intel software

THE PARALLEL UNIVERSE

インテルの新しい組込み AI アクセラレーション・エンジンを使用する

インテル® CPU における高速化された AI の新時代の到来

我々がどこまで辿り着いたか振り返る

Issue
55
2024

目次

編集者からのメッセージ 3

注目記事

インテル® CPU における高速化された AI の新時代の到来 5
Numenta は CPU 上での AI 推論を高速化する神経科学ベースのソリューションでインテルと協力

インテルの新しい組込み AI アクセラレーション・エンジンを使用する 12
インテル® AMX とインテル® XMX で AI の効率、スケーラビリティー、パフォーマンスを向上

我々がどこまで辿り着いたか振り返る 20
インテル® ソフトウェア開発ツールと oneAPI

Unified Acceleration（UXL）Foundation 24
アクセラレーター向けのオープン・スタンダード・プログラミングのコミュニティーを統合

BigDL プライバシー保護マシンラーニングを使用したトラスト TorchServe 28
インテル® トラスト・ドメイン・エクステンションズとコンフィデンシャル・コンテナで AI をセキュアに

ベイズ・ディープラーニングの量子化フレームワーク 36
Bayesian-Torch でベイズ・ニューラル・ネットワークを簡単に量子化してデプロイ

最先端のビジュアル品質検査 43
インテルにより最適化されたエンドツーエンドのモジュール式リファレンス・キットを使用した効率的な異常検出

Spack を使用してインテルにより最適化された HPC バイナリーのディストリビューション 49
重要な HPC アプリケーションに最適な実行ファイルを構築する Spack レシピ

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

編集者からのメッセージ

Henry A. Gabb インテル コーポレーション シニア主席エンジニア
HPC と並列コンピューティング分野で長年の経験があり、並列プログラミングに関する記事を多数出版しています。『Developing Multithreaded Applications : A Platform Consistent Approach』の編集者 / 共著者で、インテルと Microsoft による Universal Parallel Computing Research Centers のプログラム・マネージャーを務めました。

新しい年、再び oneAPI が受賞

The Parallel Universe のスタッフを代表して、読者の皆さんに新年のご挨拶を申し上げます。

インテル® ソフトウェア開発ツールのチームは、2023 年も HPCwire Readers' Choice Award の「ベスト HPC プログラミング・ツール / テクノロジー」（英語）を受賞しました。この賞は、oneAPI 仕様またはインテル® oneAPI ツールキットが発表以来受賞している数多くの賞の 1 つです。本誌の名誉編集者である James Reinders は、「**我々がどこまで辿り着いたか振り返る：インテル® ソフトウェア開発ツールと oneAPI**」で彼の見解を述べています。前号では、Unified Acceleration Foundation（英語）が設立されたことをお知らせしました。本号では、このことが oneAPI にとって何を意味するかについて、Codeplay Software エコシステム担当副社長および UXL Foundation ステアリング・コミッティー委員長の Rod Burns が「**Unified Acceleration（UXL）Foundation**」で詳しく説明します。

本号の注目記事として、Numenta（英語）に、同社の NuPIC* プラットフォームがどのようにインテル® AMX を活用して AI 推論を行っているかを説明する記事「**インテル® CPU における高速化された AI の新時代の到来**」の執筆を依頼しました。関連記事として、「**インテルの新しい組込み AI アクセラレーション・エンジンを使用する**」では、インテル® AMX とインテル® XMX を使用して AI パフォーマンスを向上する方法について詳しく説明します。

次に、トラスト AI と異常検出など、AI の異なる側面をカバーする 3 つの記事「**BigDL プライバシー保護マシンラーニングを使用したトラスト TorchServe**」、「**ベイズ・ディープラーニングの量子化フレームワーク**」、「**最先端のビジュアル品質検査**」が続きます。

最後に、ローレンス・リバモア国立研究所のスーパーコンピューター・パッケージ・マネージャー、Spack に関する記事「**Spack を使用してインテルにより最適化された HPC バイナリーのディストリビューション**」で締めくくります。Spack は、2023 年の HPCwire Editors' Choice Award の「ベスト HPC プログラミング・ツール / テクノロジー」（英語）を受賞しています。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

次号では、James Reinders が The Parallel Universe の編集長として戻ってきます。彼が次号の編集長を引き受けてくれたおかげで、私は安心して休暇を取ることができます。彼が本誌の舵を握るのは 2016 年以来のことです。

AI とデータサイエンス、コードの現代化、ビジュアル・コンピューティング、データセンターとクラウド・コンピューティング、システムと IoT 開発、oneAPI を利用したヘテロジニアス並列コンピューティング向けのインテル・ソリューションの詳細は、Tech.Decoded（英語）を参照してください。

Henry A. Gabb

2024 年 1 月

ポッドキャスト

生成 AI は我々の成長、仕事、生活をどのように変えるのか

聴く（英語）

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

インテル® CPU における高速化された AI の新時代の到来

Numenta は CPU 上での AI 推論を高速化する神経科学ベースのソリューションでインテルと協力

Charmaine Lai Numenta マーケティング・マネージャー

問題ステートメント

2024 年を迎えて、さまざまな業界の企業が大規模言語モデル（LLM）の可能性を目の当たりにしています。しかし、主にコスト、複雑さ、高い消費電力、データ保護への懸念により、企業は LLM を自社の業務に統合することに苦労しています。GPU はこれまで LLM にとって主要なハードウェアでしたが、IT の複雑さとコストの増加により、成長する運用を維持することは容易ではありません。さらに、世界的な需要の増加と供給不足のため、企業は現在、AI プロジェクト向けの GPU を確保するのに 1 年以上待つこともあります。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

新しいアプローチ

そこで、Numenta とインテルは、GPU への依存から企業を解放する、LLM 導入の新しいアプローチを作り出しました。Numenta Platform for Intelligent Computing（NuPIC*）は、神経科学に基づく概念をインテル® アドバンスト・マトリクス・エクステンション（インテル® AMX）命令セットにマップします。NuPIC* を使用すると、インテル® CPU 上に LLM を大規模にデプロイして、コストとパフォーマンスの劇的な向上を実現できます（第 4 世代インテル® Xeon® スケーラブル・プロセッサーのパフォーマンス・インデックス（英語）のベンチマーク P6 および P11 を参照）。

テクノロジー

インテル® AMX

インテル ® AMX は、専用のハードウェア・サポートを提供することにより行列演算を高速化します（**図 1**）。このテクノロジーは、ディープラーニング推論など、行列計算に大きく依存するアプリケーションでは特に有益です。インテル® AMX は、第 4 世代インテル® Xeon® プロセッサーで最初に導入され、新しい第 5 世代インテル® Xeon® プロセッサーでさらに高速化されました。

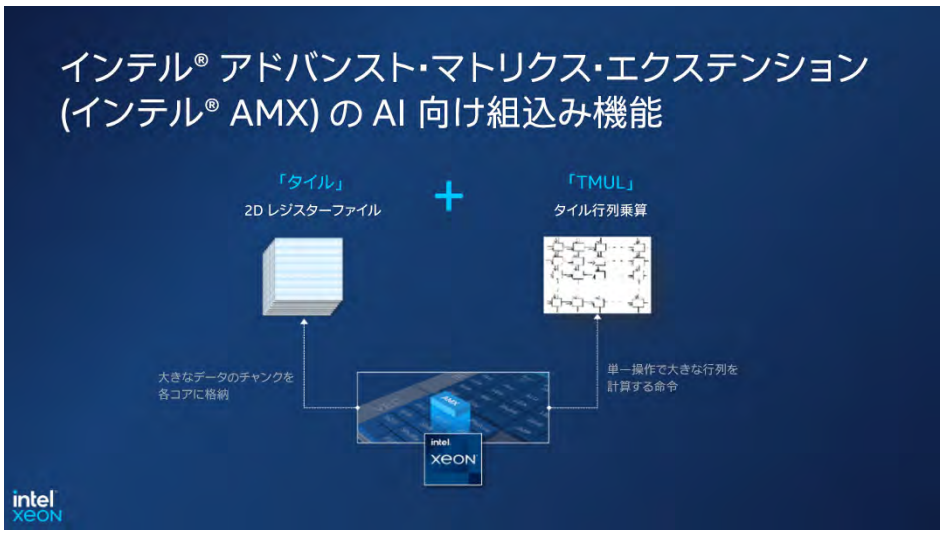


図 1. インテル® AMX 命令セットには、タイルと TMUL という 2 つの主要なコンポーネントが含まれています。タイルは、大きな行列の並列処理を可能にする 2D ユニットです。TMUL は、わずか 16 クロックサイクルで 16x16x32 の行列乗算を実行できます。

NuPIC*

NuPIC* は数十年にわたる神経科学研究に基づいて構築されており、強力な言語ベースのアプリケーションを迅速かつ容易に構築できます（**図 2**）。NuPIC* の中心となるのは、インテル® CPU 上で LLM を効率的に実行できるようにする、インテル® AMX を活用して高度に最適化された推論サーバーです。さまざまな自然言語処理のユースケースに合わせてカスタマイズできる、実用的な事前トレーニング済みモデルのライブラリーを使用して、NuPIC* 推論サーバーでモデルを直接実行できます。NuPIC* トレーニング・モジュールを使用してユーザーデータでモデルを調整し、そうしたカスタムモデルを NuPIC* モデル・ライブラリーにデプロイして推論サーバーで実行することもできます。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

標準の推論プロトコルをベースに構築された NuPIC* は、標準の MLOps パイプラインにシームレスに統合できます。このソリューションは Docker* コンテナとしてデプロイされ、スケーラブル、セキュア、高可用性、ハイパフォーマンスの環境としてインフラストラクチャー内に保持できます。カスタムモデルは完全に制御できます。データは完全にプライベートのままで、システムの外にエクスポートする必要はありません。

インテル® Xeon® CPU 上の NuPIC* が AI 推論に最適な理由

Numenta とインテルは、この物語の新たな章を開き、非常にコスト効率に優れた方法で LLM を CPU 上に大規模にデプロイできるようにしました。その理由をいくつか紹介しましょう。

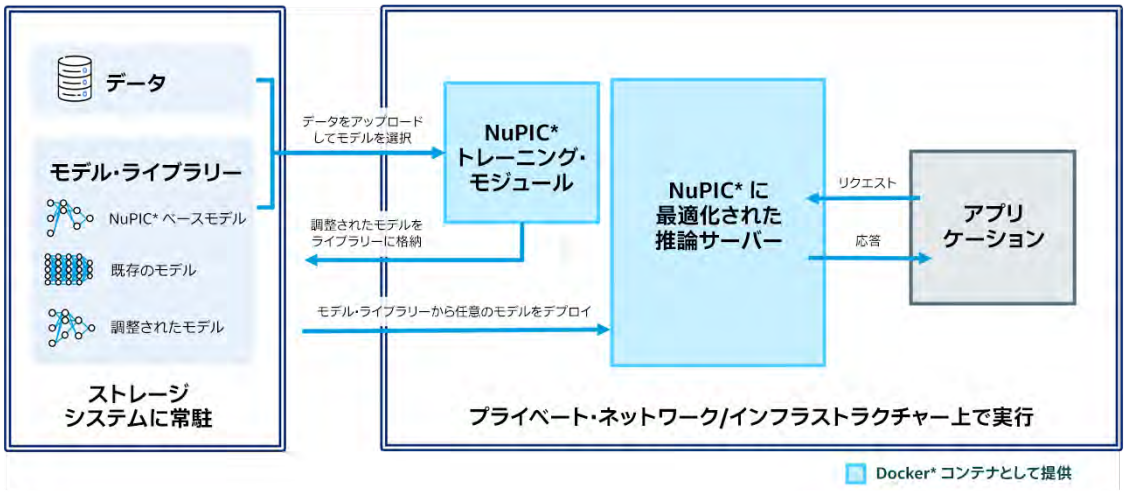


図 2. NuPIC* アーキテクチャー

パフォーマンス : NVIDIA* A100 Tensor Core GPU より 17 倍高速

Transformer 構造への最小限の変更により、NuPIC* は、インテル® AMX 対応の CPU で、前世代の CPU と比較して推論スループットを 2 桁以上向上し、GPU と比較しても大幅な速度向上を実現します(**表 1**)。BERT-Large で、第 5 世代インテル® Xeon® CPU 上の NuPIC* は、NVIDIA* A100 GPU よりも最大でおよそ 17 倍優れています。GPU で最高の並列パフォーマンスを実現するには、より大きなバッチサイズが必要です。しかし、バッチ処理は推論の実装が複雑になり、リアルタイム・アプリケーションでレイテンシーが生じます。対照的に、NuPIC* ではバッチ処理が必要ないため、アプリケーションは柔軟で、スケーラブルで、管理が容易になります。参考として、バッチサイズ 8 の NVIDIA* A100 のパフォーマンスをリストします。バッチサイズ 1 の NuPIC* は、このバッチサイズ 8 の NVIDIA* GPU 実装と比較した場合でも 2 倍以上優れています。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

シーケンスの長さ	バッチサイズ	BERT-Large での 1 秒あたりの推論数		スピードアップ
		NuPIC* 第 5 世代インテル® Xeon®	Nvidia* A100	
64	1	2891	166**	17.4x
128	1	901	128	7.0x
128	8	-	495	2.2x

表 1. 第 5 世代インテル® Xeon® CPU 上の NuPIC* と NVIDIA* A100 の推論の結果。インテル® Xeon® CPU の結果は、シングルノード、2 ソケットのインテル® Xeon® Platinum 8592+ プロセッサー、メモリー 500GB のシステムで、Ubuntu* 22.04 カーネル 5.15.0-87、Numenta Platform for Intelligent Computing V1.0、NuPIC* に最適化された BERT-Large、シーケンス長 64/128、BF16、バッチサイズ 1 を使用して、2023 年 11 月 27 日に Numenta により生成されました。NVIDIA* A100 の結果は、こちら（英語）。 は推定。**

スケーラビリティー：複数のクライアントの管理

ほとんどの LLM アプリケーションは多くのクライアントをサポートする必要があり、それぞれ推論結果が生成されます。単一の第 5 世代インテル® Xeon® プロセッサー・ベースのサーバー上の NuPIC* で、アプリケーションは高いスループットを維持しながら、多くのクライアントのリクエストを処理できます（**図 3**）。クライアントが入力をバッチ処理する必要はなく、各クライアントを完全に非同期にできます。

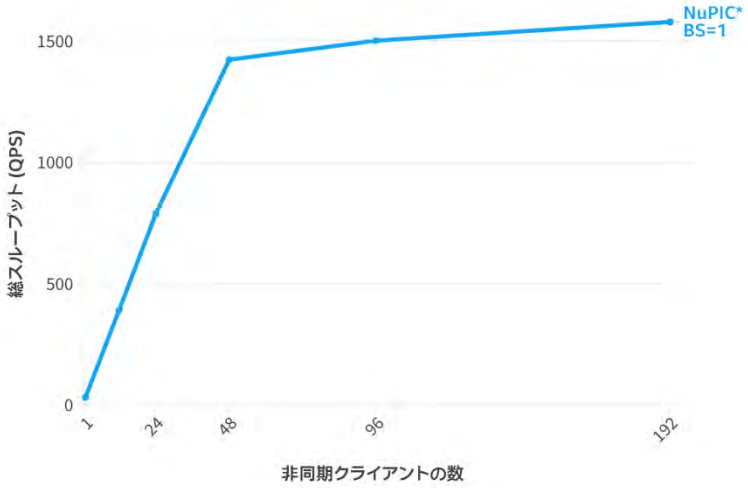


図 3. 非同期クライアントの数が増加した場合の、128 の独立した NuPIC* に最適化された BERT-Large モデルの総スループット。結果は、シングルノード、2 ソケットのインテル® Xeon® Platinum 9592+ プロセッサー、メモリー 500GB のシステムで、Ubuntu* 22.04 カーネル 5.15.0-87、Numenta Platform for Intelligent Computing v1.0、NuPIC* に最適化された BERT-Large、シーケンス長 64、BF16、バッチサイズ 1 を使用して、2023 年 11 月 28 日に Numenta により生成されました。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

柔軟性：同一サーバー上で複数のモデルを実行

多くのビジネス・アプリケーションは、それぞれ異なるタスクを解決する複数の LLM を利用しています。NuPIC* を使用すると、非同期クライアントの管理に加えて、同一サーバー上で別々のモデルを同時に非同期に実行できますが、GPU でこの処理を効率的に管理することは困難です。**図 3** に示すように、単一サーバー上で 128 の独立したモデルを実行しました。異なるモデルでは重みのセットや計算要件も異なることがありますが、インテル® CPU 上の NuPIC* では容易に処理できます。さらに、データと計算のニーズが増加したときに、追加の GPU を統合するよりもシステムに CPU を追加する方が、より簡単で制限も少なくなります。これは、クリックするだけで計算能力を追加できるクラウドベースの環境に特に当てはまります。IT の観点から見ると、CPU は GPU よりもデプロイと管理が容易です。

コストと電力効率

CPU は一般に GPU よりも手頃な価格で入手できるため、多くの企業に推奨される選択肢と言えます。しかし、AI 推論を実行する際の有効性は、処理速度の面では制限されていました。NuPIC* はインテル® CPU 上での AI 推論を高速化し、推論にかかるコストと消費電力の両方で利点をもたらします。NuPIC* とインテル® AMX により、AI アプリケーションのコストと消費電力が大幅に削減され、持続可能性とスケーラビリティーが向上します。

実装例

NuPIC* の機能を活用する方法を示すため、容易にデプロイできるものから調整が必要なものまで、さまざまなユースケースのサンプルコードがあります。一例として、NuPIC* を使用して金融コンテキストで感情分析を実行する方法を示します。NuPIC* に最適化された BERT モデルは、ニュース記事や市場レポートなどのテキストデータを処理し、それらを異なる感情（肯定、否定、中立）に分類します。金融アナリストは市場全体の感情を迅速に把握し、投資戦略や市場の動向の予測に関して、十分な情報に基づいて判断できます。

NuPIC* をインストールして起動する

Numenta は、サンプルコードおよび調整とデプロイに必要なすべてのコンポーネントを含むスクリプトを提供しています。NuPIC* をインストールした後、次のコマンドを実行してトレーニング・モジュールと推論サーバーを起動します。

Unix +

```
./nupic_training.sh --start
./nupic_inference.sh -start
```

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

Docker* イメージがダウンロードされ、Docker* コンテナが起動します。デフォルトでは、トレーニング・モジュールと推論サーバーの Docker* コンテナは、API サーバーのポート 8321 と 8000 で実行されます。サーバーと Docker* コンテナは、`--stop` コマンドでいつでも停止できます。

データセットを準備する

この例では、金融感情データセットをモデルの調整に使用します。まず、データをトレーニング・セットとテストセットに分割します。

```python
dataset_file = f"{script_dir}/../../datasets/financial_sentiment.csv"

…

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, stratify=y, random_state=42
)

train = pd.concat([y_train, X_train], axis=1)
test = pd.concat([y_test, X_test], axis=1)

train.to_csv("financial_sentiment_train_dataset.csv", index=False)
test.to_csv("financial_sentiment_test_dataset.csv", index=False)
```

NuPIC* トレーニング・モジュールを使用してモデルをデータnい調整する

データセットを分割した後、NuPIC* のトレーニング・モジュールを使用してモデルを調整できます。このプロセスでは、パフォーマンスを最適化するため、学習率、バッチサイズなどのパラメーターを変更します。ここでは、NuPIC* に最適化された BERT モデルを金融感情データセットについて調整します。

```unix
python -m nupic.client.nupic_train  --train_path  ../../datasets/financial_sentiment_train_dataset.csv
--test_path ../../datasets/financial_sentiment_test_dataset.csv --url http\://localhost:8321
```

NuPIC* のトレーニング・モジュールは、金融感情データセットのテスト部分を使用して精度スコアを出力します。必要に応じてハイパーパラメーターを調整し、満足な結果が得られるまで再トレーニングします。調整が完了すると、新しいモデルが生成されます。上記のスクリプトは、トレーニング・モジュールからモデルを自動的に取得し、`.tar.gz` ファイルとしてローカル・ディレクトリーに保存します。後で、このファイルを NuPIC* の推論サーバーにデプロイできます。

調整したモデルを NuPIC* 推論サーバーにインポートする

調整したモデルをデプロイするには、まずそのモデルを推論サーバーで利用できるようにする必要があります。具体的には、最終フェーズの出力（`model_xxx.tar.gz`）を展開して、推論サーバーのコンテナにマップされる `models` ディレクトリーに移動します。次のコマンドを使用します。`xxx` を調整したモデルに置き換えてください。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

```
Unix
cp model_xxx.tar.gz ../nupic/inference/models
cd ../../nupic/inference/models
tar -xzvf model_xxx.tar.gz
```

デプロイ : インテル® Xeon® CPU 上でのリアルタイム推論

次に、モデルをデプロイします。NuPIC* の Python* クライアントを利用して、簡単にデプロイできます。このクライアントは、エンドユーザー・アプリケーションとモデル間の通信を管理します。シンプルな API を使用して、埋め込みを自動的に生成し、推論サーバーにリアルタイムでリクエストを送信できます。NuPIC* クライアントをセットアップし、次のように推論を実行します。

```
Python
# Model name, you can use a model from NuPIC model libary, or your fine-tuned model
MODEL = "numenta-sbert-2-v1-wtokenizer"

# The URL to your inference server instance
URL = "localhost:8000"

# The supported protocol includes http and grpc
PROTOCOL = "http"

# Optional connection configuration, such as SSL protocol certificates.
CONNECTION_CONFIG = {}

#Connect to NuPIC client
numenta_client = ClientFactory.get_client(MODEL, URL, PROTOCOL, CONNECTION_CONFIG)

#Run inference
result = numenta_client.infer(["Semiconductor industry is doing great!"])
```

テキストデータを含むリクエストを API エンドポイントに送信し、感情予測を取得します。リアルタイム解析の場合、市場レポートやニュース記事を API に継続的にフィードして、感情予測を受け取ることができます。

まとめ

NuPIC* とインテル® AMX を利用して、インテル® CPU 上で LLM の素晴らしいスケーリング能力と組み合わせることにより、エンタープライズ IT に必要な汎用性とセキュリティーを得ることができます。このソフトウェアとハードウェア・テクノロジーの相乗的な組み合わせにより、CPU 上で LLM を実行できるようになるだけでなく、戦略的な利点が得られます。皆さんの環境で NuPIC* からどのようなメリットが得られるか確認したい場合は、こちらからデモをリクエスト（英語）してください。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

インテルの新しい組込み AI アクセラレーション・エンジンを使用する

インテル® AMX とインテル® XMX で AI の効率、スケーラビリティー、パフォーマンスを向上

Subarnarekha Ghosal インテル コーポレーション テクニカル・コンサルティング・エンジニア

人工知能（AI）が注目されるとともに、パフォーマンスを向上させる方法として、低精度データ型の導入と、これらのデータ型のハードウェア・サポートが求められています。低精度モデルは計算が速く、メモリー・フットプリントが小さくなることから、AI トレーニングと推論には、32 ビットのデータ型よりも低精度のデータ型が推奨されます。これらの低精度データ型を最適化およびサポートするには、ハードウェアに特別な機能と命令が必要です。インテルは、これらをインテル® CPU とインテル® GPU でそれぞれ、インテル® アドバンスト・マトリクス・エクステンション（インテル® AMX）およびインテル® Xe マトリクス・エクステンション（インテル® XMX）として提供しています。最も使用されている 16 ビット形式は、16 ビット IEEE 浮動小数点数（fp16）、bfloat16、16 ビット整数（int16）であり、最も使用されている 8 ビット形式は、8 ビット整数 (int8) と 8 ビット Microsoft* 浮動小数点数 (ms-fp8) です。これらの形式の違いの一部を**図 1** に示します。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

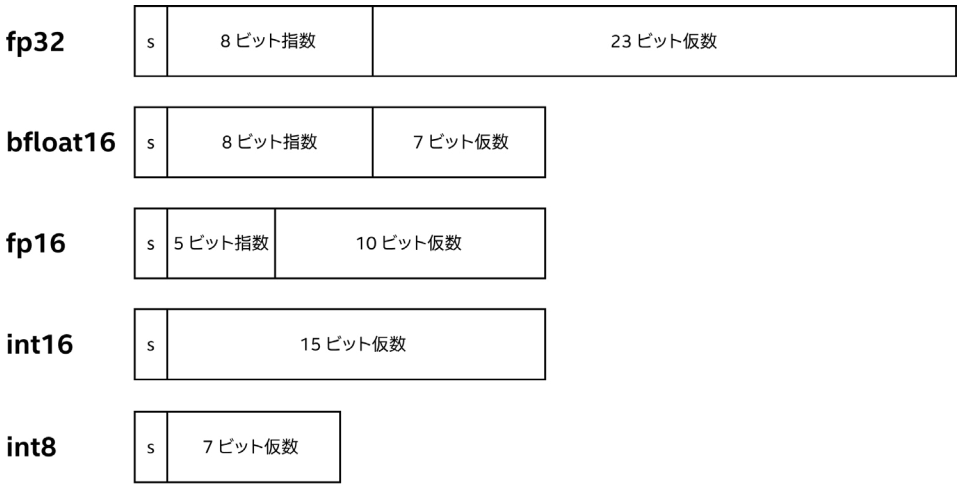


図 1. IEEE 標準データ型のさまざまな数値表現。s は符号付きビット（正の数は 0、負の数は 1）を表し、exp は指数を表します。

インテル® AMX とインテル® XMX を呼び出すプログラミング・パラダイムは、ハードウェアにより異なります。この記事では、それらをプログラムするさまざまな方法を紹介した後、これらの命令セットのパフォーマンスの利点を説明します。

インテル® AMX とインテル® XMX

インテル® AMX

インテル® AMX は、マイクロプロセッサー向けの x86 命令セット・アーキテクチャー（ISA）を拡張したものです。アクセラレーターが操作を実行できる、タイルと呼ばれる 2D レジスターを使用します。インテル® AMX の 1 つのユニットには、通常 8 つのタイルがあります。タイルは、ロード、ストア、クリア、または内積操作を実行できます。インテル® AMX は、int8 および bfloat16 データ型をサポートします。インテル® AMX は、第 4 世代インテル® Xeon® スケーラブル・プロセッサーでサポートされています（**図 2**）。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

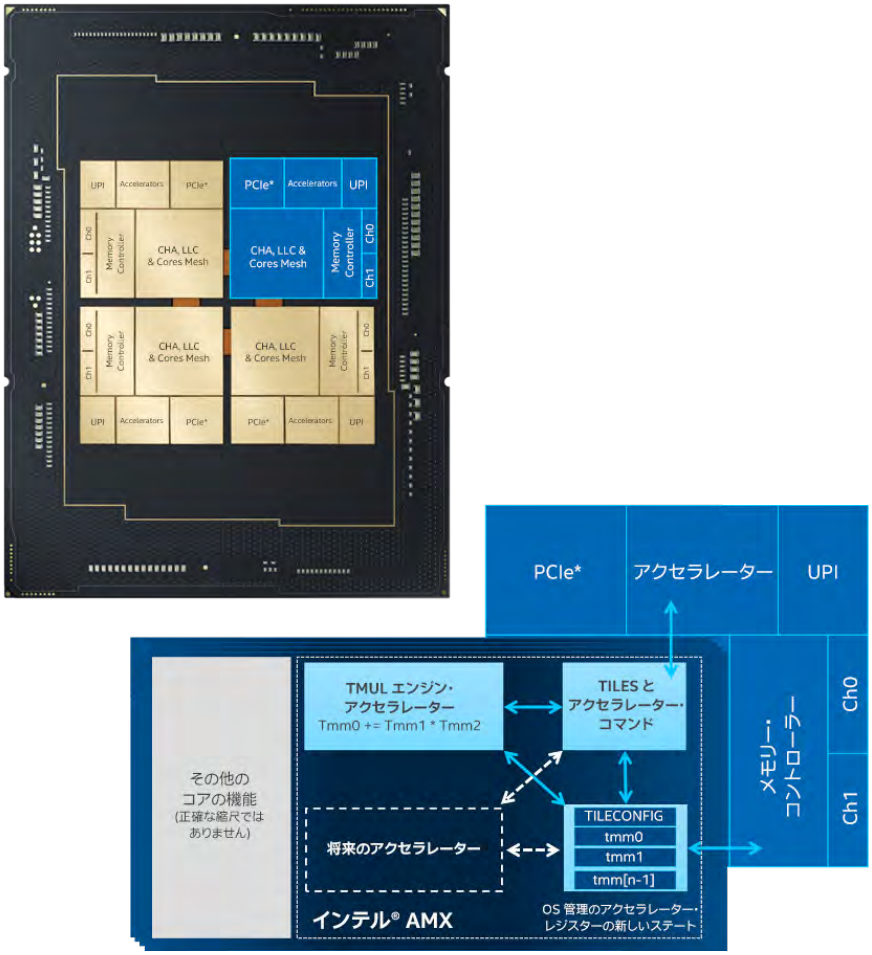


図 2. 第 4 世代インテル® Xeon® スケーラブル・プロセッサーのインテル® AMX

インテル® XMX

インテル® XMX（内積累算シストリック（Dot Product Accumulate Systolic、DPAS）とも呼ばれる）は、2D シストリック・アレイでの内積と累算命令の実行を専門としています。並列コンピューター・アーキテクチャーのシストリック・アレイは、密結合なデータ処理ユニットのホモジニアス・ネットワークです。各ユニットは、上流のユニットから受け取ったデータの関数として部分的な結果を計算し、その結果を保存して、下流のユニットに渡します。インテル® XMX は、ハードウェア世代に応じて、int8、fp16、bfloat16、tf32 などのさまざまなデータ型をサポートしています。インテル® XMX は、インテル® データセンター GPU マックス・シリーズまたはインテル® データセンター GPU フレックス・シリーズの一部です。インテル® データセンター GPU マックス・シリーズのインテル® X^e HPC 2 スタックは、**図 3** では X^e として省略されています。スタックとは、タイルの代わりに使用される用語です。インテル® データセンター GPU マックス・シリーズは 8 つのインテル® X^e スライスで構成されます。各スライスには 16 個のインテル® X^e コアが含まれています。各コアには 8 つのベクトルエンジンと 8 つの行列エンジンが含まれています。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

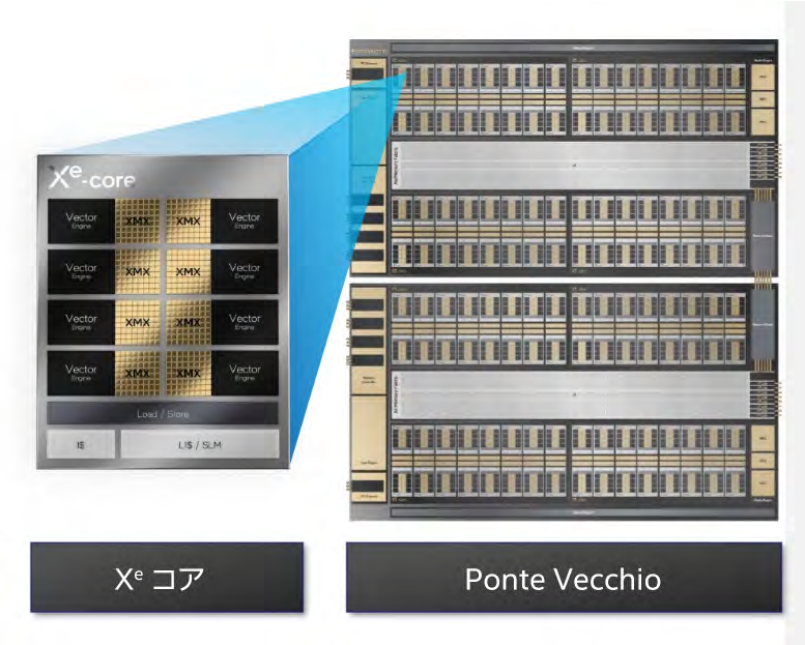


図 3. インテル® データセンター GPU マックス・シリーズ（開発コード名 Ponte Vecchio）のインテル® XMX

インテル® AMX とインテル® XMX を使用したプログラミング

ユーザーは、ディープラーニング・フレームワーク、専用ライブラリー、カスタム SYCL* カーネルから低水準の組込み関数まで、さまざまなレベルでインテル® XMX を操作できます。**図 4** は、さまざまなコーディング抽象化でインテル® AMX とインテル® XMX を呼び出す方法（後述）を示しています。これらの拡張機能を利用するには、インテル® oneAPI ベース・ツールキット 2024.0 が必要です

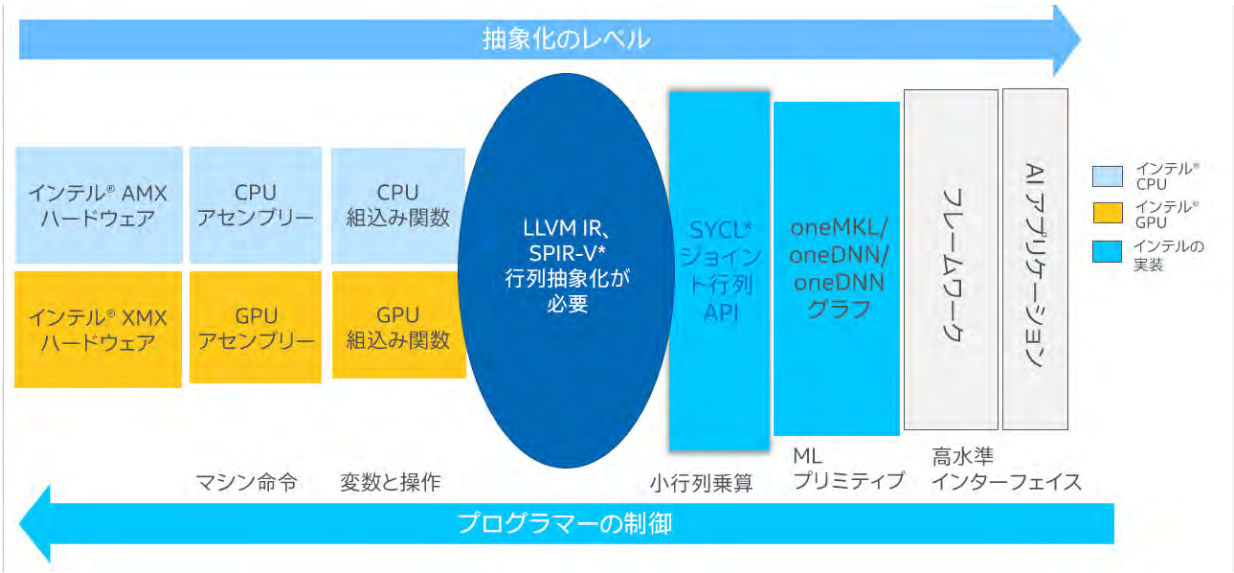


図 4. 行列計算のプログラミングの抽象化

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

SYCL* ジョイント行列拡張機能

ジョイント行列は、CPU 向けのインテル® AMX、GPU 向けのインテル® XMX、NVIDIA* Tensor コアのようなテンソル・ハードウェア・プログラミングのターゲットを統合する新しい SYCL* 拡張機能です。これらの SYCL* ジョイント行列の例（英語）は、この拡張機能の使用法を示しています。独自のニューラル・ネットワーク・アプリケーションを構築したいユーザーは、TensorFlow* などのフレームワークや、インテル® oneAPI ディープ・ニューラル・ネットワーク・ライブラリー（インテル® oneDNN）やインテル® oneAPI マス・カーネル・ライブラリー（インテル® oneMKL）などのライブラリーよりも抽象度が低いジョイント行列を使用できます。ジョイント行列は、ポータブルなだけでなく、異なるハードウェアで利用可能な最大のパフォーマンスを活用できる統合インターフェイスを提供します。ジョイント行列は、パフォーマンス、生産性、融合機能と、異なるテンソル・ハードウェアへの移植性を提供するため、ハードウェア・プラットフォームごとに異なるコードベースを維持する必要がなくなります。

「ジョイント行列」という用語は、行列がワーク項目のグループ内で共有され、各ワーク項目専用ではないことを強調しています（**図 5**）。グループスコープは SYCL* ジョイント行列構文の追加テンプレート・パラメーターとして追加されますが、現在の実装ではサブグループ・スコープのみサポートされています。行列に対する一般的な操作である、ロード、ストア、「乗算と加算」操作を実行するために必要な関数の一部は、それぞれ、`joint_matrix_load`、`joint_matrix_store`、`joint_matrix_mad` で表されます。それらは、プログラムを実行しているハードウェアに応じて、低水準のインテル® AMX またはインテル® XMX 組込み関数を呼び出します。`joint_matrix_load` 関数はメモリーから 2D タイル / レジスターにデータをロードし、`joint_matrix_store` は 2D タイルからアキュムレーター行列のデータをメモリーに格納し、`joint_matrix_mad` は 2 つの行列に乗算を実行し、結果を 3 番目の行列と累算して結果を返します。

名前空間	namespace sycl::ext::oneapi::experimental::matrix
グループスコープ、use (a, b, accumulator)、サイズ、レイアウトで定義される新しい行列データ型	template <typename Group, typename T, use Use, size_t Rows, size_t Cols, layout Layout = layout::dynamic> struct joint_matrix; enum class use { a, b, accumulator};
メモリー操作を計算から分離 enum class layout {row_major, col_major, dynamic };	• void joint_matrix_fill(Group g, joint_matrix<>&dst, T v); • void joint_matrix_load(Group g, joint_matrix<>dst, T *base, unsigned stride, Layout layout); • void joint_matrix_store(Group g, joint_matrix<>src, T *base, unsigned stride, Layout layout);
乗算と加算	• joint_matrix<>joint_matrix_mad(Group g, joint_matrix<>A, joint_matrix<>B, joint_matrix<>C);

図 5. 一般的な SYCL* ジョイント行列 API

インテル® oneDNN とインテル® oneMKL

インテル® oneAPI ディープ・ニューラル・ネットワーク・ライブラリー（インテル® oneDNN）とインテル® oneAPI マス・カーネル・ライブラリー（インテル® oneMKL）は、インテル® CPU（第 4 世代インテル® Xeon® スケーラブル・プロセッサー以降）とインテル® データセンター GPU マックス・シリーズでそれぞれ、デフォルトでインテル® AMX とインテル® XMX を使用します。インテル® oneDNN ユーザーは、コードでインテル® XMX でサポートされているデータ型を使用していることを確認し、GPU サポートを有効にしてインテル® oneDNN ライブラリーをビルドする必要があります。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

インテル® oneDNN にバンドルされている matmul_perf（英語）サンプルは、適切にコンパイルされるとインテル® AMX およびインテル® XMX を呼び出します。追加のインテル® oneDNN のサンプルは、こちら（英語）から入手できます。単純なインテル® oneMKL のサンプルは、こちら（英語）から入手できます。

インテル® AMX のパフォーマンス解析

標準の fp32 と bfloat16 で実行した行列乗算を比較してみましょう。使用するベンチマーク（英語）は、本質的には、GEMM を繰り返し呼び出す単純なラッパーで、次のオプションを渡すことができます。

- GEMM カーネルのインテル® マス・カーネル・ライブラリー（インテル® MKL）バージョン
- 単精度または倍精度の GEMM（SGEMM/DGEMM）
- 使用するスレッドの数（NUM_THREADS）
- 乗算する行列のサイズ（SIZE_N）

ベンチマーク・コードは、問題のサイズを取得し、C = A x B 行列のスペースを割り当て、A 行列と B 行列をランダムデータで初期化し、GEMM を 1 回呼び出して初期化し、設定した回数 GEMM を連続して呼び出し、後者の実行時間を測定します。bfloat16 のほうが fp32 よりも優れていることが分かります（**図 6**）。

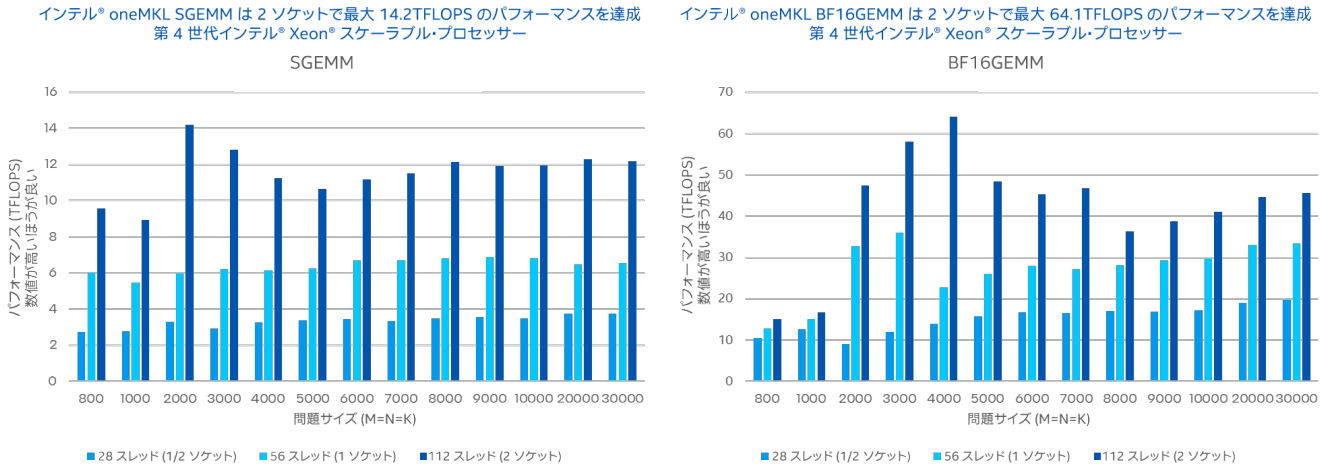


図 6. fp32 と bfloat16 データ型を使用した行列乗算パフォーマンスの比較

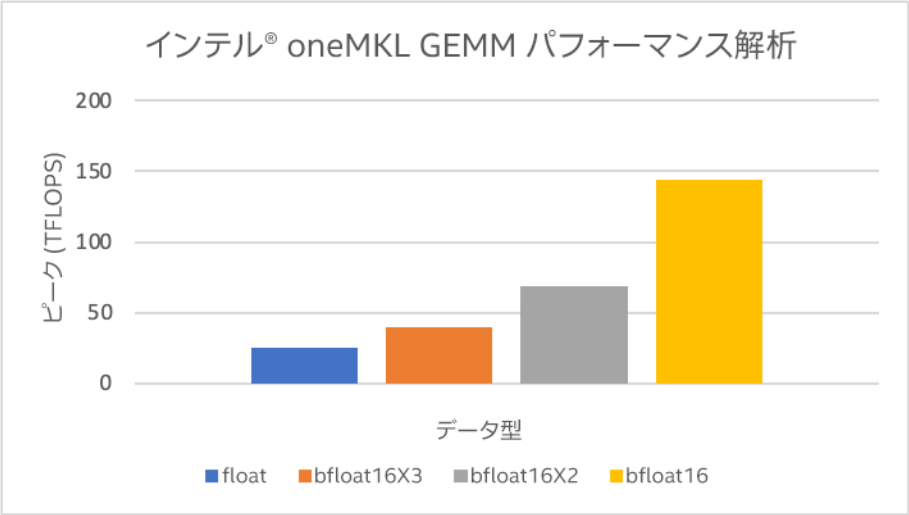
テスト：パフォーマンス結果は 2024.0 で検証しました。2023.x リリースと比較してパフォーマンスの低下は観察されませんでした。
システム構成とワークロードのセットアップ：1 ノード、2x インテル® Xeon® Platinum 8480+ プロセッサー、Denali Pass プラットフォーム、合計 DDR5 メモリー 1024GB（16 スロット/64GB/4800）、ucode 0x2b000161、インテル® ハイパースレッディング・テクノロジー無効、インテル® ターボ・ブースト・テクノロジー有効、Ubuntu* 22.04.1 LTS、5.17.0-051700-generic、1x インテル® SSD 3.5TB（OS ドライブ）、インテル® oneMKL 2023.0。正方行列（ 次元 800 から 30,000）の SGEMM および BFLOAT16GEMM のパフォーマンス。性能の測定結果はシステム構成の日付時点のテストに基づいています。また、現在公開中のすべてのセキュリティー・アップデートが適用されているとは限りません。詳細は、システム構成を参照してください。絶対的なセキュリティーを提供できる製品またはコンポーネントはありません。性能は、使用状況、構成、その他の要因によって異なります。詳細については、http://www.intel.com/PerformanceIndex/（英語）を参照してください。実際の費用と結果は異なる場合があります。

法務上の注意書き
性能は、使用状況、構成、その他の要因によって異なります。詳細については、http://www.intel.com/PerformanceIndex/（英語）を参照してください。性能の測定結果はシステム構成の日付時点のテストに基づいています。また、現在公開中のすべてのセキュリティー・アップデートが適用されているとは限りません。構成の詳細は、補足資料を参照してください。絶対的なセキュリティーを提供できる製品またはコンポーネントはありません。実際の費用と結果は異なる場合があります。インテルのテクノロジーを使用するには、対応したハードウェア、ソフトウェア、またはサービスの有効化が必要となる場合があります。
© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
* その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。
システム構成：1x インテル® データセンター GPU マックス 1550、2 ソケットのインテル® Xeon® 8480+ プロセッサーでホスト、512GB DDR5-4800 メモリー、Ubuntu* 22.04、カーネル 5.15、IFWI 2023WW28、インテル® oneMKL 2023.1、ICX 2023.1。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

インテル® XMX のパフォーマンス解析

次に、この GEMM ベンチマーク（英語）を使用して、インテル® GPU 上のインテル® XMX のパフォーマンスの利点を調査します。インテル® oneMKL は、インテル® XMX を使用して単精度 GEMM を高速化するいくつかのアルゴリズムをサポートしています。bfloat16x2 と bfloat16x3 は、bfloat16 シストリック・ハードウェアを使用して単精度 GEMM を近似する 2 つのアルゴリズムです。内部的には、単精度入力データが bfloat16 に変換され、シストリック配列と乗算されます。3 つのバージョン（bfloat16、bfloat16x2、bfloat16x3）を使用して、精度とパフォーマンスの間のトレードオフを選択できます。bfloat16 が最も高速で、bfloat16x3 が最も正確です（標準 GEMM と同様）。3 つのバージョンはすべて、標準 float を上回ります（**図 7**）。



システム構成：1x インテル® データセンター GPU マックス 1550、2 ソケットのインテル® Xeon® 8480+ プロセッサーでホスト、512GB DDR5-4800 メモリー、Ubuntu* 22.04、カーネル 5.15、IFWI 2023WW28、インテル® oneMKL 2023.1、ICX 2024.0。https://github.com/oneapi-src/oneAPI-samples/tree/master/Libraries/oneMKL/matrix_mul_mkl（英語）のコードを使用。

法務上の注意書き
性能は、使用状況、構成、その他の要因によって異なります。詳細については、http://www.intel.com/PerformanceIndex/（英語）を参照してください。性能の測定結果はシステム構成の日付時点のテストに基づいています。また、現在公開中のすべてのセキュリティー・アップデートが適用されているとは限りません。構成の詳細は、補足資料を参照してください。絶対的なセキュリティーを提供できる製品またはコンポーネントはありません。実際の費用と結果は異なる場合があります。インテルのテクノロジーを使用するには、対応したハードウェア、ソフトウェア、またはサービスの有効化が必要となる場合があります。
© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
* その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

まとめ

この記事では、最新のインテル® CPU とインテル® GPU に導入されたインテル® AMX とインテル® XMX 命令セットの概要を説明しました。インテル® AMX とインテル® XMX 命令セットは、コンパイラー組込み関数から SYCL* ジョイント行列抽象化、インテル® oneMKL とインテル® oneDNN まで、異なるレベルのプログラミングで呼び出すことができます。コーディング・パラダイムの抽象化レベルが高くなるほど、これらの命令セットの呼び出しが容易になり、呼び出しに対するプログラマーの制御が少なくなります。インテル® AMX とインテル® XMX はパフォーマンス向上のためのテクノロジーです。第 4 世代インテル® Xeon® スケーラブル・プロセッサーの GEMM ベンチマークの例では、適切な低精度データ型を使用してインテル® AMX を呼び出すだけで、パフォーマンスがどのように向上するか明確に理解できます。インテル® データセンター GPU マックス 1550 の GEMM ベンチマークの例では、適切な低精度データ型を使用してインテル® XMX を呼び出すだけで、パフォーマンスがどのように向上するか明確に理解できます。パフォーマンスの利点、使いやすさ、および低水準プログラミング、ライブラリーとフレームワークでプログラムできる機能のため、ユーザーは AI ワークロードでこれらの拡張機能をテストして使用することを推奨します。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

我々がどこまで辿り着いたか振り返る

インテル® ソフトウェア開発ツールと oneAPI

James Reinders インテル コーポレーション oneAPI エバンジェリスト兼 The Parallel Universe 名誉編集長

30 年以上前にインテルがコンパイラー、ライブラリー、パフォーマンス解析ツールをリリースしたとき、市場はあまり興味を示しませんでした。しかし、30 年以上にわたり積み上げてきた実績により、最近リリースされたインテル® ソフトウェア開発ツール 2024 は、ソフトウェア開発者にとって重要な価値をもたらすツールとして受け入れられています。2018 年に、インテルが「オープン・アクセラレーテッド・コンピューティング」を実現する oneAPI というアイデアを発表したときにも、市場は再び興味を示しませんでした。アイデア自体は良いものとして受け入れられましたが、独自路線で直面する課題は、インテルであっても克服するには大きすぎると指摘されました。その後、その指摘を覆し、このアイデアは根付いてきています。この記事では、おそらく我々すべてに影響し、アクセラレーターの開発と利用に関わるすべての人にとって非常に重要な、現時点の変化について説明します。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

2 つの重要な発表

この 2 つの発表は、標準ベースのアプローチの推進とそれらをサポートするツールの提供における進歩を証明しています。

1. Unified Acceleration Foundation によりオープンな高速計算とクロスプラットフォーム･パフォーマンスを推進 (英語)
2. インテル® ソフトウェア開発ツール 2024 がリリース（英語）

Unified Acceleration Foundation

オープン・スタンダードとオープンソースは、開かれたガバナンスが確保されてこそ真価を発揮します。Linux Foundation が Unified Acceleration（UXL）Foundation を創設したことで、オープンガバナンスが確保され、oneAPI の未来は明確になりました。オープン・スタンダードを重視し、ベンダーやアーキテクチャーに依存することなく、すべてのデバイスのサポートを受け入れるというアプローチは、インテルを含む大手ベンダーがアクセラレーテッド・コンピューティングのソフトウェア・スタックに採用していた従来のアプローチとは異なるものです。

2019 年の最初のリリースから現在に至るまで、oneAPI は複数のアーキテクチャーとベンダーから支持されており、有望なアイデアであることは明らかです。一部のベンダーは依然として抵抗していますが、より革新的なハードウェアが市場に登場するにつれて、それらのベンダーの意見は重要ではなくなるでしょう。つまり、oneAPI 仕様と oneAPI プロジェクトに対して、Linux Foundation スタイルの本格的なガバナンスを採用する時期が来たのです。インテルはすべての oneAPI オープンソース･プロジェクトを UXL Foundation に寄与しているため、UXL Foundation の方向性に対するインテルの意見は、多くの意見の 1 つにすぎません。オープンガバナンスはすべての人々にとって最善なものです。

インテル® ソフトウェア開発ツール 2024

インテルのソフトウェア開発ツールは世界中で認められ、信頼されています。インテル® コンパイラーとライブラリーには 30 年以上にわたる歴史があります。インテルのツールは、C、C++、Fortran、Python*、OpenMP*、MPI、SYCL* などオープン・スタンダードのサポートをリードしています。AI、FPGA、GPU の出現によって、インテル® ハードウェアと同様に「どこにも負けない」サポートを提供する、新しいライブラリーとツールが追加されました。2019 年の oneAPI のリリース以来、ツールは可能な限り最良の方法として、プラグインを利用してインテル以外のハードウェアをサポートできるようになっています。例えば、コードが SYCL* で記述されていれば、インテル® C/C++ コンパイラーは他社製アクセラレーター向けのコードを生成する必要があります。これは、Khronos Group Inc. の SYCL* 標準が、「単一ソース」のコーディングで、すべてのベンダーとすべてのアーキテクチャーをサポートすることを目的としているためです。そのため、インテル® コンパイラーでは、他社製アクセラレーターのサポートを追加するプラグインを許可しています。例えば、Codeplay の最新のプラグインは、NVIDIA と AMD のサポートを追加します。LLVM エコシステムにおける NVIDIA* GPU と AMD* GPU のオープンソース･サポートは､競争力を高めることに役立っています。2022 年と 2023 年に実施されたテスト（英語）では、NVIDIA と AMD のハードウェアでの CUDA* 対 SYCL* および HIP 対 SYCL* のパフォーマンスはほぼ同等であることが示されました。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

oneAPI マス・カーネル・ライブラリー（oneMKL）（英語）は、プラグインモデルも効果的に実行します。共通の API を使用して最高のパフォーマンスが得られるように、コードは、インテルのハードウェアではインテル® oneMKL に、NVIDIA* GPU では cuBLAS に、AMD* GPU では rocBLAS に、それぞれディスパッチされます。各アーキテクチャー / ベンダーのサポートを確実に行うことを目指すもう 1 つの例は、最良の結果を得ることです。

コンピューター全体を使用する

最良のパフォーマンスを得るには、ベンダーやアーキテクチャーに関係なく、システム内のすべての機能を利用するようにプログラムする必要があります。それには、オープン･スタンダード、オープンソース、オープンガバナンス、そして必要なすべての問題を解決するオープンマインドが必要です。良いニュースとして、我々が必要とする多くが現在は存在することを､前述の｢2 つの重大な発表｣が示しています。これらは 4 年前には存在しなかったものです。

多くの開発プロジェクトが、独自のアプローチを止めて、完全にオープンにすることに成功しています。我々には、オープンガバナンスを備えたオープン・スタンダードが必要です。ISO、ANSI、Khronos、OpenMP*、Linux Foundation の取り組みに感謝します。どのようなハードウェアに対しても、必要に応じて最高のサポートへのアクセスを妨げない、優れた実装が必要です。ツールをオープンソースとして提供しているインテルの取り組みにも感謝します。

なぜインテルはオープンであることに熱心なのでしょうか？ それは、アクセラレーテッド・コンピューティングを使用する革新的なソフトウェアや革新的な新しいアクセラレーター・ハードウェアの開発者が、オープンであることを望んでいると知っているからです。これは、Windows* や UNIX* ではなぜ駄目なのかと多くの人に批判されていた Linux* の初期を思い出させます。当時はオープンであることがすべてであり､現在もそれは変わりありません。私は、オープン・アクセラレーテッド・コンピューティングの主唱者であるインテル・チームの一員であることを誇りに思っています。

もちろん、インテルのハードウェア設計に対して一定の偏見（身内としての思い入れ）はありますが、将来的にパーツの生産をインテルのファブに任せる（英語）という他社の設計者によるハードウェア設計への愛情によって、その偏見は補完されます。だからこそ、インテルのオープンへの取り組みを信頼できるのです。インテルの CEO が 2 年以上前に「An Open Letter to an Open Ecosystem（オープン・エコシステム宛ての公開書簡）」（英語）を発表したのも不思議なことではありません。

オープンな未来

oneAPI のオープンなアプローチと、インテルのツールの信頼性と価値を支持するユーザー・コミュニティーが成長しています。インテルのツールには、30 年以上にわたる強力な標準化のサポートと、パフォーマンスの最適化を支援してきた歴史があります。オープンなアクセラレーターのサポートを優先するという理念に基づいて、インテルのツールはこの 4 年間 oneAPI により強化されました。oneAPI は、Linux Foundation によりオープンガバナンスとして採用されました。アクセラレーテッド・コンピューティングをプログラムする真にオープンな未来は、我々の手の中にあります。ツールとアプローチを試してみてください。UXL に参加しましょう。皆さんの興味とフィードバックは、オープンな未来をより良くするのにきっと役立ちます。オープンな未来は、我々すべてが必要としているものです。現在でもその未来をまだ疑っている人は限りなく少数であることでしょう。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

intel Developer Cloud

コードの ROI を 高める

自分だけのクラウドで
快適に作業

複数の言語をサポートし、高速なヘテロジニアス・コンピューティングを実現する、単一のオープンなプログラミング・モデルで、コードの能力を引き出しましょう。オープン・スタンダードをベースとして構築されたインテル® デベロッパー・クラウドは、AI 開発を活性化し、優れたパフォーマンスを実現するための理想的なエンタープライズ環境を提供します。

エンタープライズ、クラウド、HPC、
AI などの分野の計算を高速化。

今すぐ開始

https://www.xlsoft.com/jp/products/intel/devcloud/services.html

Unified Acceleration（UXL）Foundation

アクセラレーター向けのオープン・スタンダード・プログラミングのコミュニティーを統合

Rod Burns Codeplay Software エコシステム担当副社長

テクノロジー業界では注目すべき変化が起きており、ヘテロジニアス・アーキテクチャーへの大きな波が押し寄せています。この新たな波への移行により、多くのソフトウェアのパフォーマンスが大幅に向上する可能性があります。「ヘテロジニアス」という用語は以前から使われていましたが、特に最新の AI ソフトウェアの出現と注目度の高さにより、この用語が重視されるようになったのはつい最近のことです。ヘテロジニアスとは、複数の種類のプロセッサー、例えば CPU、GPU、FPGA、TPU などで構成されるコンピューター・アーキテクチャーを指し、多くの場合、特定の種類のワークロードを高速化するために設計されています。実際に、世界上位のスーパーコンピューターの多くはヘテロジニアス・アーキテクチャーの一部として GPU を利用しており、この割合は最近の AI ブーム以前の過去 5 年間で大幅に増加しました。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

ChatGPT* などの大規模言語モデルベースのアプリケーションは、ユーザーにインタラクティブな情報を提供する驚くべき能力を実証し、最近ではニュースの見出しを賑わしています。理解しておくべき重要なことは、これらのモデルはヘテロジニアス・アーキテクチャーでのみ実行可能であるということです。高度な並列システムを活用しているのは AI ソフトウェアだけではなく、業界全体で、HPC、データセンター、組込み環境など、研究アプリケーションと産業アプリケーションの両方の限界を押し上げるパフォーマンスを提供するために GPU が使用されています。この傾向は、科学、医療、金融などの多くの領域に当てはまります。

このことは業界にとって何を意味するのでしょうか。また、オープン・スタンダードや UXL Foundation とはどのように関係しているのでしょうか。

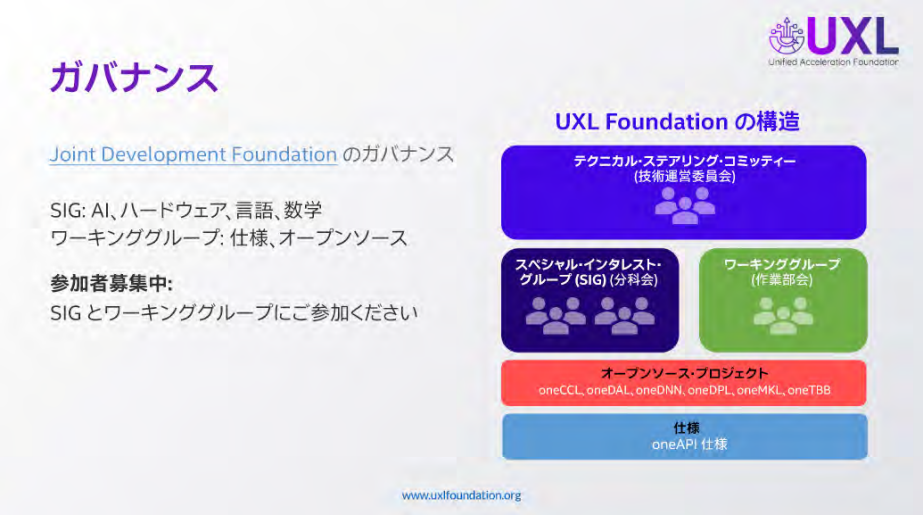
GPU の利用が一般化し、特定のワークロード（AI など）向けに設計されたプロセッサーの使用が増えることにより、ソフトウェア開発者は、マルチベンダーおよびマルチアーキテクチャーのシステム向けに高度な並列ソフトウェアを開発するという課題に直面しています。私の同僚で UXL ステアリング･コミッティー（運営委員会）のメンバーでもある Robert Cohn は、Linux* と GNU* がオープンソースとオープン・スタンダードを使用して、特にデータセンターで、CPU のソフトウェア・スタックをどのように変換したかを思い出させてくれました。その結果、開発者はソフトウェアを作成するときに、どの種類の CPU 向けにコンパイルしているのか考える必要がなくなりました。これは GPU や AI アクセラレーターではまだ実現していませんが、業界は協力して解決に取り組んでいます。

Unified Acceleration（UXL）Foundation（英語）の設立は 2023 年 9 月に発表(英語)され、ハードウェアとソフトウェアの業界全体からメンバーが集まりました。UXL Foundation は、Linux Foundation の一部であり、オープンで独立したガバナンスモデルを採用しています。UXL Foundation の目標は、オープン・スタンダードとオープンソース・ソフトウェアを中心とするアクセラレーター・エコシステムを統合し、開発者が現在および将来にわたりマルチベンダー、マルチアーキテクチャー･システムをターゲットにできるアプリケーションを構築できるようにすることです。開発者がソフトウェアを作成するときにターゲットにするプロセッサーについて考える必要がなくなれば、我々の仕事は完了です。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

UXL Foundation は、oneAPI 仕様とこの仕様を実装するプロジェクトの上に成り立っています。これらのプロジェクトはすでにマルチベンダー・ターゲットを開発者に提供しており、メンバーは今後もこれらをベースに開発を継続します。仕様とプロジェクトはインテルから UXL Foundation に提供されており、開発者が高速化されたアプリケーションとワークロードを作成するのに必要な基礎をカバーしています。これらのプロジェクトは、UXL Foundation により定義されたオープンガバナンスの原則とプロセスに基づいて運営されます。重要なことは、すべての貢献が平等に扱われ、オープンな提案や議論を通じてプロジェクトの将来を決定する上で、コミュニティーが発言権を持つことができるという点です。

UXL Foundation は、2 つの基本的な分野の共同作業を可能にする方法で組織されています。まず、プロジェクトに関する議論とフィードバックを促進することを目的とした[スペシャル・インタレスト・グループ (SIG) (分科会)](英語) があります。これらのフォーラムでは、プロジェクトの新しい開発分野とアプリケーションに関する、メンテナーとコントリビューターの両方によるプレゼンテーションやディスカッションが行われます。これらは完全にオープンなディスカッションの場であり、トピックへの参加や提案にメンバーシップは必要ありません。次に、仕様とプロジェクトに関連する共同作業を調整するワーキンググループ (作業部会) があります。これらのグループは最近結成されたもので、メンバーとコミュニティーが実行するワークパッケージを定義します。パブリック・ビルド・インフラストラクチャーや CI、プロジェクトのオープンソース・カーネルの拡張、コントリビューション・プロセスの定義など、あらゆる人に貢献する機会が提供されます。



私はほかの代表者とともにステアリング・コミッティーの委員長を務めており、各組織は UXL Foundation の方向性に関する高レベルな決定を下す際に平等な発言権と投票権を持っています。

ステアリング・コミッティーに参加できない場合でも、SIG、ワーキンググループ、ワークパッケージに参加して貢献することで、UXL Foundation の将来に対する重要な役割を同様に担うことができます。皆さんの意見は重要です。我々が間違ったことを行っていると思ったら、意見を送ってください。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

我々が正しいことを行っていると思ったときに意見を送ることも、もちろん歓迎します。我々は、UXL Foundation がコミュニティー全体のために機能することを望んでいます。

ウェブサイト（英語）および GitHub* リポジトリー（英語）で、UXL Foundation の今後の会議やイベント、ガバナンスの詳細を確認できます。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

BigDL プライバシー保護マシンラーニングを使用したトラスト TorchServe

インテル® トラスト・ドメイン・エクステンションズとコンフィデンシャル・コンテナで AI をセキュアに

Qiyuan Gong、Dongjie Shi、Peter Zhu、Yabai Hu、Wesley Du、Hanyu Jin、Arron Wang、および Junrui Zhang インテル コーポレーション
Yao Zhang および Dian Chen ByteDance

デプロイメントは、モデルを現実世界のアプリケーションに導入する AI ライフサイクルの重要なステージです。TorchServe は、PyTorch* モデルを効率的かつ柔軟にデプロイするオープンソースのツールです。運用環境では、モデルとユーザーデータの機密性と整合性を重要事項として考慮する必要があります。この記事では、インテル® トラスト・ドメイン・エクステンションズ（インテル® TDX）、コンフィデンシャル・コンテナ（CoCo）、BigDL プライバシー保護マシンラーニング（PPML）上に構築された、トラスト PyTorch* モデル・サービング・ソリューションを紹介します。これらのテクノロジーは、パフォーマンスのオーバーヘッドを最小限に抑えながら、TorchServe の機密性と整合性を保護します。このソリューションのベスト・プラクティスとして、Jeddak サンドボックスも紹介します。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

トラスト TorchServe ソリューション

コンフィデンシャル・コンピューティングは、処理中のデータの保護に重点を置いてセキュリティーとプライバシーを強化する手法です。信頼できる実行環境（TEE）は、ハードウェア・ベースのセキュリティー機能によりコンピューティング･システム内に分離された環境を作成するコンフィデンシャル･コンピューティングの基本的な構成要素です。最新の第 4 世代インテル® Xeon® スケーラブル･プロセッサーでは、インテル® TDX と呼ばれる特殊な TEE が提供され、トラストドメインと呼ばれる分離された仮想マシン（VM）内のデータを不正なソフトウェアから保護します。

CoCo（英語）は、ハードウェア TEE、リモート認証、関連ソフトウェア･テクノロジーを利用して、コンテナ化されたワークロードに機密性､整合性､信頼性を提供する Cloud Native Computing Foundation（CNCF）のサンドボックス･プロジェクトです。インテル® TDX コンフィデンシャル・コンテナ・ソリューションは CoCo ベースで、簡単に使用できます。アプリケーション開発者は、変更されていないコンテナイメージをインテル® TDX コンフィデンシャル・コンテナで実行できます。クラスターのオペレーター / 管理者は、共通のツールを使用して完全なソリューションをインストールし、Kubernetes* クラスターを管理できます。CoCo は、ランタイムだけでなくストレージやネットワーク上でも機密性の高い高価値のアプリケーション、データ、モデルに対するエンドツーエンドの保護を提供することにより、強力なセキュリティーを保証します（**図 1**）。

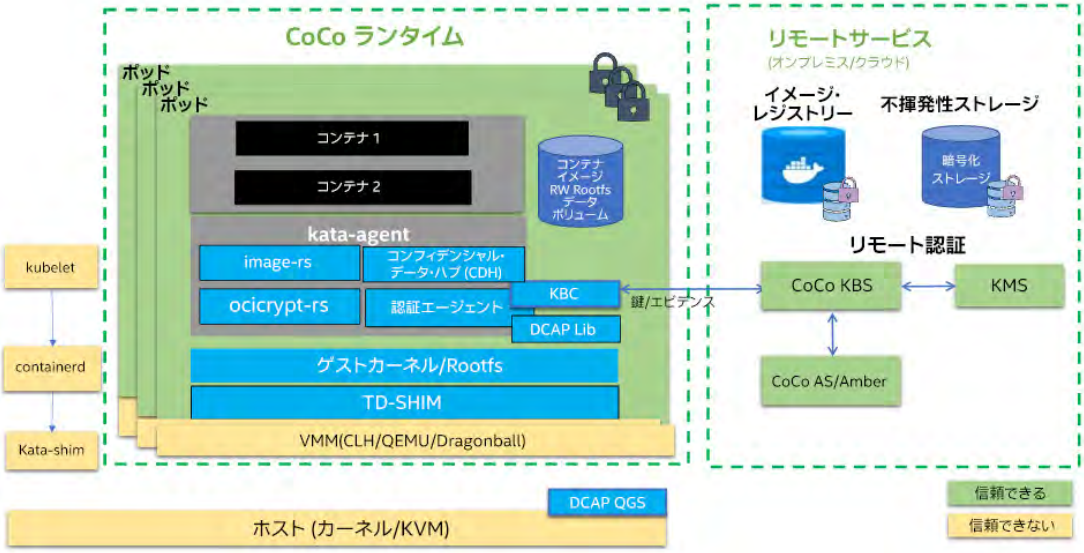


図 1. インテル® TDX コンフィデンシャル・コンテナ・アーキテクチャー

PPML（英語）は、エンドツーエンドの AI および分析アプリケーションのプライバシーとセキュリティーに対する新しい BigDL（英語）の機能です。インテル® ソフトウェア・ガード・エクステンションズ（インテル® SGX）、インテル® TDX、CoCo などの複数のセキュリティー・テクノロジーを組み合わせることにより、BigDL PPML を使用すると、ユーザーは既存のアプリケーションを変更することなく、ハードウェア保護とエンドツーエンドのセキュリティーを備えた状態で大規模に実行できます（**図 2**）。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

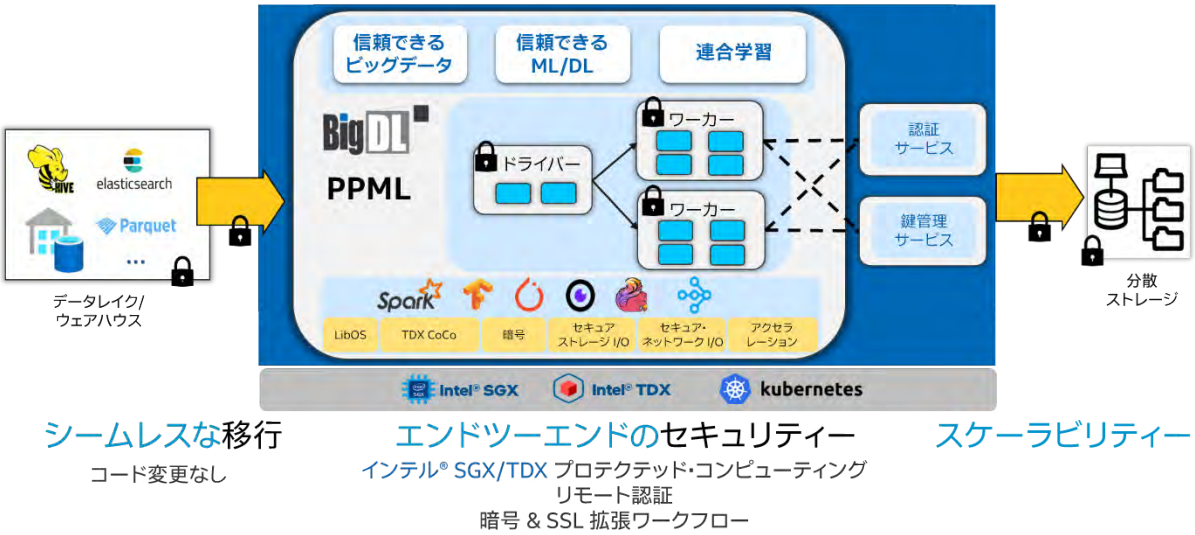


図 2. BigDL PPML アーキテクチャー

インテル® TDX CoCo をベースとし、BigDL PPML を使用して、Kubernetes* クラスターに簡単にデプロイできるトラスト TorchServe ソリューションを構築しました。**図 3** のように、次の 3 つの主要コンポーネントから構成されます。

- ロード可能なモデルが存在するモデルストア
- モデルを管理し、リクエストと応答を処理するフロントエンド
- 入力データを処理し、モデル推論を実行し、応答を生成するバックエンド

入出力データとモデルを保護するため、フロントエンドとバックエンドの両方がインテル® TDX CoCo 上で BigDL PPML を使用して実行されます。

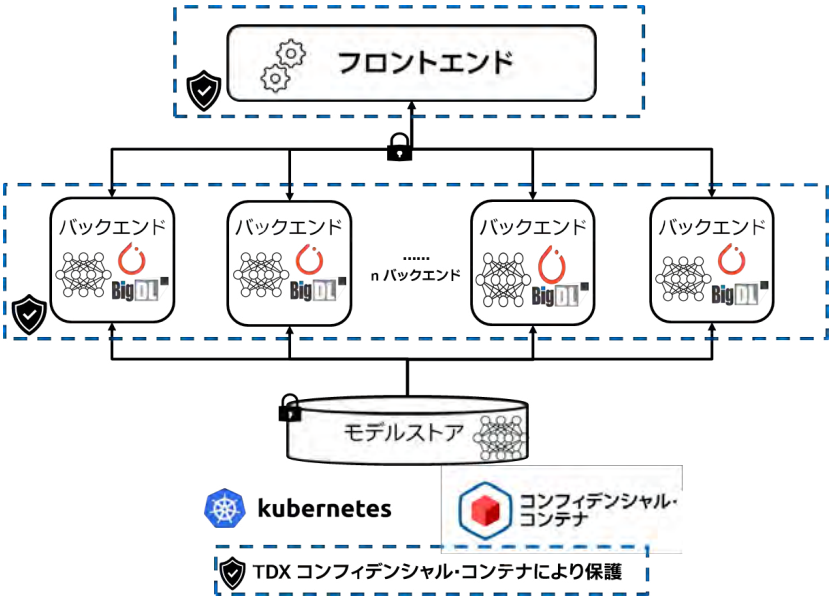


図 3. インテル® TDX コンフィデンシャル・コンテナと BigDL PPML 上のトラスト TorchServe

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

トラスト TorchServe ソリューションのワークフローは次のようになります。

1. モデルのオーナーが、モデルを準備してモデルストアに配置します。
2. モデルのオーナーが、CoCo ランタイムを使用してポッドを起動します。インテル® TDX により保護されたゲスト VM が作成され、VM 内でコンフィデンシャル・コンテナが開始します。
3. モデルのオーナーが、CoCo で TorchServe フロントエンド / バックエンドを開始します。
4. エンドユーザーが、トラスト TorchServe にサービングリクエストを送信し、推論の応答を受け取ります。

このワークフローでは、フロントエンドとバックエンドの両方がインテル® TDX CoCo により保護され、ユーザーモデルと入力データの機密性と整合性が保証されます。

セットアップ・ガイドおよびパフォーマンスのベスト・プラクティス

モデルを準備する

TorchServe でモデルをデプロイするには、特定の `mar` 形式のモデルファイル（すべてのモデルのアーティファクトを含む zip アーカイブ）が必要です。一般に、ユーザーはトレーニング済みモデル・チェックポイントとカスタム・モデル･ハンドラー･コードを準備し、TorchServe の torch-model-archiver ツールにフィードして、モデルファイルを生成する必要があります。インテル® プラットフォームのパフォーマンスを最大限に引き出せるように、BigDL Nano を使用してモデルを最適化することを推奨します。このツールは、ユーザーが PyTorch* 向けインテル® エクステンション（IPEX)、JIT、低精度のデータ型（BF16 や INT8 など）を含む、さまざまなソフトウェア最適化を活用するのに役立ちます。

環境をセットアップして設定する

トラスト TorchServe 環境をセットアップする前に、CoCo オペレーターが Kubernetes* クラスターにインストールされている（英語）ことを確認します。次に、トラスト TorchServe を実行するため、k8s ポッド内で CoCo を構成して開始します。フロントエンドとバックエンドのリソースを分離してパフォーマンスを向上するため、2 つのコンポーネントを異なるポッドに分割し、異なるセットの CPU コアを各ポッドに固定します。

Kubernetes* スタティック CPU マネージャー・ポリシーをセットアップして、kubelet 構成によるコンテナの CPU コアの固定を有効にします。

```
kubeReserved:
        cpu: "200m"
        memory: "1Gi"
systemReserved:
        cpu: "200m"
        memory: "1Gi"
cpuManagerPolicy: static
topologyManagerPolicy: best-effort
```

インテル® TDX CoCo でポッドを実行し、フロントエンドのポッドとバックエンドのポッドにリソースを割り当てるようにコンテナ・ランタイム・クラスを設定します。コンテナの CPU 固定を有効にするには、CPU リソースの `requests` と `limits` を整数の同じ値にする必要があることに注意してください。次に例を示します。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

```
…
spec:
  runtimeClassName: kata-qemu-tdx
  containers:
  - name: torchserve-backend
    image: intelanalytics/bigdl-ppml-trusted-dl-serving-gramine-ref:multipods-8G
    imagePullPolicy: Always
…
    resources:
      limits:
        cpu: "48"
        memory: "64Gi"
      requests:
        cpu: "48"
        memory: "64Gi"
```

フロントエンドとバックエンドのゲスト VM のリソースを設定します。VCPU とメモリーリソースは、前のステップのポッドの設定と同じにする必要があることに注意してください。次に例を示します。

```
default_vcpus = 48
default_memory = 65536
```

TorchServe フロントエンドとバックエンドのポッドを起動します。

TorchServe を開始する

フロントエンドとバックエンドのポッドが起動され、コンフィデンシャル・コンテナがポッド内に自動的に作成されたら、対応するコンテナ内のフロントエンドのコンポーネントとバックエンドのコンポーネントを開始できます。マルチコアまたはマルチソケット・サーバーで最適なパフォーマンスを実現するには、複数のバックエンド・ワーカーを起動し、同じ NUMA ノードの CPU コアとメモリーを各バックエンド・ワーカーにバインドすることを推奨します。TorchServe 設定ファイルでバックエンドのワーカー番号を指定し、`numactl` ユーティリティーを使用してバックエンドのワーカープロセスを開始できます。

パフォーマンス・ベンチマーク

2 ソケットの第 4 世代インテル® Xeon® スケーラブル・プロセッサーを搭載したサーバー上で BERT-Large モデルの BigDL PPML を使用して、インテル® TDX CoCo 上でトラスト TorchServe のパフォーマンスを測定しました。ベンチマークでは、フロントエンドのポッドに 2 つのコアと 5GB メモリー、バックエンドのポッドに 48 個のコアと 64GB メモリーを割り当て、12 個のバックエンド・ワーカーを開始して、各ワーカーに 4 つのコアを固定しました。BigDL Nano を使用して、インテル® アドバンスト・マトリクス・エクステンション（インテル® AMX）アクセラレーションを利用した低精度推論によりモデル・サービング・パフォーマンスを最適化しました。FP32、BF16、INT8 精度のモデルサービングのエンドツーエンドのスループットを評価しました。インテル® TDX CoCo により保護されたトラスト TorchServe ソリューションのパフォーマンスは、保護されていない場合とほぼ同じです（**図 4**）。オーバーヘッドは通常 6% 未満です。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

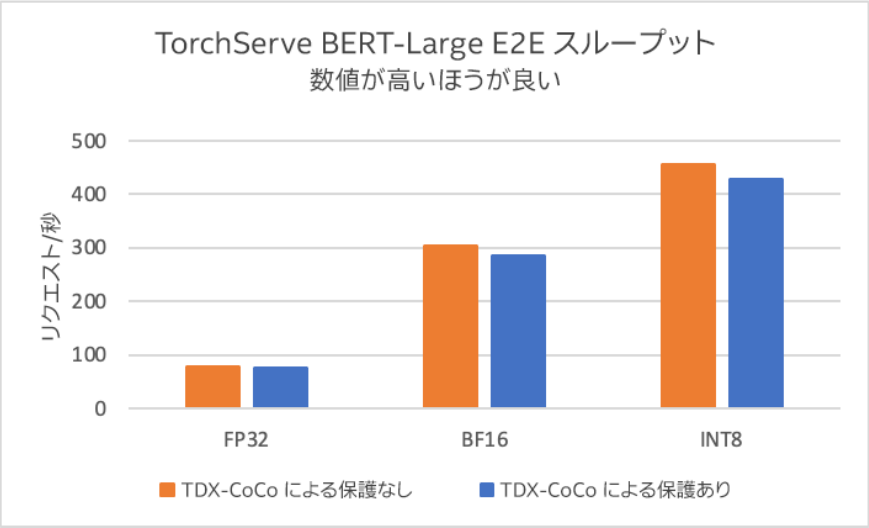


図 4. TorchServe BERT-Large のパフォーマンス

ByteDance のベスト・プラクティス

ByteDance は、社内および社外の企業ユーザーに安全かつ効率的なマシンラーニング・サービスを提供する Jeddak サンドボックスを発表しました。このソリューションは、ユーザーが AI シナリオにおけるさまざまなプライバシー・コンプライアンスの課題に対処するのを支援し、ユーザーがデータの価値を最大限に活用できるようにします。Jeddak サンドボックスは製品設計において BigDL と密接に連携し、そのセキュリティー機能とパフォーマンスの最適化を統合してユーザー・エクスペリエンスを向上させます（**図 5**）。

- BigDL Nano により提供される最適化手法を活用して、デプロイされたモデルを簡単に最適化し、モデル推論プロセスを加速して、ユーザー予測リクエストの応答レイテンシーを削減します。
- オンライン予測サービスのデプロイ方法としてインテル® TDX CoCo を採用しています。このアプローチは、サービングプロセス全体にハードウェア・ベースの保護を提供し、K8S の堅牢なコンテナ・オーケストレーション機能を組み合わせて、複雑で動的なリクエスト・ワークロードに対処するシームレスなスケーリングを実現します。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

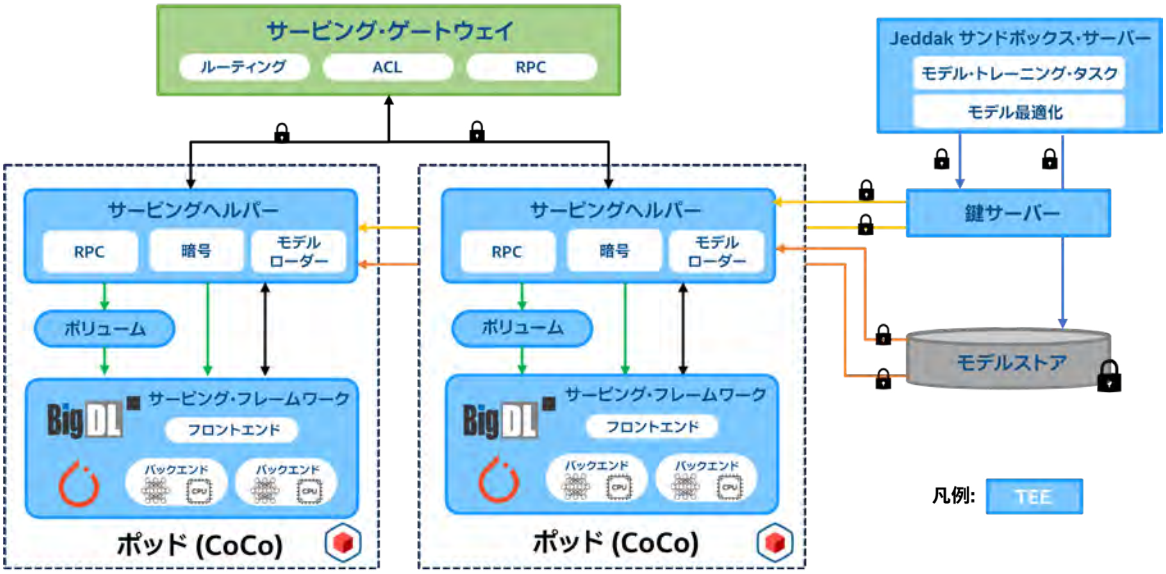


図 5. ByteDance の Jeddak サンドボックス・アーキテクチャー

Jeddak サンドボックスは、モデル予測プロセス中のさらなるセキュリティー強化を提供します。ユーザーのプライベート・データがハードウェアで保護された環境内でのみ復号化されることを保証する、ユーザーの推論リクエストのエンドツーエンド暗号化も含まれます。このソリューションにはモデルリクエストの認証および承認メカニズムも含まれており、ユーザーモデルへのアクセスを包括的に保護し、知的財産の漏洩を防ぎます。

まとめ

この記事では、サービング・コンポーネントが未承認のソフトウェアから保護されたトラストドメインに分離される、インテル® TDX 上の CoCo および BigDL PPML を利用したトラスト TorchServe ソリューションを紹介しました。インテル® TDX、CoCo、インテル® AMX アクセラレーションの長所を組み合わせることにより、ユーザーは、安全かつ柔軟でパフォーマンスの高い方法で PyTorch* モデルをデプロイできるようになります。また、トラスト TorchServe ソリューションを統合する ByteDance の Jeddak サンドボックスについても説明しました。今後も、モデルストアの暗号化 / 復号化や分散バックエンド・ワーカーのデプロイメントなどの新機能を継続的に追加し、モデル・サービング・パイプラインのエンドツーエンドのセキュリティーと柔軟性を強化します。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

ハードウェアとソフトウェアの構成

ベンチマークは次のプラットフォームでインテルにより実施されました。

プラットフォーム	第 4 世代インテル® Xeon® スケーラブル・プロセッサー
サーバー	インテル® Xeon® Platinum 8475B プロセッサー インテル® ハイパースレッディング・テクノロジー無効 512GB DRAM
ホストシステムのソフトウェア	Ubuntu* 22.04.2 LTS カーネル 5.19.0-mvp15v2+4-generic
ゲストシステムのソフトウェア	Ubuntu* 20.04 カーネル vmlinuz-5.15-plus-TDX-102cc-tdx Qemu v7.1.0 Libvirt 8.6.0-2022.11.17.mvp2.el8 (tdxlibvirt-2022.11.17) TDVF mvp6-stable202211 TDX SEAM 1.0.01.01-mvp29.el8
クラウドネイティブのソフトウェア	Kubernetes* v1.24.0 CoCo v0.5.0
DL フレームワーク	TorchServe 0.7.1 PyTorch* 1.13.1 IPEX 1.13.100 Transformers 4.29.1 BigDL 2.4.0 snapshot
テスト実施日	2023 年 06 月 19 日

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

ベイズ・ディープラーニングの量子化フレームワーク

Bayesian-Torch でベイズ・ニューラル・ネットワークを簡単に量子化してデプロイ

Keyur Ranipa インテル コーポレーション AI フレームワーク・アーキテクト
Vrushabh Sanghavi インテル コーポレーション AI フレームワーク・エンジニア
Ranganath Krishnan インテル コーポレーション AI リサーチ・サイエンティスト

ディープ・ニューラル・ネットワーク（DNN）は、優れた精度とパフォーマンスにより、医療診断、自動運転車、天気予報など、さまざまなセーフティー・クリティカル・アプリケーションにおける不可欠なコンポーネントとなっています。しかし、トレーニング・データの過剰適合や、間違った予測を過信してしまうことがあります。これにより、分布外のデータを検出する能力や、信頼できる信頼区間を含む予測の不確実性に関する情報を提供する能力が制限されます。この問題は、説明可能性と安全性が必要な実際のアプリケーションでは極めて重要です。例えば、医療診断では、モデルは予測に不確実性の尺度を示す必要があります。

データに最も適合する重みパラメーターの決定論的ポイントの推定を見つけることを目的とした従来の DNN モデルとは異なり、ベイズ・ディープ・ニューラル・ネットワーク（BNN）モデルは、ベイズの定理に基づいて重みの事後分布を学習することを目的としています（**図 1**）。ベイズモデルの重みは推論中に学習された分布からサンプリングされ、複数のモンテカルロ・サンプルにより予測の不確実性を推定します。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

図 1. 大腸組織構造診断における不確実性に基づく選択的予測

BNN は、重みのサンプリングと複数の確率的フォワードパスにより追加のメモリーと計算コストが必要になるため、決定論的モデルよりもモデルのサイズが大きくなり、推論が遅くなります。これらの欠点のため、BNN を実際のアプリケーションにデプロイする際には大きな課題が生じます。ここで量子化の出番です。量子化は、重みと有効性を 8 ビット整数（int8）などの低精度データ型で表すことにより、BNN 推論のメモリーと計算コストを削減するのに役立ちます。

ここでは、広く利用されているベイズ・ディープラーニング（BDL）向けの PyTorch* ベースのオープンソース・ライブラリー、Bayesian-Torch（英語）を使用して構築された量子化モデルを含む BDL ワークロードを紹介します。Bayesian-Torch は、BDL の低精度の最適化をサポートします。Bayesian-Torch を使用して、量子化されたベイズモデルをインテル® アドバンスト・マトリクス・エクステンション（インテル® AMX）を搭載した第 4 世代インテル® Xeon® スケーラブル・プロセッサーにデプロイすることにより、モデルの精度と不確実性の質を犠牲にすることなく、完全な精度のベイズモデルと比較して ImageNet ベンチマークで 6.9 倍の推論速度の向上を達成しました。

BNN の量子化

Bayesian-Torch は、シンプルな API（`dnn_to_bnn()`）を使用して、あらゆる DNN モデルを BNN にシームレスに変換できます。我々は、IISWC 2023（IEEE International Symposium on Workload Characterization）（英語）で発表された論文「Quantization for Bayesian Deep Learning : Low-Precision Characterization and Robustness（ベイズ・ディープラーニングの量子化 : 低精度の特性化とロバスト性）」で、包括的な量子化ワークフローを紹介しました。以下の 3 つの簡単なステップで、トレーニング後の量子化（PTQ）を BNN に適用できます。

1. **準備 :**「オブザーバーの挿入」などの前処理タスクを実行して、静的量子化のモデルを準備します。
2. **キャリブレーション :** 代表的なデータを使用してキャリブレーションを行い、キャリブレーション統計を取得します。
3. **変換 :** モデルのすべてのテンソルと演算についてスケールとゼロ点を計算し、量子化可能な関数を低精度の関数に置換することにより、完全な精度のベイズモデルを量子化モデルに変換します。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

Bayesian-Torch 量子化フレームワークには、PyTorch*（英語）のような高水準 API があります（**図 2**）。

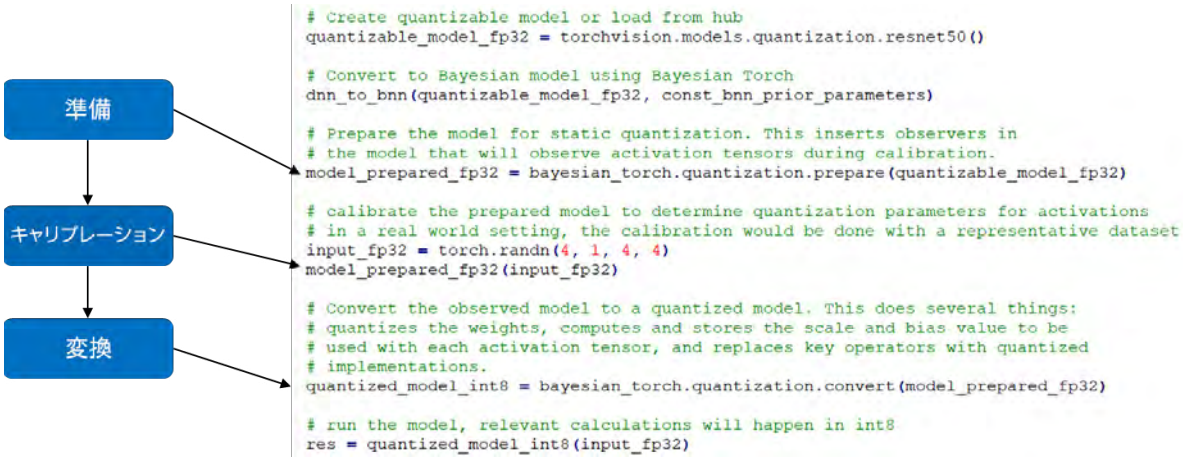


図 2. Bayesian-Torch でトレーニング後の量子化を実装するステップ

ImageNet ベンチマークを使用した ResNet-50 の実験

経験的ベイズ手法（英語）を使用して、重みの事前分布を指定し、事後分布を初期化することにより、平均場変分推論を使用してベイズ ResNet50 モデルをトレーニングしました。重みの事前分布は、Torchvision（英語）ライブラリーで利用可能な事前トレーニング済み ResNet50 決定論的モデルから初期化されます。ImageNet-1K（英語）データセットで Bayesian-Torch（英語）と PyTorch* 2.0 を使用して、初期学習率 0.0001、運動量 0.9、重みの減衰 0.0001 で SGD オプティマイザーを使用し、50 エポック実行してモデルをトレーニングしました。前述したように、トレーニング済みモデルにトレーニング後の量子化を適用し、量子化されたベイズ・ニューラル・ネットワーク（QBNN）を取得しました。

すべての検証は、AI ワークロードを高速化するインテル® AMX が搭載された第 4 世代インテル® Xeon® スケーラブル・プロセッサー上で行われました。インテル® AMX は、推論では INT8 精度を、推論とトレーニングでは Bfloat16 をサポートします。インテル® AMX は、メモリー内の大きな行列の部分行列を保持できる 2 次元レジスター（またはタイル）のセットと、タイル行列乗算アクセラレーターの 2 つのコンポーネントから成る新しい 64 ビット・プログラミング・パラダイムです。インテル® AMX を利用すると、単一の命令をタイルおよびアクセラレーター・ハードウェア上で複数のサイクルで実行できるようになります。

量子化されたベイズモデルの利点を実証するため、ImageNet-1K データセットで 1 つのモンテカルロ・サンプルを使用して、量子化前と量子化後のベイズ ResNet50 モデルの推論スループットを比較しました（**表 1**）。この結果は、量子化されたベイズモデルのスループットが完全な精度のベイズモデルよりも高く、最大 6.9 倍のスピードアップを達成していることを示しています。また、精度をほとんど低下させず、予測される不確実性のキャリブレーション誤差により定量化される不確実性の質を低下させることなく達成されています（**表 2**）。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

バッチサイズ	推論スループット ↑ (画像数/秒) BNN (FP32)	QBNN (INT8)	スピードアップ (QBNN vs. BNN)
32	218.5	420.9	1.93x
64	276.4	740.8	2.68x
128	301.1	1171.1	3.89x
256	342.7	1645.6	4.80x
512	366.4	1969.1	5.37x
1024	377.9	2358.3	6.24x
2048	395.1	2735.1	6.92x

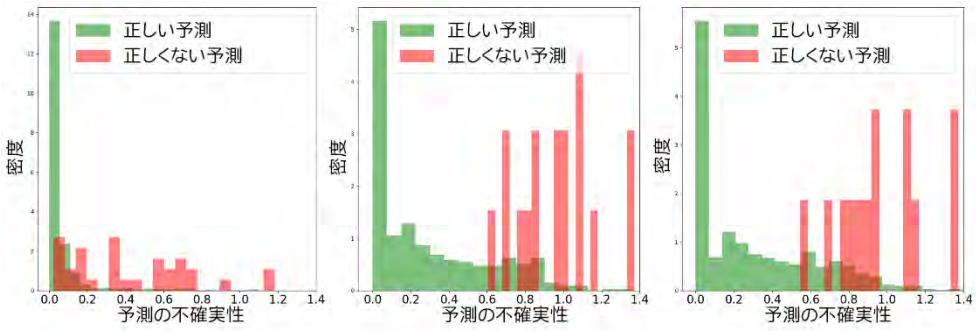
表 1. ImageNet-1K のさまざまなバッチサイズでの BNN と QBNN の推論スループットの比較

	BNN (FP32)	QBNN (INT8)
精度 ↑	76.10	75.71
UCE ↓	8.27	7.26

表 2. ImageNet での BNN と QBNN の精度と不確実性のキャリブレーション誤差（UCE）。矢印は数値が高いほうが良いか低いほうが良いかを示す。

実際のセーフティー・クリティカル・アプリケーション

医療診断、特に大腸組織構造画像分類（英語）に使用される BNN で、この量子化フレームワークを評価します。このデータセットは、ヒト大腸がんの 5,000 枚の組織構造画像のテクスチャーのコレクションです。トレーニングに 4,000 枚の画像、テストに 1,000 枚の画像を使用しました。各 RGB 画像は 150x150x3 です。モデルは画像を 8 つのクラスのいずれかに分類する必要があります。初期学習率 0.0001、運動量 0.9、重みの減衰 0.0001 で SGD オプティマイザーを使用し、50 エポック実行して、ResNet-50 アーキテクチャーの DNN モデルと BNN モデルをトレーニングしました。次に、トレーニング済みモデルにトレーニング後の量子化を適用し、QBNN モデルを取得しました。BNN のロバスト性と信頼性、後述する選択的予測および分布外（OOD）検出タスクについて、量子化後の不確実性の質を評価しました。



(a) DNN (FP32) (b) BNN (FP32) (c) QBNN (INT8)

図 3. 正しい予測と正しくない予測の不確実性の密度ヒストグラム

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

不確実性の高い予測はドメイン・エキスパートに照会し、不確実性の低い予測に基づいてモデルを評価する、選択的予測（**図 1**）を実行しました。**図 3** は、正しい予測と正しくない予測について DNN、BNN、QBNN から取得された予測の不確実性の密度ヒストグラムです。モデルが正しくない予測を行う場合には高い不確実性が得られ、正しい予測を行う場合には低い不確実性が得られるのが望ましい動作です。ベイズモデルは、DNN よりも信頼できる不確実性の推定を生成し、AI のエキスパートはモデルがいつ失敗するか知ることができることが分かりました。また、**図 3 (c)** から、不確実性の推定の質は量子化の影響を受けないことが分かります。

図 4 (a) は、照会率の関数として精度を比較したものです。モデルの精度が 99.5% を達成するときのドメイン・エキスパートへのサンプルの照会率は、標準 ResNet-50 では 27.5% であるのに対し、BNN と QBNN ではそれぞれ 6% と 8.5% のみです。これは、ドメイン･エキスパートへのデータ照会が少なくなると偽陰性と偽陽性が減ることを示しています。QBNN では DNN と比較して照会率の効率が 78% 向上しています。

Camelyon17-wilds（英語）データセットを使用して、OOD 検出のモデルをテストしました。このデータセットは、リンパ節切片の乳がん転移の全 50 枚のスライド画像から抽出された 450,000 個のパッチで構成されています。検証セットからランダムに選択された 1,000 枚の画像を OOD 検出のテストモデルに使用しました。OOD 検出は、データサンプルが分布データに属しているかどうかを識別する 2 項分類タスクです。教師なしアプローチである (つまり、モデルに OOD データの知識がない)、予測の不確実性の推定を使用して OOD データを検出するモデルの能力を評価します。**図 4 (b)** は、OOD 検出の ROC 曲線下の面積 (AUC) の比較です。BNN (AUC 0.83) は、DNN（AUC 0.41）よりも OOD サンプルを検出する能力が優れていることが分かります。また、BNN の量子化は OOD 検出の AUC にほとんど影響を与えていません。

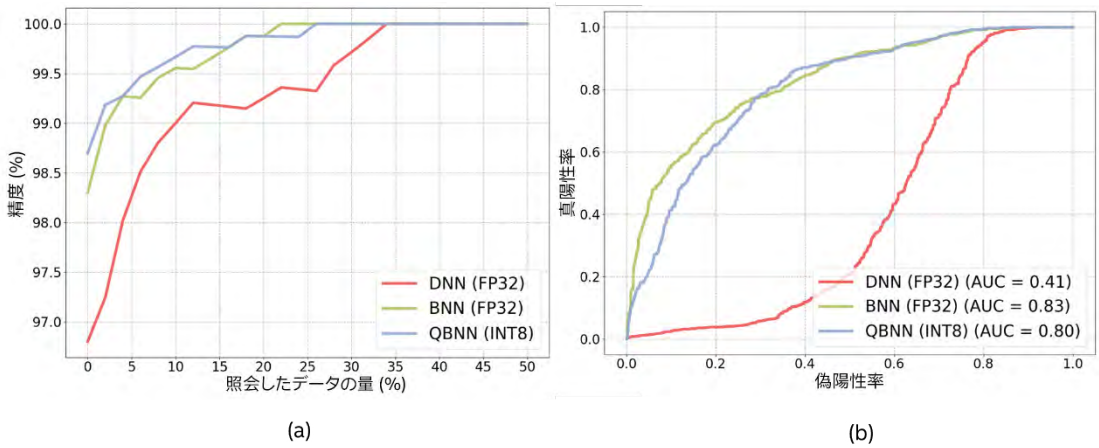


図 4. 予測の不確実性に基づいて照会したデータの量の関数としての精度

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

まとめ

この記事では、シンプルで使い慣れた API を使用して BNN のトレーニング後の量子化を可能にする、ベイズ・ディープラーニング向けの低精度の最適化フレームワークを紹介しました。Bayesian-Torch 量子化フレームワークを使用することで、最適化された 8 ビット整数（INT8）BNN は、モデルの精度、不確実性の質、ロバスト性を犠牲にすることなく、32 ビット浮動小数点（FP32）BNN と比較して最大 6.9 倍の推論スループットのスピードアップを達成し（**図 5**）、必要なメモリーを 4 分の 1 に減らしました。この結果は、大規模なデータセットと実際のアプリケーションにおける広範な実証分析を通じて得られたものです。量子化フレームワークのコードはオープンソース化されているため、実際のアプリケーションでベイズ・ディープラーニング・モデルが広範にデプロイされることを期待しています。

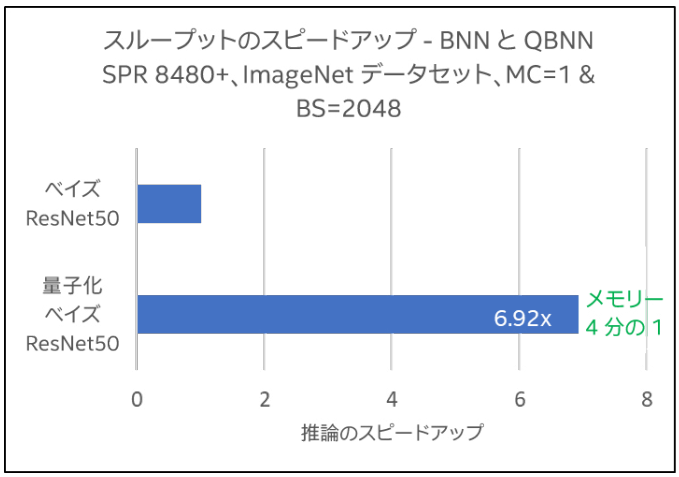


図 5. BNN と QBNN のスループットのスピードアップの比較

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

intel. software

私の見たところ…

コードに改良の余地があるようです

複数の言語をサポートし、ヘテロジニアス計算パフォーマンスを実現する、単一のオープンなプログラミング・モデルで、コードのパフォーマンスを強化しましょう。オープンな標準ベースの oneAPI は、クロスアーキテクチャーのライブラリー、コンパイラー、ツールを提供し、コードをより多くのハードウェアに対応させ、優れたパフォーマンスを実現します。

oneAPI

oneAPI を活用して、エンタープライズ、クラウド、HPC、AI などの分野の計算を高速化。

今すぐダウンロード

最先端のビジュアル品質検査

インテルにより最適化されたエンドツーエンドのモジュール式リファレンス・キットを使用した効率的な異常検出

Pratool Bharti インテル コーポレーション AI フレームワーク・エンジニア
Vrushabh Sanghavi インテル コーポレーション AI フレームワーク・エンジニア

異常検出は、時間と労力がかかるため、一般的な産業環境の大量データへの適用は制限されます。人工知能（AI）は、生産性の向上、優れた洞察、ダウンタイムの短縮、優れた製品品質を可能にすることで、産業用のモノのインターネット（IIoT）を変革しています。異常検出リファレンス・キット（英語）は、AI を活用した高解像度画像のビジュアル品質検査を提供し、生産ラインで生じる不良品のように、まれに発生する異常事象を特定します。概念的には、通常のデータポイントと統計的に異なる外れ値を検出するため、金融取引の不正検出、製造業の品質監視、スパムメールのフィルタリング、ネットワークの侵入検知などのユースケースにも適用できます。このリファレンス・キットは「そのまま」使用することも、アプリケーションの要件に合わせて変更することもできます。

異常検出にはいくつかの課題があります。生データから表現を抽出するには、特徴量エンジニアリングが必要です。従来のマシンラーニング手法は手動の特徴量エンジニアリングに依存しており、異なる設定に必ずしも一般化できるとは限りません。ラベル付きのトレーニング・データも必要になるため、データ収集とアノテーションのオーバーヘッドが増加します。異常はまれにしか発生しないため、適切なバランスのトレーニング・データを用意することは不可能です。つまり、標準的な分類手法は適していません。異常の性質は偶発的な可能性があり、欠陥はさまざまな予測不能な理由で発生する可能性があるため、異常の種類を予測することが不可能な場合があります。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

優れたパフォーマンスを達成しつつこれらの課題を克服するため、異常検出向けの、教師なし、混合方式のエンドツーエンドの調整および推論リファレンス・キットを利用することを提案します。このキットは、教師なし方式で正常なデータから正規性のモデルを学習し、モデルからの逸脱に対し異常としてフラグを立てます。このリファレンス・キットは、インテルにより最適化されたソフトウェア（英語）によって高速化され、使いやすいインテル® Transfer Learning Tool（英語）の API を使用して構築されています。

MVTec AD – 教師なし異常検出用の包括的な実際のデータセット

MVTec AD（英語）は、工業検査アプリケーションの異常検出に広く利用されているベンチマーク・データセットです。ボトル、タイル、木材など、15 の異なる物体カテゴリーの 5,000 枚を超える高解像度カラー画像から構成されており、すべてのカテゴリーで合計 60 種類の異常が含まれています（**表 1**）。各カテゴリーで、データセットは通常の画像のトレーニング・セットと 6 つの異なる種類の異常を含む画像のテストセットを提供します。異常は、物体にさまざまな変化（切れ目、傷、汚れなど）を加えることにより手動で導入されます。このデータセットは、現実の世界の異常を模倣した高難易度かつ現実的な画像セットを提供するため、異常検出手法のパフォーマンスを評価するのに役立ちます。

	Category	# Train	# Test (good)	# Test (defective)	# Defect groups	# Defect regions	Image side length
Textures	Carpet	280	28	89	5	97	1024
	Grid	264	21	57	5	170	1024
	Leather	245	32	92	5	99	1024
	Tile	230	33	84	5	86	840
	Wood	247	19	60	5	168	1024
Objects	Bottle	209	20	63	3	68	900
	Cable	224	58	92	8	151	1024
	Capsule	219	23	109	5	114	1000
	Hazelnut	391	40	70	4	136	1024
	Metal Nut	220	22	93	4	132	700
	Pill	267	26	141	7	245	800
	Screw	320	41	119	5	135	1024
	Toothbrush	60	12	30	1	66	1024
	Transistor	213	60	40	4	44	1024
	Zipper	240	32	119	7	177	1024
	Total	3629	467	1258	73	1888	-

表 1. MVTec AD データセットの概要。カテゴリーごとに、トレーニング・イメージとテストイメージの数、テストイメージに含まれる欠陥に関する情報が示される（出典（英語））。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

ビジュアル品質検査のパイプライン

このリファレンスは、分布外検出アプローチである深層特徴モデリング（英語）を使用しています。モデルは、トレーニング・データと異なる入力を検出すると、その入力を異常としてマークします。ビジュアル品質検査パイプラインには、データの前処理、特徴抽出、良画像のトレーニング、良および不良テスト画像の推論、評価メトリックの計算が含まれます。

データの前処理と特徴抽出

MVTec AD はトレーニング・データセットとテスト・データセットにすでに分割されているため、画像を正規化してサイズを変更し、特徴抽出モデルと互換性があるようにすることから始めます。すべての画像のサイズを [224, 224, 3] に変更し、ImageNet データセットの平均と標準偏差を使用して正規化します。特徴抽出では、畳み込みニューラル・ネットワークまたはその他のディープラーニング・アーキテクチャーを使用して、入力画像から高レベルの特徴を抽出します。

異常検出のトレーニング

このステップでは、抽出した特徴の正常パターンと異常パターンを区別する方法を学習して、異常を特定するようにモデルをトレーニングします（**図 1**）。ビジョンサブタスクのモデリングには 3 つのオプションがあります。

1. 事前トレーニング済みバックボーンは、ImageNet のような大規模なビジュアル・データセットでトレーニングされた ResNet-51 v1.5 を使用します。
2. シャム・ネットワーク・ベースの対照学習手法は、ラベルを使用しないでデータセットの意味のある表現を学習します。SimSiam（英語）自己教師あり学習には、1 つの基礎となる画像から 2 つの異なる拡張画像を生成できるデータローダーが必要です。目標は、両方の画像で同じ特徴を生成するようにネットワークをトレーニングすることです。ResNet モデルをバックボーンとして使用し、拡張データセットで調整して、より近い特徴の埋め込みにします。
3. SimSiam に似た、CutPaste（英語）と呼ばれる対照学習手法は、トレーニング中に使用する拡張が異なります。ResNet モデルをバックボーンとして使用し、画像パッチを切り取って大きな画像のランダムな場所に貼り付けるデータ拡張方法を適用した後、調整することにより、異常データを必要とすることなく、ハイパフォーマンスな異常検出のモデルを構築します。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

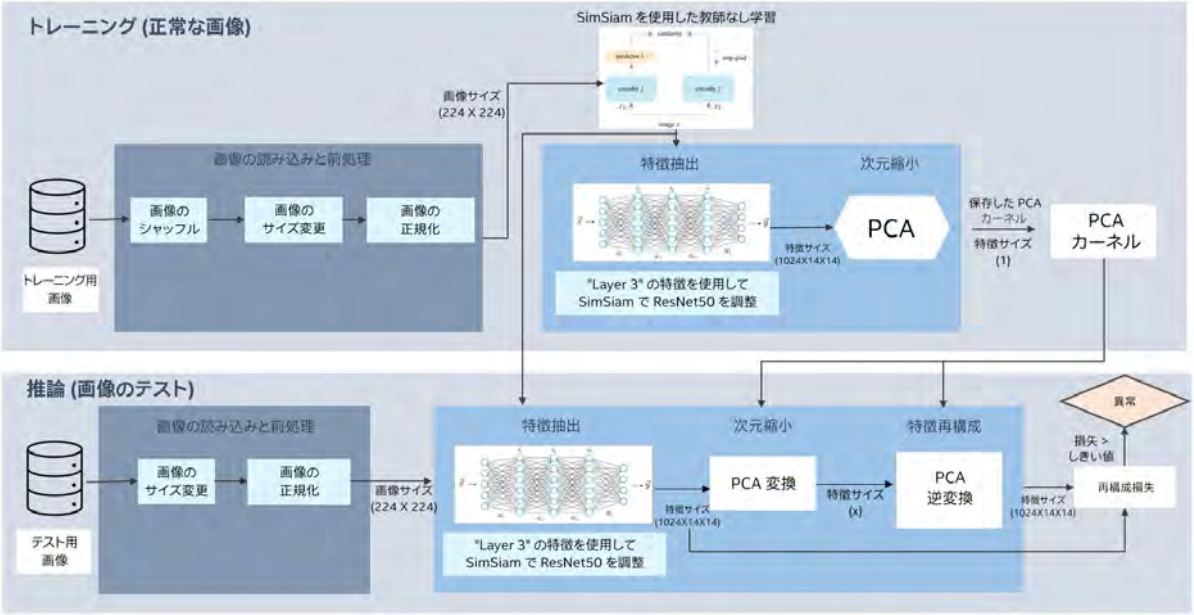


図 1. ビジュアル品質検査のパイプライン。この図は、SimSiam の自己教師ありトレーニングの例。

トレーニング段階では正常なデータのみ使用されることに注意してください。画像はロード後に、シャッフルされ、サイズが変更され、正規化されます。次に、3 つの転移学習手法のうち 1 つを使用してモデルを調整し、中間層から識別特徴を抽出します。主成分分析 (PCA) は、99% の分散を維持して次元を削減するために使用されます。行列の特異点やランク不足の発生を防ぐには、この DNN の中間特徴の前処理が必要です。

推論と評価のメトリック

推論中、テスト画像からの特徴は以前と同じネットワークを通じて生成されます。次に、以前にトレーニングした PCA カーネルを使用して PCA を実行し、逆変換を適用してオリジナルの特徴を再作成し、オリジナルの特徴ベクトルと対応する縮小埋め込みの事前画像との差のノルムである特徴再構成誤差スコアを生成します。異常を含む画像では、特徴が正常なトレーニング・セットの分布外にあるため、オリジナルの特徴を再構成する際に大きな誤差が発生します。異常な画像と正常な画像を区別する際のこれらのスコアの有効性は、受信者動作特性 (ROC) 曲線をプロットすることにより評価されます。この曲線は、分類スコアのしきい値を変化させた場合の、分類器の真陽性率と偽陽性率のプロットです。ROC 曲線下の面積 (AUROC) メトリックは、この曲線を 0 から 1 の間で要約します。1 は完璧な分類を示します。

結果

このビジュアル品質検査リファレンス・コードを実行した場合のサンプル出力を**表 2** に示します。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

Category	Test set (Image count)	AUROC	Accuracy (%)
BOTTLE	83	99.92	98.8
CABLE	150	94.36	88.67
CAPSULE	132	95.33	87.12
CARPET	117	91.65	83.76
GRID	78	86.3	82.05
HAZELNUT	110	99.25	97.27
LEATHER	124	99.9	98.39
METAL_NUT	115	93.3	90.43
PILL	167	96.02	86.83
SCREW	160	83.3	81.88
TILE	117	98.81	99.15
TOOTHBRUSH	42	96.11	88.1
TRANSISTOR	100	96.42	91.0
WOOD	79	99.3	97.47
ZIPPER	151	97.16	90.07

表 2. 異常検出リファレンス・ケースの精度

システム構成：シングルノード、デュアルソケットの第 4 世代インテル® Xeon® Platinum 8480+ プロセッサー、ソケット当たり 56 コア、インテル® ターボ・ブースト・テクノロジー有効、インテル® ハイパースレッディング・テクノロジー有効、1024GB メモリー（16x64GB）、メモリー速度 =4800MT/s、インテル® SSD DC S3710（SSDSC2BA012T4）、CentOS* 8、BIOS=EGSDCRB.86B.WD.64.2022.29.7.13.1329、CPU Governor=performance、intel-extension-for-pytorch v2.0.0、torch 2.0.0、scikit-learn-intelex v2023.1.1、pandas 2.0.1。精度 =bfloat16、バッチサイズ =32、事前トレーニング済み Resnet50 v1.50 モデルから特徴を抽出。

まとめ

この記事では、深層特徴抽出と分布外検出を使用する異常検出リファレンス・キットについて説明しました。このキットは、モデルの調整と特徴の抽出に、調整可能な、モジュール式ワークフローを使用します。モデルの調整と特徴の抽出には、インテル® Transfer Learning Tool を使用します。インテル® アーキテクチャーで最適なパフォーマンスを得るため、スクリプトは、PyTorch* 向けインテル® エクステンション、scikit-learn* 向けインテル® エクステンション、およびインテル® アドバンスト・マトリクス・エクステンション（インテル® AMX）を使用して第 4 世代インテル® Xeon® スケーラブル・プロセッサーで bfloat16 を実行するオプションが有効になっています。

このアーキテクチャーは、分布外工業異常サンプルに混合方式の、マルチモデル予測を実行するリファレンス・ユースケース（英語）で構成されています（**図 2**）。調整モジュールは、インテル® Transfer Learning Tool を使用する、個別の構成可能なモジュール式ワークフロー（英語）として利用可能です。モジュール式アーキテクチャーにより生産性が向上し、解決までの時間が短縮されます。転移学習手法は、インテル® Transfer Learning Tool の API をシームレスに使用する、独立したワークフローにより利用可能になります。設定ファイルを使用すると、ユーザーは、大幅なコード変更を行うことなく、パラメーターと設定を変更できます。特徴抽出の事前トレーニング済みモデルと中間層は柔軟に選択できます。リファレンス・キットは、デフォルトの状態で優れたパフォーマンスを提供する、インテルにより最適化された基本ツールを活用しています。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

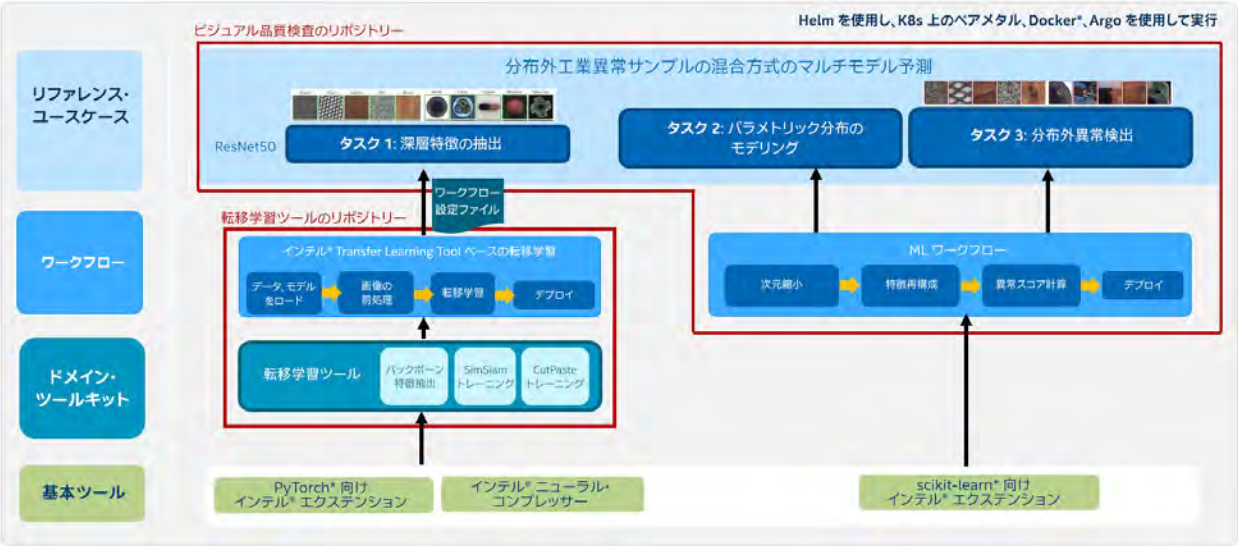
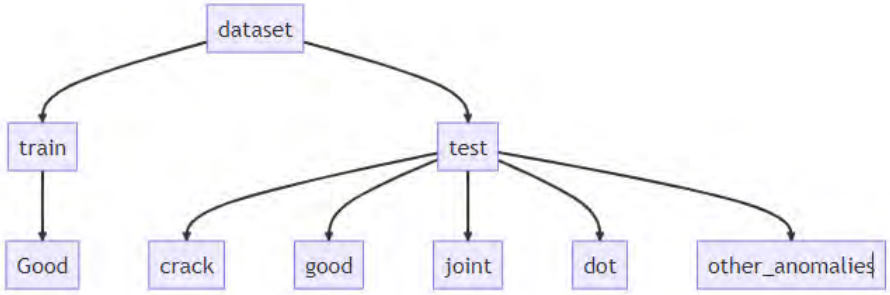


図 2. リファレンス・キットのアーキテクチャー

このリファレンス・キットをカスタマイズするための、調整可能な構成とパラメーターが公開されています。ユーザーは、YAML 設定ファイルを使用して、コードを変更または検索することなく、モデルのトレーニングのハイパーパラメーター、データ型、パス、データセット設定を変更できます。リファレンス・キットは、トレーニングとテスト用の画像をフォルダー構造に配置するだけで、データセットに簡単に導入できます。



このアプローチでは、正常な画像のみトレーニングに使用されることに注意してください。例えば、Kaggle の大理石表面の異常検出データセット（英語）に対して実行するには、データセットをダウンロードした後、Good フォルダーのみ含まれるように train フォルダーを更新します。train フォルダー内の異常な画像を含むサブフォルダーは、対応する test フォルダーに移動するか、削除します。

独自の事前トレーニング済みのカスタマイズされたモデルをプラグインすることもできます。詳細は、readme（英語）を参照してください。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

Spack を使用してインテルにより最適化された HPC バイナリーのディストリビューション

重要な HPC アプリケーションに最適な実行ファイルを構築する Spack レシピ

S. M. Iman Gohari, Ph.D. インテル コーポレーション ソフトウェア・イネーブリング & 最適化エンジニア

ハイパフォーマンス・コンピューティング（HPC）ユーザーは、パフォーマンス、互換性、精度をテストするため、異なるソフトウェアやパッケージのバージョン、依存関係、設定に対処する必要があります。この作業には時間がかかるため、ローレンス・リバモア国立研究所が主体となり、スーパーコンピューター・パッケージ・マネージャー (Spack)（英語）と呼ばれる効率的なパッケージ・マネージャーが開発されました。Spack は、その多くの利点（オープンソースで、シンプルな仕様ベースの構文を提供し、パッケージの作者 / メンテナーに同じパッケージの異なるビルドを処理する Python* インターフェイスを提供している、など）により、HPC コミュニティーの注目を集めています。Spack が 2023 年の HPCwire Editors' Choice Award の「ベスト HPC プログラミング・ツール / テクノロジー」（英語）を受賞したことは注目に値します。

この記事では、インテルのツール、ライブラリー、設定を使用してインテル® ハードウェアのパフォーマンスを最大限に引き出すことが難しい 3 つの重要な HPC アプリケーションで、最適な実行ファイルを構築する Spack レシピとガイドラインを示します。現在の作業における「最適な実行ファイル」という用語は、Spack を利用してそれらのツールを適切に使用し、インテル® HPC の顧客とパートナーに、インテル® Xeon® スケーラブル・プロセッサーで最適なパフォーマンスを達成しつつターゲットのオープンソース・アプリケーションを簡単に構築する一般的な方法を提供することを指します。これらのレシピは、Amazon Web Services*（AWS*）および Google Cloud

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

Platform*（GCP*）上の第 4 世代インテル® Xeon® スケーラブル・プロセッサーでテストされ、インテルにより最適化され、手動でビルドされたバイナリー（以下、「リファレンス・ビルド」と呼びます）と同等のパフォーマンスが得られています。

ターゲット・アプリケーション

HPC アプリケーションは、科学、エンジニアリング、ビジネスにおける複雑な問題を解決します。Spack を利用可能なパッケージのリスト（英語）は増え続けています。産業、研究、学術分野で広く使用されているため、さまざまな HPC 業界にわたる、次のアプリケーションが現在の作業に選択されています。

- Weather Research and Forecasting Model（WRF）（英語）は、大気のモデリングと運用予測のために設計された数値気象予測モデルです。WRF のビルドプロセスは、一連の依存関係が含まれているため、「かなり」複雑なものと考えられています。そのため、ビルドを簡素化する Spack の能力を評価するのに役立ちます。
- Large-scale Atomic/Molecular Massively Parallel Simulator（LAMMPS）（英語）は、材料のモデリングに重点を置いた古典的な分子動力学シミュレーターですが、異なるスケールの汎用粒子シミュレーターとして機能します。LAMMPS のビルドプロセスは簡単ですが、そのパフォーマンスはコンパイラー、命令セット、パフォーマンス・ライブラリーなどの選択に左右されます。そのため、ビルドプロセス中に発生する潜在的なパフォーマンスの低下を追跡するのに役立ちます。
- Open Source Field Operation and Manipulation（OpenFOAM）（英語）は、数値のモデリング、および主に数値流体力学向けの連続体力学の前処理と後処理のためのオープンソースのツールボックス・スイートです。コンパイラー間の依存関係があるため、コンパイルプロセスに比較的時間がかかります。そのため、コンパイラー間の依存関係や長いコンパイルを処理する Spack の能力を評価するのに役立ちます。

システム構成

クラウド・サービス・プロバイダー（CSP）（AWS* や GCP* など）は、オンプレミスのシステムで発生する初期コストや導入の困難さを回避しながら、アプリケーションとデータを処理するオンデマンドで、オープンアクセスの、スケーラブルなコンピューティング・リソースをユーザーに提供します。この記事では、公開されている CSP のインスタンスを使用して Spack レシピのパフォーマンスを評価し、読者がレシピを再現できるようにしています。**表 1** と**表 2** はそれぞれ、AWS* と GCP* のインスタンスの概要です。提供された Spack ガイドラインが包括的で有用であり、異なるインテル® アーキテクチャーでテストされていることを保証するため、2 つの CSP で提供されている 4 つの異なる世代のインテル® Xeon® プロセッサーを含むインスタンスでテストを行いました。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

システム構成	AWS* c5n.18xl	AWS* c6i.32xl
テスト実施者	Intel Corporation	Intel Corporation
CSP/ リージョン	AWS* us-east-2	AWS* us-east-2
インスタンス・タイプ	c5n.18xl	c6i.32xl
vCPU 数	72	128
ソケット	2	2
コアごとのスレッド数	2	2
NUMA ノード数	2	2
反復数と結果の選択	上位 3 つの反復の中央値	上位 3 つの反復の中央値
アーキテクチャー(開発コード名)	Skylake	Ice Lake
メモリー容量	192GB 2666MT/s	256GB 3200MT/s
SKU	インテル® Xeon® Platinum 8124M	インテル® Xeon® Platinum 8375C
ストレージタイプ	Amazon* FSx Luster	Amazon* FSx Luster
OS	CentOS* 7 (Core)	CentOS* 7 (Core)
カーネル	kernel-3.10.0-1160.88.1.el7.x86_64	kernel-3.10.0-1160.88.1.el7.x86_64
参考資料	Amazon* EC2 C5 インスタンス	Amazon* EC2 C6i インスタンス

表 1. AWS* インスタンスのシステム構成の詳細

システム構成	GCP* c2-stand-60	GCP* c3-highcpu-176
テスト実施者	Intel Corporation	Intel Corporation
CSP/ リージョン	GCP* us-central1	GCP* us-central1
インスタンス・タイプ	c2-stand-60	c3-highcpu-176
vCPU 数	30	176
ソケット	2	2
コアごとのスレッド数	2	2
NUMA ノード数	2	4
反復数と結果の選択	上位 3 つの反復の中央値	上位 3 つの反復の中央値
アーキテクチャー(開発コード名)	Cascade Lake	Sapphire Rapids
メモリー容量	240GB	352GB
SKU	インテル® Xeon® CPU @ 3.10GHz	インテル® Xeon® Platinum 8481C @ 2.70GHz
ストレージタイプ	標準永続ディスク	標準永続ディスク
OS	CentOS* 7 (Core)	CentOS* 7 (Core)
カーネル	3.10.0-1160.81.1.el7.x86_64	3.10.0-1160.81.1.el7.x86_64
参考資料	C2 マシンシリーズ	C3 マシンシリーズ

表 2. GCP* インスタンスのシステム構成の詳細

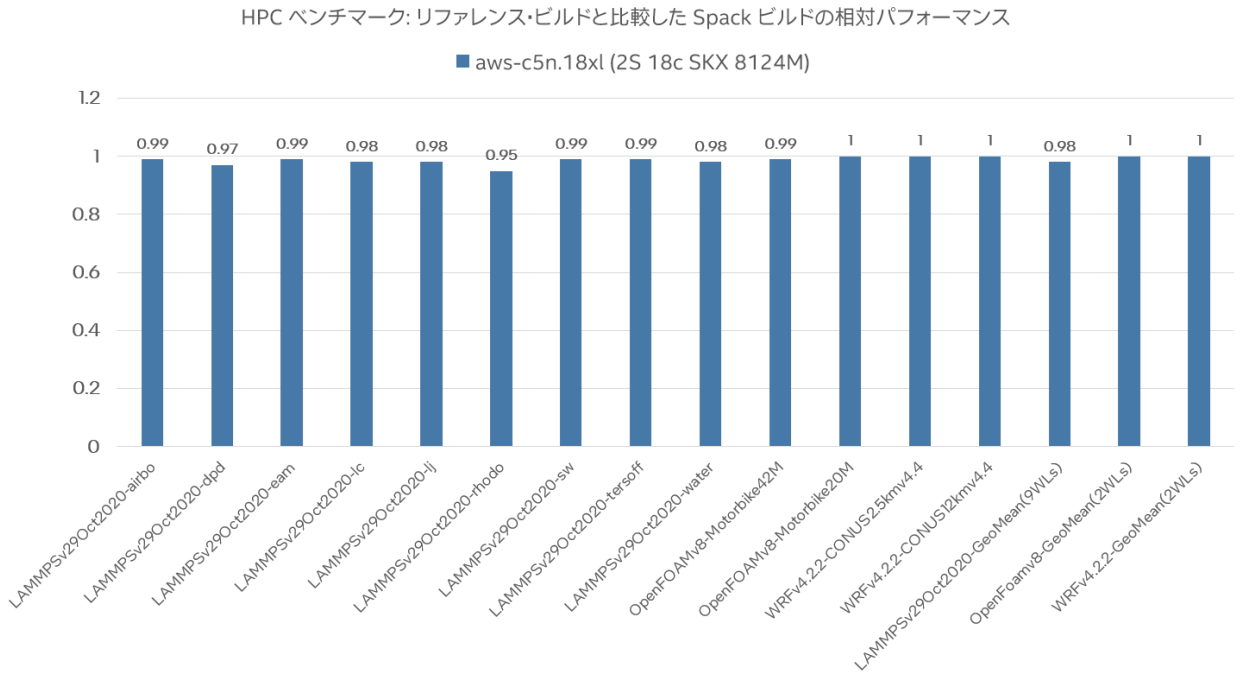
© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

パフォーマンス解析

前述の 3 つのアプリケーションの 13 のワークロードのパフォーマンスを、次の点を考慮して評価しました。

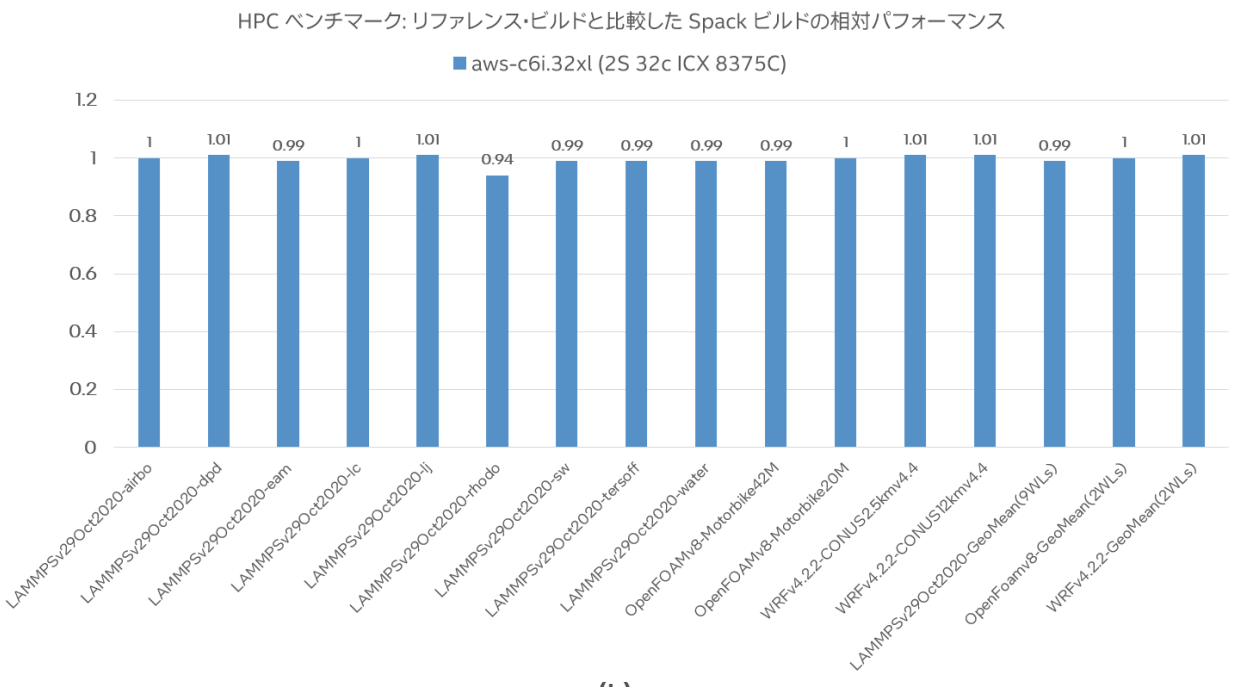
- Spack バイナリーを対応するリファレンス・ビルドと比較する際、同様の（または同じ）コンパイラーやアプリケーションのバージョンを利用して、ソフトウェア・スタックの変更を最小限に抑える。
- 同じベンチマーク・ツール、環境変数 / 設定、ワークロード・タイプ / バージョン、入力データセット、その他を使用して、各ビルドのパフォーマンスを測定する。

図 1 は、AWS* インスタンス上の対応するリファレンス・ビルドと比較した Spack ビルドのパフォーマンスを示しています。実行ごとの変動に伴う 10% の許容誤差を考慮すると、リファレンス・ビルドの予想パフォーマンスが、調査したすべてのワークロード間で Spack ビルドを使用して（ほぼ）再現されています。同じテストと実行プロセスを GCP* のインスタンスで実行したときの、対応するパフォーマンス結果を**図 2** に示します。繰り返しますが、リファレンス・ビルドと Spack バイナリーのパフォーマンスはほぼ同等です。



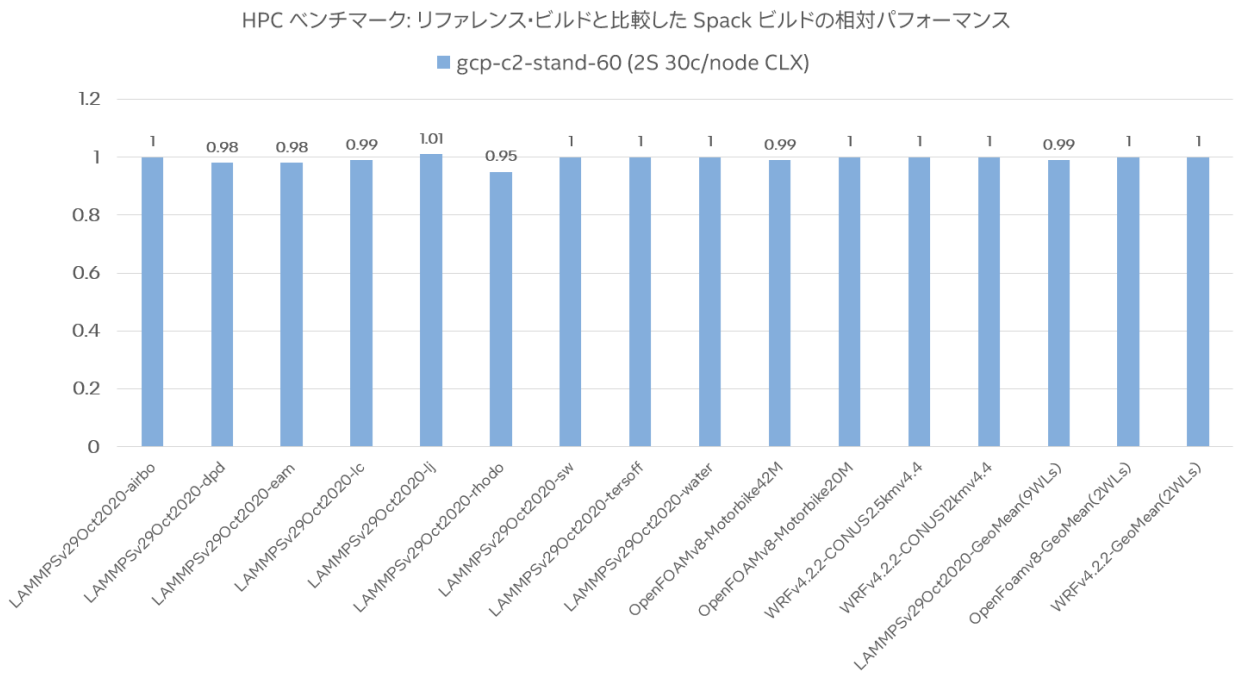
(a)

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。



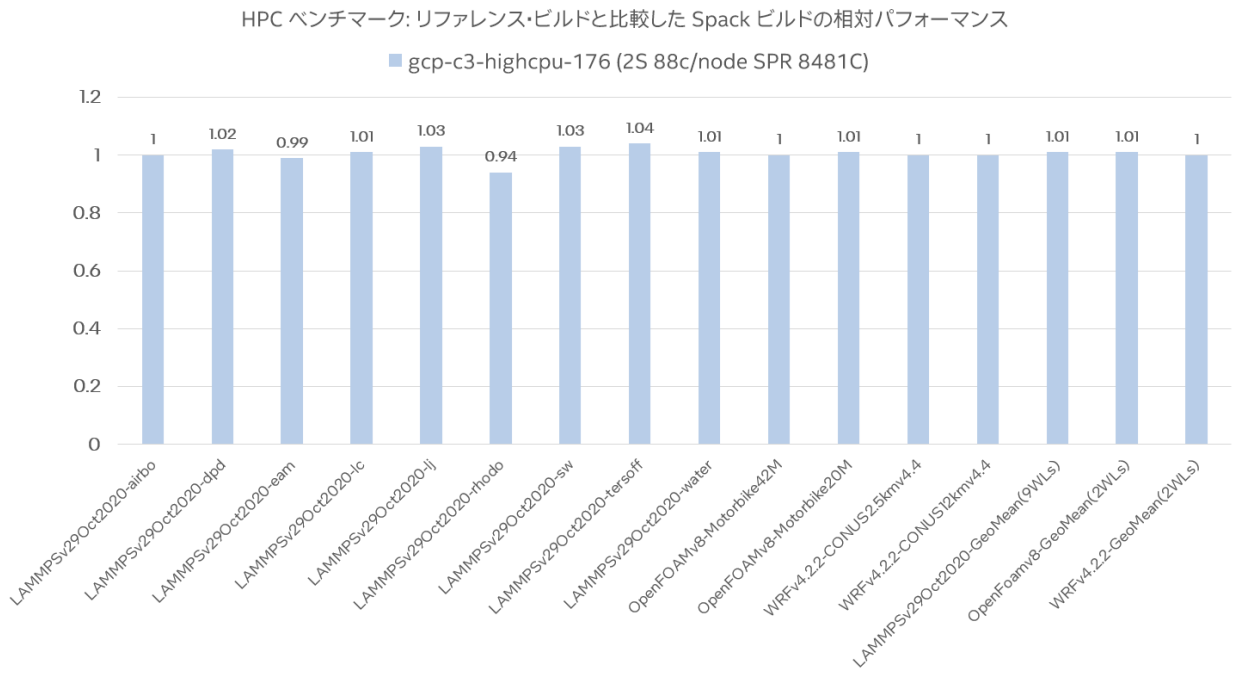
(b)

図 1. AWS* でテストした HPC ベンチマークの Spack ビルドとリファレンス・ビルドの相対パフォーマンス：(a) c5n.18xlarge、(b) c6i.32xlarge



(a)

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。



(b)

図 2. GCP* でテストした HPC ベンチマークの Spack ビルドとリファレンス・ビルドの相対パフォーマンス : (a) c2-stand-60、(b) c3-highcpu-176

調査対象の HPC アプリケーションには、リファレンス・ビルドと比較するための客観的に定義された「ベースライン」構成が存在しないことに注意してください (WRF のコンパイルには 70 を超えるオプションが用意されているためです)。主観的な結論を避けるため、「インテルにより最適化されたバイナリーとベースライン」の比較は行っていません。

Spack レシピ

HPC アプリケーションのエンドツーエンドのインストールは、システム・アーキテクチャー、依存関係の可用性、アプリケーションのコンパイル時間などによって異なります。Spack は、そのようなインストール (および必要なすべての依存関係) を合理化することにより、「使いやすさ」を提供します。手動プロセスと比較してエンドツーエンドの時間のオーバーヘッドはありません。**表 3 ~ 5** は、v0.20.0 リリース (英語) を使用して LAMMPS、OpenFOAM、WRF のパフォーマンス解析セクションの結果を取得するために使用した Spack レシピの概要です。これらの重要な HPC アプリケーション向けの最適なバイナリーを構築するには、Spack レシピ (または同様のもの) を使用することを推奨します。Spack レシピを使用する前提条件として、基本的な Spack インストール (英語) とインテル® oneAPI パッケージのインストール (英語) が必要なことに注意してください。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

アプリケーション	LAMPS
ビルドコマンド	$spack install lammps@20201029 +asphere +class2 +kspace +manybody +misc +molecule +mpiio +opt +replica +rigid +user-omp +user-intel %intel@2021.6.0 ^intel-oneapi-mkl ^intel-oneapi-mpi target=skylake_avx512
バージョンとワークロード	V20201029。airebo、dpd、eam、lc、lj、rhodo、sw、tersoff、water
ノート	Spack のソースコードは変更していません。

表 3. LAMMPS の最適なビルド向けの Spack レシピ

アプリケーション	OPENFOAM
ビルドコマンド	$spack install openfoam-org@8%intel@2021.6.0 ^intel-oneapi-mpi target=skylake_avx512 ^scotch%gcc
バージョンとワークロード	V8。 MotorBike 20M（250 反復）、MotorBike 42M（250 反復）
ノート	Spack のソースコードは変更していません。

表 4. OpenFOAM の最適なビルド向けの Spack レシピ

アプリケーション	WRF
ビルドコマンド	$spack install wrf@4.2.2%intel@2021.6.0 build_type=dm+sm ^intel-oneapi-mpi ^perl%intel ^libxml2%intel ^tar%oneapi@2022.1.0 ^jasper@2.0.32 ^m4%gcc@8.5.0
バージョンとワークロード	V4.2.2。CONUS-12km（v4.4）、CONUS-2.5km（v4.4）
ノート	WRF のタイミングを改善する推奨オプション（英語）を Spack ソースコード（英語）に反映できます。推奨オプションは精度に影響を与える可能性があることに注意してください（詳細（英語））。

表 5. WRF の最適なビルド向けの Spack レシピ

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

まとめ

HPC アプリケーションは、さまざまな複雑な問題を解決するのに役立ちますが、アプリケーションの構築とテストのプロセスは、単調でコストがかかり、時間もかかります。Spack は、HPC パッケージのユーザーに、自動構築と依存関係を追跡するシンプルな、オープンソースの、仕様ベースのインターフェイスを提供します。この記事では、Spack を使用して、最適なパフォーマンスを達成しつつ、重要な HPC アプリケーションを簡単に効率良く構築できることを説明しました。パフォーマンス解析は、AWS* と GCP* で動作している 4 つの異なる世代のインテル® Xeon® プロセッサー上で行いました。この記事で説明した Spack レシピ（または同様のもの）を使用して、紹介した HPC アプリケーション向けの最適なバイナリーを構築することを推奨します。

その他の Spack の資料（英語）

- Spack : HPC ソフトウェアのための柔軟なパッケージ・マネージャー
- GitHub* の Spack リポジトリー
- チュートリアル : Spack 101
- Spack のドキュメント
- Spack のインテル® oneAPI パッケージ

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。
コンパイラーの最適化に関する詳細は、最適化に関する注意事項を参照してください。

intel software

THE PARALLEL UNIVERSE

インテルのテクノロジーを使用するには、対応したハードウェア、ソフトウェア、またはサービスの有効化が必要となる場合があります。詳細については、OEM または販売店にお問い合わせいただくか、http://www.intel.co.jp/ を参照してください。

実際の費用と結果は異なる場合があります。

インテルは、サードパーティーのデータについて管理や監査を行っていません。ほかの情報も参考にして、正確かどうかを評価してください。

最適化に関する注意事項 : インテル® コンパイラーでは、インテル® マイクロプロセッサーに限定されない最適化に関して、他社製マイクロプロセッサー用に同等の最適化を行えないことがあります。これには、インテル® ストリーミング SIMD 拡張命令 2、インテル® ストリーミング SIMD 拡張命令 3、インテル® ストリーミング SIMD 拡張命令 3 補足命令などの最適化が該当します。インテルは、他社製マイクロプロセッサーに関して、いかなる最適化の利用、機能、または効果も保証いたしません。本製品のマイクロプロセッサー依存の最適化は、インテル® マイクロプロセッサーでの使用を前提としています。インテル® マイクロアーキテクチャーに限定されない最適化のなかにも、インテル® マイクロプロセッサー用のものがあります。この注意事項で言及した命令セットの詳細については、該当する製品のユーザー・リファレンス・ガイドを参照してください。注意事項の改訂 #20110804。

性能に関するテストに使用されるソフトウェアとワークロードは、性能がインテル® マイクロプロセッサー用に最適化されていることがあります。

SYSmark* や MobileMark* などの性能テストは、特定のコンピューター・システム、コンポーネント、ソフトウェア、操作、機能に基づいて行ったものです。結果はこれらの要因によって異なります。製品の購入を検討される場合は、他の製品と組み合わせた場合の本製品の性能など、ほかの情報や性能テストも参考にして、パフォーマンスを総合的に評価することをお勧めします。構成の詳細は、補足資料を参照してください。性能やベンチマーク結果について、さらに詳しい情報をお知りになりたい場合は、http://www.intel.com/benchmarks/（英語）を参照してください。

性能の測定結果はシステム構成の日付時点のテストに基づいています。また、現在公開中のすべてのセキュリティー・アップデートが適用されているとは限りません。詳細は、システム構成を参照してください。絶対的なセキュリティーを提供できる製品またはコンポーネントはありません。

本資料は、（明示されているか否かにかかわらず、また禁反言によるとよらずにかかわらず）いかなる知的財産権のライセンスも許諾するものではありません。

インテルは、明示されているか否かにかかわらず、いかなる保証もいたしません。ここにいう保証には、商品適格性、特定目的への適合性、および非侵害性の黙示の保証、ならびに履行の過程、取引の過程、または取引での使用から生じるあらゆる保証を含みますが、これらに限定されるわけではありません。

© Intel Corporation. Intel、インテル、Intel ロゴ、その他のインテルの名称やロゴは、Intel Corporation またはその子会社の商標です。

* その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。