N,yVlN0ⱠPX1O[fO
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

O[fbX1
IEICE Technical Report
CPSY2022-38,DC2022-97(2023-03)

Intel Xeon 及び富士通 A64FX 上での ビット削減版準同型暗号ライブラリの評価

西　将輝[†]　宍戸　哲平[†]　李　欣怡[†]　木村　啓二[†]　佐野健太郎[††]

[†] 早稲田大学
[††] 理化学研究所
E-mail: †{masaki170401,teppei14860623}@kasahara.cs.waseda.ac.jp, ††xylee@toki.waseda.ac, †††keiji@waseda.jp,
††††kentaro.sano@riken.jp

あらまし　深層学習が様々な場面で利用されるようになり，それと共に学習データや入力データ，推論結果，さらには学習モデルの保護が問題となりつつある．そのため，データを暗号化したまま演算処理できる準同型暗号による深層学習が注目されている．しかしながら，準同型暗号による演算コストは高く，これまでに, 並列化やハードウェアアクセラレータの利用といった様々な高速化手法が提案されている．一方筆者等は，深層学習の推論処理において多くのビット数が必要としないことに注目し, ビット削減版準同型暗号を利用した深層学習推論を提案してきた．本稿ではまず，ビット削減版の深層学習処理を準同型暗号ライブラリ SEAL と SEAL を用いた深層学習フレームワーク HE-Transformer に実装し，Intel Xeon プロセッサ上で評価した．さらに，富士通 A64FX 上でビット削減版準同型暗号を実装し評価を行なった．評価の結果，CryptoNets を用いた MNIST データセットの分類においてオリジナルの nGraph-HE2 に対し Intel Xeon 上で最大で 9.37 倍の速度向上が得られた．また，富士通 A64FX 上で行列積を評価した結果，Intel Xeon W-2145 と比較して，最大 1.08 倍の速度向上が得られた．

キーワード　準同型暗号 深層学習 SIMD 並列化 A64FX

Evaluation of Bit-Reduced Homomorphic Encryption Library on Intel Xeon and Fujitsu A64FX

Msaki NISHI[†], Teppei SHISHIDO[†], Xinyi LI[†], Keiji KIMURA[†], and Kentaro SANO[††]

[†] Waseda University
[††] RIKEN
E-mail: †{masaki170401,teppei14860623}@kasahara.cs.waseda.ac.jp, ††xylee@toki.waseda.ac, †††keiji@waseda.jp,
††††kentaro.sano@riken.jp

1. はじめに

深層学習は計算機の重要な利用用途の一つとなり様々な場面で利用されている。それと共に深層学習で個人の健康情報や財務情報などの秘密情報を扱うことも増え，学習に用いるデータ，推論処理に対する入力データや得られた推論結果の第三者への漏洩が問題となっている．さらには，深層学習のモデルもサービスプロバイダの重要な知的財産であり，これの漏洩も重篤な経済的損失を招きうる．

上記の問題に対し，データを暗号化したまま演算可能な準同型暗号 [1] による深層学習が，秘密データを秘匿したまま深層学習の，特に推論処理を可能とする技術として注目されている．例えばこれまでに，Microsoft SEAL [2] 等の準同型暗号ライブラリや Intel AVX を用いて SEAL を高速化する Intel HEXL ライブラリ [3], 及びこれらを利用した深層学習フレームワークである HE-Transformer [4]〜[6] などが開発されてきた．

しかしながら，準同型暗号では平文を暗号化する過程でデータ量が著しく増大してしまい，演算コストが非常に高くなってしまうという問題がある．そのため，ハードウェアアクセラレータによる高速化手法がこれまで提案されてきた．また準同型暗号の演算はデータ並列性が大きく前述の SEAL，HEXL，HE-Transformer でも OpenMP によるマルチスレッドや AVX に

よる SIMD による並列処理が行われてきた．これらのソフトウェアは内部演算が 64bit で行われており，SIMD 演算を行う場合は利用可能なレーン数，すなわち並列性が制限されてしまう。

一方，深層学習では，特にその推論処理で多くのビット数を必要としなくても精度を保てるという報告がある [7], [8]. ビット数を削減することで，SIMD レーン数の増加やメモリ使用量の減少によるメモリアクセスの向上などの高速化が期待できる．そのため，筆者等はビット数を半減させ SIMD 化による高速化を実現したビット削減版準同型暗号を提案してきた [9].

本稿ではまず，ビット削減版準同型暗号を Intel Xeon の CPU と AVX512 命令セットで実装し，HE-Transformer [4]〜[6] を利用して深層学習推論を評価した結果を報告する．さらに，Scalable Vector Extension (SVE) 命令セットを持つ富士通 A64FX 上でビット削減版準同型暗号を実装し評価したのでその評価結果を報告する．

本論文の構成は次の通りである.2. 節で準同型暗号について説明し，3. 節で準同型暗号技術の関連研究を紹介する．4. 節で本研究の提案手法であるビット削減版準同型暗号とこれを用いた深層学習推論について説明する.5. 節で準同型暗号を用いた深層学習推論の評価について述べ，6. 節でビット削減版準同型暗号を富士通 A64FX で評価について述べる．最後に第 7. 節でまとめる．

2. 準同型暗号

準同型暗号は公開鍵 pk によって暗号化され，秘密鍵 sk によって復号される公開鍵暗号の一種である．本稿では準同型暗号の演算を示す時，以下のように表記することとする．

$$Enc(A) + Enc(B) = Enc(A + B) \tag{1}$$

$$Enc(A) \times Enc(B) = Enc(A \times B) \tag{2}$$

加算と乗算が可能な完全準同型暗号（FHE）を 2009 年に Gentry がを発表した [1]．さらに，上限付きで加算と乗算の計算が可能な SomeWhat 準同型暗号（SHE）が登場した．しかし，SHE では暗号文同士の乗算によりノイズが増大し閾値を超えると復元できなくなる．乗算の上限回数を確保するにはパラメータサイズを大きくする必要があり，乗算を行うごとに暗号文のサイズが大きくなる．そのため，実用上は乗算が 2，3 回程度の計算に限定されてしまう．

事前に決定されたパラメータ (Level) によって，乗算可能な回数が決められる暗号方式を Leveled 準同型暗号と呼ぶ．パラメータ (Level) を大きくすると乗算可能な回数が増加するが，処理時間と暗号文のサイズが増加する．しかし，SHE ほどは増加しないため，乗算回数をより多くできる．

演算を行いノイズが増加した暗号文に対して，ノイズを削除する Bootstrapping 処理を行うことで，演算を行なっていない暗号文と同程度までノイズを削減可能となる．これにより，SHE や Leveled 準同型暗号に Bootstrapping をはさむことで閾値を超えることなく復元が可能となり，FHE の構成が示された．

2.1 CKKS 方式 [10]

本節では本稿で対象とする CKKS 方式について説明する．CKKS 方式は整数だけでなく，実数や複素数の計算が可能な方式である．CKKS 方式では平文に対してパラメータ scale を利用したスケーリングを行い固定小数点として扱うことで，実数や複素数の計算を可能としている．scale は暗号文乗算毎に指数的に増加するため，rescale と呼ばれる操作により増加した scale を適切に調整する．

円分多項式を $\Phi_M(X)$ としたとき，CKKS 方式の平文空間は $\mathbb{Z}[X]/(\Phi_M(X))$ と表せる．これにより，複数の値を 1 つの平文として表現可能になる．これはパッキングと呼ばれ，複数要素に対する同一演算の大幅な高速化が可能となる．複数の値を 1 つの平文にパッキングする操作を encode と呼び，その逆を decode と呼ぶ．また，円分多項式の次数 N は，2.2 節で説明する NTT を適用させるために 2 の冪乗の値を取る．SEAL 上で次数 N は poly_modulus と呼ぶ．パッキングでは，複数の値をそれぞれスロットと呼ばれる値格納用の領域に配置する．一つの平文に格納できる要素数をスロット数と呼び，CKKS の場合は円分多項式の次数 N が，複素数を表現するスロット数となる．従って，実数を表すとき $N/2$ (N の半分) がスロット数となる．

暗号文は，coefficient modulus と呼ばれるパラメータで指定されたビット数の素数を用いた余剰で表現される．coefficient modulus は異なる素数の組み合わせで表現され，これをモジュラススイッチングチェーンと呼ぶ．この素数の組み合わせの数に対応して乗算可能な回数が決まり，これを level と呼ぶ．この level が Leveled 準同型暗号の名前の由来となっている．rescale と呼ばれるノイズを減らす処理によって level が減る．

準同型暗号では，パッキング処理が行われた暗号文を演算する．従って，スロットごとの演算が一度に行われる．例えば，スロット数を N として，要素数 N のベクトル $a([a_1, a_2, ..., a_N])$, $b([b_1, b_2, ..., b_N])$ をそれぞれ encode し暗号化した暗号文の加算と乗算を考える．スロット $i(1 \leqq i \leqq N)$ には，それぞれ a_i, b_i が格納されているとすると，加算と乗算はそれぞれ以下のように表せる．

$$\begin{aligned} &Enc([a_1, a_2, .., a_N]) + Enc([b_1, b_2, .., b_N]) \\ &\quad = Enc([a_1 + b_1, a_2 + b_2, .., a_N + b_N]) \end{aligned} \tag{3}$$

$$\begin{aligned} &Enc([a_1, a_2, .., a_N]) \times Enc([b_1, b_2, .., b_N]) \\ &\quad = Enc([a_1 \times b_1, a_2 \times b_2, .., a_n \times b_N]) \end{aligned} \tag{4}$$

さらに，このパッキングにより，スロットに格納されている値の順番をずらす，以下のような rotate と呼ばれる処理が存在する．

$$Enc([a_1, a_2, ..., a_N]) \rightarrow rotate(1) \rightarrow Enc([a_N, a_1, .., a_{N-1}]) \tag{5}$$

2.2 NTT(数論変換)

本節では準同型暗号で重要な演算である NTT(数論変換) について説明する．多項式の次数を N としたとき，多項式乗算には $O(N^2)$ の時間計算量を要する．しかし，多項式の項数が 2 の

冪乗のとき，FFT(高速フーリエ変換) を利用して多項式乗算の高速化が可能になる．さらに，特定の素数を法とした余剰体上で行われる FFT を数論変換 (Number Theoretic Transform: NTT) という [11]．多項式の係数列を NTT 形式に変換した場合，多項式乗算を $O(NlogN)$ で得ることができる．準同型暗号ではこれがボトルネックの一つとして知られており，Intel HEXL は AVX512 によって NTT を高速化している．

3. 関 連 研 究

3.1 CryptoNets

CryptoNets [12], [13] は Microsoft によって開発された準同型暗号を利用したニューラルネットワークである．MNIST データセットを対象とする CryptoNets と，CIFAR-10 データセットや CalTech-101 データセットを対象とする Low-Latency CryptoNets（LoLa）の 2 つのバージョンがある．

3.2 nGraph-HE

準同型暗号のライブラリは C++ などで記述されることが多く，そのままではニューラルネットワークを構築するのに高度な知識が求められる．nGraph-HE [4] は準同型暗号ライブラリ (SEAL の CKKS 方式) を TensorFlow で利用可能にするグラフコンパイラとして開発された．nGraph-HE を改良した nGraph-HE2 [5] では CKKS に特化した最適化を施したり，実数を複素数にパッキングしてスループットを 2 倍にするなどの最適化をおこなっている．nGraph-HE2 [5] では，MNIST データセットを分類する CryptoNets と ImageNet データセットを分類する MobileNetV2 の評価が行われた．

4. ビット削減版準同型暗号と深層学習推論

4.1 ビット削減版準同型暗号ライブラリ

本研究で提案しているビット削減版準同型暗号ライブラリは，SEAL のバージョン 3.6.6 [2] の CKKS 方式，及び HEXL のバージョン 1.1.0 [3] を元にしている．両者とも内部の変数や演算は 64bit 整数で行われる．本ライブラリでは，ビット削減による高速化の他，SIMD 化や並列化による高速化をおこなっている．

ビット削減によりセキュリティを保ったまま適切な演算を行うにはパラメーターを適切に設定する必要があるため．ここではまず，使用するパラメータについて詳しく説明する．

CKKS 方式を利用する上で，表 1 に示すパラメータを設定する必要がある．poly_modulus は多項式の次数，すなわち，平文および暗号文の長さを決定するパラメータである．poly_modulus は 2 の冪乗の値をとる．この値によって，coeff_modulus の総数が限定される．poly_modulus が大きいほど実行時間が増加してしまうが，スロット数が増えることによるスループットの向上や level の上限が上がることによる複雑な演算への対応などが可能となる．次に coeff_modulus は，暗号文の多項式係数の法となる素数の組みを表すパラメータである．例えば，$\{29, 29, 29, 29\}$ とすると暗号文多項式の係数は 29bit 前後の素数 4 つで構成される．そのため，coeff_modulus を適切に設定することで，暗号文空間を 32 ビット以下に抑えることが可能となる．32 ビット以下の異なる素数を十分に確保できるため，準同型暗号は 32bit で実装可能である．深層学習は 16 ビットが使われることもあるが，16 ビット以下の異なる素数を十分に確保できないため，準同型暗号を 16bit で実装するのは困難である．scale は，実数を整数にスケーリングするためのパラメータである．security level は安全性を決定する値で 128bit，192bit，256bit を設定する．この値が大きいほど高い安全性を確保できるが，coeff_modulus の総数を制限する．本稿では 128bit を使用する．

表 1　CKKS 方式で設定するパラメータ

パラメータ名	概要
poly_modulus	多項式の次数
coeff_modulus	暗号文空間の法
scale	実数を整数にスケーリングするための値
security level	安全性を決める値

開発したビット削減版準同型暗号ライブラリで 32 ビットで実装されているものは，平文空間と暗号文空間の中のデータ，及び準同型暗号の各種基本処理 (encode，decode，暗号化，復号化，多項式余剰加算，多項式余剰乗算，rescale，ノイズを調整する relinearize など) を実装するための各種演算などである．なお，32 ビット同士の乗算の解は 64 ビットデータとして保持している．

本ライブラリは AVX512 命令セットによる SIMD 化も行っている．SEAL バージョン 3.6.6 は，HEXL が組み込まれており AVX512 命令セットによる SIMD 化が施されている．ビット削減版準同型暗号ライブラリーはこの HEXL を参考に，32 ビットベースの AVX512 命令で SIMD 化した．SIMD 化を行っているのは，多項式剰余乗算と多項式剰余加算，多項式剰余，数論変換 (NTT)，逆数論変換 (逆 NTT) である．これにより SIMD 演算幅が 8(=512bit/64bit) から 16(=512bit/32bit) と倍増している．

その他，本ライブラリは主に以下の 3 つの高速化を実装している：(1) rotate で必要な galois 鍵の生成関数の並列化，(2) rotate やノイズを削減する relinearize などで実行される鍵交換関数の 4 重ループになっている乗算と加算部分などの SIMD 化，(3) encode 時に配列を barrett reduction によって余剰を計算するループを SIMD 化．

4.2 深層学習推論

本稿では，ビット削減版準同型暗号ライブラリを利用した深層学習推論の実装も行った．具体的には HE-Transformer の中で利用されている SEAL を本ライブラリに置き換えることで対応している．HE-Transformer は SEAL バージョン 3.4.5 に対応しているが，本ライブラリでは SEAL バージョン 3.6.6 をベースにしているため，まず HE-Transformer を SEAL バージョン 3.6.6 に対応させた．その後，ビット削減版準同型暗号ライブラリに対応させた．nGraph-HE2 [5] では CryptoNets と MobileNetV2 を実装し評価をおこなっている．本稿では nGraph-HE2 と比較するために，同様に CryptoNets と MobileNetV2 を実装し評価をおこなった．

5. 深層学習評価

本節では，ビット削減版準同型暗号ライブラリで実装した深

層学習推論の評価結果について述べる．

5.1 評価環境

比較対象として，SEAL バージョン 3.4.5 に対応している nGraph-HE2 (以下，nGraph-HE2 (SEAL v3.4.5) と表記) と SEAL バージョン 3.6.6 に対応している nGraph-HE2 (以下，nGraph-HE2 (SEAL v3.6.6) と表記) も評価を行った．本節では 2 つのマシン上で評価を行った（表 2）．

表 2　評価環境

マシン名	項目	設定値
Intel Xeon W-2145 (Skylake-W)	動作周波数	3.7GHz
	コア数	8
	キャッシュ	Li1/L1d：32KiB / core
		L2：256KiB / core
		L3：11MiB / all cores
	メモリ	100GiB
Intel Xeon Gold 6326 (IceLake-SP)	動作周波数	2.9GHz
	コア数	16
	キャッシュ	Li1：32KiB / core
		L1d：48KiB / core
		L2：512KiB / core
		L3：24MiB / all core
	メモリ	256GiB

5.2 CryptoNets(MNIST) の評価

まず，CryptoNets を評価する上で設定したパラメータを表 3 に示す．rescale は本処理の実現に必要な最小値である $level = 7$ とした．scale は，本計算では少数点以下の値が重要なため 29 ビット (2^{29}) とした．

表 3　CryptoNets のパラメータ

パラメータ名	値
poly_modulus	2^{14}
security_level	128 bit
coeff_modulus	{29,29,29,29,29,29,29,29}
scale	2^{29}
バッチサイズ	2^{13}

5.2.1 Skylake-W での評価結果

Skylake-W で CryptoNets を評価した結果を図 1 に示す．8 スレッド利用時に，nGraph-HE2 (SEAL v3.6.6) は nGraph-HE2 (SEAL v3.4.5) の 4.02 倍の性能となった．nGraph-HE2 (SEAL v3.4.5) のベースである SEAL バージョン 3.4.5 は SIMD 化が実装されていないため，AVX512 命令セットによる SIMD 化と SEAL のバージョンアップにより性能向上でが得られた．さらに，提案手法は nGraph-HE2 (SEAL v3.6.6) に対して 2.33 倍の速度向上を得た．これは SIMD 幅 (レーン数) が 2 倍になったことと 4.1 節で述べたと独自最適化による．Skylake-W 上で nGraph-HE2 (SEAL v3.4.5) と提案手法を比較して最大の速度向上を示したのは 8 スレッドで，9.37 倍の性能向上を得た．

5.2.2 IceLake-SP での評価結果

IceLake-SP で CryptoNets を評価した結果を図 2 に示す．IceLake においてもビット削減及びコア数増加による性能向上を得ることが出来た．提案手法は 16 スレッドで，nGraph-HE2 (SEAL v3.6.6) と比較して 1.80 倍の性能向上となった．

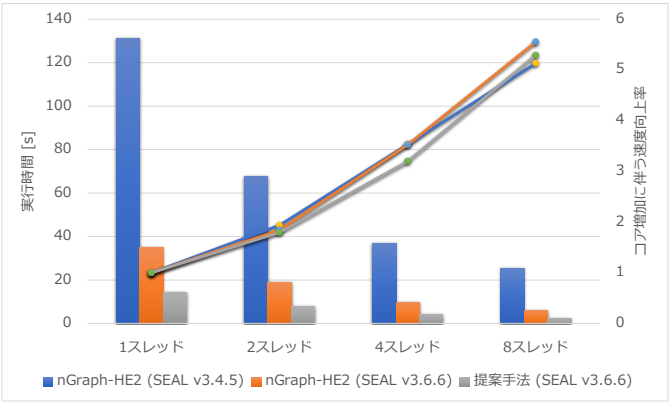

図 1　Skylake-W 上での CryptoNets の評価

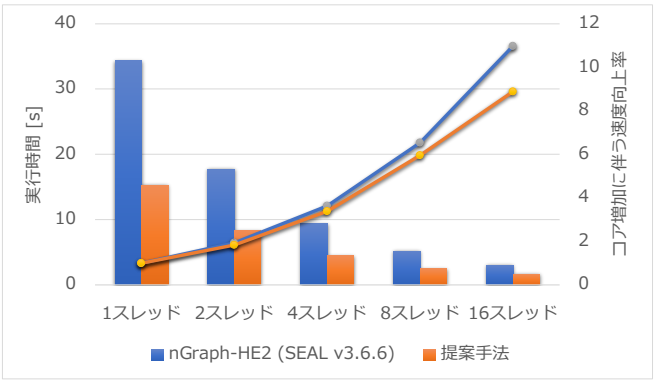

図 2　IceLake-SP 上での CryptoNets の評価

5.2.3 精度 (正解率)

提案手法が推論精度に及ぼす影響を評価した（表 4）．それぞれ同じモデルを使用したため，正確率はすべて一致した．CKKS 方式の準同型暗号を使用した場合，実数を整数にスケーリングする際に誤差が発生するが，MNIST データセットの分類において，影響は出ないことがわかる．

表 4　CryptoNets の精度 (正確率)

計算方法	正確率 [%]
暗号化しない通常計算	98.80
nGraph-HE2 (SEAL v3.4.5)	98.80
nGraph-HE2 (SEAL v3.6.6)	98.80
提案手法	98.80

5.2.4 L3 キャッシュヒット率の調査

Skylake-W で L3 キャッシュについて調査した結果を図 3 に示す．nGraph-HE2 (SEAL v3.6.6) ではスレッド増加に伴い L3 ヒット率が低下しているのがわかる．これはコンフリクトミスが原因と考えられる．Skylake-W では，L3 キャッシュの連想度は 8 スレッドで 11way しかない．MNIST データセットはデータ局所性が低く，複数のコアが同時にメインメモリの広い範囲にアクセスする傾向があるため，11way では不足となる．対して，提案手法ではビット削減により 4 スレッド，8 スレッドの

L3 ヒット率が SEAL v3.6.6 使用時に対して高くなる．以上より，メモリアクセスの点からも提案手法の有効性が確認できた．

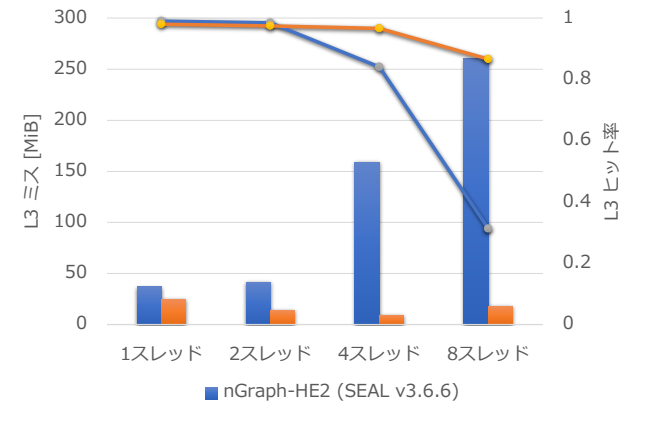

図 3　Skylake-W 上での CryptoNets の L3 キャッシュ

5.3　MobileNetV2(ImageNet) の評価

ImageNet データセットを分類する MobileNetV2 の評価結果について述べる．本評価のメモリ消費量は大きく 100GiB では不足するため，256GiB のメモリを持つの IceLake-SP でのみ評価を行った．

使用したパラメータを表 5 に示す．MobileNetV2 を実現するために必要な rescale の回数に合わせて $level = 3$ とした．

表 5　MobileNetV2 のパラメータ

パラメータ名	値
poly_modulus	2^{12}
security_level	128 bit
coeff_modulus	{29,21,21,29}
scale	2^{21}
バッチサイズ	2^{12}

評価結果を表 6 に示す．表中の model(0.35-96 や 0.35-128) に関して，前半の値は Width Multiplier のことでネットワークの厚みのような意味を持つ．後半の値は Resolution Multiplier のことで画像の解像度 (128 の時は，128×128) を意味する．

表 6　MobileNetV2 の実行時間

model	評価対象	実行時間 [s]
0.35	nGraph-HE2	264.44
- 96	提案手法	240.21
0.35	nGraph-HE2	457.90
- 128	提案手法	415.64

表 6 より本評価で提案手法により実行時間が短縮し，model(0.35-96) と model(0.35-128) 両方で約 10% の速度向上を得たことがわかる．CryptoNets の時ほどの性能向上ではないが，これは，MobileNetV2 が畳み込み層の代わりに bottleneck 層を利用することで軽量化したモデルであり，内部で準同型暗号の乗算や rescale が実行される回数が少ないのが理由と考えられる．

精度に関しては，ImageNet データセットを分類する MobileNetV2 はニューラルネットワークが大きいため，僅かに正解率に誤差が生じる．15 回計測したところ，一番低い正解率は暗号化しない時の正解率より 0.07% 低く，一番高い正解率は暗号化しない時の正解率より 0.07% 高くなった．このように誤差は小さい．

6.　富士通 A64FX での評価

本節では，ビット削減版準同型暗号を富士通 A64FX へ実装し評価した結果について述べる．

6.1　富士通 A64FX への実装

富士通が開発した A64FX プロセッサーは High Performance Computing (HPC) 向けに設計され，Armv8-A および Scalable Vector Extension for Armv8-A (SVE) 命令セットに準拠している [14]．スーパーコンピュータ富岳はこの A64FX を採用している．A64FX の仕様は表 7 の通りである．

表 7　富士通 A64FX の仕様

項目		説明
命令セット		Armv8.2-A SVE 512bit
演算コア数		48 コア
動作周波数		1.8GHz，2.0GHz，2.2GHz
キャッシュ	L1D/core	64KiB, 4way
	L2/CMG	8MiB, 16way
メモリ		HBM2 32GiB, 1024GB/s

4. 1 節で述べたビット削減版準同型ライブラリの SIMD 演算部分を SVE に対応し富士通コンパイラ（clang モード）でビルドした．使用したオプションは`-Nclang -Ofast -fopenmp` である．本稿では富岳の 1 ノードのみ使用して評価し，準同型暗号の基本処理と行列積プログラムを富士通 A64FX と Intel Xeon (Skylake-W と IceLake-SP) で比較した．

6.2　準同型演算の基本処理の評価

図 4 に準同型演算の基本処理の実行時間（マイクロ秒）を IceLake-SP（AVX512）と A64FX（SVE）で比較した結果を示す．

多くの演算で Icelake-SP の方が実行時間が短いが，これは IceLake-SP のブースト時の動作周波数が 3.5GHz と，A64FX の 2.2GHz より高く，1 スレッド実行時の性能差による．一方，図 4 の add（多項式余剰加算）及び mul（多項式余剰乗算）は A64FX の方が性能が高く，IceLake-SP に対してそれぞれ 2.1 倍，1.19 倍の性能となった．SVE には乗算の上位ビットを求める命令がある一方で，AVX512 命令セットではいくつかの命令を組み合わせてこれを実現する必要がある．これによる命令数の差が性能差の要因となっている．

6.3　行列積演算の評価

本評価で使用した行列積の実装方法は HElib のアルゴリズム [15] で提案されている手法を利用したものであり，文献 [9] の評価でも利用した．行列積の処理として，その周辺処理である encode（暗号化），rescale，及び decode（復号化）も含んでいる．A64FX，SkyLake-W，及び IceLake-SP の 3 つの環境で行列積プログラムを評価した結果を表 8 に示す．本評価では各環

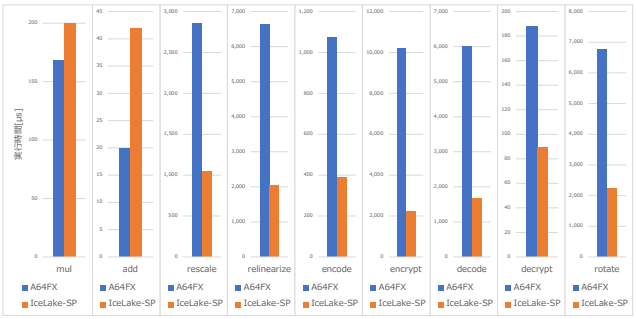

図 4　基本準同型演算の比較 (IceLake-SP vs A64FX)

境で使用可能な最大コア数で並列処理を行った。

表 8　行列積プログラムの評価結果 ($N = 2^{14}$, $L = 14$, security level=128bit)

環境	encode, 暗号化	行列積 (乗算, 加算, rotate, relinea-rize)	rescale	復号, decode	合計
SkyLake-W (8 threads)	133.3 ms	591.7 ms	5.18 ms	14.4 ms	744.6 ms
A64FX (48 threads)	156.8 ms	481.8 ms	6.64 ms	41.3 ms	686.5 ms
IceLake-SP (16 threads)	64.9 ms	318.6 ms	2.59 ms	12.9 ms	398.9 ms

表より A64FX の実行時間は 686.5ms となり，SkyLake-W の 744.6ms より速く，IceLake-SP の 398.9ms より遅いことがわかる．A64FX の実行プロファイルを確認したところ，キャッシュアクセス待ち時間が実行時間の 40% 強を占めている一方で，L1 データキャッシュミス率は 3% 程度であった．また，コンパイラが生成したアセンブリコードは多くのメモリアクセス命令が存在した．A64FX の load-to-use レイテンシは SVE ロード命令の場合で 11 cycle であり [14]，メモリアクセスが多いとパイプライン効率が低くなる可能性がある．レジスタ再利用性向上によるパイプライン効率の向上が本評価で用いたようなタイプのプログラムの性能向上では必要であると考えられる．

7. ま と め

本稿では，ビット数削減準同型暗号計算を準同型暗号ライブラリ SEAL 及びその高速化ライブラリ HEXL に実装した．これに HE-Transformer を利用することで，ビット削減版準同型暗号ライブラリーを用いた深層学習推論処理を実現した．Intel Xeon 上での評価の結果，既存手法である nGraph-HE2 と比較して最大で 9.37 倍の速度向上を得た．

また，ビット削減版準同型暗号ライブラリーを富士通 A64FX の SVE 命令セットを用いて実装評価を行った．評価の結果は，準同型暗号の基本処理において，IceLake-SP と 1 コアで比較して多項式余剰乗算は 1.19 倍，多項式余剰加算は 2.1 倍の性能を得た．その一方で，その他の基本処理や行列積では IceLake-SP に対して低い性能となり，行列積では 0.58 倍の性能となった．プロファイル結果や生成されたアセンブリの調査より，SVE 命令の利用と共にレジスタ再利用性の向上による性能向上の可能性があることを確認した．

8. 謝　　辞

本研究は，スーパーコンピュータ「富岳」の高度化・利用拡大枠利用課題「PCI を用いた機能拡張による付加価値検討」の一環として行われたものです．

文　　献

[1] C. Gentry, “Fully homomorphic encryption using ideal lattices,” Proceedings of the forty-first annual ACM symposium on Theory of computing, pp.169–178, 2009.

[2] “Microsoft SEAL (release 3.6),” https://github.com/Microsoft/SEAL, Nov. 2020. Microsoft Research, Redmond, WA.

[3] F. Boemer, S. Kim, G. Seifu, F.D. deSouza, V. Gopal, et al., “Intel HEXL (release 1.1),” https://github.com/intel/hexl, May 2021.

[4] F. Boemer, Y. Lao, R. Cammarota, and C. Wierzynski, “ngraph-he: a graph compiler for deep learning on homomorphically encrypted data,” Proceedings of the 16th ACM International Conference on Computing Frontiers, pp.3–13, 2019.

[5] F. Boemer, A. Costache, R. Cammarota, and C. Wierzynski, “ngraph-he2: A high-throughput framework for neural network inference on encrypted data,” Proceedings of the 7th ACM Workshop on Encrypted Computing & Applied Homomorphic Cryptography, pp.45–56, 2019.

[6] F. Boemer, R. Cammarota, D. Demmler, T. Schneider, and H. Yalame, “Mp2ml: A mixed-protocol machine learning framework for private inference,” Proceedings of the 15th International Conference on Availability, Reliability and Security, pp.1–10, 2020.

[7] K. Hwang and W. Sung, “Fixed-point feedforward deep neural network design using weights +1, 0, and − 1,” 2014 IEEE Workshop on Signal Processing Systems (SiPS), pp.1–6, 2014.

[8] M. Courbariaux, Y. Bengio, and J.-P. David, “Training deep neural networks with low precision multiplications,” 2014. https://arxiv.org/abs/1412.7024

[9] 宍戸哲平，西将暉，李欣怡，木村啓二，“演算ビット数削減による準同型暗号ライブラリ seal の高速化,” 電子情報通信学会技術報告，第 121 巻，pp.91–96，2022.

[10] J.H. Cheon, A. Kim, M. Kim, and Y. Song, “Homomorphic encryption for arithmetic of approximate numbers,” International Conference on the Theory and Application of Cryptology and Information SecuritySpringer, pp.409–437 2017.

[11] J.W. Cooley and J.W. Tukey, “An algorithm for the machine calculation of complex fourier series,” Mathematics of computation, vol.19, no.90, pp.297–301, 1965.

[12] R. Gilad-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 The 33rd International Conference on Machine Learning, eds. by M.F. Balcan and K.Q. Weinberger, vol.48, pp.201–210, Proceedings of Machine Learning Research, PMLR, New York, New York, USA, 20–22 Jun 2016. https://proceedings.mlr.press/v48/gilad-bachrach16.html

[13] A. Brutzkus, R. Gilad-Bachrach, and O. Elisha, “Low latency privacy preserving inference,” International Conference on Machine LearningPMLR, pp.812–821 2019.

[14] Fujitsu, “A64fx microarchitecture manual,” Sept. 2021.

[15] S. Halevi and V. Shoup, “Algorithms in helib,” Annual Cryptology ConferenceSpringer, pp.554–571 2014.