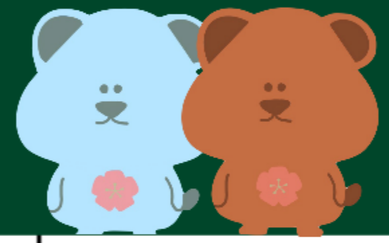


Post-training Quantization on Diffusion Models

고급심화 장윤서

EWHA EURON

목차

EWHA EURON

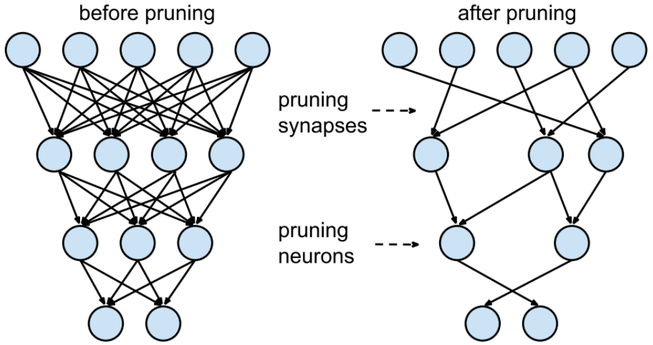
Primary knowledge

EWHA EURON

모델 경량화 기법

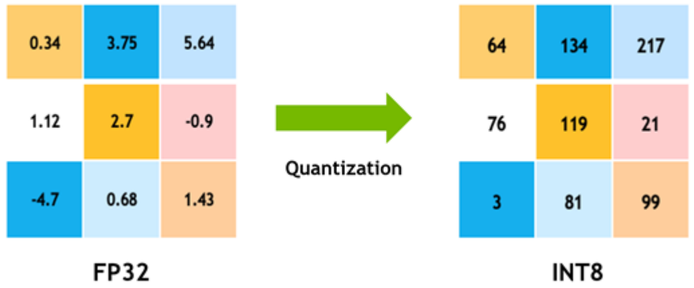
pruning

0에 가까운 애들 다 지워버리자.



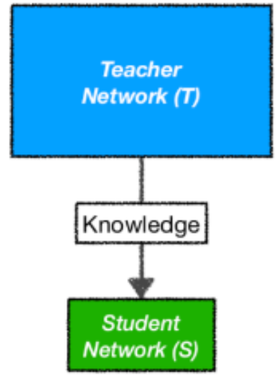
quantization

소수점 모델 파라미터를 정수화해서 연산량, 메모리 사용량 줄이자



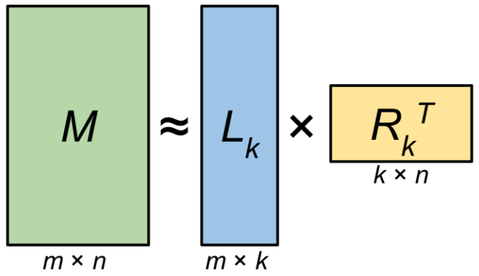
Knowledge distillation

원본 DL 모델 (크고 학습많이한 model)이 teacher, 이 모델을 다 쓸필요없거나 device에 안맞을 때 딱 task 에 맞는 작은 student model 만듬. teacher model의 prediction을 loss에 넣어서 학습



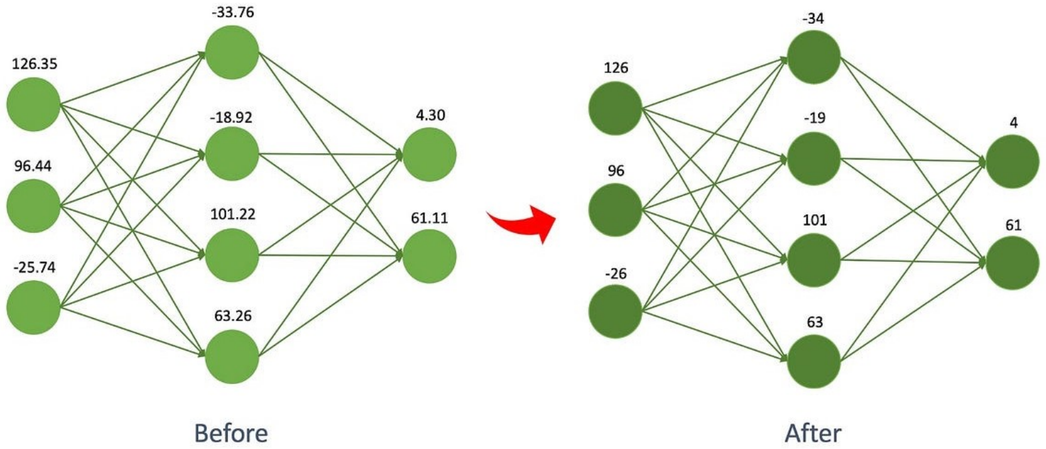
low-rank factorization

NxM이 너무 크니까 Nxk @ kxM 으로 줄인 것 (matrix factorization)



Quantinazation

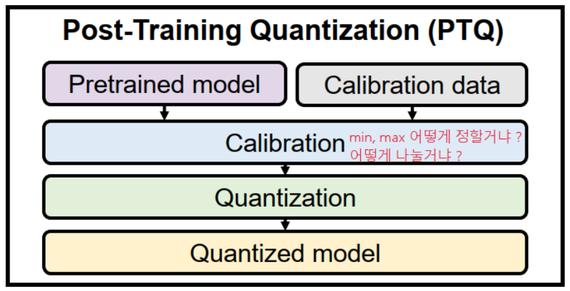
- Quantinazation, what is it?

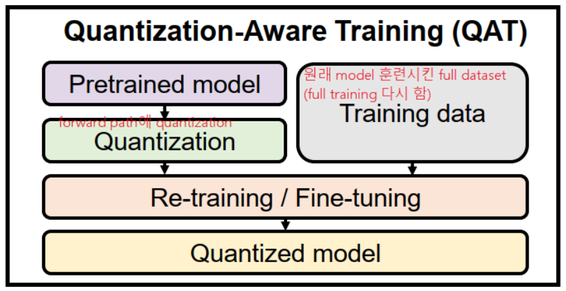


- 모델의 파라미터를 lower bit로 표현함으로서 계산과 메모리 access 속도를 높이는 경량화 기법
- 보통 32비트 부동소수점 연산을 8비트 정수로 변환하는 방식 사용
- pytorch, tensorflow의 default data type = fp32

quantization-aware training

training 이 완료된 후 quantization을 하면서 다시 training 해 나가면서 weight, activation 의 quantization 값을 조정해나가는 방법





Post training Quantinazation

post-training quantization은 학습이 끝난 모델을 quantization 하는 방법. model의 weight 뿐 아니라 model 에 입력을 넣었을 때 나오는 activation들도 quantization 함. 이때 원 모델과 비교평가하는데 쓰이는 training data를 calibration data라고 함

Introduction

EWHA EURON

Background

- **Rise of Diffusion generative models**
  - Diffusion model이 비전 생성 분야에서 새 트렌드로 떠오름
  - - 이미지 외 오디오, 비디오 그래프 등 다양한 형식의 데이터 생성 가능
  - - Downstream task implementation (super-resolution 등) 에 있어서 유연함
  - - 대부분의 task에서 SoTA 모델이였던 GAN을 압도한 성능 보임

- **Why do we need to accelerate diffusion model?**

1. Denoising process takes long iteration

   (DDPM은 모델을 1000번 정도 통과시켜야 함, GAN이 모델을 1번만 통과해도 좋은 결과가 나오는 것에 비해 느림)

2. High inference cost for generating sample images

   ➔ 생성 결과물이 좋은 반면 시간과 노이즈로부터 새 이미지를 만들어가는 과정에 많은 비용이 소모됨

   ➔현실세계 적용이 어려움

Background

- two orthogonal factors slowing down the denoising process

  i) lengthy iterations for sampling images from noise -> fast sampling

  ii) a cumbersome network for estimating noise in each iteration -> network Compression

  -> Previous work only focus on the former but overlook the latter

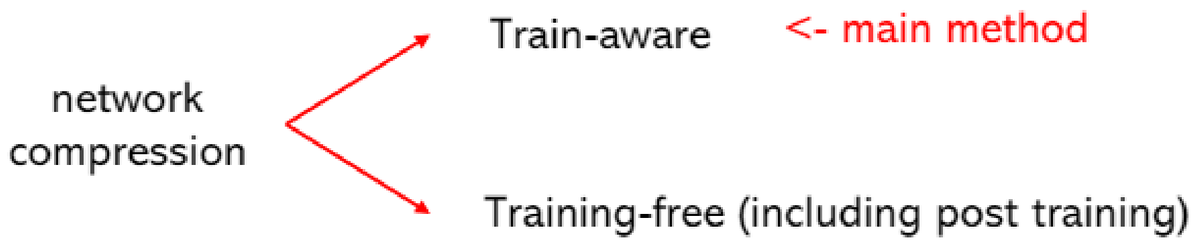
- Previously DM acceleration methods : Focus on faster sampling strategies

  ▪ propose faster step size schedules for VP diffusions with relatively good quality/diversity metrics (Chen et al, San-Roman et al)

  ▪ adopt implicit phases in the denoising process (Song et al)

  ▪ derive analytical approximations to simplify the generation process (Bao et al Lu et al)

Background



- **Train-aware method :** training the original model and then fine-tuning the quantized/pruned compressed model and fine-tuning the quantized/pruned compressed model

- **Training-aware compression not suitable for DM**

  1) training data are not always ready-to-use (due to privacy and commercial concerns)

  2) the training process is extremely expensive

Background

Introducing post-training quantization (PTQ) as DM compression method

- training-free manner
- speed up the computation of the denoising process
- reduce resources to store the weight

Challenges

PTQ methods are designed for single-time-step scenarios
Diffusion model has multiple time steps (output distribution change with time step)

Background

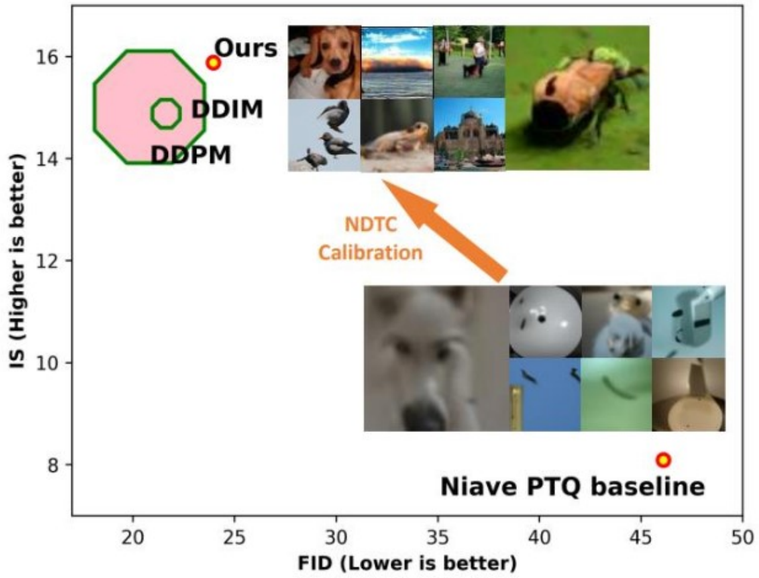
Due to multiple time step of DM, applying previous PTQ method is not affective

Post-Training for Diffusion Models (PTQ4DM) , Normally Distributed Time-step Calibration (NDTC)

DM- specific calibration module applied in PTQ4DM

1. samples a set of time-steps from a skew normal distribution
2. generates calibration samples in terms of sampled time-steps by the denoising process

- **Performance summary on ImageNet64**



FID score ; Similarity between Test/GT image feature output from model (Frechet distance)

Inception Score(IS) : 생성모델 평가지표, 생성된 이미지의 sharpness(명료도)와 diversity 결과의 곱

center (not the boundary) of the dot corresponds to the model performance

size of the dots denotes theoretical inference time

Contribution

I. introduce PTQ into DM acceleration for the first time

II. observe the performance drop induced by PTQ for DMs and explore PTQ from different aspects and propose PTQ4DM

III. PTQ4DM can quantize the pretrained diffusion models to 8-bit without significant performance loss for the first time can serve as a plug-and-play module for other SoTA DM acceleration methods

Related Work

Background

Due to multiple time step of DM, applying previous PTQ method is not affective

Previous works

Focused on finding shorter sampling trajectories while maintaining the DM

Performance

- grid search and find an effective trajectory (Chen et al)
- trajectory searching as a dynamic programming problem (Watson et al)
- non-Markovian diffusion processes which reverse process can be much faster to sample from (Song et al)
- com- press the reverse denoising process into a single-step model (Luhman & Luhman)

Our model

Implementing these models requires additional training after obtaining a pretrained DM

Diffusion models can be further accelerated through network compression for each noise estimation iteration

orthogonal path with previously developed models

can be deployed as a plug- and-play module for those methods

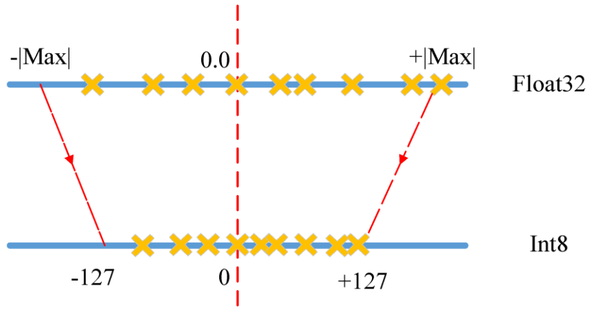
Post-training Quantization

Quantization

F : FP32 형식 모델 출력

Q : INT8 형식 모델 출력

F, Q의 관계식 :



논문의 예시 : uniform quantization

Xint : Q , X : F

Quantization 파라미터 scaling factor s and zero point z

$$F = \alpha \times Q + \beta$$

$$x_{int} = \text{clamp}(\lfloor \frac{x}{s} \rceil - z, p_{min}, p_{max}).$$

scaling factor α, zero point β

Clamp : [x/s]-z를 pmin, pmax 범위 좌표값으로 변환함

Quantization

- Quantization-aware training (QAT)
  quantization in the network training phase
- Post- training quantization (PTQ)
  quantizes the network after training

PTQ consumes much less time and computation resources therefore widely used in network deployment

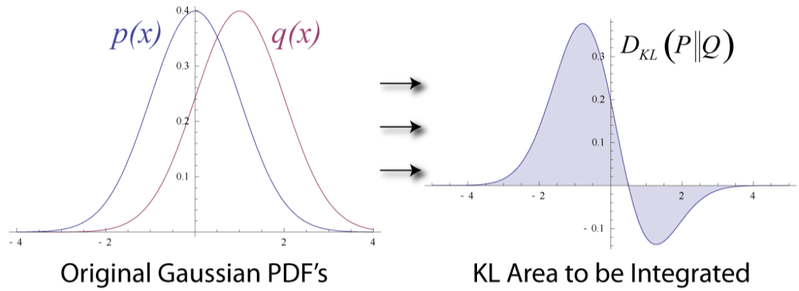
Post-training Quantization

Calibration

$$x_{int} = \text{clamp}(\lfloor \frac{x}{s} \rceil - z, p_{min}, p_{max}).$$

$$D_{KL}(P||Q) = \sum_i P[i] * log \frac{P[i]}{Q[i]}$$

KL divergence



Original Gaussian PDF's　　KL Area to be Integrated

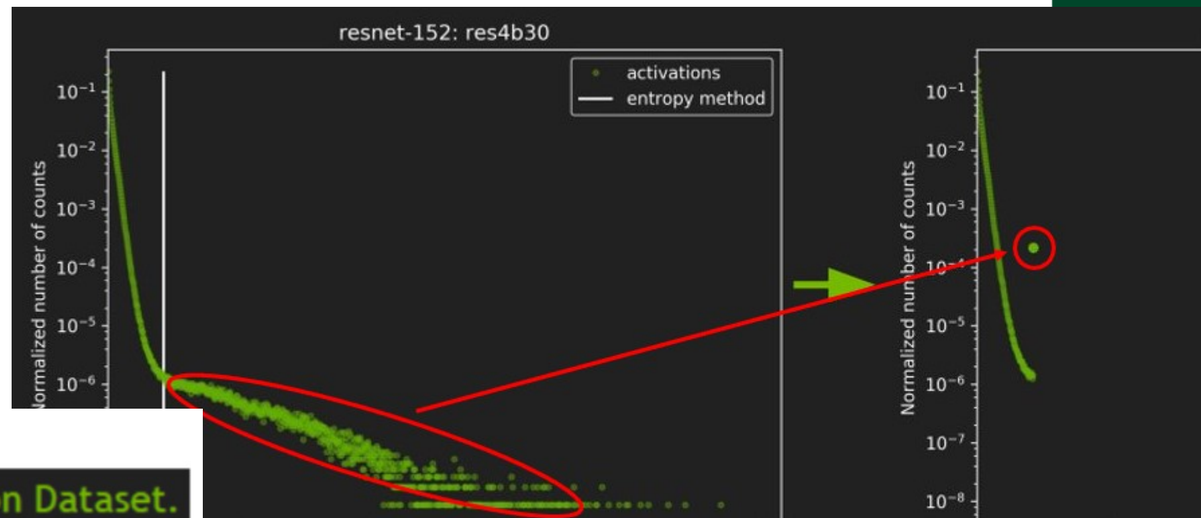
Quantization 파라미터 s, z는 원본값과 quantization된 값의 MSE를 줄이는 방식으로 결정

(= FP32 layer 출력값의 분포 P와 INT8 layer 출력값의 분포 Q의 차이)

L1 distance, cosine distance, KL divergence 등을 MSE 대신에 사용하기도 함

Post-training Quantization 과정

1. Calibration dataset을 입력해 FP32 layer의 출력 분포 P 계산
2. Calibration 통해 INT8 layer 생성
3. Calibration dataset을 입력해 INT8 layer의 출력 분포 Q 계산
4. P, Q 차이가 최소화도록 Quantization 파라미터 최적화



Calibration algorithm

- Run FP32 inference on Calibration Dataset.

PTQ on Diffusion Models

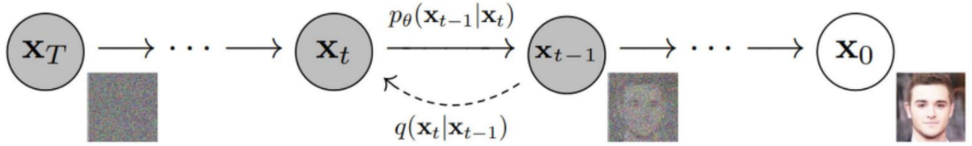
EWHA EURON

Preliminaries - Diffusion Models

Diffusion process

Reverse process : noise에서
이미지를 조금씩 추정해가는 과정

Forward process : 원본 이미지에서
noise를 점점 더해가는 과정

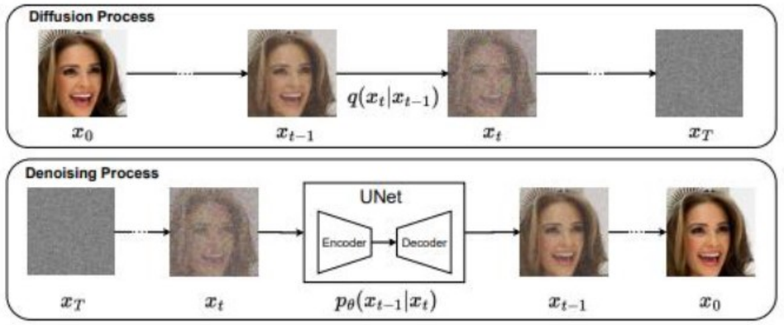


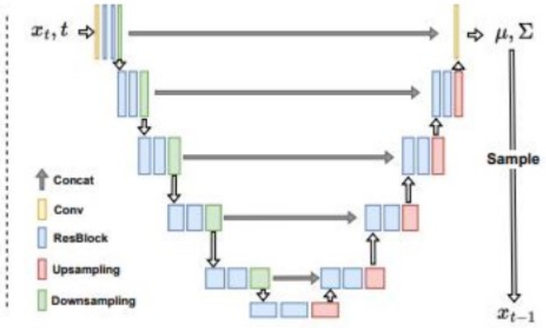
[Figure 1] Diffusion process

[Figure 2-1] Backward process

[Figure 2-2] Forward process

Diffusion model





- U-net 구조 : input / output이 동일한 형태여야 하기 때문에 이에 적절한 U-Net구조를 채택
- 생성에 있어서 모델이 참고할 추가 condition이 입력될 수 있음

이미지의 픽셀값에 노이즈를 조금씩 더해가거나 노이즈로부터 이미지를 조금씩 복원해가는 과정을 통해 이미지를 생성하는 모델

EWHA EURON

Preliminaries - Diffusion Models

Diffusion model training

실제 Noise

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

모델이 예측한 Noise

시간 t 마다 이미지 픽셀값에 첨가된 Noise 값을 계산할 수 있다면 Noise 이미지 xT가 주어졌을때 이로부터 진짜 이미지 x0를 복원하는 것이 가능

➔각 단계에서 이미지 픽셀에 추가된 노이즈를 예측함으로싸 노이즈에서 원본 이미지를 추정하는 Reverse process를 학습

실제 noise와 모델이 예측한 noise의 차이가 Loss가 됨

모델은 이 Loss를 최소화하도록 학습

Preliminaries - Diffusion Models

Diffusion Probabilistic model (DPM)

Forward(diffusion) process

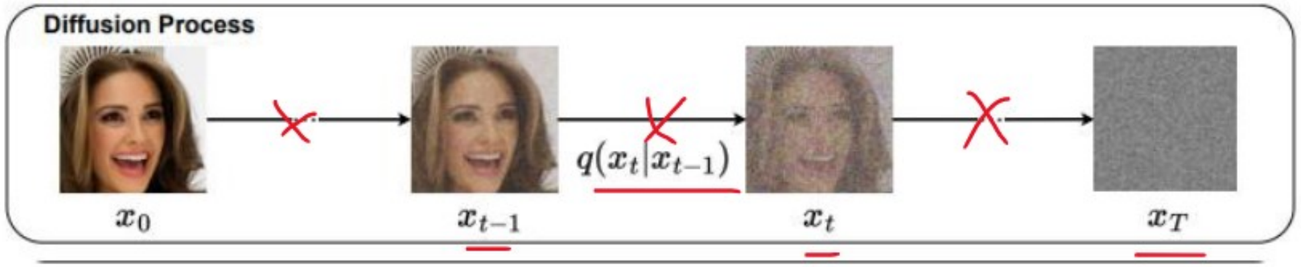
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \qquad (2)$$

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \qquad (3)$$

1. real data distribution x0 ~ q(x0) given
2. Add Scheduled Gaussian noise variance β1 , ..., βT

(0, 1) (T is sufficiently large T ~ ∞)

3. produce a sequence of latent x1, ..., xT which is fixed to a Markov chain



Preliminaries - Diffusion Models

Diffusion Probabilistic model (DPM)

Markov Chain

특정 상태의 확률은 오직 과거의 상태에 의존한다는 성질을 가진 이산 확률과정

0 번째 부터 t 번째 까지의 모든 data(; 과거)가 있을 때, (t+1)번째 데이터(; 미래)는 t 번째 데이터에만 영향을 받는다 = 직전 데이터만 가지고 방금 수행된 transition을 추정할 수 있다

마르코프 성질

$$P(X_1, X_2, \ldots, X_n) = P(X_1) \times P(X_2|X_1) \times P(X_3|X_2) \times P(X_4|X_3) \times \cdots \times P(X_n|X_{n-1})$$

$$P(X_{t+1}|X_t, X_{t-1}, \ldots, X_2, X_1) = P(X_{t+1}|X_t)$$

$$P(X_1, X_2, \ldots, X_n) = P(X_1) \times P(X_2|X_1) \times P(X_3|X_2, X_1) \times \cdots \times P(X_n|X_{n-1}, X_{n-2}, \ldots, X_1)$$

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \qquad (3)$$

X0이 주어졌을 때 X1 ~ XT까지 결합 확률 분포는 마르코프 성질에 의해 n=1…T에 대해 각 Xn-1가 존재할 때 Xn가 존재할 조건부 확률 P(Xn|Xn-1)의 곱으로 나타낼 수 있다

EWHA EURON

Preliminaries - Diffusion Models

Diffusion Probabilistic model (DPM)

Forward(diffusion) process

Markov Chain Monte Carlo을 통해 연산하면 x0의 조건에서 xt를 직접 샘플링할 수 있음

$$\text{Let } \alpha_t = 1 - \beta_t \text{ and } \bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i:$$

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I}) \qquad (4)$$

4. parameterize a neural network to approximate q

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)) \qquad (5)$$

q(xt−1|xt) 가 원본 확률분포인 q(x0)에 의존하므로 다음과 같이 q(xt−1|xt)의 추정값인 p(xt-1|xt)를 parameterize할 수 있다

Preliminaries - Diffusion Models

Training process

5. Compute Lvlb

$$L_{\text{VLB}} = \mathbb{E}_q[\underbrace{D_{\text{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_T))}_{L_T} \underbrace{- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}$$
$$+ \sum_{t=2}^{T} \underbrace{D_{\text{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}}] \quad (7)$$

Lvlb : utilize the variational lower bound to optimize the negative log-likelihood

Lt−1 is a KL-divergence to directly compare pθ(xt−1|xt) to diffusion process posterior when x0 is given

6. mean, var computation

$$\tilde{\beta}_t := \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \cdot \beta_t \quad (8)$$

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t \mathbf{x}_0) := \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1 - \bar{\alpha}_t} \mathbf{x}_0 \quad (9)$$

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) := \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}). \quad (10)$$

Then goes to denoising process by iterative sampling new xt−1 from pθ(xt−1|xt) until we receive x0.

EWHA EURON

Preliminaries – PTQ

Training process

Quantization error Lquant formula

$$X_{sim} = s(\text{clamp}(\lfloor \frac{X_{fp}}{s} \rceil - z, p_{min}, p_{max}) + z), \quad (11)$$

$$L_{quant} = \text{Metric}(X_{sim}, X_{fp}), \quad (12)$$

Xsim : dequantized tensor

Xfp : full-precision tensor

Metric : metric function to evaluate the distance of Xsim and Xfp

Minimize the quantization error

$$\arg\min_{s,z} L_{quant}. \quad (13)$$

Exploration on Operation Selection

Select which operations in the network should be quantized

CNN base computation : Follow previous PTQ methods

Quantized

- computation-intensive convolution layers and fully-connected layers

keep full-precision

- special functions such as SiLU and softmax

Generation of μ, Σ, or xt−1 : select throught experiments

(At each denosing iteration, DM gets inputs xt and t computes μ, Σ and pass it to next iteration to estimate xt-1)

-> 성능을 크게 해치지 않으므로 quantize

Table 1. Exploration on operation selection for 8-bit quantization. The diffusion model is for unconditional ImageNet 64x64 image generation with a cosine noise schedule. DDIM (250 timesteps) is used to generate 10K images. IS is the inception score.

	IS	FID	sFID
FP	14.88	21.63	17.66
quantize μ	15.51	21.38	17.41
quantize Σ	15.47	21.96	17.62
quantize x_{t-1}	15.26	21.94	17.67
quantize μ+Σ+x_{t-1}	14.94	21.99	17.84

Exploration on Calibration Dataset

Calibration data samples are needed to calculate loss between quantize tensor and original tensor.

Distribution of the collected calibration samples should be as close as possible to the distribution of the real data

Why collection of Calibration samples so important?

- Over fitting issue
- small size (extremely sensitive collection required)

Exploration on Calibration Dataset

Issues collecting calibration samples for DM

1. DM denoising network generates it' s own input data

- The calibration samples can be collected from the training data set. However, the training dataset in the diffusion model is x0, which is not the denoising network' s actual input.
  - The real input is the generated samples xt.

2. DM has multitime-step scenario

Time step t (t = 0, 1, · · · , T)

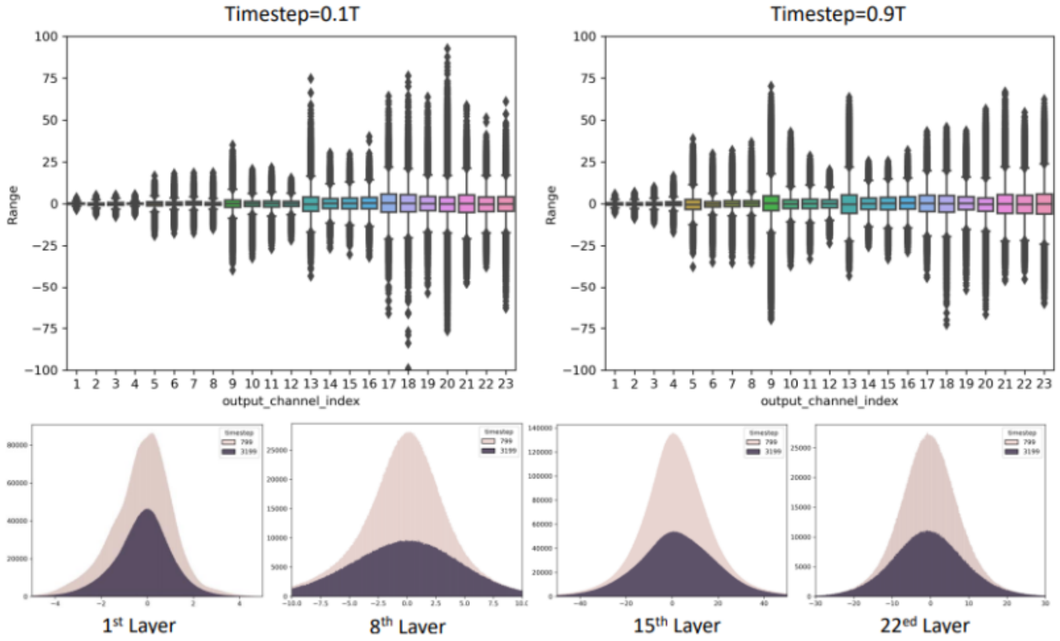
- novel and effective calibration dataset collection method for particular multitimestep scenario is required

Analysis on PTQ Calibration and DMs

Observation 0: Distributions of activations changes along with time-step changing.

실험 0 : 출력된 노이즈의 분포가 타임 스탬프에 따라 차이가 있는지 확인한다

Let t1 = 0.1T and t2 = 0.9T



결론
previous PTQ calibration methods
[22, 17] inapplicable for DM

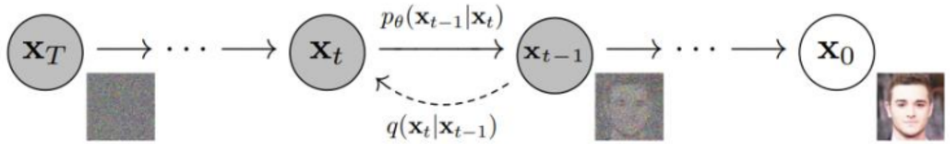
1. overall activation distributions of the noise estimation network by boxplot
2. layer-wise distributions via histogram

Analysis on PTQ Calibration and DMs

Observation 1: Generated samples in the denoising process are more constructive for calibration.

실험 1 : raw 이미지 인풋과 노이즈 인풋 중

어느 경우가 더 나은 calibration data일까?



input noises for diffusion process

디노이징 과정에서 뽑힌 샘플이 제일 좋은 성능 보임

➔ training-mimic baseline으로 명명

Table 2. Results of calibration using noise (input of the denoising process), image (input of the diffusion process), and samples generated by Eq. 3 (Mimicking the diffusion model training).

	IS ↑	FID ↓	sFID ↓
Noise Samples	13.92	33.15	20.38
Image Samples	6.90	128.63	90.04
Training-mimic	12.91	34.55	25.18

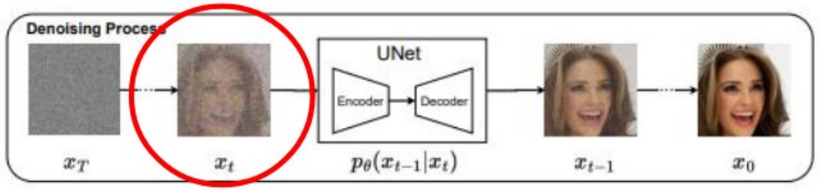
➔ 결론

Image + Gaussian Noise

Analysis on PTQ Calibration and DMs

Observation 2: Sample xt close to real image x0 is more beneficial for calibration.

실험 2 : 디노이징 과정에서 몇번째 타임 스탭의 이미지가
성능이 좋을까?
i.e., t = 0.0T, 0.2T, · · · , 1.0T



결론 :
PTQ calibration helps more when the time-step t approaches the real image x0

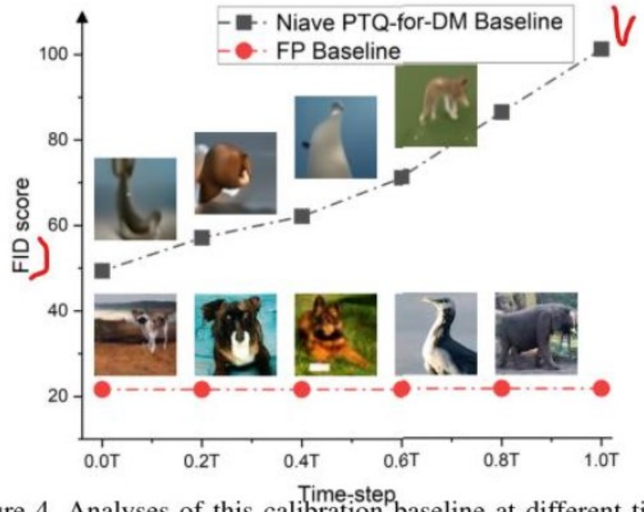


Figure 4. Analyses of this calibration baseline at different time-steps. FP Baseline denotes the 32-bit model, which does not require to be calibrated.

직관적 설명
distribution of outputs of network $p_\theta(x_{t-1}|x_t)$ is similar to real images' distribution

Analysis on PTQ Calibration and DMs

Observation 3: Instead of a set of samples generated at the same time-step, calibration samples should be generated with varying time-steps.

실험 3 : DM의 calibration data는 타임 스탭의 다양성을 고려해 각 다른 타임 스탭에서 수집되어야 한다는 가설 검증

Ob2 : Image + Gaussian Noise calibration data

Ob3 : real image x0 에 가까운 Image + Gaussian Noise calibration data

결론 :

calibration samples should reflect the time-step discrepancy

Table 3. Quantitative results of the intuitive baselines for the observations and our proposed *NDTC* calibration method. With our method, the performance of PTQ for DM has been significantly improved, even exceeding full-precision DM performance w.r.t.IS and sFID.

	IS ↑	FID ↓	sFID ↓
Full precision DDIM	14.88	21.63	17.66
Baseline in Observation 2	11.92	49.37	41.33
Baseline in Observation 3	14.99	26.19	19.51
NDTC (ours)	**15.68**	**24.26**	**17.28**

Analysis on PTQ Calibration and DMs

Conclusion

Calibration data for PTQ should be…

(1) generated by the denoising process (from noise xT) with the full-precision diffusion model;

(2) relatively close to x0, far away from xT ;

(3) covered by various time-steps

Note that (2) and (3) are a pair of trade-off conditions, which can not be satisfied simultaneously

Normally Distributed Time-step Calibration

Algorithm 1 Normally Distributed Time-step Calibration Collection (*DNTC*) Algorithm.

Input: The size of calibration set N, and a mean of the Normal distribution μ, and the full-precision noise estimation network $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ in Eq. 5.

Output: Obtain a Calibration Set $\mathcal{C}$.

Collecting Calibration Set:
for $i = 1$ to N **do**
  Sample t_i from distribution $\mathcal{N}(\mu, \frac{T}{2})$ in Eq. 15;
  Round down t_i into a integer, *i.e.*, $t_i = \lfloor t_i \rfloor$;
  Clamp t_i between $[0, T]$, *i.e.*, $t_i = \text{Clamp}(0, T, t_i)$;
  Produce sample on t_i time-step:
  for $t = T$ to t_i **do**
    Generate a Gaussian Noise $\mathbf{x}_T$ as initialization;
    Sample $\mathbf{x}_{t-1}$ using $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$;
  end for
  Output sample x_{t_i};
end for
Output a calibration set $\mathcal{C} = \{\mathbf{x}_{t_i}\}_{i=1}^{N}$.

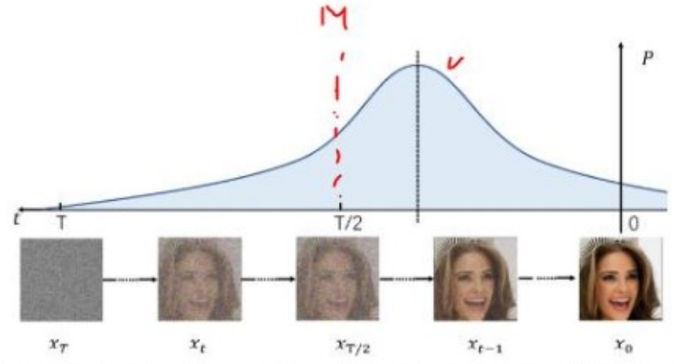


Figure 5. A general illustration of sampling time-steps following a distribution over the range of the denoising time-step.

NDTC
(Normally Distributed Time-step Calibration)

DM specific calibration set collection method

calibration set {xti } are generated by the denoising process (for condition 1)

- time-step ti are sampled from a skew Normal distribution (for balancing conditions 2 & 3)

$$t_i \sim \mathcal{N}(\mu, \frac{T}{2}) \ \ (i = 1, 2, \cdots, N), \qquad (15)$$

1. generate a set of sampled {ti} over the timestep range (satisfying condition 3)
2. μ is less than or equal to the median of timestep, T 2 (satisfying condition 2).

Exploration on Parameter Calibration

Table 4. Exploration on calibration metric for 8-bit quantization. We set p=2.4 for MSE metrics.

	IS ↑	FID ↓	sFID ↓
L1 distance	7.38	100.52	63.01
Cosine distance	12.85	34.81	23.75
KL divergence	11.74	47.27	45.08
MSE	13.76	30.46	19.42

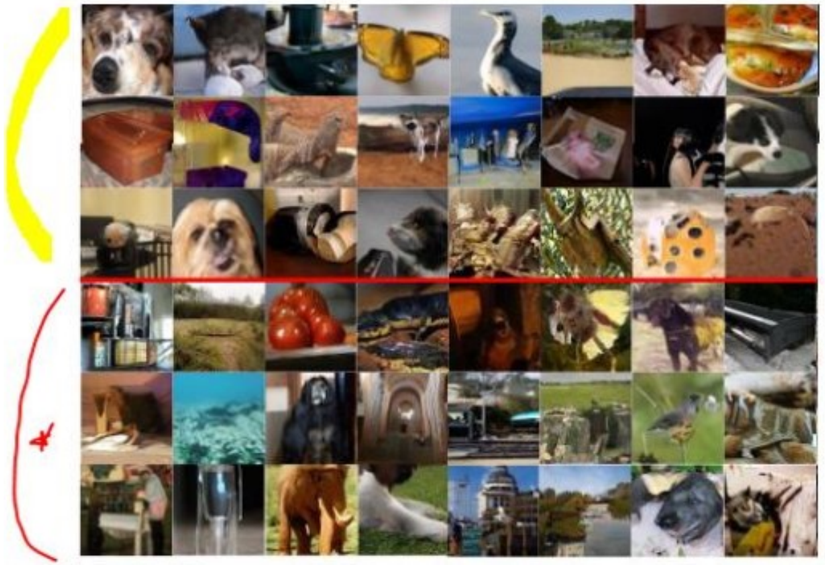


Figure 6. Non-cherry-picked generated samples. **(Upper)** Samples synthesized by full precision DDPM [11]. **(Bottom)** Samples synthesized by 8-bit model quantized by our method. Note that PTQ4DM can directly output an 8-bit diffusion with the pre-trained 32-bit diffusion model as input in a *training-free* manner.

- 최적의 calibration Loss 계산 방법 실험-> MSE 채택
- 실험에서 얻은 방식을 종합해 PTQ4DM 모델 개발

실제 이미지 생성 실험

-> 생성된 이미지가 DDPM과 유사한 퀄리티를 보여줌

More Experiments & Appendix

EWHA EURON

Model Performance

Task :
generating CIFAR10 32 × 32 images & ImageNet 64 × 64 images

PTQ4DM : 8-bit quantized with 1024 calibration sample data

1) DDPM 모델 베이스 PTQ4DM quantized 한 모델을 원본 DDPM 모델과 비교

2) DDIM 모델 베이스 PTQ4DM quantized 한 모델을 원본 DDIM 모델과 비교

-> Quantization 모델이 원본 모델과 유사한 성능을 보임

Table 5. Experiment on 8-bit quantized diffusion models generating CIFAR10 image or ImageNet image.

Task	Method	IS ↑	FID ↓	sFID ↓
ImageNet 64x64 DDIM 100 steps	FP	15.38	21.70	17.93
	PTQ4DM	15.52	24.92	17.36
ImageNet 64x64 DDIM 250 steps	FP	14.88	21.63	17.66
	PTQ4DM	15.88	23.96	17.67
ImageNet 64x64 DDPM 4000 steps	FP	15.93	20.82	17.42
	PTQ4DM	15.28	23.64	17.29
CIFAR 32x32 DDIM 100 steps	FP	9.18	10.05	19.71
	PTQ4DM	9.31	14.18	22.59
CIFAR 32x32 DDIM 250 steps	FP	9.19	8.91	18.43
	PTQ4DM	9.70	11.66	19.71
CIFAR 32x32 DDPM 4000 steps	FP	9.28	7.14	17.09
	PTQ4DM	9.55	7.10	17.02

FP : full-precision DM (quantization x)

DDPM / DDIM
DDPM이 초반 모델, DDIM은 DDPM을 발전시켜 iteration step 수를 좀 더 줄인것

Nonuniform Distribution Selection

Choosing the best Distribution by experiments

Nonuniform distribution :

Poisson , 지수분포

hyperparameter selection experiments :

정규 분포 중 최적의 평균,

분산 하이퍼파라미터 선택

T : time step

Table 6. Hyperparameter selection for non-uniform distribution in Algorithm 1.

Task	Method	IS ↑	FID ↓	sFID ↓
	FP	14.88	21.63	17.66
Other Nonuniform Distributions	Poisson	13.29	34.54	25.84
	Exponential	12.87	39.91	30.04
Normal Distribution with Different Mean μ, and Variance σ	$\mu = \frac{T}{2},\ \sigma = 0.5\sqrt{\frac{T}{2}}$	15.45	25.11	17.35
	$\mu = \frac{T}{2},\ \sigma = 1.0\sqrt{\frac{T}{2}}$	15.65	24.83	18.90
	$\mu = \frac{T}{2},\ \sigma = 2.0\sqrt{\frac{T}{2}}$	15.85	24.27	17.92
	$\mu = \frac{1.5T}{2},\ \sigma = \sqrt{\frac{T}{2}}$	12.63	39.09	35.81
	$\mu = \frac{1.0T}{2},\ \sigma = \sqrt{\frac{T}{2}}$	15.65	24.83	18.90
	$\mu = \frac{0.5T}{2},\ \sigma = \sqrt{\frac{T}{2}}$	15.88	23.96	17.67

-> skew Normal distribution was applied to final model

Actual Acceleration

Latency(ms)

original network vs quantized Network

Hardware

on Nvidia RTX A6000 GPU

Table 7. Inference speed test with Nvidia RTX A6000.

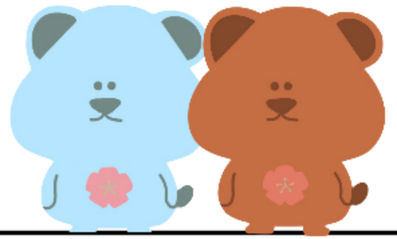
Task	Batch Size	FP32	INT8
ImageNet	1	9.80	4.99
64x64	16	64.42	28.16
CIFAR	1	5.92	2.98
32x32	16	23.15	14.13

Result

-> 8-bit quantization is about 2x faster

speedup can be more significant on NPU

QNA



EWHA EURON