TFHE 기반 CNN 연산 최적화를 위한 비산술연산의 양자화 기술 연구

남기빈[1], 정헌희[1], 이동주[1], 백윤흥[1]
[1]서울대학교 전기정보공학부, 서울대학교 반도체 공동연구소
{kvnam, hhjung, djlee}@sor.snu.ac.kr, ypaek@snu.ac.kr

Quantization of Non-Arithmetics to Optimize CNNs over TFHE

Kevin Nam[1], Heonhui Jung[1], Dongju Lee[1], Yunheung Paek[1]
[1]Dept. of Electrical and Computer Enginering and Inter-University Semiconductor Research Center(ISRC), Seoul National University

요 약

동형암호는 주목받는 차세대 프라이버시 보존 기술이며, 이를 활용한 신경망 연산 연구들이 많이 수행되고 있다. 여러 체계들 중 TFHE는 비산술 연산을 직접 연산할 수 있으나 다른 체계들보다 매우 느리다는 단점이 있다. 본 연구는 정확도 하락을 최소화하며 성능 개선을 통해 다른 체계인 CKKS보다 빠른 TFHE기반 CNN 연산이 가능하도록 하는 TFHE 비산술 연산의 양자화 기술을 소개한다.

1. 서론

동형암호는 데이터를 암호화 상태로 연산할 수 있는 암호체계다. 사용자는 이런 장점을 사용하여 프라이버시 이슈 없이 원격 클라우드 기반 신경망 연산 서비스들을 수행할 수 있다. 하지만 일반적으로 산술연산만 지원하는 동형암호는 비산술연산은 직접 지원하지 않는다. 이들을 산술 근사식으로 대체 연산할 수 있으나[1], 신경망의 정확도 하락으로 이어지곤 한다. 반면, TFHE라는 동형암호 체계를 활용하면 비산술연산을 근사 없이 연산할 수 있다. 하지만, TFHE는 다른 동형암호 체계 (예: CKKS) 보다 훨씬 느린 성능을 갖고 있다.

본 연구는 대표적인 신경망과 비산술 연산인 CNN과 ReLU/MAX에 대한 양자화 기술을 다룬다. 구체적으로, 본 연구는 신경망 연산 최적화를 위해 기존 CNN을 양자화하는 기술을 동형암호 특성에 맞게 적용하는 방법을 소개한다. 기존 TFHE의 산술연산에 대한 DQ 연구를 출발점으로, 본연구의 추가 최적화를 통해 CKKS보다 빠르고 정확한 CNN 연산이 가능함을 확인할 수 있다.

2. 이론적 배경

2.1 동형암호, 그리고 TFHE

동형암호 체계들[2,3]은 2^{10} 이상의 차수를 갖는 큰 다항식 형태의 암호문을 활용한다. 일반적으로 큰 차수(N)의 다항식을 쓸수록 더 정교한 데이터 타입을 암호화할 수 있으나, 연산 속도가 느려진다. 랜덤 a와 비밀키 s, 그리고 보안성을 위해 추가되는 noise e를 활용하여, $c = (a \cdot s + m + e, a)$ 라는 수식을 통해 message m을 암호화한다. 암호문 연산을 거듭할수록 noise이 커지면 올바른 복호화에 영향을 주기 때문에, noise를 작게 만들어주는 재부팅(bootstrapping, BTS)이라는 연산이 필요하다. 재부팅은 수없이 많은 다항식 연산들로 구성되며, 가장 느린 동형암호 함수로 알려져있다. 많이 쓰이는 체계로, CKKS가 있다.

	CKKS [2]	TFHE [3]
암호문 크기	MBs	KBs
message precision	실수, 복소수	실수, 정수, **bool**
지원 연산	산술(덧셈/곱셈)	산술/**비산술**
BTS (PBS) 속도	**느림**	**빠름**
BTS(PBS) 빈도	**낮음**	**높음**

표 1 : CKKS와 TFHE 비교

TFHE는 CKKS와 다르게 BTS대신에 Programmable Bootstrapping(PBS)를 수행한다. 이는 재부팅 과정에서 원하는 비산술 연산을 동시에 수행할 수 있는 연산이다. <표 1>은 CKKS와 TFHE에 대한 주요 비교사항들을 나타낸다. 여기서 BTS/PBS의 속도와 빈도에 대해 주목할 필요가 있다. PBS는 BTS에 비해서 연산 속도가 매우 빠르지만, 훨씬 높은 연산 빈도를 필요로 한다. 그 이유는,

CKKS는 noise가 지나치게 커질 때만 BTS를 수행하면 되지만, TFHE는 비산술 연산을 수행해야할 때마다 PBS를 수행해야 하기 때문이다. 비산술 연산을 수행할 수 있는 능력이 역설적으로 큰 연산 부하로 이어진다는 뜻이다. 또다른 특징으로 TFHE는 boolean 단위로 암호화할 수 있다. 이를 활용한 bit-wise 암호화를 통해 정확한 연산이 가능하다.

2.2 동형암호 기반 CNN 연산 과정

원격 클라우드에서 CNN에 동형암호를 적용하여 연산하는 것은 다음과 같이 이루어질 수 있다. 우선, 사용자가 자신의 프라이버시에 해당하는 입력 이미지를 암호화하여 클라우드로 전송한다. 클라우드는 전송받은 암호문을 활용해서 CNN연산을 수행한다. 이 때 CKKS가 쓰였다면 비산술연산은 산술 근사식으로 연산한다[1]. 또한 convolution의 경우, weight 및 bias 값에 해당하는 모델 파라미터는 클라우드 소유이므로, 암호화하지 않은 평문으로 사용한다. 즉, convolution은 암호문과 평문간의 연산으로 이루어지는데, 이 연산은 암호문간 연산보다 빠르기도 하며, 상호 프라이버시가 보존되므로 효율적이다.

2.3 CNN의 양자화 (Quantization)

CNN은 대표적인 신경망 중 하나이며, 수없이 많은 연산들로 구성되어 있다. 동형암호를 적용하지 않은 평문 CNN이라도 이런 많은 연산수로 인한 부하가 크며, 이를 줄이기 위한 양자화 기술들이 등장했다. CNN 양자화에는 연산 일부 생략, 혹은 데이터 타입 (data-type) 간소화 등 다양한 방법들이 존재한다. 그 중, **데이터 타입 양자화 (이후 DQ라고 칭함)**는 각 연산 수행시간을 단축시켜주는 효과가 있다. 주로 convolution과 같이 곱셈 연산에 적용하는데, 예를 들어 32-bit를 8-bit로 DQ하면, 이론상 곱셈을 16배 빨리 수행할 수 있다.

물론, 0.7을 0.5로 양자화하면 0.2만큼의 오차가 생기듯 DQ도 연산 오차를 일으킨다. 하지만 이로 인한 CNN 추론 정확도 하락은 그리 크지 않으며, layer 별 적용 및 비연속적 quantization을 통해 더욱이 정확도 하락을 방지할 수 있다고 한다. 또한 DQ는 training 과정 혹은 그 후에 이루어지는데, 본 연구는 training 후에 이루어지는 post-training, 즉 추가 학습 과정이 필요없는 DQ를 타겟으로 삼는다.

3. 실험적 관찰 및 기존 연구 소개

3.1 평문 DQ 효율성과 동형암호 CNN 연산 분석

평문 CNN에서는 ReLU와 Max 연산은 단순 비교 연산 하나로 구성된다. 이는 컴퓨터 입장에서 덧셈 및 곱셈에 비해 훨씬 빠르게 수행할 수 있다[5]. 따라서, 평문 CNN에 DQ를 사용할 경우 위에서 소개한 대로 주로 곱셈에 적용한다[4]. 하지만 동형암호의 경우, 연산간 상대적 성능이 뒤바뀐다. <표2>는 32-bit data를 CKKS와 TFHE의 단일 연산 성능 분석이다. CKKS는 HEaaN라이브러리를 활용했으며, TFHE는 TFHE-rs 라이브러리를 사용했다. 32-bit 데이터를 온전히 암호화 하기 위해CKKS의 경우 $N=2^{17}$, TFHE는 32개의 $N=2^{10}$를 사용했다. ReLU 및 Max는 두 체계 모두 CKKS 곱셈 대비 4배 이상 느린 것을 확인할 수 있다. 물론, 한 암호문에 여러 데이터를 한번에 암호화하는 packing 기술 등 최적화를 적용하여 신경망 연산할 수 있으나, 그 결과도 크게 다르지 않다. <표3>는 VGG-16 신경망 연산을 분석한 결과이다. CKKS만 사용하는 경우 6.11% 정확도 하락과 함께 2,982초가 필요한 반면, TFHE를 혼용해서 사용하는 최신 연구기법을 적용한 경우 정확도 하락 없이 31,000초가 필요하다. 결과적으로, 정확한 비산술 연산을 위해 TFHE를 추가로 사용하는 것은 정확도 하락을 방지할 수 있지만, 약 10배 느리다는 것을 알 수 있다.

32-bit	CKKS	TFHE
곱셈	3 ms	-
ReLU	33ms	13 ms
Max	30ms	

표 2 : 동형암호 체계별, data 크기별 단일 연산 속도 분석. CKKS의 ReLU와 Max는 최신 논문 근사식을 활용[1]

	정확도 (%)	시간 (초)
CKKS	92.64 → 87.53	2,982
TFHE	92.64 → 92.64	31,000
SHE[5]	92.64 → 92.11	4,192

표 3 : VGG-16 모델을 활용한 CIFAR-10 추론 성능

3.2 기존 TFHE 신경망 DQ 연구

<표3>의 SHE[5]는 TFHE-CNN의 산술연산에 사용되는 모델 파라미터에 DQ를 적용한 기존 연구 사례이다. 구체적으로, 모델 파라미터를 log2 scale로 DQ하는데, 5는 $4(2^2)$로, 0.3은 $0.25(2^{-2})$로 DQ하는 것을 의미한다. 이런 값들에 대한 곱은, 단순 bit-shift에 해당하기 때문에, bit-wise 암호화한 TFHE-NN에서는 아무런 cost가 발생하지 않는다. 그 결과, <표3>의 4,192초에 해당하는 시간은 대부분 비산술연산에 해당하는 시간이라 할 수 있다. 정

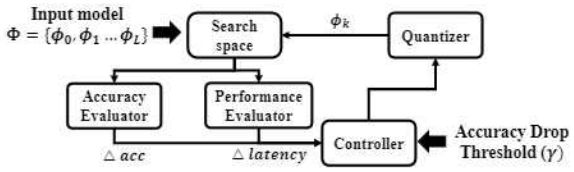

그림 1 비산술함수 양자화를 위한 QNAS 구조

확도 하락이 0.53%로 매우 적지만 CKKS와 비교하면, 아직 속도는 아직 약 1.4배 느린것을 확인할 수 있다. 이런 SHE의 결과에서 시작하여, 남은 비산술연산들을 DQ하면 CKKS보다 낮은 정확도 하락과 빠른 성능을 동시에 이룰 수 있을 것이라 기대된다.

4. QNAS 기반 TFHE 비산술연산 DQ 기술

4.1 비산술연산 DQ의 오차 분석

앞서 언급했듯, 평문 CNN에서는 큰 의미가 적을 수 있으나 TFHE로는 큰 부하가 발생하는 비산술연산들에 대해 DQ를 적용하는 것에 대해, 산술연산에 적용하는 DQ보다 적은 연산 오차를 발생시킴을 확인할 수 있다. 우선, 직관적인 비교로써, ReLU 및 MAX는 모두 두 수 A와 B에 대해 어떤 수가 더 큰지 비교하는 연산으로 구성되어 있다. 이 때 데이터 정밀도 차이가 발생하더라도, A보다 작던 B가 B와 같아질 수는 있어도 B보다 커질 수가 없기에, DQ가 적용되어도 연산 결과에 차이가 발생하지 않는다.

보다 구체적으로, A와 B의 DQ로 인해 오차 e_1와 e_2가 발생하면, 곱셈 결과 $AB + Ae_2 + Be_1 + e_1e_2$가 나온다. 한편, 한 수에 대한 ReLU 및 MAX 연산의 결과는 오차가 e_1 혹은 e_2만 발생한다. 즉 산술연산보다 비산술연산의 DQ가 적은 오차를 발생시킨다.

4.2 Quantization Neural Architecture Search

위와 같은 관찰, 실험 및 분석을 바탕으로 우리는 다음과 같은 TFHE 비산술 연산 DQ 기술을 고안했다. [그림 1]은 본 연구가 제안하는 비산술함수 양자화를 수행하기 위해 구현한 Quantization Neural Architecture Search (QNAS) 구조다. 우선, 모델이 입력된 후, 비산술함수를 포함한 layer들을 search space로 하여, 데이터의 각 비트를 양자화하며 성능과 정확도에 대한 영향을 측정한다. 이 때, 정확도 측정의 dataset의 validation set 내에서 1000장을 임의 추출하여 측정한다. 만약 사용자가 설정한 정확도 하락 한계를 넘는 경우 해당 비트는 양자화하지 않으며 다음 비트를 양자화하도록 구현한다. 이 때 양자화는 최하위 비트부터 시작한다. 양자화하는 비트수를 최대화 하는 것을 목적으로 진행하는 이 NAS를 통해 최종적으로 우리는 TFHE-NN의 비산술연산들을 DQ하는데 있어 설정한 최대 정확도 하락 상한 내 최대한의 성능 향상을 누릴 수 있다.

5. 실험 결과

제안한 기술의 성능을 점검하기 위해 Intel Xeon-Gold 6326와 DRAM 1TB가 탑재된 서버에서 TFHE-rs 라이브러리를 활용해서 성능을 측정했다.

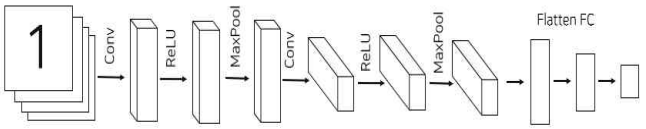

그림 2 ReLU를 활용한 LeNet-5 모델 구조

	양자화된 bit 수
ReLU 1	32 → 16 ~ 24
MaxPool 1	32 → 18 ~ 25
ReLU 2	32 → 12 ~16
MaxPool 2	32 → 11 ~ 18
ReLU 3	32 → 6 ~ 11
ReLU 4	32 → 6 ~ 9

표 4 : QNAS를 통한 LeNet-5 비활성화 함수 양자화 결과

모델들은 Keras의 pre-trained 모델들을 사용했으며, 정확도 측정은 testset의 1000장을 통해 산출 했다.

우선, QNAS의 Quantization 결과 예시를 확인해 보자. <표4>는 [그림 2]의 ReLU를 활용한 LeNet-5 모델로 MNIST dataset 추론하는 과정에 대해 측정한 결과에 대한 분석 결과이다. 최초 LeNet-5는 Sigmoid를 사용하지만, Keras의 pre-designed LeNet-5는 ReLU를 사용하는 버전을 사용하며, 이것이 더 정확도가 높다고 한다. 결과를 통해 우리는 QNAS가 최초 32-bit 데이터를 최소 6-bit, 최대 25-bit 형태로 DQ하는 것을 확인할 수 있다. 대체적으로 입력단에 가까운 layer일수록 정확도에 큰 영향을 미치기 때문에 많은 bit 수를 남기고, 출력단에 가까울수록 과감한 DQ가 되는 것으로 추측된다.

단위 : %	평문	CKKS	TFHE	SHE	Ours
LeNet-5	98.9%	98.2%	98.9%	98.5%	98.4%
VGG-16	92.6%	87.5%	92.6%	92.1%	91.9%
ShuffleNet	71.4%	60.2%	71.4%	69.4%	67.7%

표 5 : CNN 추론 정확도 결과 (%)

단위 : 초	CKKS	TFHE	SHE	Ours
LeNet-5	10	89	9	4
VGG-16	2,982	31,000	4,192	2,099
ShuffleNet	14,694	126,000	18,000	10,421

표 6 : CNN 추론 성능 결과 (seconds)

여러 모델들에 대한 성능 및 정확도는 <표5>와 <표6>에서 확인할 수 있다. LeNet-5는 MNIST,

VGG-16은 CIFAR-10, ShuffleNet은 ImageNet dataset을 사용했고, 각각 QNAS를 사용하는데 있어 허용 정확도하락 상한 Y를 0.5%, 2%, 그리고 5%로 두고 측정하였다. 'Ours'는 SHE에 우리의 비산술연산 DQ기술을 추가한 결과다. 실험 결과 Ours는 CKKS보다 항상 높은 정확도를 유지하며 성능도 더 뛰어났다. SHE보다 약간 정확도가 낮았지만 성능이 약 2배 더 좋다. 우리 목표대로, CKKS보다 빠르면서도 높은 정확도를 이루었으며, 정확도 하락 정도가 SHE와 크게 차이나지 않는다는 점에서, 우리의 비산술연산 DQ기술이 효율적임을 확인할 수 있다.

6. 토론 및 고찰

본 연구를 통해 구현된 QNAS와 TFHE 비산술연산의 양자화가 성능 및 정확도 모두의 측면에서 CKKS보다 좋다는 점을 확인할 수 있었다. 하지만, 논의가 필요한 하는 점들이 있다.

첫째로, 작은 데이터셋인 MNIST의 경우 고차다항식을 활용한 근사를 사용할 필요가 없다. LoLa[7]의 경우 ReLU를 x^2의 단순한 식으로 근사해서 CKKS로 LeNet-5를 적은 정확도 하락과 함께 2.2초 안에 연산한다. 즉, 데이터셋이 작은 경우는 CKKS가 TFHE를 쓰는것보다 좋다고 할 수 있다. 하지만 실생활에서 쓰이는 모델들과 데이터셋은 ImageNet과 같이 큰 경우가 대부분이며, 이럴 경우는 LoLa와 같은 접근은 모델을 무용지물 수준으로 만드는 정확도 하락을 일으키기 때문에, 다른 실험결과들과 같이 우리 방법을 적용하는 것이 더 효과적이다.

다음으로, 근사식을 적용하여 adaptive training을 추가로 수행하는 방법들이 있는데, 이 경우 추가 학습이 필요하다. 하지만 QNAS의 수행시간은 4시간 이내인 반면 추가 학습은 GPU를 사용하여 며칠씩 걸리기 때문에 우리 접근이 더 효율적이라 할 수 있다. 또한 기존 모델보다 좋은 정확도를 보장하지 않는다는 점에서 큰 한계점이 있다. 분명 가치있는 연구 방향이지만, 우리 연구방향과 직접적으로 비교하기는 어려움이 있는 것으로 판단된다.

7. 결론

본 연구는 TFHE CNN의 비산술연산을 양자화하는 방법에 대해 다루었다. 기존에 알려진 바로는 TFHE는 CKKS보다 정확도 하락은 적지만 성능이 느렸다. 하지만 우리 연구를 추가로 적용한 결과 CKKS보다 최대 2.5배 빠르면서도 높은 정확도를 얻을 수 있었다. 본 연구에 이어 여러 체계를 섞어서 사용하는 등 추가적인 연구방향들이 가능할 것으로 보이기에 더 많은 연구들이 촉진되었으면 한다.

6. ACKNOWLEDGEMENT

이 논문은 연구 수행에 있어 2024년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원과 (RS-2023-00277326) 정보통신기획평가원의 지원을 받았으며 (No.2021-0-00528, 하드웨어 중심 신뢰계산기반과 분산 데이터보호박스를 위한 표준 프로토콜 개발), 2024년도 BK21 FOUR 정보기술 미래인재 교육연구단, 반도체 공동연구소 지원의 결과물이다. 또한, 연구장비를 지원하고 공간을 제공한 서울대학교 컴퓨터연구소에 감사드린다.

참고문헌

[1] Lee, Eunsang, et al. "Low-complexity deep convolutional neural networks on fully homomorphic encryption using multiplexed parallel convolutions." International Conference on Machine Learning. PMLR, 2022.

[2] CHEON, Jung Hee, et.al. "Homomorphic encryption for arithmetic of approximate numbers", ASIACRYPT 2017, Hong Kong, China, December 3-7, 2017, p. 409-437.

[3] CHILLOTTI, et.al. "TFHE: fast fully homomorphic encryption over the torus", Journal of Cryptology, 2020, 33.1: 34-91.

[4] Ekman, Magnus. Learning deep learning: Theory and practice of neural networks, computer vision, natural language processing, and transformers using TensorFlow. Addison-Wesley Professional, 2021.

[5] Lou, Qian, and Lei Jiang. "She: A fast and accurate deep neural network for encrypted data." Advances in neural information processing systems 32 (2019).

[6] Nagel, Markus, et al. "A white paper on neural network quantization." arXiv preprint arXiv:2106.08295 (2021).

[7] Brutzkus, Alon, Ran Gilad-Bachrach, and Oren Elisha. "Low latency privacy preserving inference." International Conference on Machine Learning. PMLR, 2019.