실제 클라우드 환경에서의 프라이버시 보존형 AI 성능 분석

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

Revealing The Performance of Privacy-Preserving AI on a Real Cloud Environment

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

요 약

클라우드 기반 AI서비스들은 사용자의 프라이버시 유출 위험이 있다. 이에 다양한 기술들을 활용한 프라이버시 보존형 AI 서비스 기술들이 등장했다. 하지만 기존 연구 결과들을 살펴보면, 실제 클라우드 서비스 환경과는 다소 차이가 있다. 본 연구는 이런 비현실적인 가정들을 파악하고 실제 클라우드 서비스 환경에서 여러 프라이버시 보존형 AI 서비스들의 성능을 비교분석한 결과를 제시한다.

1. 서론

4차 산업혁명의 핵심 요소인 AI는 빠르게 발전해나가고 있으며, 대규모 AI의 순기능을 활용할 수 있는 클라우드 기반 AI서비스들이 우리의 일상생활에 자리잡고 있다. 하지만 음성인식, 사진변환 등 편리한 기능들을 제공하는 AI서비스들을 사용함에 있어 수없이 많은 사용자들의 개인정보가 클라우드에 노출되기에, 개인정보 보호, 즉 프라이버시에 대한 우려들이 전세계적으로 주목받기 시작했다. 이런 우려에 대응하기 위해 동형암호(Homomorphic Encryption, HE), 다자간연산(Secure Multi-Party Computation, SMPC), 차분 프라이버시 등 다양한 프라이버시 보호 기술들이 등장했다.

그 중, 많은 연구자들은 HE와 MPC를 혼용하는 프로토콜들[1-3]을 개발하는데 노력하였다. 그들의 연구결과에 따르면 이 접근법은 정확도를 떨어뜨리지 않으면서 매우 빠른 성능을 보여준다. Delphi[1]는 HE로 1,000초 이상 걸리는 ResNet-32 신경망 연산을 10초 내에 수행하며, 평문과 동일한 정확도를 나타낸다고 한다.

하지만 이들의 연구 결과는 다소 비현실적인 실험환경을 전제로 한다는 점에서 한계가 있다. 예를 들어, Delphi에서 사용자와 클라우드 모두의 환경이 고성능 서버급으로 구성한다. 본 논문은 해당 예제뿐 아니라 기존 연구들이 전제로 하는 비현실적인 실험환경들에 대해 서술하고, 그들이 이런 선택을 하게 된 이유에 대해 분석할 것이다. 또한 사용자와 클라우드의 성능, 네트워크 대역폭 등 현실적인 실험환경에서 이들의 성능이 어떻게 나오는지에 대한 분석 결과를 보여주고자 한다.

	전처리	온라인
동형암호 (HE)	거의 없음	매우 큼
가블드 서킷 (GC)	많이 가능	매우 작음
비밀 분할	많이 가능	매우 작음

표 1 : 여러 SMPC의 전처리 가능성과 온라인 연산량

2. 이론적 배경

2.1 다자간연산 (SMPC)

SMPC는 당사자들이 입력을 비공개로 유지하면서 입력에 대한 함수를 공동으로 계산할 수 있는 프로토콜들을 의미한다. 동형암호를 이용하여 한명의 데이터를 암호화한 상태로 다른 사람이 연산할 수 있다는 점에서 동형암호 역시 SMPC의 일종이라고 할 수 있다. 이 외에도 로직 게이트들을 엮어 만든 회로를 연산할 수 있는 가블드 서킷 (Garbled Circuit, GC), 값에 노이즈를 섞어 상호 값을 숨기는 덧셈형 비밀 분할 등 다양한 프로토콜들이 존재한다.

이들의 연산은 전처리와 온라인으로 나눌 수 있다. 전처리는 사용자가 입력데이터를 제공하지 않은 상태에서, 입력값과 상관없는 연산들을 양쪽이 미리 수행할 수 있는 것들을 처리하는 것을 의미한다. 예

를 들어, GC의 경우 회로의 정보를 암호화하여 저장하는 Garbled Table을 생성해야 하는데, 이는 입력값과 무관하게 회로 정보만 있다면 수행할 수 있는 과정이다. 즉, 서비스를 사용하지 않는 idle 시간에 미리 많이 수행해놓는 만큼, 실제 사용자가 서비스를 요청할 때 (입력을 넣을 때)부터의 온라인 시간이 줄어든다는 장점이 있으며, [표 1]과 같이 기술별로 그 분포가 상이하다. 또한, 산술연산만 표현가능한 HE와 비밀분할과 다르게, GC는 비산술연산들을 로직회로로 표현하여 연산할 수 있어 근사할 필요 없이 정확한 연산을 수행하여 AI서비스의 정확도 하락을 유발하지 않는다는 장점을 갖고 있다. 하지만, 로직회로를 구현하기 위해 비트단위로 연산을 표현해야하는 만큼, 되려 산술연산에 있어 큰 연산부하를 유발하기도 한다.

2.2 SMPC기반 AI서비스 기술과 실험 환경

Delphi는 신경망의 산술연산은 HE와 비밀분할로, ReLU와 같은 비산술연산은 GC로 수행하며, 대부분의 연산을 전처리로 수행함으로써, 온라인 시간을 최소화하는 접근을 택함으로써 HE 기반 AI서비스 대비 약 100배 빠른 성능을 달성했다. 하지만 Delphi의 실험환경은, 앞서 서술했듯이 사용자가 서버급 성능을 지니고 있다는 다소 비현실적인 설정을 전제로 한다. Delphi 프로토콜은 적극적인 전처리를 통한 이점을 극대화하는 것에 집중되어 있기에, 사용자단에 매우 큰 스토리지가 있다는 전제를 할 수 밖에 없었다. 하지만 신경망 연산 1회당 사용자단에 약 40GB[2]의 전처리된 데이터가 필요할때, 일반적으로 사용자들이 사용하는 휴대폰, 스마트 기기 등의 용량 (e.g., 128GB)을 고려하면 모든 메모리를 AI서비스를 위해 사용하더라도, 예를 들어, Chat GPT와 대화를 4번만 빠르게 할 수 있고, 그 후에는 전처리된 데이터가 없기에, 온라인 시간에 전처리도 수행하게 되어 동형암호만큼 느려진다.

이러한 점을 고려해서 Cryptonite은 온라인 시간을 다소 늘리더라도, 사용자단에 저장해야하는 전처리 데이터 용량을 최소화할 수 있도록 GC 프로토콜을 수정하는 접근을 제안했다. 그들은 사용자가 메모리 용량이 제한적인 태블릿 환경에서 AI서비스를 연속적으로 사용한다 가정하였으며, 실험 결과 동일 환경에서의 Delphi보다 평균적으로 빠른 성능을 달성했다 (최초 query는 전처리를 한 Delphi가 빨랐지만, 연속적인 AI서비스 활용으로 인해 전처리한 데이터가 모두 소진되어 온라인 시간이 늘어나기 때문이다).

하지만 우리의 분석 결과, Cryptonite 역시 업로드/다운로드 네트워크 대역폭을 임의로 조절해가며 사용할 수 있다는 다소 비현실적인 가정을 하는 등 문제가 있음을 발견했다. Delphi 보다 많은 통신량이 발생하기에 업로드/다운로드 할때마다 해당 대역폭을 극대화하는 접근을 선택한 것이다. 하지만 일반적으로 클라우드 서비스들 업체들은 업로드 대비 다운로드 대역폭을 크게 할당한다[5-6]. 이는 일반적으로 스트리밍 서비스와 같이 사용자들이 클라우드 내 정보를 다운받는 서비스들이 주로 사용되기 때문이다. 또한 업로드를 통해 외부 데이터가 클라우드로 진입하는 것은 보안 등 이유로 보다 복잡한 과정을 통해 제한적으로 이루어져야 하기 때문이라는 이유도 있다. 이런 대역폭할당은 서비스 업체에서 사용자마다, 그리고 서비스마다 다르게 실시간으로 조절해줄 수 없는 하드웨어 설정이며, 일반적으로 총 1Gbps의 대역폭이 있다면 800Mbps를 다운로드에, 200Mbps를 업로드에 할당한다고 한다. 이런 점에서 그들의 네트워크 설정은 비현실적이라 할 수 있다.

2.3 현실적인 (실제) 클라우드 환경

우리는 보다 현실적인 사용자-클라우드 환경에서의 SMPC기반 AI서비스 성능을 측정하기 위해 양측 환경들에 대한 현실적이 기준들을 살펴본 결과, 다음과 같이 정리해보았다. 앞서 언급한 메모리 용량, 그리고 네트워크 대역폭 외에도 다음과 같은 사항들을 고려해보았다.

첫째, 사용자와 클라우드 환경의 연산 성능이 있다. 클라우드에는 많은 쓰레드와 캐시를 지닌 CPU와 강력한 GPU 기반 연산을 수행하는 것이 일반적인 한편, 사용자단은 상대적으로 약한 프로세서를 활용하는 것이 일반적이다. 이에 우리는 스마트폰 성능의 사용자단과 고성능 서버급 성능의 클라우드 환경을 전제로 실험을 수행하고자 한다.

둘째, 혼잡도 등 다양한 네트워크 성질을 생각해볼 수 있다. Delphi와 Cryptonite는 통신 시간 측정을 직접하는 대신, 대역폭과 Round Trip Time (RTT)를 설정하면 수식을 활용한 간단한 시뮬레이터를 구현하여 사용하였다. 하지만 실제 통신은 훨씬 다양한 요인들의 영향을 받는다. ChatGPT와 같이 해외 클라우드를 기반으로 하는 서비스의 경우 국내/외 여러 노드들을 지나며 네트워크 패킷 손실률이

<1.5%정도 발생하며[4], 많은 사용자로 인한 클라우드 외로 나가는 통신에 대한 요청량이 늘어나면 각 노드에서 대기열이 발생하고[5], 이는 handshake RTT 지연시간의 27%까지도 늘어난다[6].

	사용자 (C)	클라우드 (AWS)	
		Weak	Strong
기기	Galaxy S23	c5.9xlarge	p4d.24xlarge
CPU	8 core	36vCPU	96vCPU
GPU	Adreno 740	없음	NVIDIA A100
RAM	8GB	72GB	1,152GB
Storage	128GB	20TB	20TB

표 2 : 사용자 / 클라우드 실험 환경 개요

3. 현실적인 환경 속 SMPC 수행 시간 측정 결과

3.1 우리의 실험 환경

우리는 [표 2]와 같이 실제 AWS 클라우드 컴퓨팅[4] 환경을 활용한 실험 환경을 구성했다. 사용자는 Galaxy S23 휴대폰으로 CPU뿐 아니라 모바일용 GPU까지 지닌 Snapdragon 8 Gen2라는 고성능 휴대폰 칩셋을 사용한다. 클라우드 환경으로는 두가지를 활용했는데, Weak의 경우 GPU가 없고, 36vCPU 정도로 사용자단과 비슷한 컴퓨팅 파워를 지닌 환경이다. 이 환경은 Delphi가 가정했듯이 사용자와 클라우드가 비슷한 성능의 환경일 경우 결과를 보여주기 위해 설정한 것이다. 물론, 이 경우에도 storage의 경우 클라우드가 훨씬 크다. Strong의 경우 NVIDIA A100 GPU를 갖춘 환경으로 클라우드가 컴퓨팅 성능이 더 강한 현실적인 환경을 가정했다.

네트워크 환경의 경우 문서에 따르면 약 1Gbps까지의 매우 빠른 대역폭을 제공한다고 한다. 보다 현실적인 설정을 위해, 우리는 미국 내 AI서비스 회사가 있다는 가정하에 US-West의 인스턴스들을 활용했으며, 네트워크 벤치마크 결과, 약 1.2%의 패킷 손실률과 50ms의 RTT, 그리고 혼잡도로 인한 24%의 통신 부하가 추가됨을 확인할 수 있었다.

아래의 실험들은 통신 시뮬레이터를 활용한 Delphi와 Cryptonite와 다르게 실제로 AWS 클라우드를 활용해서 수행하였다. 따라서 위 요소들 외에도 다른 요인들이 있을 수 있지만, 모두 실험수치에 반영되었음을 다시한번 강조한다.

3.2 SMPC기반 AI서비스 실험 결과

우리는 ImageNet 데이터셋을 활용한 ResNet-18 신경망 연산을 100번 연속으로 수행한 것의 평균치를 측정하였다. Delphi, Cryptonite 프로토콜과 함께 순수 HE 방식중 정확도 하락 없는 state-of-the-art

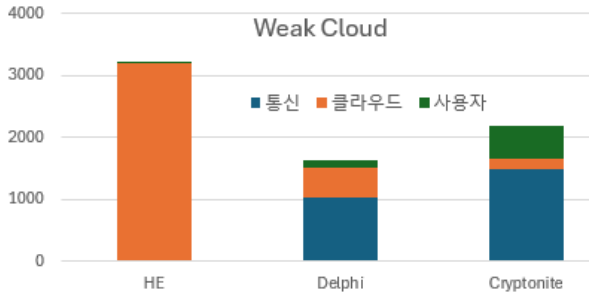

그림 1 Weak Cloud 실험 결과

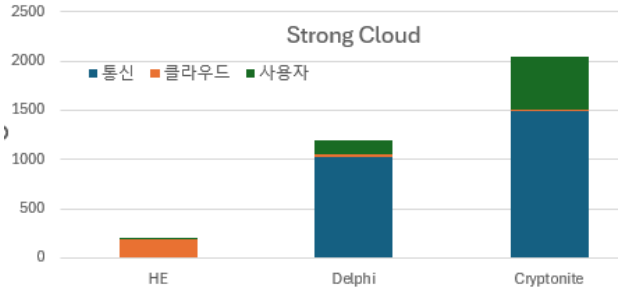

그림 2 Strong Cloud 실험 결과

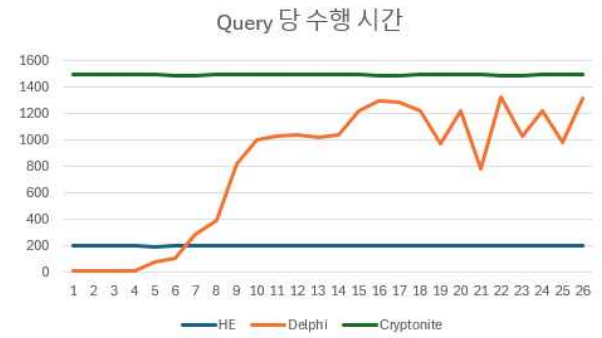

그림 3 Query 당 수행 시간 (Strong Cloud)

인 LOHEN[7]과 비교해보았다.

실험 결과는 [그림 1]과 [그림 2]와 같다. Weak Cloud에서는 예상과 같이 HE의 성능이 제일 느렸지만, Cryptonite가 Delphi보다 오히려 느렸다. 이는 Cryptonite의 주된 최적화 기법인 가변적인 네트워크 조절이 실제 클라우드 서비스 환경에서는 불가능하기 때문에 그들이 발표한 실험결과와 다르게 되려 Delphi보다 느려진 것이다. 또한 전반적인 통신부하가 크게 나타나는데, 이런 결과는 Cryptonite가 가정한 비현실적인 네트워크 설정은 실제 클라우드에서 사용할 수 없다는 점을 시사한다.

또한, Strong Cloud에서는, HE의 성능이 상대적으로 매우 빠른 것을 확인할 수 있다. 이는 두가지 이유를 통해 확인할 수 있다. 우선, Delphi가 최초 몇 번의 신경망 연산은 미리 전처리된 데이터를 통해 수행시간에 이점을 취할 수 있지만, 대부분인 나머지 연산은 미리 처리되지 않아 함께 연산을 수행해야 하기에 평균 수행시간이 많이 늘어났기 때문이다. 둘째로는, HE의 경우는 사용자는 암호화하고 결

과를 복호화하는 것 외에는 모든 연산이 클라우드에서만 이루어지기 때문에 클라우드 환경이 강할수록 성능 개선이 크게 나타나기 때문이다. 이러한 점은 상대적으로 느리다고 알려진 HE가 오히려 클라우드가 사용자보다 강한 컴퓨팅 환경을 지녔다는 현실적인 가정하에서는 다른 SMPC보다 효율적이라는 점을 나타내는 결과라 할 수 있다.

그림 3은 Query 당 수행시간 결과를 나타낸다. Delphi는 처음에 전처리한 몇 개 덕분에 수행시간이 매우 짧지만, 이들이 소모됨에 따라 수행시간이 급격히 늘어난다. Cryptonite와 HE는 전처리의 영향을 받지 않아 거의 균일한 수행시간을 나타낸다.

4. 토론 및 고찰

위 실험들을 통해 우리는 메모리 스토리지, 네트워크, 그리고 사용자/클라우드의 컴퓨팅 성능이 현실적인 상황에서 HE와 두가지 SMPC 기술들의 성능을 확인할 수 있었다. 이와 더불어 몇가지 논의할 사항들에 대해 서술하고자 한다.

본 연구가 제시하는 것은 보편적인 관점에서 AI 서비스가 활용하고 있는 환경에서의 SMPC 성능을 측정한 결과를 확인하는 것이 중요하다는 점이다. 사용자단 성능이 이미 클라우드 급으로 좋다면, 굳이 클라우드 컴퓨팅을 활용할 이유가 없다. 또한, 클라우드 서비스 업체가 제공하기 힘든 네트워크 대역폭의 가변적인 할당을 활용한다고 전제로 하는 것은 현재 일반 사용자에게는 사실상 불가능한 기능이다.

실제 클라우드 환경에서는 HE가 항상 좋다는 점을 시사하고자 하는 것도 아니다. 언급한 두가지 SMPC 외에도 Cheetah[3] 등 다양한 프로토콜들 개발되고 있으며, 보다 네트워크가 다소 안정적인 환경, 혹은 클라우드에 GPU가 사용이 불가능한 상황 등 우리 실험 환경과 다르지만 충분히 현실적인 다른 상황들이 있을 수 있으며, 이런 환경에서는 다른 SMPC 프로토콜이 더 좋은 성능을 나타낼 수 있다.

특정 환경에서 유리한 기술이 있을 수 있지만, 보편성까지 갖추기는 어렵다고 생각한다. 오히려 다양한 환경속에서 꾸준한 성능을 나타내는 기술이 보편적으로 받아드려지고 상품화되기 쉬울 수 있으며, 또한 각 환경에 따라 우월한 기술들을 선별적으로 사용하는 방법도 있을 수 있다.

5. 결론

본 연구는 기존 SMPC기반 AI서비스 연구들이 가정한 실험환경의 비현실적인 가정들을 파악하고, 실제 클라우드 서비스를 기반으로 하여 3가지 SMPC 기법들의 성능분석 결과를 제시한다. 이런 실제 수치들을 바탕으로, SMPC기반 AI 서비스가 연구논문 주제를 넘어서 실제 보편화되기 위한 연구들의 기반이 될 수 있었으면 한다.

6. ACKNOWLEDGEMENT

이 논문은 연구 수행에 있어 2025년도 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, No.RS-2023-00277060, 개방형 엣지 AI 반도체 설계 및 SW 플랫폼 기술개발, IITP-2023-RS-2023-00256081), 2024년도 BK21 FOUR 정보기술 미래인재 교육연구단, 반도체 공동연구소 지원의 결과물이다. 또한, 연구장비를 지원하고 공간을 제공한 서울대학교 컴퓨터연구소에 감사드린다.

참고문헌

[1] Mishra, Pratyush, et al. "Delphi: A cryptographic inference system for neural networks."*Proceedings of the 2020 Workshop on Privacy-Preserving Machine Learning in Practice*. 2020.

[2] Garimella, Karthik, et al. "Characterizing and optimizing end-to-end systems for private inference."*Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*. 2023.

[3] Huang, Zhicong, et al. "Cheetah: Lean and fast secure {Two-Party} deep neural network inference."*31st USENIX Security Symposium (USENIX Security 22)*. 2022.

[4] https://docs.aws.amazon.com

[5] Guo, Chuanxiong, et al. "Pingmesh: A large-scale system for data center network latency measurement and analysis."*Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*. 2015.

[6] https://cloud.google.com/vpc

[7] Nam, Kevin, et al. "LOHEN: Layer-wise Optimizations for Neural Network Inferences over Encrypted Data with High Performance or Accuracy."*Cryptology ePrint Archive*(2025).