스케줄링을 통한 동형암호 딥러닝 모델 가속

하회리 [1], 이동주 [1], 백윤흥 [2]
[1] 서울대학교 전기정보공학부 박사과정
[2] 서울대학교 전기정보공학부 교수

{Wrha, djlee} @sor.snu.ac.kr, ypaek@snu.ac.kr

Accelerating Homomorphically Encrypted Deep Learning Model with Scheduling

Whoi Ree Ha[1], Dongju Lee[1], Yunheung Paek[1]
[1]Dept. of Electrical and Computer Engineering and Inter-university Semiconductor Research Center, Seoul National University

요 약

동형암호는 MLaaS (Machine Learning as a Service)가 만연한 이 시대에 각광받고 있는 프라이버시 보호 기술 중 하나이다. 하지만 동형암호를 적용하게되면 데이터 크기가 굉장히 커지고, 비싼 연산으로 인하여 큰 overhead 가 발생한다. 따라서 더 효율적인 스케줄링을 통하여 이 overhead 를 최소화 하였으며, 실험결과 총 latency 의 18%를 감소할 수 있었다.

1. Introduction

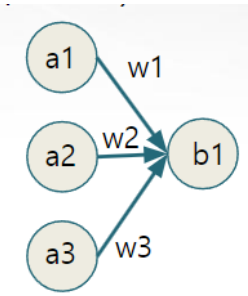
MLaaS (Machine Learning as a Service)이 현실에서 많이 사용됨에 따라 사용자의 원본 데이터가 service-provider 의 클라우드로 전송되고 있다. 이는 심각한 프라이버시 문제를 초래한다. 따라서 대표적인 프라이버시 보호 기술 중 하나로 동형암호가 각광받고 있다. 동형암호는 암호화된 상태에서 연산할 수 있는 암호이며 따라서 동형암호가 적용된 모델에는 사용자의 원본 데이터가 아닌 암호화된 데이터를 사용할 수 있다.[1] 그러므로 사용자의 원본 데이터가 3 자의 클라우드로 전송될 필요가 없어지고, 암호화된 데이터만이 클라우드로 전송된다.

그러나 동형암호는 보안성을 유지하면서 암호화된 상태에서 연산을 하기 위해 데이터 크기가 매우 커지며, 비싼 알고리즘을 수행해야 된다. 실제로 동형암호가 적용된 모델들을 보면 데이터 크기는 많게는 10,000 배 커지고[2], 모델의 수행시간은 1,000 배 이상 느려 진다. [3]

이 연구에서는 동형암호가 적용된 모델을 가속하기 위해 동형암호 연산 중 많은 수행 시간을 차지하는 rescale 연산을 최소화하여 모델을 가속한다. 이 때, rescale 연산을 무작정 최소화 하는 것이 아닌 최소화 하는 과정에서 발생할 수 있는 overhead 도 감안하여 스케줄링을 하였다. 실험결과 스케줄링 최적화를 통하여 총 18%의 전체 수행 시간을 줄일 수 있었다.

2. Motivation

Deep Learning(DL)의 특징은 곱셈 뒤에 덧셈, 혹은 합 연산이 뒤 따른다는 것이다.



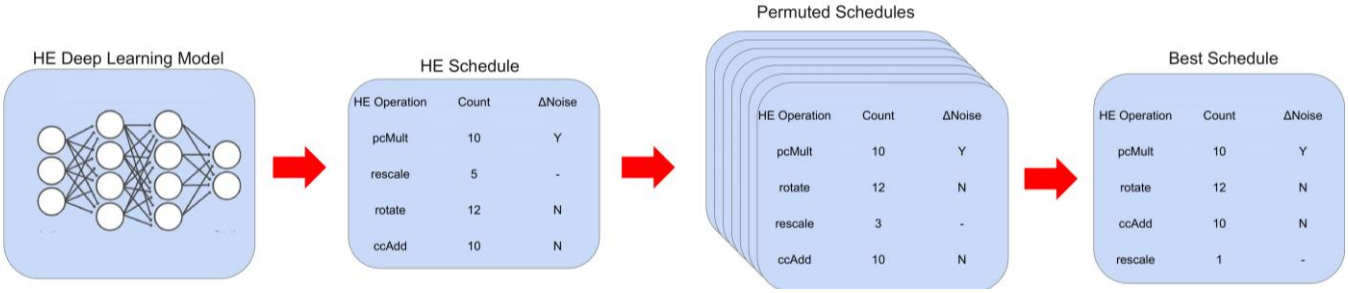
<그림 1> 모델 예제

<그림 1>과 같은 모델의 일부분을 연산할 때, a1, a2, a3 는 각 각 w1, w2, w3 와 곱해지며 각 곱해진 값의 합을 최종 결과값으로 가진다. 이 특성은 Fully Connected (FC), Recurrent Neural Network (RNN), Convolution Neural Network (CNN), Transformer 가리지 않고 나타나는 특징이며, 대부분 하나의 레이어에서 element-wise 곱셈 후 각 element 들을 합하게 된다.

동형암호가 적용된 상태에서는 암호의 동형성과 결과의 정확도를 유지하기 위해서 곱셈 후 relinearize 또는 rescale 이라는 연산을 추가로 수행한 후 합 연산을 수행한다. 동형암호의 기본

형태는 polynomial 이기 때문에 곱셈을 하게 되면 1 차식이 2 차식이 된다. 따라서 이 2 차식을 다시 1 차식으로 변형해 주는 것이 필요하면 이를 relinearze 연산이라고 정의한다. Rescale 연산은 암호문의 곱셈을 하였을 때, 암호문의 noise 가 곱해져서 커지는 것을 최소화 하기 위해 도입된 연산이다. 곱셈 후의 암호문의 noise 를 최소화하지 않으면, 다음 곱셈 시 noise 가 기하급수적으로 증가하기 때문에[4], 동형암호 연산의 한계에 더 급히 도달한다. 따라서 이 두 연산은 동형암호 scheme 에 필수적이지만, 굉장히 비싼 연산이며 많게는 전체 수행시간의 70% 이상을 차지한다. [5] 그 중 rescale 연산은 이론적으로 다른 연산을 수행하고 난 뒤에 진행해도 문제가 없다. 특히 noise 를 증가시키지 않는 동형암호 연산들이 존재하며 이를 특정한 후, rescale 연산을 뒤로 미루어 rescale 이 최소화될 때 연산할 수 있다면 동형암호화된 딥러닝 모델의 연산을 가속할 수 있다.

3. Design



<그림 2> overview

<그림 2>는 이 연구의 전체적인 overview 를 도식화 하였다. 첫 번째로 동형암호가 적용된 딥러닝 모델을 정적 분석하여, 이 모델의 HE schedule 을 뽑아낸다. 이 때, 각 HE operation 이 noise 를 바꾸는지를 annotate 한다. 이를 통하여 rescale 연산이 특정 연산 뒤로 미루어 질 수 있는 지를 확인할 수 있다. 또한 각 연산이 몇 번 반복 수행되는지도 annotate 한다. 같은 AI 모델이여도 어떤 암호 scheme 을 사용하는지, 또는 어떤 packing method 를 사용하는지에 따라서 각 연산의 수가 달라지기 때문에, 일괄적으로 처리할 수 없고, 매번 정적 분석을 통하여 각 연산의 수행 수를 확인하여야 된다. 이 연산수는 굉장히 중요한데, rescale 연산을 하면 동형암호 데이터의 크기가 어느 정도 줄어든다. 따라서 rescale 전의 연산은 rescale 후의 연산보다 느리다. 물론 rescale 은 비싼 알고리즘이기 때문에 자체적으로 수행 시간을 많이 차지하지만, rescale 을 한 번 줄이는 것 보다 1,000 번 반복되는 특정 연산을 rescale 후에 처리하는 것이 전체적으로는 더 빠를 수 있다.

그 후, 이 정보들을 가지고 rescale 을 뒤로 미룰 수 있는 곳을 파악하여 permuted schedules 를 생성한다. 이 때, 기본적으로 rescale 을 제외한 다른 연산들의 수행 수는 변하지 않는다. 하지만 rescale 의 수는 암호문의 숫자에 따라서 변하는데, 이는 마찬가지로 모델, 암호문의 parameter 등이 영향이 미치기 때문에 매 application 마다 다르다. 하지만 rescale 의 수는 암호문의 수와 같기 때문에 특정 연산 뒤로 미루어 졌을 때의 rescale 의 수를 쉽게 구할 수 있다.

또한 단순하게 모든 경우의 수에 따라 permutation 을 하게 되면, 모델에 따라서 어떤 경우는 너무 많은 permutation 이 생성될 수 있고, 기하급수적으로 증가할 수 있기 때문에 scalable 하지 못하다. 따라서 이 연구에서는 rescale 과 이후 연산이 바뀌었을 때, rescale 의 증감 그리고 바뀐 연산의 count 를 고려하여 rescale 의 수가 오히려 더 많아지거나, count 가 rescale 의 감소량에 비해 너무 큰 경우를 제외한다. 이 때, 각 연산의 중요도를 설정하여 threshold 를 정한다. 이 중요도는 각 동형암호 연산의 latency 에 비례한다. 이를 통하여 실험적으로 총 latency 를 구해보기 전에 비효율적인 schedule 을 filter 할 수 있다.

마지막으로 permutated 된 schedule 을 각 각 구현하여 실험적으로 수행 시간을 찾고, 그 중 제일 좋은 경우를 택한다.

4. Experiments

이번 실험에서는 Microsoft 의 SEAL [6]을 사용하여 동형암호 딥러닝 모델을 구현하였으며, scheme 은 CKKS 를 사용하였다. 모델은 LoLa[2] 에서 제시한 모델 중 MNIST 모델에 대하여 실험을 진행하였다.

모델	Latency(us)
LoLa-MNIST	1,038,594
LoLa-MNIST-opt	854,952
Lola-MNIST-min	1,108,184

<표 1> 실험결과

<표 1>은 실험결과는 나타낸다. LoLa-MNIST 는 baseline 으로 기본적인 동형암호 scheme 을 사용한 모

델이며 총 1,038,594 us 의 수행시간이 걸렸다. LoLa-MNIST-opt 는 이 연구에서 제시한 최적화 기법을 사용한 결과 값이다. 동형암호 연산의 스케줄을 수정하여 총 latency 를 18% 감소시킬 수 있었다. LoLa-MNIST-min 은 rescale 에만 중점을 두어 rescale 의 수를 최소화한 스케줄을 적용하였을 때 결과이다. Rescale 의 수는 기본 모델 대비 144 번 감소하였고, LoLa-MNIST-opt 보다도 12 번 더 감소하였다. 하지만 design 장에서 서술하였듯, rescale 은 전체 암호문의 크기를 줄여주기 때문에, rescale 후의 연산들이 조금 빨라진다. Rescale 을 뒤로 미루어 rescale 이 감소하는 것만 신경 쓸 경우, 더 큰 암호문에 대해서 연산을 수행해야 되고, 특정 경우에는 rescale 의 수를 줄여 얻는 이득보다 더 큰 overhead 가 발생할 수 있다.

5. Conclusion

이 연구에서는 MLaaS 에서 발생할 수 있는 프라이버시 문제를 해결하기 위해 사용할 수 있는 동형암호의 overhead 를 최소화하려고 하였다. 동형암호가 적용된 딥러닝 모델은 데이터 크기가 커지고 비싼 알고리즘을 수행해야되기 때문에 매우 큰 overhead 가 발생한다. 특히 매 번 곱셈마다 수행해야되는 rescale 의 경우 이 overhead 의 많은 부분을 차지하고 있으며 이 rescale 을 최소화하는 방향으로 스케줄을 최적화 하였다. 하지만 단순히 rescale 의 수만 줄일 경우, application 마다 다르겠지만, 오히려 overhead 가 증가할 수 있기 때문에, 이를 고려하여 최적화를 수행ㅎ였다. 실험결과 단순히 스케줄을 바꾸어 총 18% 의 latency 를 감소시켰다.

6. Acknowledgement

이 논문은 2024 년도 BK21 FOUR 정보기술 미래인재 교육연구단, 반도체 공동연구소, 정부(과학기술정보통신부)의 재원으로 한국연구재단(RS-2023-00277326), 23 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원(No.2021-0-00528, 하드웨어 중심 신뢰계산 기반과 분산 데이터보호박스를 위한 표준 프로토콜 개발), 2023 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원(No.RS-2023-00277060, 개방형 엣지 AI 반도체 설계 및 SW 플랫폼 기술개발), 2024 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원(IITP-2023-RS-2023-00256081), 2024 년도 정부(산업통상자원부)의 재원으로 한국산업기술기획평가원 (No. RS-2024-00406121, 자동차보안취약점기반위협분석시스템개발(R&D)) 에 의하여 지원되었음.

참고문헌

[1] Albrecht, M., Chase, M., Chen, H., Ding, J., Goldwasser, S., Gorbunov, S., Hoffstein, J., Lauter, K., Lokam, S., Micciancio, D., et al. Homomorphic encryption standard. 2018.

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

[3] Gilad-Bachrach, Ran, et al. "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy." International conference on machine learning. PMLR, 2016.

[4] J. H. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, “A full rns
variant of approximate homomorphic encryption,” in Selected Areas in
Cryptography – SAC 2018, C. Cid and M. J. Jacobson Jr., Eds. Cham:
Springer International Publishing, 2018, pp. 347–368

[5] Samardzic, Nikola, et al. "F1: A fast and programmable accelerator for fully homomorphic encryption." MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture. 2021.

[6] Chen, Hao, Kim Laine, and Rachel Player. "Simple encrypted arithmetic library-SEAL v2. 1." Financial Cryptography and Data Security: FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA, Sliema, Malta, April 7, 2017, Revised Selected Papers 21. Springer International Publishing, 2017.