準同型暗号と隔離実行環境を用いたプライバシ保護畳み込みニューラルネットワーク

大西 隆太郎† 鈴木 拓也‡ 山名 早人§

†早稲田大学基幹理工学部 〒169-8555 東京都新宿区大久保 3-4-1

‡早稲田大学大学院基幹理工学研究科 〒169-8555 東京都新宿区大久保 3-4-1

§早稲田大学理工学術院 〒169-8555 東京都新宿区大久保 3-4-1

E-mail: †‡§{ryutaro634, t-suzuki, yamana}@yama.info.waseda.ac.jp

あらまし 近年，クラウドコンピューティングが関心を集めている．しかし，クラウド上では，ユーザデータのプライバシの侵害やプログラム及びプログラムパラメータといった知的財産の侵害が懸念されている．この懸念を払拭する技術として，データの機密性を保護する準同型暗号（以下，HE）や，データやコードの機密性・完全性を保護する隔離実行環境（以下，TEE）がある．両技術にはそれぞれ固有の欠点があり，それらを克服すべく両技術を組み合わせる試みが行われてきた．本稿では，両技術の組み合わせ方による実行パフォーマンス及びデータ保護能力の違いを明らかにすることを目的として，畳み込みニューラルネットワークの推論処理を対象に，組み合わせ方ごとの実行レイテンシの比較を実機上で行うと共に，データ保護能力を定性的に比較した．結果，「リッチ実行環境（REE）において HE 上の処理を行い，TEE 内で平文上の処理を行う組み合わせ」が，最も高い実行パフォーマンスとなり，全処理を REE 内で HE 実行する場合と比較して 50%の実行レイテンシとなることを確認した．本稿では，さらに，データ保護能力を最大限に高めることを目的に，ユーザデータに加えプログラムパラメータを準同型暗号化し TEE 内で処理を実行する手法の実行パフォーマンスを評価した．結果，上記の最も高い実行パフォーマンスを示す組み合わせと比較して，実行レイテンシが 5.5 倍となった．

キーワード プライバシ保護技術，準同型暗号，隔離実行環境，クラウドコンピューティング，Intel SGX，畳み込みニューラルネットワーク

1. はじめに

近年，低導入コストや低運用コスト，スケールの容易性を理由に，ユーザが所有するデータ（以下，UD：User Data）に対しクラウド上で処理を行うクラウドコンピューティングが関心を集めている．しかし，従来の暗号技術ではクラウド上で UD を平文で扱う必要があり，個人情報に関わるデータを扱う際には，UD へのプライバシ侵害が懸念される．また，UD を処理するプログラムを提供するパーティ（以下，プログラム提供者）が存在する場合は，プログラムやプログラムパラメータ（以下，PP：Program Parameter）の知的財産的価値への侵害も懸念される．これらの懸念を解消するためのデータ保護技術として，準同型暗号（以下，HE：homomorphic encryption）[1]と隔離実行環境（以下，TEE：trusted execution environment）[2]が挙げられる．

HE は，暗号化されたデータに対し，復号することなく演算実行を可能にする性質を持つ．そのため，HE を用いて暗号化（以下，HE 暗号化）されたデータに対して，機密性を保護しつつ計算を行うことができる．しかし，HE には，(1)平文と比較して計算コストが高い，(2)実行可能な演算が限定される，(3)暗号文のデータサイズが大きく通信コストが高い，(4)暗号文の完全性は保証しない，といった問題点がある．

TEE は，通常の環境（以下，REE：Rich Execution Environment）での実行と異なり，OS から独立した環境に配置したデータやコードの機密性や完全性を保護しつつプロセス実行を可能にする．そのため，HE と同様にクラウドコンピューティングにおけるデータ保護手段とされている．本稿で注目する Intel SGX[3]には，(1)サイドチャネル攻撃に対し脆弱である[4]，(2)割り当てられる物理メモリの制限に起因するページングオーバヘッドが存在する，(3)Intel SGX に対応したコーディングの難度が高い，といった問題点がある．

HE と TEE には，それぞれ前述した問題点がある．そこで，両技術を組み合わせ，互いの利点を残しつつ欠点を補うことで，UD とプログラム及び PP を保護しつつ効率的に処理を行うクラウドコンピューティングに関する研究が近年行われている．これまで提案された組み合わせ方は，それぞれデータ保護能力と実行パフォーマンスの間でトレードオフが存在する[5]．しかし，組み合わせ方による実行パフォーマンスの違いの定量的な比較は，これまで実施されておらず，HE と TEE の研究を進める上で，パフォーマンス差を明かにすることは重要である．

そこで，本稿では，HE と TEE の新たな組み合わせ方を提案した上で，各組み合わせ方による実行パフォーマンスを定量的に測定し，データ保護能力とのトレードオフを明らかにする．本稿の貢献を以下に示す．

- クラウドコンピューティングにおける HE と TEE の組み合わせについて，先行研究での組み合わせ方を拡張し，新たな組み合わせ方を提示すること．
- HE と TEE の各組み合わせ方について，横断的に実行パフォーマンスを測定し，データ保護能力とのトレードオフを明らかにすること．
- データ保護要件に基づいて，最適な組み合わせ方について議論すること．

本稿は以下の構成である．2 節で HE と TEE の背景知識について説明する．3 節で HE と TEE を組み合わせた関連研究について述べる．4 節で組み合わせ方の比較を行うための提案手法を説明し，5 節で評価実験について示し，結果に関する考察を 6 節で述べる．最後に，7 節で本稿のまとめを行う．

2. 背景知識

2.1. 準同型暗号（HE）

HE が実行できる演算は加算と乗算であり，それぞれ準同型加算，準同型乗算と呼ばれる．

HE には複数の種類がある．まず，準同型加算のみもしくは準同型乗算のみをサポートする部分 HE である．これに対し，任意回数の準同型加算と準同型乗算を実行可能なものを完全準同型暗号（以下，FHE：fully HE）と呼び，2009 年に Gentry ら[1]によって提案された．FHE では，HE 暗号化されたデータにノイズが含まれており，準同型演算が行われるたびにノイズが増加する．ノイズ量が閾値を超えると正常に復号できなくなるため，ノイズを削減するために Bootstrapping という処理が導入されているが，実行時間が長く，使用した際にパフォーマンスの大きな低下を招く．

対して，Bootstrapping を用いず，準同型演算の回数に制限を設ける Leveled HE があり，複数回の準同型加算と事前に設定したパラメータに依存した一定数回（レベル）の準同型乗算をサポートする．Leveled HE には整数上での演算が可能な BGV 方式，固定小数点数を扱える CKKS 方式[6]などがある．特に，CKKS 方式は固定小数点数を扱える性質を持ち，畳み込みニューラルネットワーク（以下，CNN：convolutional neural network）や機械学習に適している．CNN や機械学習は HE の適用先として有望視されている[7]ため，本稿では CKKS 方式を用いる．

クラウドコンピューティングに HE を適用すると，データの機密性を保護できる一方で，本稿で用いる CKKS 方式では，以下の欠点が存在する．

1. 加算・乗算以外の演算をサポートしない．
2. 悪意を持って書き換えられた計算結果を本来の計算結果と区別できない．
3. HE 上の演算は処理時間が長い[1]．
4. 平文と比較して，暗号文はデータサイズが$10\sim10^6$倍程度大きく[5]，通信コストが高い．

2.2. 隔離実行環境（TEE）

現在利用可能な TEE には，例えば，Intel SGX や Arm TrustZone，AMD SEV などが存在する．HE と組み合わせた先行研究では，Intel SGX が多く用いられてきたため，本稿でも Intel SGX を用いる．Intel SGX では，メモリ上の保護領域をエンクレーブと呼ぶ．エンクレーブにはデータやコードは暗号化された状態で格納され，プロセス実行の際は CPU 上で復号した上で演算を行う．Intel SGX では，データやコードの機密性及び完全性を保護できる点や，CPU 上の計算は平文で行うため，HE と比較して計算コストが低い点や任意の演算を実行できる点が長所である．また Intel SGX では公式の開発用ライブラリである SGX SDK が用意されている．

しかし，Intel SGX には以下の欠点が存在する．

1. コンピュータの物理的特性から内部情報を取得するサイドチャネル攻撃に対し脆弱であり[4]，データやコードの機密性侵害のリスクがある．
2. エンクレーブに割り当てられるメモリサイズに制限があることに起因するページングオーバヘッドのために，スケーラビリティが低い[2]．
3. エンクレーブ内でシステムコールを使用できないといった制約があるため，Intel SGX に対応するコーディングの難度が高い．

3 の問題点に関しては，実行バイナリを修正することなく Intel SGX 上で実行できるようにする Gramine-SGX[3]というシステムを用いることで回避することが可能であり，本稿でも Gramine-SGX を用いる．Gramine-SGX では，マニフェストと呼ばれる，実行バイナリやエンクレーブ内に読み込む必要のあるファイル，さらにエンクレーブのサイズ等を指定するためのテキストファイルを用意する．

3. 関連研究

本節では，筆者らの CSS2022 の論文[5]に基づき，TEE 内 HE 実行型，REE 内 HE 実行型， REE 内 HE 実行＆TEE 内平文実行型の三つの組み合わせ方について

[1] 平文上の演算と比較して，準同型加算で$10\sim10^5$倍，準同型乗算で$10\sim10^7$倍程度である[5]．

[2] ただし，第三世代の Intel Xeon Scalable Processor ではメモリ制限が 8~512GB に緩和されている．(Intel, “第 3 世代インテル® Xeon® スケーラブル・プロセッサー・ファミリーの概要,” https://www.intel.co.jp/content/www/jp/ja/products/docs/processors/xeon/3rd-gen-xeon-scalable-processors-brief.html, (参照 2023-01-06).)

[3] “Introduction to Gramine,” Gramine SGX - Read the Docs, https://gramine.readthedocs.io/en/stable/, (参照 2022-12-30).

説明する．

3.1. TEE 内 HE 実行型

TEE 内 HE 実行型では，TEE で HE 上の処理を行う．TEE 内 HE 実行型の流れを図 1 に示す．ユーザとプログラム提供者，クラウドサーバの 3 パーティが存在し，プログラム提供者がまずクラウドサーバ上の TEE を初期化した上で，TEE に UD に対する処理を行うためのプログラムをセットする．ユーザは HE 鍵を生成し，公開鍵を用いて UD を暗号化し，クラウドサーバ上の TEE へ送る．TEE で HE 上の処理が行われた後，暗号化された処理結果がユーザに返され，ユーザは秘密鍵を用いて処理結果を復号し得る．

本実行型の目的は，TEE で処理を行うことによる計算過程の完全性の保護と，HE による UD の機密性の保護である．その一方で，欠点として計算コストが増加する点が挙げられる．

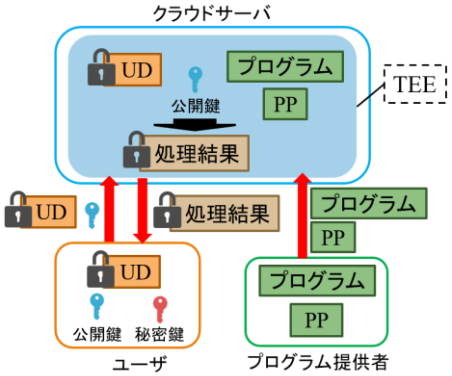

図 1 TEE 内 HE 実行型の流れ

3.2. REE 内 HE 実行型

この組み合わせ方では，REE において HE 上の処理を行う．REE 内 HE 実行型の流れを図 2 に示す．ユーザとプログラム提供者，クラウドサーバの 3 パーティが存在し，まずクラウドサーバ上の TEE において HE 鍵が生成される．公開鍵が，クラウドサーバ上の REE やユーザ，プログラム提供者に渡される．ユーザとプログラム提供者は，それぞれ公開鍵を用いて UD と PP を暗号化し，クラウドサーバ上の REE に送る．クラウドサーバでは REE において HE 上の処理が行われた後，暗号化された処理結果が TEE に送られ，秘密鍵を用いて復号される．そして，復号された処理結果が TEE からユーザへ送られ，ユーザは処理結果が得る．

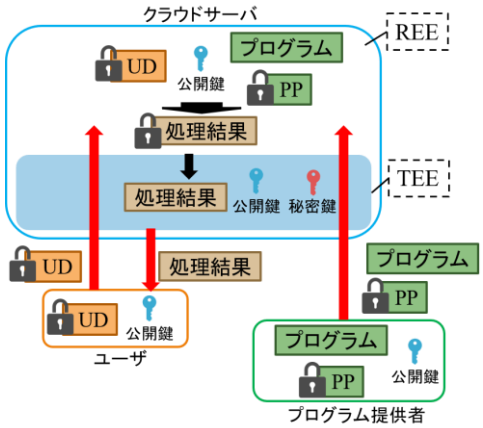

図 2 REE 内 HE 実行型の流れ

REE 内 HE 実行型の目的は，HE 鍵を TEE で生成することによる鍵管理用の信頼サーバの排除と HE 暗号化による UD 及び PP の機密性の保護である．その一方で，UD に対する処理は REE で行うことから計算過程の完全性は保証しない点が欠点として挙げられる．

3.3. REE 内 HE 実行＆TEE 内平文実行型

この組み合わせ方では，REE 内での HE 上の処理実行と，TEE での中間暗号文を復号し平文上の処理実行を組み合わせて処理を行う．REE 内 HE 実行型＆TEE 内平文実行型の流れを図 3 に示す．ユーザとクラウドサーバの 2 パーティが存在し，まずクラウドサーバ上の TEE において HE 鍵が生成される．公開鍵が，クラウドサーバ上の REE やユーザに渡される．ユーザは公開鍵を用いて UD を暗号化し，クラウドサーバ上の REE に送る．REE では HE 上で処理が行われるが，比較等の HE では対応できない処理は，一度 TEE へ中間暗号文を送り，復号した上で平文上の処理を行う．TEE での処理後は再度 HE 暗号化を施した上で REE に暗号文を送り処理を進める．最後に処理結果の暗号文は，TEE 内で復号された上でユーザへ送信され，ユーザは処理結果を得る．

REE 内 HE 実行＆TEE 内平文実行型の目的は，REE と TEE で処理分担することによる処理の高速化及び推論精度の向上である．その一方で，REE における計算過程の完全性については保証されない点が欠点として挙げられる．

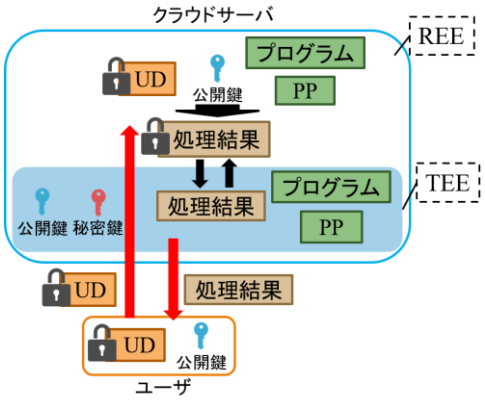

図 3 REE 内 HE 実行＆TEE 内平文実行の流れ

3.4. サイドチャネル攻撃

REE と同様に TEE は，サイドチャネル攻撃に対して脆弱であり，データを平文の状態で扱うことは，データの機密性の侵害リスクにつながる．そこで，データ

を HE 暗号化し処理することで，データに対するサイドチャネル攻撃を回避できることが期待される．各組み合わせ方で，HE 暗号化によって機密性が保護されるデータと保護されないデータを表 1 にまとめる．

表 1 HE によるサイドチャネル攻撃に対するデータ機密性保護

組み合わせ方	保護対象	保護対象外
TEE 内 HE 実行	UD，中間結果，PP*	PP
REE 内 HE 実行	UD，PP*，中間結果	鍵関連情報，PP
REE 内 HE 実行＆TEE 内平文実行	UD，REE 内の中間結果，PP*	TEE 内の中間結果，PP

（*は HE 暗号化した場合を示す）

3.5. 関連研究のまとめ

本節で示した三つの組み合わせ方において，データ保護能力と実行パフォーマンスはトレードオフの関係にある．データ保護の観点から比較すると，TEE 内 HE 実行＞REE 内 HE 実行＞REE 内 HE 実行＆TEE 内平文実行の順で優れていると考えられる．また，精度やレイテンシといったパフォーマンスの観点から比較すると REE 内 HE 実行＆TEE 内平文実行＞REE 内 HE 実行＞TEE 内 HE 実行の順で優れていると考えられる．

4. 組み合わせ方の比較手法の提案

4.1. 本研究の目的

3 節で示した通り，これまで様々な組み合わせ方が提案されてきた．しかし，組み合わせ間の性能比較が同一環境では行われていない．そこで，組み合わせごとのデータ保護能力と実行パフォーマンスのトレードオフを明らかにすることが本研究の目的である．

4.2. 本研究が対象とする HE と TEE の組み合わせ方とデータ保護能力

まず，HE 暗号化の適用有無に関して，表 2 に示す 4 つのパターンが存在する．さらに，HE 暗号化適用パターンのそれぞれについて，TEE 内と REE 内のどちらで実行するかの 2 パターンが存在する．ただし，本稿では，PP のみを HE 暗号化するパターンを除外する．なぜなら，CNN の推論処理において，PP のみを暗号化した場合，一層目畳み込み層を除いて各層の入力は暗号文となるため，それらの層は REE 内 UD・PP 暗号文型と同等になり，測定の必要性が低いためである．したがって，本稿では，表 3 に示すように 6 通りの組み合わせ方を想定する．そして，この 6 通りの組み合わせ方のそれぞれのデータ保護能力を表 3 にまとめる．

各層によって適用するパターンを変更する場合，REE-TEE 間のデータ転送や，TEE 内でのデータの復号及び暗号化といった処理が発生する場合がある．これらのオーバヘッドに関しては 5 節の評価実験にて示す．

表 2 HE 暗号化適用パターン

		PP の HE 暗号化	
		無	有
UD の HE 暗号化	無	UD・PP 平文型	（対象外）
	有	UD 暗号文型	UD・PP 暗号文型

表 3 HE と TEE の各組み合わせのデータ保護能力（△はサイドチャネル攻撃を考慮しない場合は保護できることを示す）

組み合わせ方	UD の機密性	PP の機密性	計算処理の完全性
REE 内 UD・PP 平文	—	—	—
REE 内 UD 暗号文	✓	—	—
REE 内 UD・PP 暗号文	✓	✓	—
TEE 内 UD・PP 平文	△	△	✓
TEE 内 UD 暗号文	✓	△	✓
TEE 内 UD・PP 暗号文	✓	✓	✓

5. 評価実験

5.1. 概要

HE と TEE の組み合わせ方の実行パフォーマンスを測定するために，クラウドコンピューティングにおけるキラーアプリケーションである CNN の推論処理を評価対象とした．実験では，まず 4.2 項で示した各組み合わせ方の CNN の推論レイテンシの測定を実施した（5.5 項）．さらに，処理ごとに適用する組み合わせ方を変える場合のオーバヘッドを測定するために，REE と TEE 間のデータ転送レイテンシの測定（5.6 項）と TEE 内での HE 暗号化及び復号レイテンシの測定（5.7 項）を実施した．最後に柔軟に組み合わせ方の比較を行うために暗号文レベルを 1 とした時のレイテンシの測定（5.8 項）を実施した．

5.2. データセット

CNN の推論対象として，MNIST [8]データセットを用いた．MNIST データセットは，手書きの数字の画像データセットであり，それぞれ 28×28 ピクセルの 8bit グレースケール形式で，0 から 9 までの 10 個の整数のクラスのいずれかに分類される．MNIST データセットは，60,000 枚の学習用画像と 10,000 枚のテスト用画像で構成される．

5.3. CNN のネットワーク構成

評価実験で用いる CNN のネットワーク構成は，Ishiyama ら[9]と同一で，構成を表 4 に示す．活性化関数には ReLU 関数を用いた．暗号文上での推論では ReLU 関数を二次の近似多項式を用いた．Ishiyama ら[9]は，HE 上の演算回数を削減し，処理を高速化する

表 4 ネットワーク構成（[9]に基づく）

層	層の説明	入出力サイズ
畳み込み 1	フィルタ数:5，サイズ:5×5 ストライド:(2,2)，パディング:なし	入力:28×28×1，出力:12×12×5
バッチノーマライゼーション	畳み込み 1 に融合される	入力:12×12×5，出力:12×12×5
活性化 1	ReLU 関数 (暗号文上では二次の近似多項式)	入力:12×12×5，出力:12×12×5
畳み込み 2	フィルタ数:50，サイズ:5×5 ストライド:(2,2)，パディング:なし	入力:12×12×5，出力:4×4×50
バッチノーマライゼーション	畳み込み 2 に融合される	入力:4×4×50，出力: 4×4×50
活性化 2	ReLU 関数 (暗号文上では二次の近似多項式)	入力:4×4×50，出力:4×4×50
全結合	10 個のニューロンへの加重和を計算	入力:4×4×50，出力:1×1×10

ために，バッチノーマライゼーション層の畳み込み層への融合と，活性化関数の近似多項式の二次の項の係数を他の層の係数に乗算するという戦略を取っており，本稿でも同様の戦略を採る．また，HE には，複数の平文データを一つの暗号文データにまとめ，SIMD 形式の計算を行えるパッキングという性質があり，本研究では Ishiyama ら[9]と同様にパッキングを用いて一枚の入力画像を一つの暗号文に暗号化した．

5.4. 実験環境及び HE パラメータ

評価実験で用いた計算機の構成を表 5 に示す．

表 5　実験に使用した計算機の構成

CPU	Intel Xeon Gold 6334 (3.60GHz)
Hyper-threading	有効
コア数	16
メモリサイズ	128GB
OS	Ubuntu 20.04.5 LTS
カーネルバージョン	5.4.189
g++バージョン	9.4.0
PRM サイズ	2GB

評価実験は全てシングルスレッドで行い HE ライブラリは，Microsoft SEAL（バージョン 3.6.6）[4]を用いた．HE 方式は，固定小数点を扱える CKKS 方式を採用している．SEAL を利用した際の HE パラメータに関して，poly_modulus_degree は 16,384，スロット数は 8,192，Scale factor は 30，最大初期レベルは 5 としており，評価実験では最大初期レベル以外は同一のパラメータを用いた．

5.5. CNN の推論レイテンシ

5.5.1. 実験目的

条件を様々変えた上で CNN モデルの各層における実行レイテンシを測定し，複数の組み合わせ方の推論レイテンシの比較を行うことを目的とする．

5.5.2. 実験方法

MNIST データセット内の 20 枚の画像に対し，1 枚ごとに推論処理を行い，そのレイテンシの平均を求めた．測定対象は，画像が CNN モデルに入力されてから結果が出力される間に，各層での処理に要したレイテンシとする．TEE 内 UD・PP 平文型，TEE 内 UD 暗号文型，TEE 内 UD・PP 暗号文型では TEE で評価を行うため Gramine-SGX を用いた．

5.5.3. 実験結果

表 6 に，各パターンにおける推論レイテンシを示す．

5.6. REE-TEE 間のデータ転送レイテンシの評価

5.6.1. 実験目的

REE と TEE の両方で UD に対する処理を行う場合に

表 6 CNN 推論レイテンシ

層	各パターンにおけるレイテンシ [ms]					
	REE 内 UD ・PP 平文	REE 内 UD 暗号文	REE 内 UD ・PP 暗号文	TEE 内 UD ・PP 平文	TEE 内 UD 暗号文	TEE 内 UD ・PP 暗号文
畳み込み 1	3.97	22,989	35,008	6.31	29,897	46,048
バッチノーマライゼーション						
活性化 1	0.04	3,025	5,388	0.07	3,684	6,473
畳み込み 2	201.79	341,700	663,661	324.71	488,113	978,755
バッチノーマライゼーション						
活性化 2	0.74	15,206	26,625	1.08	18,415	32,211
全結合	5.81	14,502	15,917	7.34	17,862	19,650
全体	212.36	397,422	746,599	339.51	557,972	1,083,137

[4] Microsoft, “Microsoft SEAL (release 3.6),” https://github.com/microsoft/SEAL, (参照 2022-12-30).

は，REE-TEE 間でデータの転送処理が発生する．本実験では，REE-TEE 間のデータ転送レイテンシを測定し，そのオーバヘッドを考慮して組み合わせの比較を行えるようにすることを目的とする．

5.6.2. 実験方法

REE-TEE 間のデータ転送レイテンシの測定では SGX SDK（バージョン 2.18）を用いた．SGX SDK を用いたプログラムでは，REE で実行するコードと TEE で実行するコードがある．まず REE 側から TEE の初期化を行った後，TEE 側で行いたい処理を REE 側から関数として呼び出す．この関数呼び出しを ECALL という．対して，TEE 側から REE 側の関数の呼び出すことを OCALL と呼び．OCALL は ECALL によって呼び出された関数内で行われる．

ECALL と OCALL を用いる際に，関数の引数として様々なサイズの unit64_t 型配列を与えられる．そこで，ECALL によって呼び出した関数の実行時間を REE→TEE 方向のデータ転送レイテンシ，OCALL によって呼び出した関数の実行時間を TEE→REE 方向のデータ転送レイテンシとして測定した．

unit64_t 型配列のデータサイズは，レベル 0 から 5 の暗号文のサイズである 256[KiB]から 1,536[KiB]までとした．なお，暗号文のデータサイズは式(1)で求まる．また，結果は 400 回測定した内の 201～400 回目の測定値の平均とした．これは，試行回数が少ない段階では測定値が大きくなることから，測定値が安定した後のデータに基づいて議論するためである．

$$\text{size} = 2 \times \text{poly modulus degree} \times (\text{レベル} + 1) \times 8[\text{Byte}] \quad (1)$$

5.6.3. 実験結果

評価結果を表 7 に示す．なお TEE→REE のレイテンシは，測定時のオーバヘッドを除いた値とする．

表 7 REE-TEE 間のデータ転送レイテンシ

データサイズ [KiB]	データ転送レイテンシ[μs]	
	REE→TEE	TEE→REE
256	23.39	14.02
512	43.68	44.32
768	78.61	71.32
1,024	139.53	99.54
1,280	168.06	123.87
1,536	186.25	155.67

5.7. TEE 内での HE の暗号化及び復号レイテンシ

5.7.1. 実験目的

REE 内 HE 実行&TEE 内平文実行型のように HE 暗号化していたデータを TEE で復号し，平文上での処理後，再度暗号化を行う場合には， TEE での HE 暗号化及び復号オーバヘッドを考慮する必要がある．本実験の目的は， TEE での HE 暗号化及び復号のレイテンシを測定し，組み合わせ方の比較に役立てることである．

5.7.2. 実験方法

HE 暗号化の際は，double 型配列をエンコードした上で，HE 上の暗号文に暗号化する必要がある．復号も同様に，暗号文を復号した後，デコードし double 型配列に変換する．本実験では要素数が 8,192 の double 型配列を 100 個用意し，各配列に対しエンコード&暗号化及び復号&デコードを行い，その平均レイテンシを求めた．double 型の配列を用いた理由は，SEAL では float 型の配列に対応していないためである．復号&デコードはレベル 0 の暗号文に対して行った．これは，デコード及び復号する際の暗号文のレベルは 0 とするのが最も計算量が小さくなるためである．また，各層で消費されるレベルは 1 であり，少なくともレベル 1 で暗号化を行う必要があるため，エンコード&暗号化はレベル 1 を対象として実験を行った．

5.7.3. 実験結果

TEE での HE 上のエンコード&暗号化及び復号&デコードのレイテンシの評価結果を表 8，表 9 に示す．

表 8 TEE 内エンコード&暗号化レイテンシ

暗号文のレベル	エンコード[ms]	暗号化[ms]
1	5.62	155.54

表 9 TEE 内復号&デコードレイテンシ

暗号文のレベル	復号[ms]	デコード[ms]
0	11.17	2.92

5.8. CNN の各層における各レベルでのレイテンシ

5.8.1. 実験目的

CKKS 方式では，暗号文のレベルが高いほどレイテンシが長くなる．したがって， TEE 内 UD・PP 平文型と他の組み合わせ方を併用する場合，TEE 内で暗号文を復号するために，要求レベルを削減できる．そこで，各層への入力暗号文のレベルを 1 とした時の各層でのレイテンシを測定し，HE と TEE の組み合わせ方の比較をより柔軟に行うことを本実験の目的とする．

5.8.2. 実験方法

MNIST データセット内の 20 枚の画像に対し，1 枚ごとに推論処理を行い，その平均レイテンシを求めた．CNN の各層の処理後に復号及び暗号化処理を追加することで暗号文のレベルをリセットした．消費されるレベルは各層で共通して 1 であるため，初期レベルは 1 とした．測定対象の組み合わせ方は，REE 内 UD 暗号文型，REE 内 UD・PP 暗号文型，TEE 内 UD 暗号文型，TEE 内 UD・PP 暗号文型の 4 つである．

5.8.3. 実験結果

CNN の各層の実行レイテンシを表 10 に示す．

表 10 レベル 1 での各層の実行レイテンシ

層	各パターンにおけるレイテンシ [ms]			
	REE 内 UD 暗号文	REE 内 UD・PP 暗号文	TEE 内 UD 暗号文	TEE 内 UD・PP 暗号文
畳み込み 1	5,923	9,564	7,601	12,369
活性化 1	914	1,588	1,122	1,924
畳み込み 2	166,154	325,802	239,673	477,937
活性化 2	9,137	15,896	11,296	19,196
全結合	14,562	15,958	18,121	19,678

6. 考察

6.1. 既存の組み合わせ方のレイテンシ比較

先行研究における組み合わせ方と本稿で提案する組み合わせ方の対応を以下に示す.

1. TEE 内 HE 実行型：TEE 内 UD 暗号文型
2. REE 内 HE 実行型：REE 内 UD・PP 暗号文型
3. REE 内 HE 実行＆TEE 内平文実行型：REE 内 UD 暗号文型と TEE 内 UD・PP 平文型の融合

REE 内 HE 実行＆TEE 内平文実行型の CNN の処理への適用に関して，畳み込み層や全結合層では REE 内で HE 上の処理を，活性化層では中間結果を TEE 内に転送後暗号文の復号，平文上の処理，暗号化を行う場合の推論レイテンシを考える.

まず，REE-TEE 間のデータ転送レイテンシについて. 活性化 1 ではレベル 0 の暗号文が 5 個，活性化 2 ではレベル 0 の暗号文 50 個，REE から TEE へ転送される. 対して，活性化層の処理後は，活性化 1 ではレベル 1 の暗号文が 5 個，活性化 2 ではレベル 1 の暗号文が 50 個，TEE から REE へ転送される. 式(1)より，レベル 0 の暗号文のサイズは 256[KiB]，レベル 1 の暗号文のサイズは 512[KiB]であることから，表 7 より転送レイテンシの合計は高々3.963[ms]となる.

次に TEE 内における復号及び暗号化処理のレイテンシについて. 活性化 1 での処理前はレベル 0 の暗号文を 5 個，活性化 2 での処理前はレベル 0 の暗号文を 50 個，全結合の後にはレベル 0 の暗号文を 10 個デコード及び復号する. また，活性化 1 での処理後は配列を 5 個，活性化 2 での処理後は配列を 50 個，レベル 1 の暗号文へのエンコード及び暗号化を行う. よって，表 8，表 9 より合計でレイテンシは 9,780[ms]となる. また，初期レベルを 1 とした場合の REE 内 UD 暗号文型の畳み込み層や全結合層の処理時間に関しては表 10 を，TEE 内 UD・PP 平文型の活性化層の処理時間に関しては表 6 を参照すると，REE 内 HE 実行＆TEE 内平文実行型の CNN 推論レイテンシは 196,424[ms]と求められる. 対して，表 6 より，最後の処理結果の復号レイテンシを含めると TEE 内 HE 実行型の CNN 推論レイテンシは 558,112[ms]，REE 内 HE 実行型の CNN 推論レイテンシは 746,740[ms]である. よって，3.5 項で述べた予想と異なり，実行パフォーマンスは REE 内 HE 実行＆TEE 内平文実行型＞TEE 内 HE 実行型＞REE 内 HE 実行型の順で優れていることが分かった.

REE 内 HE 実行＆TEE 内平文実行型が最も優れている理由としては，活性化層の処理を平文上で実行する点に加え，その際に暗号文のレベルをリセットするため，他の組み合わせより要求される暗号文のレベルを削減できる点が挙げられる. また，REE 内 HE 実行＆TEE 内平文実行型は TEE を一切用いない REE 内 UD 暗号文型と比較して，実行レイテンシが 50%になる.

次に，TEE 内 HE 実行型が REE 内 HE 実行型よりもレイテンシが短かった理由としては，実験で用いた計算機において，TEE に割り当てられるメモリが従来よりも増大したため，TEE 内 HE 実行型の懸念点であったページングオーバヘッドが改善された点がまず考えられる. また，両パターンにおいて最もレイテンシが長い処理は畳み込み 2 である. 畳み込み 2 のレイテンシを REE 内 UD 暗号文型と比較すると，TEE 内 HE 実行型では 1.43 倍，REE 内 HE 実行型では 1.94 倍であったため，畳み込み 2 における暗号文同士の演算のコストが処理全体のレイテンシに大きく影響することが分かる. ただし，REE 内で処理を行う場合では，TEE 内での処理と異なり GPU 等の豊富な計算資源を活用でき，REE 内 HE 実行型のパフォーマンスが TEE 内 HE 実行型を上回ることが期待できる.

筆者らの CSS2020 の論文[5]では言及していない組み合わせ方である TEE 内 UD・PP 暗号文型は，他の組み合わせ方と比較してサイドチャネル攻撃があっても UD や PP の機密性を保証し，また処理の完全性も保証するため，データ保護能力が最も高い. しかし，推論レイテンシが最も長く，REE 内 HE 実行＆TEE 内平文実行型のレイテンシの算出値と比較すると 5.5 倍であった.

6.2. CNN 推論を対象とした組み合わせ方の候補

CNN モデルの層の中で，学習過程によって得られるパラメータを含むのは畳み込み層と全結合層であり，活性化層では，学習用データや学習結果に関わらず，予め指定された活性化関数（ReLU 関数や Sigmoid 関数など）や，暗号文上ではその近似多項式を用いて計算を行う. プログラム提供者の知的財産保護の観点から検討すると畳み込み層や全結合層における PP は保護対象となるが，活性化関数の近似多項式の係数に関する情報は保護対象外と考えられる. そこで，データ

保護要件を考慮した CNN の層ごとの計算処理パターン適用案を表 11，表 12 に示す．

表 11 畳み込み層，全結合層パターン適用案

		サイドチャネル攻撃	
		想定有	想定無
完全性侵害	保護有	TEE 内 UD・PP 暗号文	TEE 内 UD・PP 平文
	保護無	REE 内 UD・PP 暗号文	TEE 内 UD 暗号文

表 12 活性化層パターン適用案

		サイドチャネル攻撃	
		想定有	想定無
完全性侵害	保護有	TEE 内 UD 暗号文	TEE 内 UD 平文
	保護無	REE 内 UD 暗号文	REE 内 UD 暗号文

表 11 は畳み込み層及び全結合層において UD と PP を保護し，表 12 は活性化層において UD を保護することが前提となっている．表 11，表 12 を基に，本稿の CNN モデルの層ごとにパターンを適用した場合の推論レイテンシの算出結果を表 13 に示す．

表 13 適用時の推論レイテンシ

		サイドチャネル攻撃	
		想定有	想定無
完全性侵害	保護有	1,066,692[ms]	340[ms]
	保護無	732,958[ms]	561,345[ms]

HE 上の処理が含まれる場合，さらに各層の処理後に TEE で復号及び暗号化を行い，要求される暗号文の初期レベルを 1 にすることで，TEE 内で中間結果を平文にするリスクを甘受する代わりに，レイテンシの削減が期待できる．その時の CNN の推論レイテンシの算出結果を表 14 に示す．ただし，全体で REE-TEE 間のデータ転送レイテンシを含めた値とする．

表 14 TEE 内復号及び暗号化を行った場合の推論レイテンシ

		サイドチャネル攻撃	
		想定有	想定無
完全性侵害	保護有	541,820[ms]	340[ms]
	保護無	380,801[ms]	294,868[ms]

表 14 より，TEE における復号及び暗号化処理によって CNN の推論レイテンシは表 13 の値と比較して，47～49%の削減が見込めることが分かる．

7. おわりに

本稿では，クラウドコンピューティングにおける，HE と TEE の複数の組み合わせ方について議論した．CNN の推論レイテンシに加え，その他予想されるオーバヘッドの測定し，組み合わせ方の比較を行った結果，同一環境での実行パフォーマンスは REE 内 HE 実行＆TEE 内平文実行型＞TEE 内 HE 実行型＞REE 内 HE 実行型の順で優れていることが分かった．また，TEE 内 UD・PP 暗号文型は，データ保護能力が高い反面，CNN の推論レイテンシでは REE 内 HE 実行＆TEE 内平文実行型の 5.5 倍となることが分かった．最後に，TEE による暗号文のレベルのリセットによりレイテンシが 47～49%短縮されることが示された．

本稿では，層ごとに処理を分割して推論レイテンシを測定した．そのため，今後の課題は，実際に REE と TEE の間でデータ転送を行い，処理を実行できるシステムを構築し，評価することである．

参 考 文 献

[1] C. Gentry, “Fully Homomorphic Encryption Using Ideal Lattices,” in Proceedings of the 41 annual ACM symposium on Theory of computing, pp. 169‐178, 2009.

[2] GlobalPlatform, “Introduction to Trusted Execution Environment,” https://globalplatform.org/wp-content/uploads/2018/05/Introduction-to-Trusted-Execution-Environment-15May2018.pdf, (参照 2022-12-30).

[3] F. McKeen, I. Alexandrovich, A. Berenzon, C. V. Rozas, H. Shafi, V. Shanbhogue, and U. Savagaonkar, “Innovative instructions and software model for isolated execution,” in Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy, article no. 10, pp. 1‐8, 2013.

[4] N. Alexander, P. N. Bideh, and J Brorsson, “A survey of published attacks on Intel SGX,” arXiv preprint arXiv:2006.13598, 2020.

[5] 大西隆太郎，鈴木拓也，山名早人，“準同型暗号と隔離実行環境の組み合わせに関するサーベイ，” コンピュータセキュリティシンポジウム 2022(CSS2022)，2A2-I-1，pp.1-8，2022.

[6] J. H. Cheon, A. Kim, M. Kim, and Y. Song, “Homomorphic encryption for arithmetic of approximate numbers,”iIn Proceedings of International conference on the theory and application of cryptology and information security, pp. 409‐437, 2017.

[7] A. Wood, K. Najarian, and D. Kahrobaei, “Homomorphic Encryption for Machine Learning in Medicine and Bioinformatics,” ACM Computing Surveys, vol. 53, issue. 4, Article 70 (July 2021), pp. 1–35, 2020.

[8] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, 86 (11) (1998), pp. 2278-2344, 1998.

[9] 石山琢己，鈴木拓也，山名早人，“準同型暗号上での畳み込みニューラルネットワーク推論に対する Channel Pruning の適用，” 第 14 回データ工学と情報マネジメントに関するフォーラム (DEIM2022)，J33-4, 2022.