딥러닝 프라이버시에 관한 연구

노시현 [1], 이병영 [2]
[1] 서울대학교 전기정보공학부 석박통합과정
[2] 서울대학교 전기정보공학부 교수

sihyeonroh@snu.ac.kr, byoungyoung@snu.ac.kr

A Study on Deep Learning Privacy

Si-Hyeon Roh[1], Byoung-Young Lee[2]
[1]Dept. of Electrical and Computer Engineering, Seoul National University
[2] Dept. of Electrical and Computer Engineering, Seoul National University

요 약

딥러닝은 선형 연산과 비선형 연산을 조합하여 목표로 하는 시스템을 잘 표현할 수 있는 함수를 찾기 위해 사용하며, 이미지 분류 및 생성, 거대 언어 모델 및 객체 인식의 영역에서 활발하게 사용되고 있다. 그러나 딥러닝 연산을 위해서는 모델과, 연산을 수행하고자 하는 데이터가 하나의 공간에 저장되어야 한다. 모델과 데이터를 데이터 소유자가 관리할 경우, 데이터 소유자가 모델 데이터의 프라이버시를 침해할 수 있으며, 이는 모델을 적대적 예제 생성 공격에 취약하도록 만드는 원인이 된다. 한편 모델과 데이터를 모델 소유자가 관리할 경우, 모델 소유자는 데이터의 프라이버시를 침해하여 데이터 소유자의 정보를 악의적으로 이용할 수 있다. 본 논문에서는 딥러닝 모델과 데이터의 프라이버시를 모두 보호하기 위해 주어진 딥러닝 모델의 암호화와 복호화를 수행하는 EncNet 을 구현하였으며, MNIST 와 Cifat-10 데이터셋에 대하여 실효성을 테스트하였다.

1. 서론

최근 병렬 연산을 위한 하드웨어와 소프트웨어 연구의 발전으로, 대규모 병렬 연산을 사용하는 딥러닝을 기반으로 하는 서비스가 크게 확장되었으며, 이에 따라 보안 사고의 경로와 침해의 영향력 또한 증가하고 있다. 대표적으로 모델과 데이터의 프라이버시가 강조되고 있다. 오픈소스로 공개되는 모델의 경우 적대적 예제 생성[1]을 통한 입력 값의 생성은 딥러닝 모델의 신뢰도를 크게 떨어뜨리며, 이에 따라 해당 모델을 사용하는 시스템의 안정성을 크게 손상시킨다. 모델 역변환 공격[2]은 생성형 모델을 사용하여 오픈소스 모델에 대해 모델 학습에 이용된 데이터들의 정보들을 알아낼 수 있음을 보였다. 위와 같은 이유로 사용자의 데이터와 모델의 프라이버시를 모두 지킬 수 있는 연산 시스템의 요구가 증대되었으며, 위와 같은 배경에서 기밀 연산 하드웨어의 도움을 받거나 [3, 4], 동형 암호와 다자 연산 프로토콜[5]을 사용한 딥러닝 모델 및 데이터 프라이버시 보호 연구가 대두되었다. 연산의 특성 상, 딥러닝의 연산자인 모델과 피연산자인 데이터는 연산 과정 중 필연적으로 하나의 연산 유닛에 있을 수밖에 없는데, 기밀 연산 하드웨어는 보안 목적으로 설계된 하드웨어의 도움을 받아 이 과정을 안전하게 수행한다. 한편 다자 연산 프로토콜에서는 서로 접점이 없는 다자들 간에 연산과 피연산자를 분배하여 연산을 수행하며, 수행 내용은 동형암호를 사용함으로써 각 연산을 수행자는 연산 내용과 데이터를 알 수 없도록 보호한다. 그러나 딥러닝 프라이버시 보호를 대표하는 위의 두 방식은 보안 프로토콜을 실행하기 위한 막대한 추가 연산과 네트워크 자원이 필요하거나 기밀 계산을 수행할 수 있는 하드웨어의 지원이 필요하다. 이에 따라 연산 자원의 부담 없이 적은 성능 저하를 경험하며 딥러닝 서비스 이용자들이 쉽게 사용할 수 있는 새로운 프라이버시 보호 방식이 필요하다.

2. 배경

2.1 기밀 연산

일반적인 CPU 에서는 연산 수행 시 악의적인 공격자에 의해 연산 수행 과정 및 내용이 유출될 수 있다. 이에 대한 근본적인 문제 원인으로 공격자가 실행시키는 악성 프로그램의 수행이 희생자의 취약한 프로그램과 동일한 하드웨어에서 수행되고 있다는 점이

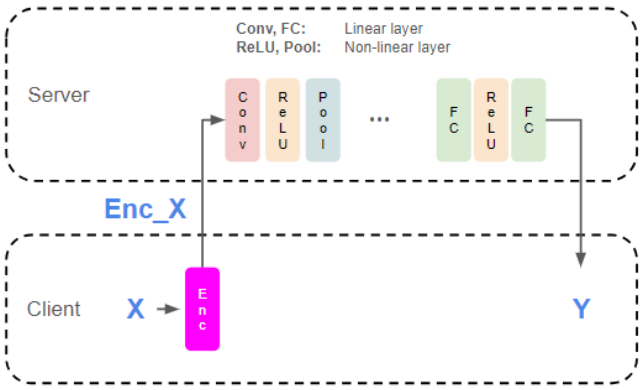
지적되었으며, 때문에 각 프로그램이 실행되는 하드웨어 차원에서의 격리가 수행되지 않기 때문에 커널을 점령하거나 마이크로 아키텍처의 특성을 사용한 공격[6]에 취약하다는 특성을 보여왔다. 기밀연산의 한 예로 Intel 의 SGX 모듈은 보안적으로 안전하게 격리된 환경에서 프로그램을 구동시킬 수 있는 환경을 제공하며, 악의적인 의도를 지닌 호스트가 연산의 내용을 알아볼 수 없도록 막는다. 그러나 외부 장치의 사용이 제한되고 연산 자원이 많지 않기 때문에 높은 성능을 필요로 하는 작업에는 제약이 따른다.

2.2 딥러닝

딥러닝은 알려지지 않은 함수를 추론하기 위해 선형 연산과 비선형 연산으로 구성된 모델을 생성하고, 목표 함수의 피연산자에 따른 기대되는 결과 값과 모델 연산 결과 사이의 오차를 손실함수로 두어, 손실함수가 줄어드는 방향으로 경사 하강법을 사용해 선형 연산자들을 원소 값들을 변형하며 목표 함수를 찾는 방법이다. 딥러닝은 일반적으로 막대한 양의 행렬곱 연산이 수반되기 때문에 행렬곱 연산의 병렬성을 사용하여 그래픽 프로세서와 같이 막대한 병렬 연산을 수행할 수 있는 장치와 함께 사용한다.

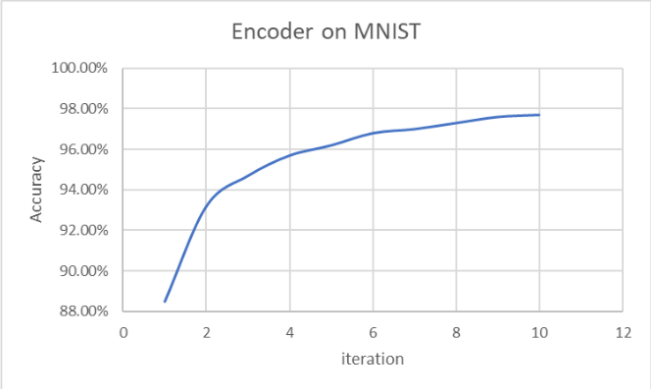
3. 본론

본 논문에서는 딥러닝의 모델과 데이터를 보호하기 위한 방법으로 EncNet 을 설계하였다.

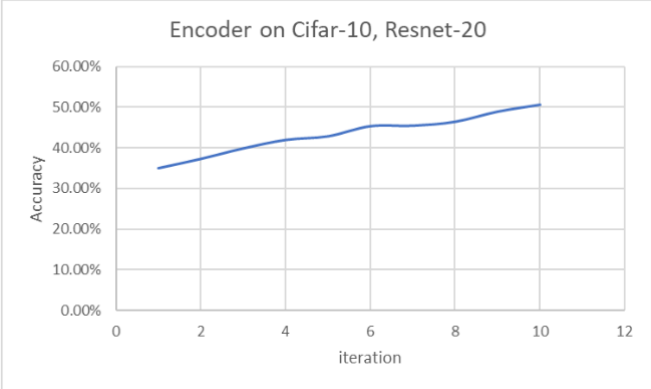


(그림 1) EncNet 의 디자인

EncNet 은 모델을 제공하는 서버와 데이터를 소유하고 있는 클라이언트 사이의 안전한 정보 교환을 목적으로 한다. 클라이언트는 이를 위해 암호화 모델을 갖는다. 암호화 모델은 클라이언트가 소유한 데이터(X)를 암호화하며, 암호화한 클라이언트의 데이터(Enc_X)를 서버로 보내면, 서버는 암호화된 데이터(Enc_X)를 서버가 소유한 모델과 함께 연산하여 결과값 Y 를 클라이언트에게 반환한다. 암호화 모델의 목표는 Enc_X 를 서버에 보내 반환 받은 결과 값과 X 를 서버에 보내 반환 받은 결과 값의 오차를 줄이는 것이며, 이는 클라이언트가 소유한 데이터(X)에서 서버의 모델에서 연산의 결과에 영향을 주지 않는 부분을 변형한 데이터(Enc_X)를 주는 것으로 이해할 수 있다. 위 설계를 위해 행렬곱 연산 3 회와 비선형 연산 3 회(ELU x2, Softmax x1)로 이루어진 EncNet 을 구성하고, MNIST 와 Cifar-10 데이터셋에 대하여 1 회의 컨볼루션 연산과 행렬곱 연산으로 구성된 간단한 모델과 Resnet-20 모델에 대해 결과의 정확도와 성능을 측정하였다.



(그림 2) MNIST 데이터셋, CNN 모델 수행 결과



(그림 3) Cifar-10 데이터셋, Resnet-20 모델 수행 결과

그 결과, MNIST 데이터셋의 경우 암호화 모듈을 사용하지 않고 추론을 수행한 결과 정확도가 99.8%였으나, 암호화를 적용한 후 10 회의 반복 학습동안 추론한 결과값이 약 98%까지 도달하였으며, 1.8%p 의 정확도 손실이 발생하였다. 한편, Resnet-20 모델에서 Cifar-10 데이터셋을 추론한 결과, 암호화 모듈이 없었을 때는 91.3%에 달하는 정확도가 암호화 모듈을 사용하고 나서 10 회 반복 학습에도 약 50%대의 정확도를 보여 추론에 있어서 40%p 이상의 정확도 저하를 보였다.

<표 1> 암호화 모델 학습 수행 시간

반복 횟수	소요 시간(초)	소요 시간(초)
1	3.4	15.2
2	6.7	30.1
3	9.9	44.9
4	13.1	59.9
5	16.3	74.8
6	19.5	89.8
7	22.7	104.8
8	25.9	120
9	29.1	135.7
10	32.3	150.1

암호화 모델을 학습하는데 소요되는 시간은 보호하려는 추론 모델의 크기에 좌우되며, 그림 2, 3 에서 제시한 정확도 향상과 표 1 의 학습에 소요된 시간을 참조할 때, MNIST 에 대한 간단한 모델에 적용한 암호화 모델의 학습 속도가 월등히 빠르다는 것을 확인할 수 있다.

4. 결론

본 연구에서 설계한 EncNet 은 간단한 모델과 데이터에 대해서는 암호화 모델을 적용하지 않았을 때와 비슷한 성능을 보였으나, 복잡한 모델과 데이터에 대해서는 모델을 적용하지 않았을 때에 비해 정확도에 큰 감소가 있었다. EncNet 에서 구하고자 하는 함수는 주어진 모델과 데이터에서, 데이터로부터 모델의 추론 결과에 영향을 적게 주는 요소를 변형해주는 것이며, 이러한 작업을 잘 수행하기 위해서는 EncNet 의 모델 복잡도를 증가시키면 원하는 결과를 얻을 수도 있으나, 이는 클라이언트 측의 막대한 연산 오버헤드를 유발하므로 고려하지 않았다.

모델의 정확도와 높은 성능을 보장하기 위해서는 암호화 모델의 동작을 분석하여 이에 맞는 복호화 모델을 추가해주는 것을 고려해볼 수 있을 것이다.

참고문헌

[1] Ian J. Goodfellow "Explaining and Harnessing Adversarial Examples " ICLR, San Diego, 2015, 11

[2] Yuhng Zhang “The Secret Revealer: Generative Model-Inversion Attacks Against Deep Neural Networks” CVPR, Seattle, 2020, 9

[3] Floran Tramer “Slalom: Fast, Verifiable and Private Execution of Neural Networks in Trusted Hardware” ICLR, New Orleans, 2019, 19

[4] Stvros Volos “Graviton: Trusted Execution Environments on GPUs” OSDI, California, 2018, 17

[5] Bria Knott “CrypTen: Secure Multi-Party Computation Meets Machine Learning” NeurIPS, Vancouver, 2021, 23

[5] Moritz Lipp “Meltdown: Reading Kernel Memory from User Space” Usenix Security, Baltimore, 2018, 19