DeepSynk

Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference

김민준

2024. 02. 21.

TABLE OF CONTENTS

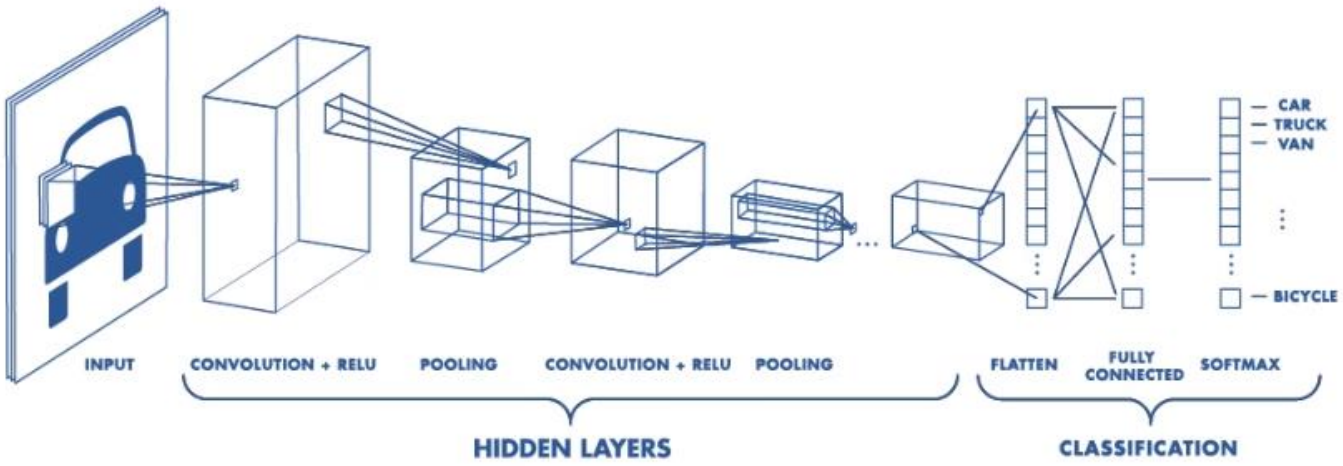
Chapter I

Background

- Alexnet의 등장 이후 -> CNN은 분류 탐지 정확도에 초점
- 모델의 복잡성, 계산 효율성에 대한 고려↓

Convolutional Neural Network





DeepSynk

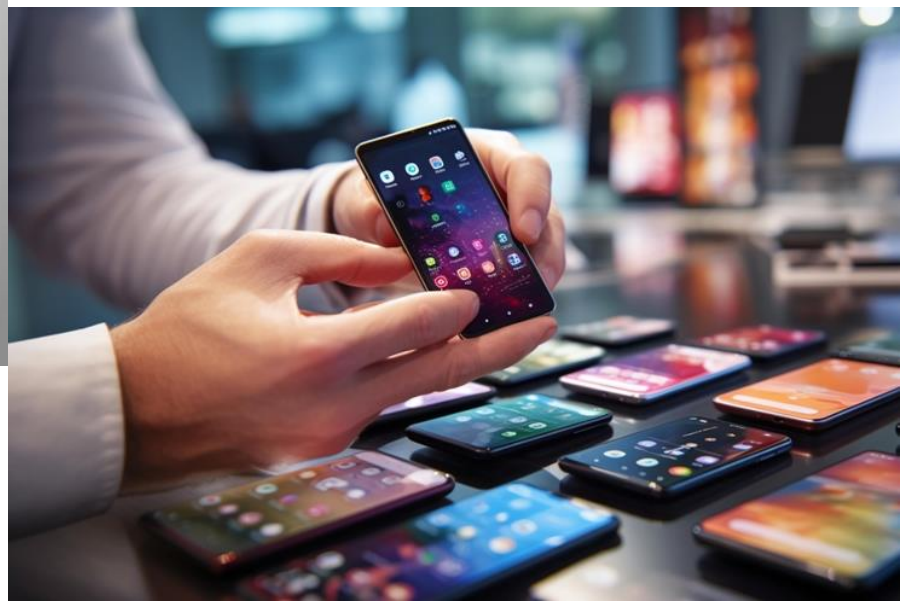
Chapter I

Background

- 스마트폰, AR/VR 디바이스, 드론과 같은 모바일 플랫폼에 CNN 적용

➔ 온디바이스 메모리 대응할 작은 크기의 모델

➔ 사용자 참여를 유도할 낮은 지연시간 필요



DeepSynk

Background

● 기존 해결방법

1. MobileNet, SqueezeNet과 같은 새로운 네트워크 아키텍처

2. CNN의 가중치와 중간값을 양자화(quantize)

➔ 32bit 부동 소수점을 낮은 bit로 표현, 정확도<->지연시간 trade off

● 한계점

1. AlexNet, VGG, GoogleNet 성능을 높이기 위해 과도한 파라미터화

➔ 양자화 시 상당한 압축률, 그러나 커다란 정확도 손실 (개념 증명 용도)

2. MobileNet과 같은 정확도, 지연시간 트레이드오프에 효율적인 모델 아키텍쳐를 양자화 하는 것이 의미가 있을 것

Chapter II

Quantization schema

DeepSynk

Quantization scheme

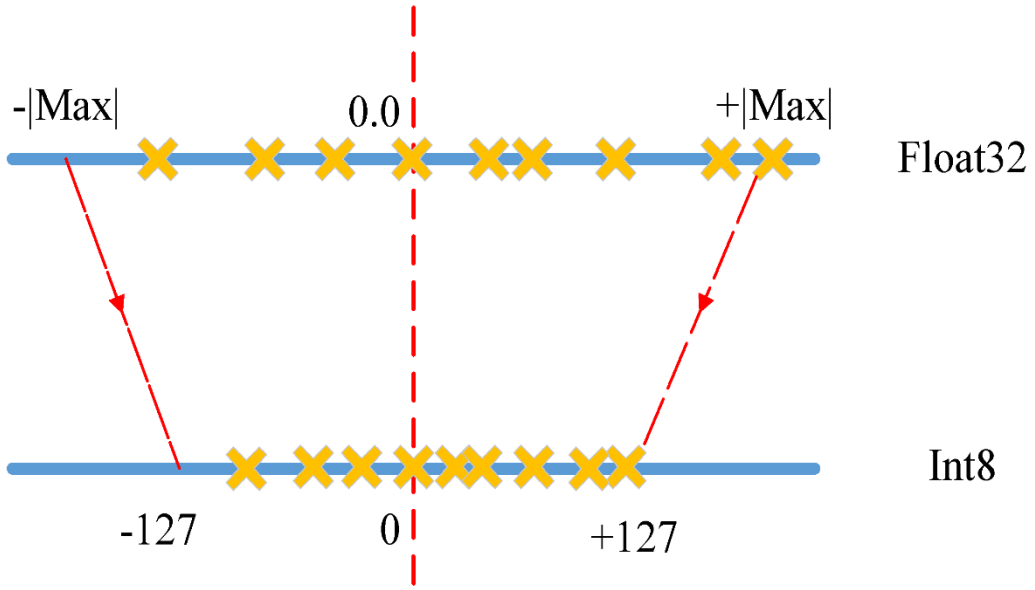
양자화 스키마 -> 실수를 정수로 변환, 양자화된 값에 대해 정수 연산만을 사용하여 모든 연산을 효율적으로 구현

$$r = S(q - Z)$$

$$q = r/S + Z$$

q: quantized value (양자화된 값)

R: real value (실제 값)



- Quantization parameter -

S: Scale (양자화된 값에서 실제 값으로 변환하는 데 사용되는 step의 크기)

Z: Zero point (부동 소수점 값이 양자화된 값으로 변환될 때의 오프셋)

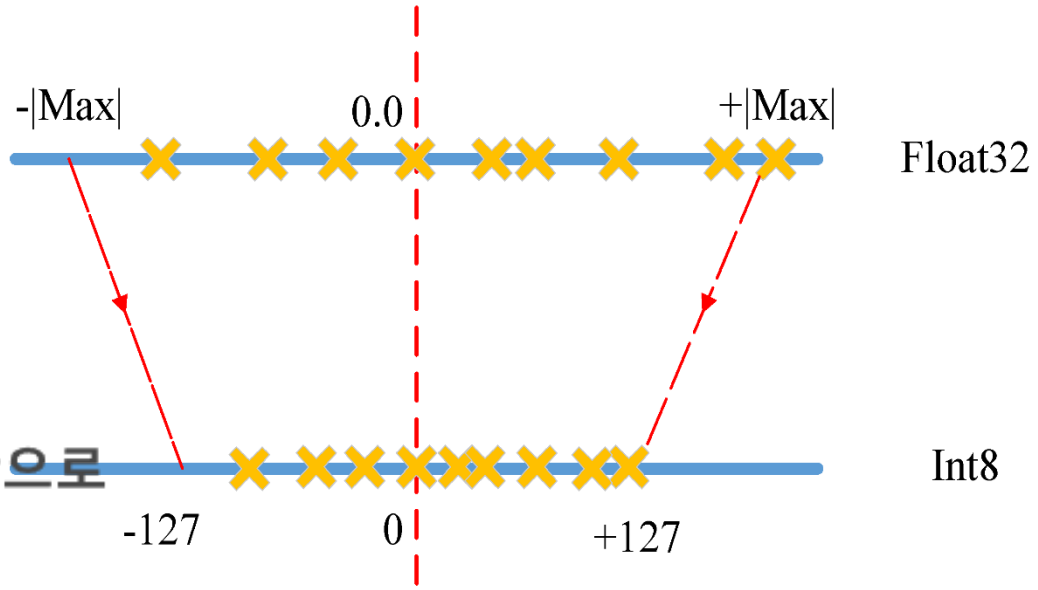
Quantization scheme

양자화 스키마 -> 실수를 정수로 변환, 양자화된 값에 대해
정수 연산만을 사용하여 모든 연산을 효율적으로 구현

S = 0.1, Z = 5, r = 0.7

Q = r/S + Z

Q = 0.7/0.1 + 5 = 12



반대로 실수값 복원도 가능

➔ 모든 산술 연산이 정수 산술 만으로 수행될 수 있도록 보장

메모리 효율성, 연산 효율성 증대

Chapter III

Quantized inference framework

DeepSynk

Integer-arithmetic-only matrix multiplication

$$r_\alpha^{(i,j)} = S_\alpha(q_\alpha^{(i,j)} - Z_\alpha).$$

multiplication of two square N × N matrices of real numbers, r1 and r2, with their product represented by r3 = r1r2. We denote the entries of each of these matrices rα (α = 1, 2 or 3), 1<=i, j<=N and the quantization parameters with which they are quantized as (Sα, Zα).

Integer-arithmetic-only matrix multiplication

행렬 곱셈의 정의에 따른 계산:

$$S_3(q_3^{(i,k)} - Z_3) = \sum_{j=1}^{N} S_1(q_1^{(i,j)} - Z_1) S_2(q_2^{(j,k)} - Z_2), \quad (3)$$

수식의 재작성

$$q_3^{(i,k)} = Z_3 + M \sum_{j=1}^{N} (q_1^{(i,j)} - Z_1)(q_2^{(j,k)} - Z_2),$$

이때의 M 값

$$M := \frac{S_1 S_2}{S_3}.$$

정규화된 형태

$$M = 2^{-n} M_0$$

DeepSynk

Integer-arithmetic-only matrix multiplication

식의 재구성

$$q_3^{(i,k)} = Z_3 + M \left(NZ_1Z_2 - Z_1a_2^{(k)} - Z_2\bar{a}_1^{(i)} + \sum_{j=1}^{N} q_1^{(i,j)} q_2^{(j,k)} \right)$$

where

$$a_2^{(k)} := \sum_{j=1}^{N} q_2^{(j,k)}, \quad \bar{a}_1^{(i)} := \sum_{j=1}^{N} q_1^{(i,j)}.$$

Chapter IV

Quantized training framwork

DeepSynk

Quantized training framework

양자화된 신경망 학습하는 일반적인 접근 방식

➔ 먼저 부동 소수점으로 훈련한 다음 양자화된 가중치를 얻는 것

➔ 대규모 모델에서는 잘 작동, 작은 모델에서는 정확도 저하

이유

➔ 각 출력 채널의 가중치 범위 사이의 큰 차이(작은 채널 상대 오차 ↑)

➔ 이상값 가중치 값은 양자화 후 가중치의 부정확화 심화

제안

➔ 순방향 패스에서 양자화 효과를 시뮬레이션 하는 접근 방식을 제안

➔ 훈련 중에도 양자화 효과를 시뮬레이션히야 정확도 감소 방지

➔ Weights: 입력과 컨볼루션 전에 양자화

➔ Activations: 활성화 함수가 컨볼루션 or FC layer의 output에 적용되고 난 뒤(혹은 여러 layer의 concat 후) 양자화

Quantized training framework

$$\begin{aligned}
\text{clamp}(r; a, b) &:= \min\left(\max(x, a), b\right) \\
s(a, b, n) &:= \frac{b-a}{n-1} \\
q(r; a, b, n) &:= \left\lfloor \frac{\text{clamp}(r; a, b) - a}{s(a, b, n)} \right\rceil s(a, b, n) + a,
\end{aligned} \tag{12}$$

양자화를 수행하는 함수

➔ 양자화 수준과 클램핑 범위에 의해 매개변수화

➔ 주어진 양자화 범위 내에서 양자화 수준에 따라 실수를 정수로 변환하는 역할

Chapter V

Experiments

DeepSynk

양자화된 훈련의 효과

Act.	type	accuracy		recall 5	
		mean	std. dev.	mean	std.dev.
ReLU6	floats	78.4%	0.1%	94.1%	0.1%
	8 bits	75.4%	0.1%	92.5%	0.1%
	7 bits	75.0%	0.3%	92.4%	0.2%
ReLU	floats	78.3%	0.1%	94.2%	0.1%
	8 bits	74.2%	0.2%	92.2%	0.1%
	7 bits	73.7%	0.3%	92.0%	0.1%

Table 4.3: Inception v3 on ImageNet: Accuracy and recall 5 comparison of floating point and quantized models.

- 7비트, 8비트로 양자화된 인셉션 모델 비교

정확도 – 지연시간 trade off1

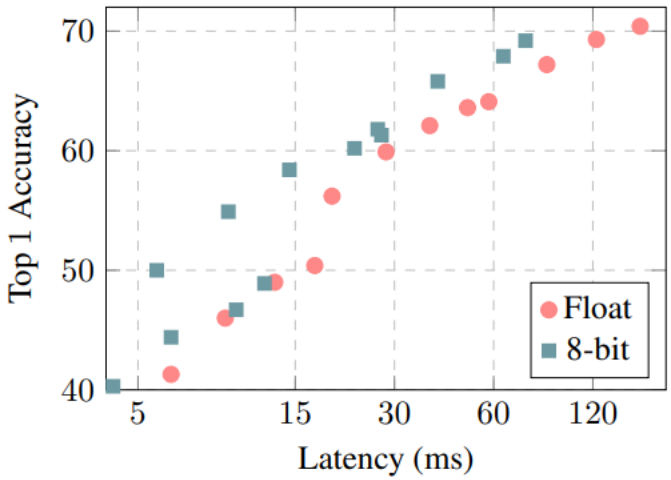


Figure 4.1: ImageNet classifier on Qualcomm Snapdragon 835 big cores: Latency-vs-accuracy tradeoff of floating-point and integer-only MobileNets.

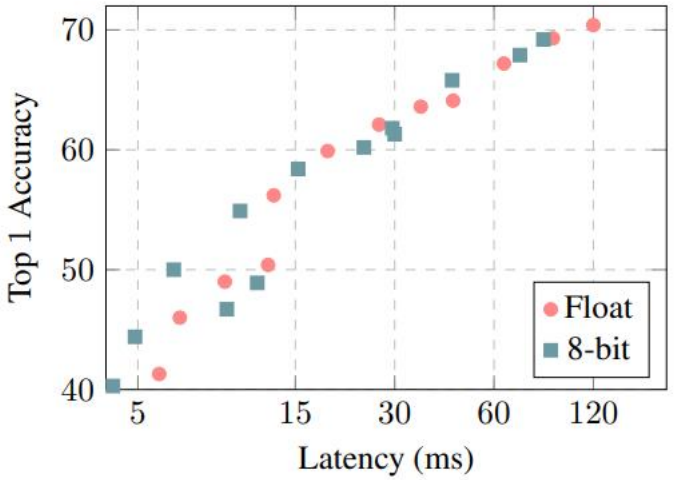


Figure 4.2: ImageNet classifier on Qualcomm Snapdragon 821: Latency-vs-accuracy tradeoff of floating-point and integer-only MobileNets.

Integer-only가 floating-point보다 동일한 런타임이 주어졌을 때, 더 높은 가속도를 달성

DeepSynk

정확도 – 지연시간 trade off2

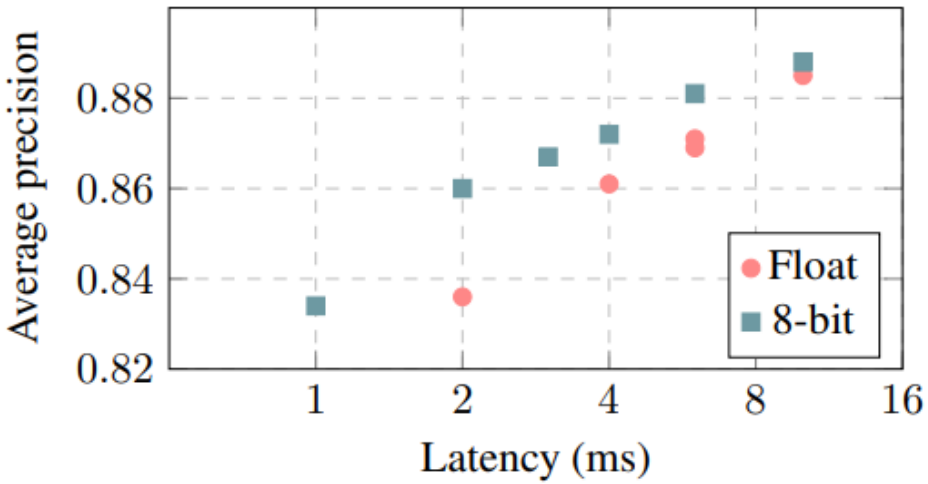


Figure 4.3: Face attribute classifier on Qualcomm Snapdragon 821: Latency-vs-accuracy tradeoff of floating-point and integer-only MobileNets.

Face detection에서도 마찬가지로 Integer-only가 floating-point보다 동일한 런타임이 주어졌을 때,더 높은 가속도를 달성

Thank you

김민준