음 (Eq.7)

$$\tilde{b} = 2^e - \log_2 \tilde{Q}_{max} + \log_2 (2 - 2^{-m}) - 1$$

Preliminaries

Floating-Point Quantization Process – Compare and quantize.

FP Quantization에서는 $X_R^{''}$을 quantization level과 비교한 다음 양자화 하는 추가 단계가 있음

수식으로는 아래와 같이 나타냄 (Eq.8)

$$X_{\mathrm{FP}} = \tilde{\alpha} \cdot v \cdot \left\lfloor \frac{X_{\mathrm{R}}''}{\tilde{\alpha} \cdot v} \right\rfloor$$

여기서 $X_R^{''}$은 Clipping된 실수 값 변수(Eq.5)이고, $\tilde{a}$는 텐서별 FP scaling factor, v는 2의 integer power

Preliminaries

Floating-Point Quantization Process – Compare and quantize.

Eq.8에서, $\frac{X_R''}{\tilde{a}}$ 의 크기에 따라 $X_R'' \cdot 2^{\tilde{b}}$ 와 동일한 Quantization level v를 선택. (Eq.9)

$$v = \begin{cases} 2^{\lfloor \log_2 |\mathbf{X}_{\mathrm{R}}''| + \tilde{\mathrm{b}} \rfloor - \mathrm{m}} & \text{if } \lfloor \log_2 |\mathbf{X}_{\mathrm{R}}''| + \tilde{\mathrm{b}} \rfloor \geq 1 \\ 2^{1-m} & \text{otherwise} \end{cases}$$

$$X_{\mathrm{FP}} = \tilde{\alpha} \cdot v \cdot \left\lfloor \frac{X_{\mathrm{R}}''}{\tilde{\alpha} \cdot v} \right\rfloor$$

Preliminaries

Floating-Point Matrix Multiplication

FP 양자화된 변수를 사용한 행렬 곱셈은 아래와 같이 정의됨. (Eq.10)

$$\mathbf{O}_{out}^{i,k} = \mathbf{X}_{\mathrm{FP}}^{i,:}\mathbf{W}_{\mathrm{FP}}^{:,k} = \tilde{\alpha}_{\mathbf{x}}\tilde{\alpha}_{\mathbf{w}}^{k}\tilde{\mathbf{X}}_{\mathrm{FP}}^{i,:}\tilde{\mathbf{W}}_{\mathrm{FP}}^{:,k}$$

Eq.10에서 $X_{FP}^{i,:}$는 activation matrix에서 i 번째 행을 나타내고, $W_{FP}^{i,:}$는 weight matrix에서 k 번째 열을 나타냄

따라서, output matrix의 각 요소 $O_{out}^{i,k}$는 두 개의 실수 $\tilde{a}_X$와 $\tilde{a}_X^k$의 곱으로,
양자화된 activation 및 weight vector를 곱하는 것으로 계산 됨

Method

Joint Format and Max Value Search

Post-training quantization(PTQ)의 objective는
양자화로 인해 pre-trained real-valued network에 들어가는 perturbation을 최소화하는 것! (Eq.11)

$$\min \mathbb{E}[\mathcal{L}(\mathbf{X}_{\mathrm{R}} + \delta\mathbf{X}) - \mathcal{L}(\mathbf{X}_{\mathrm{R}})]$$
$$(\delta\mathbf{X} = \mathbf{X}_{\mathrm{FP}} - \mathbf{X}_{\mathrm{R}})$$

해석하면, X_R의 변화에 따른 Cost를 최소화하겠다는 것!

Method

Joint Format and Max Value Search

논문에서는 이전 연구(Choukroun et al, 2019; Wu et al, 2020)에서 제시한 설정을 채택해서, 양자화된 모델의 중간 출력의 변화와, Eq.11 간의 양의 상관 관계를 가정함

따라서, 양자화된 레이어의 중간 출력($\hat{O}$)과 원래 레이어의 출력(O) 간의 거리를 최소화하면,
Eq.11을 최소화하는 것으로 이어짐

결국 objective loss metric은 아래와 같이 정의됨 (Eq.12)

$$\min\left(\hat{\mathbf{O}} - \mathbf{O}\right)^2$$

검색 프레임워크에서는 Eq.12가 최적의 FP 양자화 함수를 찾기 위해 사용

Method

Joint Format and Max Value Search

부동소수점 양자화가 어려운 이유는 Quantization format과 clipping range에 대한 민감성 때문이라고 함

이전 연구(Kuzmin et al, 2022)에서는 FP quantization-aware training(QAT)에 대한 FP 형식과 최대값 모두를 gradient로 학습하는 방식을 제안했지만, PTQ에서 over-filtting 문제가 발생하고, Naïve MinMax 방법보다 정확도가 심각하게 나빠진다고 함.

이러한 문제를 해결하기 위해,

optimal FP format과 clipping range를 함께 결정하는 검색 기반 알고리즘을 제안

Method

Joint Format and Max Value Search

검색 프로세스는 Eq.12를 최소화하는 지표를 사용하여 레이어 별로 진행

각 하위 모듈에 해당하는 행렬 곱셈의 출력을 $O = XY$ 로 나타냄

검색 프로세스의 알고리즘을 간단하게 설명하면
두 부분으로 나눌 수 있음

1. 각 레이어 l 의 intermediate raw output을 저장하기 위한 forward propagation

2. Eq.12를 최소화하여, 각 레이어에 대한 최적의 FP 형식과 bias를 3번의 라운드 동안 반복적으로 업데이트

이 검색 기반 프레임워크를 FP Quantization Baseline이라고 이름 짓고, 6, 8-bit에서 SOTA를 달성했다고 함.

Algorithm 1 FPQ baseline

Input: Calibration dataset, Full-precision Model M, Quantization format search space R_X (e.g., $R_X = \{E3M0, E2M1, E1M2\}$ for FP4), number of round $n = 3$,
Output: FP q Quantized model
for l in 1^{st} to L^{th} layer in M **do**
    Forward & collect raw output $O^l = X^l Y^l$ of layer l;
end for
for l in 1^{st} to L^{th} layer in M **do**
    Initialize the FP format search space w.r.t X^l and Y^l as $R_{\mathbf{X}} = \{r^1_{\mathbf{X}}, r^2_{\mathbf{X}}, ..., r^t_{\mathbf{X}}\}$ and $R_{\mathbf{Y}} = \{r^1_{\mathbf{Y}}, r^2_{\mathbf{Y}}, ...r^t_{\mathbf{Y}}\}$.
    Initialize bias $\tilde{b}^i_{\mathbf{X}}, \tilde{b}^i_{\mathbf{Y}}$ with Eq.7 for each format candidate $r^i_X \in R_{\mathbf{X}}$ and $r^i_{\mathbf{Y}} \in R_{\mathbf{Y}}$.
    Generate search space of $\tilde{b}_{\mathbf{X}}$ in t formats to be $[\gamma_1 \tilde{b}^{init}_{\mathbf{X}}, \gamma_2 \tilde{b}^{init}_{\mathbf{X}}]$ and $\tilde{b}_{\mathbf{Y}}$ to be $[\gamma_1 \tilde{b}^{init}_{\mathbf{Y}}, \gamma_2 \tilde{b}^{init}_{\mathbf{Y}}]$.
    for 0 to n **do**
        Search for $\tilde{b}^i_{\mathbf{X}}$ w.r.t each $r^i_{\mathbf{X}}$ that minimizes Eq.12
        Search for $r^i_{\mathbf{X}} \in R_{\mathbf{X}}$ that minimizes Eq.12
        Search for $\tilde{b}^i_{\mathbf{Y}}$ w.r.t each $r^i_{\mathbf{Y}}$ that minimizes Eq.12
        Search for $r^i_{\mathbf{Y}} \in R_{\mathbf{Y}}$ that minimizes Eq.12
    end for
end for

Method

Pre-Shifted Exponent Bias

논문에서 FP format의 지수 부분에 대한 처리를 개선하기 위해 고안한 기법

일반적으로 FP format에서 지수 부분은 숫자의 범위를 결정하는데,

Pre-Shifted Exponent Bias는 지수 부분을 조정하여 Quantization 과정에서 더 나은 성능을 내도록 도움

이는 특히 high inter-channel varience(LLM에서 각 Channel의 Activation 값들이 서로 다른 분포를 가짐)를 효과적으로 다루면서, 텐서 별 양자화와 거의 동일한 효율을 유지

논문의 실험에서는
Pre-Shifted Exponent Bias 방법을 Joint format and Max value search framework와 결합한 것을 FPQ라고 표기

Experiments

Commonsense reasoning task에서
LLaMA-7B, 13B에 대한 FPQ의 효과를 평가

데이터 전처리 및 점수 계산은
EleutherAI의 평가도구를 기반으로 함

8-bit precision에서는 Int Quant를 제외하고
거의 비슷한 성능

하지만 4/4/4-bit precision으로 내려가면,
FPQ가 다른 방법들을 전부 이겨버리고,

16/16/16-precision의 LLaMA에 비해
7B에서 8.2%, 13B에서 5.2%의 성능 감소밖에
일어나지 않았음

Quant Method	#Bits (E/W/A)	# Calib	BoolQ	PIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	Avg.
LLaMA-7B Full-precision	16/16/16	-	75.1	78.7	56.9	69.9	75.3	41.9	66.3
MinMax INT Quant	8/8/8	32	64.3	66.8	40.5	57.4	59.0	29.6	52.9
MinMax FP Quant (E4M3)	8/8/8	32	74.9	78.6	56.8	69.5	75.5	41.6	**66.1**
SmoothQuant (Xiao et al., 2022)	16/8/8	512	74.0	77.5	55.0	69.6	74.4	37.4	64.6
FPQ baseline	8/8/8	32	75.8	78.3	55.9	69.5	75.6	41.3	**66.1**
FPQ	8/8/8	32	75.6	78.2	56.6	70.2	74.6	40.7	66.0
MinMax INT Quant	4/4/16	32	64.1	76.1	51.6	66.3	72.4	40.0	61.7
MinMax FP Quant (E2M1)	4/4/16	32	73.0	77.9	55.2	69.1	73.6	40.9	64.9
GPTQ (Frantar et al., 2023)	4/4/16	128	73.3	77.9	54.9	67.9	72.7	37.4	64.0
FPQ baseline	4/4/16	32	74.8	77.9	55.6	69.5	75.2	41.0	**65.7**
FPQ	4/4/16	32	74.2	77.8	55.8	69.9	74.9	40.4	65.5
MinMax INT Quant	4/4/8	32	50.4	56.5	27.9	46.5	36.1	21.2	39.7
MinMax FP Quant (E2M1/E4M3)	4/4/8	32	73.0	77.5	55.0	69.3	73.6	40.9	64.9
FPQ baseline	4/4/8	32	75.0	77.6	55.9	69.9	74.3	39.4	65.3
FPQ	4/4/8	32	75.0	77.7	55.5	69.8	74.5	39.9	**65.4**
MinMax INT Quant	4/4/4	32	54.1	51.7	25.6	49.8	24.7	22.9	38.1
MinMax FP Quant (E2M1)	4/4/4	32	47.3	53.1	25.7	50.7	25.1	22.4	37.4
SmoothQuant (Xiao et al., 2022)	16/4/4	512	54.1	62.8	41.5	52.6	50.6	32.9	49.1
LLM-QAT (Liu et al., 2023)	16/4/4	(QAT)	63.5	64.3	55.6	52.9	50.3	30.2	52.8
FPQ baseline	4/4/4	32	57.4	56.6	30.2	51.1	37.7	23.2	42.7
FPQ	4/4/4	32	64.2	73.5	47.8	63.7	65.9	33.6	**58.1**
LLaMA-13B Full-precision	16/16/16	-	77.9	79.2	59.9	72.6	77.4	46.4	68.9
MinMax INT Quant	8/8/8	32	60.6	69.6	46.0	61.5	63.3	32.8	55.6
MinMax FP Quant (E4M3)	8/8/8	32	78.0	79.1	60.0	72.3	77.2	47.1	**68.9**
SmoothQuant (Xiao et al., 2022)	16/8/8	512	76.5	78.0	58.0	72.1	76.3	45.5	68.2
FPQ baseline	8/8/8	32	78.0	79.1	59.9	72.3	77.2	47.1	**68.9**
FPQ	8/8/8	32	78.1	78.5	59.1	72.4	76.4	46.1	68.4
MinMax INT Quant	4/4/8	32	52.1	65.0	36.4	53.9	52.3	29.0	48.1
MinMax FP Quant (E2M1/E4M3)	4/4/8	32	78.0	78.9	58.0	71.6	76.0	44.8	**67.9**
FPQ baseline	4/4/8	32	76.2	78.2	57.9	71.9	75.1	43.9	67.2
FPQ	4/4/8	32	76.4	78.5	58.2	72.1	75.2	44.7	67.5
MinMax INT Quant	4/4/4	32	54.5	52.7	25.5	51.1	25.3	22.1	38.5
MinMax FP Quant (E2M1)	4/4/4	32	45.8	51.7	25.5	49.5	25.0	22.8	36.7
SmoothQuant (Xiao et al., 2022)	16/4/4	512	57.6	61.3	56.0	52.6	49.9	25.1	50.4
FPQ baseline	4/4/4	32	54.3	57.7	35.7	52.2	41.1	25.7	44.5
FPQ	4/4/4	32	71.9	74.8	53.3	66.7	71.7	39.9	**63.1**

Table 1: Zero-shot performance on common sense reasoning tasks with LLaMA (Touvron et al., 2023) models. We denote E/W/A as the bit-width of word embeddings, model weight and activations, respectively.

Experiments

GLUE benchmark에서,
BERT 모델에 적용한 FPQ의 효과를 평가

Full-precision BERT model은
huggingFace에서 가져온 BERT-base를 사용

4/4/4-bit precision에서 BercQ(2021)과 비교해서 평균 정확도 개선율이 44.3%, Qdrop(2022)보다 7.9% 개선
심지어 Full-precision과 비교해서도 3.6% 밖에 차이가 안남.

4/4/8-bit 세팅에서 양자화 과정에 4096개 데이터 사용한 MREM-S, MREM-P는 Full-precision과 비교해서 1.6%/1.5% 감소,

하지만 4/4/8-bit 세팅의 FPQ는
거의 하락하지 않음 (-0.1%)

Quant Method	#Bits (E/W/A)	# Calib	$\text{MNLI}_{-\text{m}}$	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg.
(Full-precision)	32-32-32	-	84.9	91.4	92.1	93.2	59.7	90.1	86.3	72.2	83.7
MinMax INT Quant	8/8/8	128	77.0	89.9	88.9	92.9	51.8	88.2	83.8	71.5	80.5
MinMax FP Quant (E2M5)	8/8/8	128	78.9	90.8	88.6	92.9	52.7	88.4	84.3	69.0	80.7
MinMax FP Quant (E3M4)	8/8/8	128	84.5	90.9	91.5	93.2	58.3	**89.3**	87.7	71.8	83.4
MinMax FP Quant (E4M3)	8/8/8	128	**84.7**	90.9	**91.7**	93.0	58.6	**89.3**	86.5	**72.2**	83.4
MinMax FP Quant (E5M2)	8/8/8	128	84.1	90.9	91.4	**93.6**	58.1	89.2	87.5	71.8	83.3
FPQ baseline	8/8/8	128	84.6	90.9	**91.7**	93.1	58.6	**89.3**	**88.0**	**72.2**	83.5
FPQ	8/8/8	128	84.6	**91.0**	91.6	93.3	**58.8**	**89.3**	**88.0**	**72.2**	**83.6**
MinMax INT Quant	6/6/6	128	31.9	62.0	52.8	58.8	0.0	12.7	32.1	52.7	37.9
MinMax FP Quant (E2M3)	6/6/6	128	43.5	85.4	79.4	90.5	45.2	86.0	66.9	59.9	69.6
MinMax FP Quant (E3M2)	6/6/6	128	83.9	90.8	90.8	92.2	58.2	88.6	87.0	**72.2**	83.0
MinMax FP Quant (E4M1)	6/6/6	128	84.4	90.2	90.1	92.2	58.2	89.2	85.3	69.7	82.4
FPQ baseline	6/6/6	128	**84.6**	**90.9**	91.2	**93.2**	**58.8**	88.7	87.5	70.8	**83.2**
FPQ	6/6/6	128	84.5	90.8	**91.6**	93.1	57.3	**89.3**	**88.7**	71.8	**83.2**
MinMax INT Quant	4/4/8	128	33.1	63.8	60.1	49.3	0.0	44.0	50.2	49.1	43.7
MinMax FP Quant (E2M1)	4/4/8	128	60.6	70.9	77.4	79.9	5.5	78.6	46.8	56.6	59.5
MREM-S (Bai et al., 2022)	4/4/8	4096	83.5	90.2	**91.2**	91.4	55.1	89.1	84.8	71.8	82.1
MREM-P (Bai et al., 2022)	4/4/8	4096	83.4	90.2	91.0	91.5	54.7	89.1	86.3	71.1	82.2
FPQ baseline	4/4/8	128	84.4	**90.6**	**91.4**	**92.9**	58.6	83.7	88.2	**73.3**	82.9
FPQ	4/4/8	128	**84.5**	**90.6**	91.1	**92.7**	**58.8**	**89.3**	**88.7**	**73.3**	**83.6**
MinMax INT Quant	4/4/4	128	31.8	39.7	50.5	49.1	0.0	6.7	31.6	54.5	32.9
MinMax FP Quant (E2M1)	4/4/4	128	33.6	54.0	50.6	50.8	0.0	0.0	31.6	52.0	34.1
BrecQ (Li et al., 2021)	8/4/4	4096	31.9	62.3	50.7	50.9	0.9	6.4	31.7	52.3	35.8
QDrop (Wei et al., 2022)	8/4/4	4096	71.4	79.0	76.8	88.1	40.9	81.9	79.2	60.7	72.3
FPQ baseline	4/4/4	128	38.9	68.3	55.3	83.6	10.6	0.0	43.8	55.2	44.5
FPQ	4/4/4	128	**82.3**	**89.2**	**86.6**	**91.5**	**52.6**	**85.5**	**83.8**	**69.0**	**80.1**

Table 2: Results on the GLUE development set with BERT (Bai et al., 2022) model. We denote E/W/A as the bit-width of word embeddings, model weight and activations, respectively.

Experiments

연구 결과에 따르면,

제안하는 FPQ 활성화 방법을
Vision-Transformer에도 적용할 수 있음

FPQ를 ViT에 적용하고,
ImageNet 분류 task에서 FPQ를 FP-PTQ
baseline 및 ViT에 대한 SoTA 방법과 비교

ViT에 FPQ를 적용한 결과가 언어모델과 일치함

이전의 SoTA 방법은 low-bit로 양자화 할 때,
정확도 유지가 잘 안됨

W/A	Quant Method	Deit-S	Deit-B	ViT-S
Full-prec	-	79.9	81.8	81.4
6/6	PTQ4ViT(Yuan et al., 2022)	76.3	80.3	78.6
6/6	APQ-ViT(Ding et al., 2022)	77.8	80.4	79.2
6/6	MinMax FP Quant (E3M2)	79.3	81.7	80.7
6/6	FPQ baseline	79.43	81.7	80.9
6/6	FPQ	**79.5**	**81.8**	**81.1**
4/4	PTQ4ViT(Yuan et al., 2022)	34.1	64.4	42.6
4/4	APQ-ViT (Ding et al., 2022)	43.6	67.5	48.0
4/4	MinMax FP Quant (E2M1)	0.4	0.1	0.1
4/4	FPQ baseline	6.57	0.71	0.3
4/4	FPQ	**75.0**	**79.4**	**73.2**

Table 3: Comparison on the ImageNet dataset with vision transformer structures.

Experiments

FPQ에 대한 Calibration Size의 영향을 비교

Calibration Size를
32 / 64/ 128 / 256 으로 변화시키면서 테스트

MNLI, QQP에 대한 평가는 Calibration Size에 대해 견고함, CoLA는 분산이 더 큼

32개 Calib data와 같이 제한적인 경우에도 괜찮은 정확도를 보임

E/W/A	#Calib	MNLI-M	QQP	CoLA
4/4/4	32	81.5	**89.4**	44.4
4/4/4	64	81.8	89.4	47.9
4/4/4	128	**82.3**	89.2	52.6
4/4/4	256	81.9	89.0	**52.9**
6/6/6	32	**84.8**	90.8	55.0
6/6/6	64	84.7	**90.9**	**58.2**
6/6/6	128	84.5	90.8	57.3
6/6/6	256	84.6	90.8	57.6

Table 4: Ablation studies of different calibration sizes.

Conclusion

4-bit precision Floating-Point Post-Training Quantization를 Transformer architecture에 적용

- LLaMA, BERT 모델에 적용하여 이전 기술 대비 매우 적은 손실로 양자화

Quantized LLaMA, BERT 코드를 깃허브에서 제공하고 있음

추후에 KULLM 개선 방향에 고려해볼 수 있겠다?
(될 지 안 될 지 검토가 필요하지만)

감사합니다
Q&A