Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.33, NO.3, Jun. 2023 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2023.33.3.375

완전 동형 암호에서의 정밀한 맥스 풀링 연산*

이 은 상†‡
세종대학교 (교수)

Precise Max-Pooling on Fully Homomorphic Encryption*

Eunsang Lee†‡
Sejong University (Professor)

요 약

완전동형암호는 암호화된 데이터에 대한 대수적 연산을 지원하며, 최근에는 최대값 함수 등의 비대수적 연산도 근사하는 방법이 연구되고 있다. 그러나 아직 4개 이상의 숫자에 대한 정밀한 맥스 풀링 근사 연구는 이루어지지 않았다. 본 연구에서는 최대값 함수 근사 다항식의 합성을 활용하여 정밀한 맥스 풀링 근사 기법을 제안하였으며, 이를 이론적으로 분석하여 높은 정밀도를 증명하였다. 실험 결과, 제안하는 근사 맥스 풀링은 1ms 이내의 작은 분할 실행 시간과 이론적 분석과 일치하는 높은 정밀도를 보여주었다.

ABSTRACT

Fully homomorphic encryption enables algebraic operations on encrypted data, and recently, methods for approximating non-algebraic operations such as the maximum function have been studied. However, precise approximation of max-pooling operations for four or more numbers have not been researched yet. In this study, we propose a precise max-pooling approximation method using the composition of approximate polynomials of the maximum function and theoretically analyze its precision. Experimental results show that the proposed approximate max-pooling has a small amortized runtime of less than 1ms and high precision that matches the theoretical analysis.

Keywords: Post-Quantum Cryptography, Fully Homomorphic Encryption, Residue Number System Variant Cheon-Kim-Kim-Song (RNS-CKKS), Max-Pooling, Deep Learning

I. 서 론

동형암호(homomorphic encryption)는 암호화된 데이터에 대해 대수적인 연산을 가능하게 하는 암호 방식이다. 과거에는 동형암호를 이용해 암호화된 데이터에 대해서 제한된 횟수의 동형 연산만 수행할 수 있었다. 그러나 Gentry가 2009년에 처음으로 암호화된 데이터에 대한 동형 연산을 무제한으로 수행할 수 있는 암호를 개발하였으며 이 암호 방식은 완전동형암호(fully homomorphic encryption)라 불린다[1]. 완전동형암호가 처음 개발된 이후로 수행 시간 및 정밀도 등의 측면에서 성능이 크게 향상되었고 컴퓨터 비전, 자연어 처리, 통계 처리 등 다양한 응용에 적용되고 있으며 현재는 표준화 작업이 진행 중이다.

완전동형암호는 비트 단위로 암호화를 하는 비트-단위 완전동형암호 및 단어 단위로 암호화를 하는 단어-단위 완전동형암호로 분류된다. 최근 단어-단위 완전동형암호는 딥러닝 응용에서 광범위하게 사용되

Received(04. 24. 2023), Modified(05. 24. 2023), Accepted(05. 24. 2023)
* 이 논문은 2023년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (No. 2022R1I1A1A0106828412). 또한, 이 논문은 2023년도 세종대학교 교내연구비 지원에 의한 논문임.
† 주저자, eslee3209@sejong.ac.kr
‡ 교신저자, eslee3209@sejong.ac.kr(Corresponding author)

고 있으며 본 연구에서는 단어-단위 완전동형암호에 초점을 둔다. 본 연구에서는 특히 가장 유망한 단어-단위 완전동형암호인 RNS-CKKS(Residue Number System Variant Cheon-Kim-Kim-Song) 스킴[2,3]에 초점을 둔다.

완전동형암호는 암호화된 상태에서 대수적인 연산을 지원하지만 비대수적 연산을 사용하는 응용도 많이 있다. 대표적으로 딥러닝에서는 맥스 풀링(max-pooling)을 많이 사용하는데 이는 비대수적 연산으로서 암호화된 상태에서 정확히 계산될 수는 없다. 최근 여러 연구에서는 최대값 함수를 다항식으로 근사한 후 그 다항식을 대신 암호화된 상태에서 연산하는 방법을 연구하였다[4-6]. Lee 등[6]은 암호화된 상태에서 두 숫자에 대한 최대값의 근사값을 여러 다항식의 합성 다항식을 사용하여 계산할 것을 제안하였으며 이는 8~20 비트의 높은 정밀도를 달성한다. 이는 현재까지 알려진 최대값 함수 근사 방법 중 가장 높은 보안성 수준 및 빠른 수행시간 성능을 가진 기술로 알려져있다.

두 숫자에 대한 최대값을 연산하는 방법과는 달리 4개 이상의 숫자에 대한 맥스 풀링 연산에 대한 연구는 충분히 이루어지지 않았다. 기존 여러 연구에서는 맥스 풀링 연산을 평균 풀링(average pooling) 혹은 합계 풀링(sum pooling)등 다른 연산으로 대체하는 방식을 채택하였는데 이는 새로 디자인된 네트워크에 대해 새로 트레이닝을 해야한다는 단점이 있으며 분류 정확도가 떨어질 수 있다[7,8]. 본 연구에서는 처음으로 RNS-CKKS 스킴으로 암호화된 데이터에 대해 4개 이상의 숫자에 대한 정밀한 맥스 풀링 근사 기법을 연구한다. 본 연구가 기여한 바는 다음 세 가지이다.

1) 단어-단위 완전동형암호에서 4개 이상의 숫자에 대한 정밀한 맥스 풀링 근사 기법을 처음으로 제안한다. 이는 연구 [6]에서 제안한 최대값 함수 근사 다항식을 합성하는 방식이다.

2) 제안하는 맥스 풀링 근사 기법의 정밀도에 대해 이론적으로 분석한다. 주어진 정밀도 파라미터 α에 대해 2^t개의 숫자에 대한 제안하는 근사 맥스 풀링은 최소 $\alpha-\log_2 t$의 정밀도를 달성함이 증명된다.

3) 마지막으로 실험을 통해 제안하는 맥스 풀링 근사 기법의 수행 시간과 정밀도를 계산한다. 본 실험은 대표적인 RNS-CKKS 라이브러리 중 하나인 Lattigo 라이브러리[9]에서 수행된다. 본 실험을 통해 4개 숫자에 대한 제안하는 근사 맥스 풀링 연산은 1ms 이내의 작은 분할 실행 시간(amortized runtime)을 가짐을 보인다. 또한, 주어진 정밀도 파라미터 α에 대해 $\alpha-1$ 이상의 최소 정밀도를 가짐을 알 수 있는데 이는 제안하는 근사 맥스 풀링에 대한 이론적인 분석과 일치한다. 충분히 큰 정밀도 α를 사용하면 제안하는 근사 맥스 풀링 연산은 분류 정확도를 떨어뜨리지 않으면서 딥러닝에 활용될 것이 기대된다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 완전동형암호 RNS-CKKS 스킴 및 이 스킴에서의 최대값 연산에 대한 배경이론을 설명하며 3장에서는 제안하는 고정밀도 근사 맥스 풀링을 제안한다. 4장에서는 실험을 통해 제안하는 근사 맥스 풀링의 수행시간 및 정확도를 분석하고 5장에서는 결론을 맺는다.

II. 배경이론

본 장에서는 완전동형암호 RNS-CKKS 의 개념과 RNS-CKKS에서의 최대값 함수 근사 방법에 대해 소개한다.

2.1 RNS-CKKS

RNS-CKKS는 대표적인 완전동형암호 중 하나로서 암호화된 데이터에서의 고정소수점 대수적인 연산을 지원한다. 주어진 정수 N에 대해 $n=N/2$개의 실수(혹은 복소수)로 이루어진 벡터가 하나의 암호문에 암호화된다. 동형 연산은 벡터의 성분별 덧셈 및 성분별 곱셈을 지원한다. 만약 벡터 $\boldsymbol{u}\in\mathbb{R}^n$의 암호문을 단순히 $[\boldsymbol{u}]$로 표현한다면 RNS-CKKS 스킴의 동형 덧셈($\oplus$), 스칼라 곱셈($\odot$), 넌스칼라 곱셈($\otimes$)은 다음 식을 만족한다.

$$[\boldsymbol{u}]\oplus[\boldsymbol{v}]=[\boldsymbol{u}+\boldsymbol{v}]$$
$$[\boldsymbol{u}]\odot\boldsymbol{v}=\boldsymbol{u}\odot[\boldsymbol{v}]=[\boldsymbol{u}\cdot\boldsymbol{v}]$$
$$[\boldsymbol{u}]\otimes[\boldsymbol{v}]=[\boldsymbol{u}\cdot\boldsymbol{v}]$$

여기서 $\boldsymbol{u}\cdot\boldsymbol{v}$ 기호는 두 벡터의 성분별 곱셈을 수행한 벡터를 나타낸다.

2.2 RNS-CKKS에서의 최대값 연산

RNS-CKKS에서는 암호화된 상태에서 덧셈 및

곱셈과 같은 대수적인 연산만 지원하며 두 숫자 a,b의 최대값 $\max(a,b)$을 동형적으로 계산하는 연산을 지원하지 않는다. 따라서, $\max(a,b)$의 정확한 값을 동형적으로 계산하는 대신 $\max(a,b)$를 근사하는 다항식 $p(a,b)$를 계산해야한다.

연구 [6]에서는 부호 함수에 대한 여러 미니맥스 다항식(minimax approxmate polynomial)의 합성다항식을 사용하여 $\max(a,b)$를 근사하는 방법을 제안하였다. 주어진 정밀도 파라미터 α 및 두 숫자 $a,b\in[0,1]$에 대해 연구 [6]에서 제안한 다항식 $p_\alpha(a,b)$는 다음 조건을 만족시킨다.

$$|p_\alpha(a,b)-\max(a,b)|\le 2^{-\alpha} \tag{1}$$

또한, 정밀도 파라미터 $\alpha=8\sim20$에 대해 [6]에서 제안한 근사 최대값 함수가 정밀도 조건 (1)을 만족시키는 것이 실험을 통해 검증되었다.

III. 고정밀도 근사 맥스 풀링

기존에 완전동형암호로 암호화된 상태에서 두 숫자에 대한 최대값을 근사하는 연산을 수행하는 연구는 이루어졌지만 4개 이상의 숫자에 대해 최대값을 정밀하게 계산하는 연구는 아직 이루어지지 않았다. 딥러닝에서 널리 사용되는 맥스 풀링 함수는 크기가 2×2 혹은 3×3 등 4개 이상의 숫자에 대한 최대값을 계산해야하는 경우가 많다. 본 연구에서는 동형암호로 암호화된 상태에서 4개 이상의 숫자에 대한 최대값을 근사하는 알고리즘을 처음으로 제안하고 정밀도를 분석한다.

어떤 자연수 t에 대해 2^t개의 숫자 $a_1,a_2,\cdots,a_{2^t}$에 대한 최대값을 근사적으로 구하는 경우를 생각하자. 이 때, 제안하는 근사 다항식 $P_{\alpha,t}(a_1,a_2,\cdots,a_{2^t})$는 다음과 같이 재귀적으로 정의된다.

$$P_{\alpha,t}(a_1,a_2,\cdots,a_{2^t})=\begin{cases}p_\alpha(a_1,a_2) & t=1\\ p_\alpha(P_{\alpha,t-1}(a_1,\cdots,a_{2^{t-1}}),P_{\alpha,t-1}(a_{2^{t-1}+1},\cdots,a_{2^t})) & t\ge2\end{cases}$$

이 때, 맥스 풀링 근사 다항식 $P_{\alpha,t}$에 대해 다음 정리가 성립한다.

정리 1 [맥스 풀링 근사 다항식 정리] 주어진 정밀도 파라미터 α에 대해 2^t개의 숫자 $a_1,a_2,\cdots,a_{2^t}$가 $a_i\in[(t-1)2^{-\alpha},1-(t-1)2^{-\alpha}]$ for $1\le i\le 2^t$ 를 만족시킨다고 하자. 그러면 다음 부등식이 성립한다.

$$|P_{\alpha,t}(a_1,a_2,\cdots,a_{2^t})-\max(a_1,a_2,\cdots,a_{2^t})|\le t\cdot2^{-\alpha}. \tag{2}$$

증명. 먼저, $t=1$인 경우에는 부등식 (1)에 의해 바로 성립한다. 이제 귀납적 증명을 위해 $P_{\alpha,t}$가 부등식 (2)를 만족시킬 때, $P_{\alpha,t+1}$도 또한 부등식 (2)를 만족시킴을 보이자. 즉,

$$|P_{\alpha,t+1}(a_1,\cdots,a_{2^{t+1}})-\max(a_1,\cdots,a_{2^{t+1}})|\le(t+1)\cdot2^{-\alpha}$$
$$\text{for}\quad a_1,\cdots,a_{2^{t+1}}\in[t\cdot2^{-\alpha},1-t\cdot2^{-\alpha}] \tag{3}$$

가 성립함을 보일 것이다. 귀납 가정에 의해 다음 두 부등식이 성립한다.

$$|P_{\alpha,t}(a_1,\cdots,a_{2^t})-\max(a_1,\cdots,a_{2^t})|\le t\cdot2^{-\alpha}, \tag{4}$$

$$|P_{\alpha,t}(a_{2^t+1},\cdots,a_{2^{t+1}})-\max(a_{2^t+1},\cdots,a_{2^{t+1}})|\le t\cdot2^{-\alpha}. \tag{5}$$

일반성을 잃지 않고

$$\max(a_1,\cdots,a_{2^t})\le\max(a_{2^t+1},\cdots,a_{2^{t+1}}) \tag{6}$$

이 성립한다고 하자. 그러면 부등식 (4), (6)에 의해

$$\begin{aligned}P_{\alpha,t}(a_1,\cdots,a_{2^t})&\le\max(a_1,\cdots,a_{2^t})+t\cdot2^{-\alpha}\\&\le\max(a_1,\cdots,a_{2^{t+1}})+t\cdot2^{-\alpha}\end{aligned} \tag{7}$$

가 성립한다. 또한, 부등식 (5)에 의해

$$\begin{aligned}P_{\alpha,t}(a_{2^t+1},\cdots,a_{2^{t+1}})&\le\max(a_{2^t+1},\cdots,a_{2^{t+1}})+t\cdot2^{-\alpha}\\&=\max(a_1,\cdots,a_{2^{t+1}})+t\cdot2^{-\alpha}\end{aligned} \tag{8}$$

이 성립한다. 따라서 부등식 (7)와 (8)에 의해

$$\begin{aligned}&\max(P_{\alpha,t}(a_1,\cdots,a_{2^t}),P_{\alpha,t}(a_{2^t+1},\cdots,a_{2^{t+1}}))\\&\quad\le\max(a_1,\cdots,a_{2^{t+1}})+t\cdot2^{-\alpha}\end{aligned} \tag{9}$$

가 성립한다. 한편, 부등식 (5), (6)에 의해

$$\begin{aligned}\max(a_1,\cdots,a_{2^{t+1}})-t\cdot 2^{-\alpha} &\le \max(a_{2^t+1},\cdots,a_{2^{t+1}})-t\cdot 2^{-\alpha}\\ &\le P_{\alpha,t}(a_{2^t+1},\cdots,a_{2^{t+1}})\\ &\le \max(P_{\alpha,t}(a_1,\cdots,a_{2^t}),P_{\alpha,t}(a_{2^t+1},\cdots,a_{2^{t+1}}))\end{aligned} \quad (10)$$

이 성립한다. 부등식 (9), (10)을 다음과 같이 하나의 부등식으로 정리할 수 있다.

$$\begin{aligned}&|\max(P_{\alpha,t}(a_1,\cdots,a_{2^t}),P_{\alpha,t}(a_{2^t+1},\cdots,a_{2^{t+1}}))-\max(a_1,\cdots,a_{2^{t+1}})|\\ &\quad\le t\cdot 2^{-\alpha}.\end{aligned} \quad (11)$$

여기서 $P_{\alpha,t}(a_1,\cdots,a_{2^t})$의 범위를 생각해보자. $a_1,\cdots,a_{2^t}$는 $[t\cdot 2^{-\alpha},1-t\cdot 2^{-\alpha}]$ 범위에 속하며 $\max(a_1,\cdots,a_{2^t})$도 물론 $[t\cdot 2^{-\alpha},1-t\cdot 2^{-\alpha}]$ 범위에 속한다. 이 때, $\max(a_1,\cdots,a_{2^t})$의 범위와 부등식 (4)를 고려하면 $P_{\alpha,t}(a_1,\cdots,a_{2^t})$는 $[0,1]$ 안에 속한다는 사실을 알 수 있다. 마찬가지로 $P_{\alpha,t}(a_{2^t+1},\cdots,a_{2^{t+1}})$도 $[0,1]$ 안에 속하게 된다. 따라서 부등식 (1)에서 a,b에 각각 $P_{\alpha,t}(a_1,\cdots,a_{2^t})$와 $P_{\alpha,t}(a_{2^t+1},\cdots,a_{2^{t+1}})$를 대입하면

$$\begin{aligned}&|P_{\alpha,t+1}(a_1,\cdots,a_{2^{t+1}})-\max(P_\alpha(a_1,\cdots,a_{2^t}),P_\alpha(a_{2^t+1},\cdots,a_{2^{t+1}}))|\\ &\quad\le 2^{-\alpha}\end{aligned} \quad (12)$$

가 성립한다. 부등식 (11)과 (12) 및 삼각부등식으로부터

$|P_{\alpha,t+1}(a_1,\cdots,a_{2^{t+1}})-\max(a_1,\cdots,a_{2^{t+1}})|\le (t+1)\cdot 2^{-\alpha}$가 성립함을 알 수 있다. 이는 처음에 보이고자 하는 부등식 (3)와 동일하다. 따라서 귀납적 증명을 통해 정리 1이 증명되었다. □

정리 1에 의하면 주어진 정밀도 파라미터 α에 대해 제안하는 근사 맥스 풀링 다항식 $P_{\alpha,t}(a_1,a_2,\cdots,a_{2^t})$는 $\max(a_1,a_2,\cdots,a_{2^t})$를 $\alpha-\log_2 t$ 이상의 정밀도로 근사한다. 정밀도 파라미터 α를 충분히 높이면 근사 오차가 매우 줄어들기 때문에 제안하는 근사 맥스 풀링 다항식은 맥스 풀링 연산을 사용하는 다양한 딥러닝 응용에 효과적으로 활용될 것을 기대할 수 있다.

IV. 실험 결과

본 장에서는 $t=2$인 경우, 즉, 4개의 숫자에 대한 경우에 대해 제안하는 근사 맥스 풀링 연산을 실제 구현하여 수행시간 및 정밀도 성능을 분석한다.

4.1 실험환경

본 실험은 대표적인 오픈소스 RNS-CKKS 스킴 라이브러리인 Lattigo 라이브러리에서 수행된다. 컴퓨터 환경은 AMD Ryzen Threadripper PRO 3995WX 2.096 GHz이며 512 GB RAM이 있고 Ubuntu 20.04 운영체제를 사용한다.

4.2 파라미터

본 실험에서는 다항식 차수 파라미터 $N=2^{17}$을 사용한다. 그러면 암호화되는 벡터의 성분 개수는 $n=2^{16}$이다. 파라미터 N을 큰 값으로 잡은 이유는 맥스 풀링을 사용하는 컨볼루션 신경망 응용에서는 큰 N을 사용하는 경우가 많기 때문이다[10,11].

RNS-CKKS 스킴의 비밀키의 해밍 무게(Hamming weight)는 192로 설정한다. 특수 모듈러스(special modulus)와 기초 모듈러스(base modulus)로는 55 비트 소수를 사용하며 그 외의 모듈러스로는 50 비트 소수를 사용한다. 특수 모듈러스는 5개를 사용하는데, 이 다수의 특수 모듈러스는 연산의 속도를 빠르게 한다. 스케일링 팩터(scaling factor)는 2^{50}으로 설정한다.

암호문의 레벨은 사용하는 정밀도 α에 따라 다른 레벨을 사용한다. 본 실험에서는 정밀도 $\alpha=10,12,14$에 대한 근사 맥스 풀링 연산을 수행한다. 이 정밀도 파라미터에 대한 근사 다항식 $p_\alpha(a,b)$는 각각 레벨을 11, 13, 15개 사용하며 $P_{\alpha,2}(a,b,c,d)$는 각각 22, 26, 30개의 레벨을 사용한다. 따라서, 정밀도 $\alpha=10,12,14$에 대해 암호문 레벨을 각각 22, 26, 30개로 설정한다.

4.3 입력

본 실험에서 $P_{\alpha,2}(a,b,c,d)$를 계산할 때, a,b,c,d의 값으로는 [0.1, 0.9]에서 각각 랜덤으로 선택한다. 이 때, RNS-CKKS 암호문 하나는 $n=2^{16}$개의 성분을 가진다. 따라서 정확히는 2^{16}개의 성분 각각을 [0.1, 0.9]에서 랜덤으로 선택하는 것이다. 입력의 범위를 [0, 1]이 아닌 [0.1, 0.9]로 선택한 이유

는 RNS-CKKS 암호문 연산 자체가 오차를 수반하므로 오차가 발생하더라도 [0, 1] 범위를 넘지 않도록 하기 위함이다.

4.4 수행시간

Table 1.은 정밀도 파라미터 α에 따른 수행 시간 및 분할 실행 시간을 나타낸다. 분할 실행 시간은 전체 연산 시간을 암호문의 벡터 성분 개수 $n=2^{16}$으로 나눈 값이다. 즉, 분할 실행 시간은 하나의 데이터(실수 혹은 복소수) 당 평균 수행시간을 의미한다.

α에 따라 실행시간은 증가하며 분할 실행 시간은 $\alpha=10,12,14$ 모두에 대해 1ms 내의 작은 시간이 소요됨을 알 수 있다. 본 실험은 CPU 한 코어를 사용한 것임을 감안할 때, GPU나 하드웨어 가속기를 사용한다면 실행 시간이 각각 적어도 100배, 1000배 이상 가속될 것을 기대할 수 있다[12-13].

Table 1. Runtime and amortized runtime of the proposed approximate max pooling operation according to precision parameter α

α	runtime	amortized runtime
10	24s	0.366ms
12	39s	0.595ms
14	59s	0.9ms

4.5 정밀도

Table 2.는 정밀도 파라미터 α에 따른 최소 정밀도, 최대 정밀도, 평균 정밀도, 및 중앙값 정밀도를 나타낸다. 최소 정밀도, 최대 정밀도, 평균 정밀도 및 중앙값 정밀도는 각각 최소 오차, 최대 오차, 평균 오차 및 오차의 중앙값에 대해 $-\log_2$ 함수를 계산한 값이다.

Table 2. Minimum, maximum, average, and median precisions according to precision parameter α

α	min prec.	max prec.	average prec.	median prec.
10	9.18	27.73	12.11	12.48
12	11.64	40.52	14.80	15.15
14	13.69	32.98	16.62	16.94

α가 커짐에 따라 근사 맥스 풀링의 정밀도가 증가한다. 특히, α가 14인 경우 최소 정밀도가 13.69 비트이며 평균 정밀도는 16.62 비트인데 이는 실제 컨볼루션 신경망에서 높은 정확도를 달성하기에 충분히 높은 정밀도이다[11].

또한, 3장에서 제안하는 근사 맥스 풀링 다항식이 주어진 정밀도 파라미터 α에 대해 최소 정밀도 $\alpha-\log_2 t$를 달성함을 이론적으로 증명하였으며 $t=2$인 경우는 $\alpha-1$의 최소 정밀도를 달성하게 된다. Table 2.는 이 이론적인 분석이 실제 실험적으로도 성립함을 보여준다.

V. 결 론

본 연구에서는 완전동형암호에서 4개 이상의 숫자에 대한 정밀한 근사 맥스 풀링 연산을 처음으로 제안하였으며 정밀 근사 최대값 연산의 합성을 사용하여 정밀하게 맥스 풀링 연산을 근사하였다. 또한, 제안하는 맥스 풀링 근사 기법의 정밀도를 이론적으로 증명하였다. 마지막으로 대표적인 RNS-CKKS 라이브러리인 Lattigo에서의 실험을 통해 제안하는 근사 맥스 풀링 연산의 성능을 분석하였다. 실험 결과 4개 숫자에 대한 제안하는 근사 맥스 풀링 연산은 1ms 이내의 빠른 분할 실행 시간 성능을 가지며 높은 정밀도를 가진다. 특히, 정밀도 파라미터 α에 대해 $\alpha-\log_2 t$의 최소 정밀도를 가짐을 이론적으로 증명하였는데 Lattigo 라이브러리 상에서의 실험 결과도 이 이론적 분석과 일치함을 보여준다.

References

[1] C. Gentry, "Fully homomorphic encryption using ideal lattices," *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, pp. 169-178, May 2009.

[2] J. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," *Proceedings of ASIACRYPT 2017*, LNCS 10624, pp. 409-437, Dec. 2017.

[3] J. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, "A full RNS variant of

approximate homomorphic encryption," *Proceedings of Selected Areas in Cryptography 2018*, LNCS 11349, pp. 347-368, Aug. 2018.

[4] J. Cheon, D. Kim, D. Kim, H. Lee, and K. Lee, "Numerical method for comparison on homomorphically encrypted numbers," *Proceedings of ASIACRYPT 2019*, LNCS 11922, pp. 415-445, Dec. 2019.

[5] J. Cheon, D. Kim, and D. Kim, "Efficient homomorphic comparison methods with optimal complexity," *Proceedings of ASIACRYPT 2020*, LNCS 12492, pp. 221-256, Dec. 2020.

[6] E. Lee, J. Lee, J. No, and Y. Kim, "Minimax approximation of sign function by composite polynomial for homomorphic comparison," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 6, pp. 3711-3727, Aug. 2021.

[7] R. Cilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: applying neural networks to encrypted data with high throughput and accuracy," *Proceedings of International Conference on Machine Learning 2016*, pp. 201-210, June 2016.

[8] R. Dathathri, O. Saarikivi, H. Chen, K. Laine, K. Lauter, S. Maleki, M. Musuvathi, and T. Mytkowicz, "CHET: an optimizing compiler for fully-homomorphic neural-network inferencing," *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 142-156, June 2019.

[9] Lattigo v3, Online: https://github.com/tuneinsight/lattigo, Aug. 2022, EPFL-LDS, Tune Insight SA.

[10] J. Lee, H. Kang, Y. Lee, W. Choi, J. Eom, M. Deryabin, E. Lee, J. Lee, D. Yoo, Y. Kim, and J. No, "Privacy-preserving machine learning with fully homomorphic encryption for deep neural network," *IEEE Access*, vol. 10, pp. 30039-30054, Mar. 2022.

[11] E. Lee, J. Lee, J. Lee, Y. Kim, Y. Kim, J. No, and W. Choi, "Low-complexity deep convolutional neural networks on fully homomorphic encryption using multiplexed parallel convolutions," *Proceedings of International Conference on Machine Learning 2022*, pp. 12403-12422, June 2022.

[12] W. Jung, S. Kim, J. Ahn, J. Cheon, and Y. Lee, "Over 100x faster bootstrapping in fully homomorphic encryption through memory-centric optimization with GPUs," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2021, no. 4, pp. 114-148, Aug. 2021.

[13] S. Kim, J. Kim, M. Kim, W. Jung, J. Kim, M. Rhu, and J. Ahn, "BTS: an accelerator for bootstrappable fully homomorphic encryption," *Proceedings of the 49th Annual International Symposium on Computer Architecture*, pp. 711-725, June 2022.

〈저 자 소 개〉

이 은 상 (Eunsang Lee) 정회원
2014년 8월: 서울대학교 전기정보공학부 학사
2020년 8월: 서울대학교 전기정보공학부 박사
2020년 9월~2022년 8월: 서울대학교 전기정보공학부 박사후연구원
2022년 9월~현재: 세종대학교 소프트웨어학과 조교수
〈관심분야〉 동형암호, 포스트 양자 암호, 프라이버시-보호 머신러닝