aws DEV DAY
ONLINE JAPAN

E - 3

ディープラーニング推論を高速化せよ
~量子化技術とインテル®DLブーストのご紹介~

インテル株式会社
APJデータセンター・グループ・セールス
AIテクニカル・ソリューション・スペシャリスト
大内山　浩

intel

In Partnership with intel

本セッションはディープラーニングの推論に関する技術的な Deep Dive セッションです

対象者

- 量子化技術を詳しく知りたい方
- インテル® ディープラーニング・ブーストを使いたい方
- AI の技術寄りの方

内容

- 前提技術であるモデルの量子化
- インテル® ディープラーニング・ブースト
- 具体的な実行手順 (デモ)

インテルAI
概要

モデルの
量子化とは

DLブース
ト/デモ


インテルAI
概要
DLブース
ト/デモ
まとめ

インテル製品で AI への取り組みを加速

発見
可能性と次のステップ

データ
セットアップ、取り込み、クリーニング

開発
分析 / AI を使用するモデル

導入
実稼動へ & 反復

エコシステム

インテル® AI ビルダーズ・プログラム
100 以上の垂直的 / 水平的エコシステムのソリューション

最適化されたクラウド
Amazon Web Services*

AI に最適化された構成
intel select solution

ソフトウェア

データ分析
50 以上の最適化されたソフトウェア・プラットフォーム

マシンラーニング
Python* 向けインテル® ディストリビューション
ANALYTICS ZOO

ディープラーニング
TensorFlow PyTorch
OpenVINO

ハードウェア

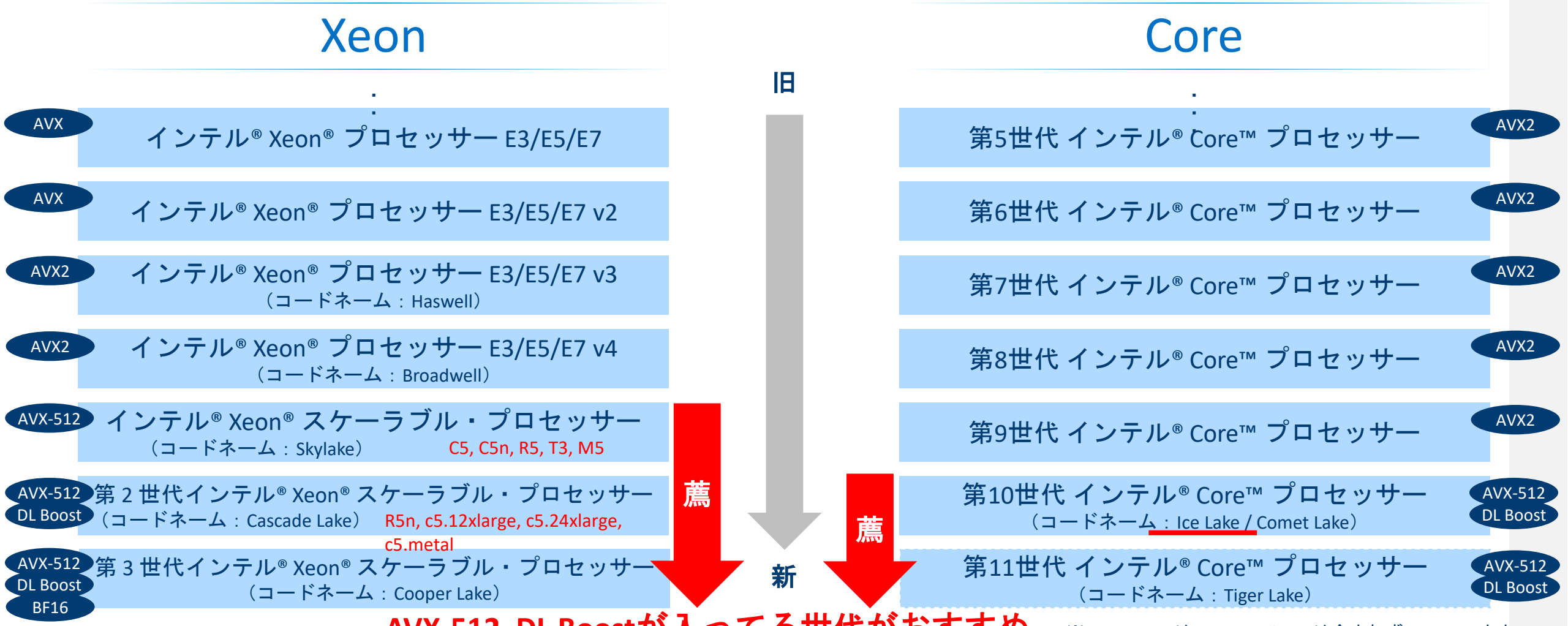
転送
intel Ethernet
BAREFOOT
intel Silicon Photonics

格納
intel OPTANE DC PERSISTENT MEMORY
intel OPTANE DC SOLID STATE DRIVE
インテル® 3D NAND SSD

処理
intel CORE i7 10TH GEN
intel XEON
Xe
intel MOVIDIUS
habana An Intel Company

すべての製品、コンピューター・システム、日付、および数値は、現在の予想に基づくものであり、予告なく変更されることがあります。最適化に関する注意事項

インテル® CPUの世代をどれだけ知っていますか？



AVX-512, DL Boostが入ってる世代がおすすめ

※ Comet Lakeは、AVX512/VNNIは含まれず、AVX2のまま

https://www.intel.co.jp/content/www/jp/ja/architecture-and-technology/avx-512-overview.html

インテル® AI ソフトウェア: マシンラーニングとディープラーニング

Developer TOols
App Developers
SW Platform Developer

Topologies & Models
Data Scientist

Frameworks
Data Scientist

Graph
ML Performance Engineer

Kernel
ML Performance Engineer

Machine learning	Deep LEarning	Management Tools
ANALYTICS ZOO	OpenVINO	Containers
▪ **Intel Distribution for Python** (SKlearn, Pandas)	TensorFlow PyTorch	kubernetes Kubeflow Deep Learning Reference Stack
▪ Intel Data Analytics Acceleration Library (Intel DAAL) ▪ Intel Math Kernel Library (Intel MKL)	▪ Intel Machine Learning Scaling Library (Intel MLSL) ▪ Intel® Deep Neural Network Library (DNNL)	Data Analytics Reference Stack
CPU	CPU ▪ GPU ▪ FPGA ▪ 専用	Architect & DevOps

Red font products are the most broadly applicable SW products for AI users

AIライフサイクル
~モデルの構築／モデルの運用~

インテルはAIのライフサイクル全体にわたってお客様と連携

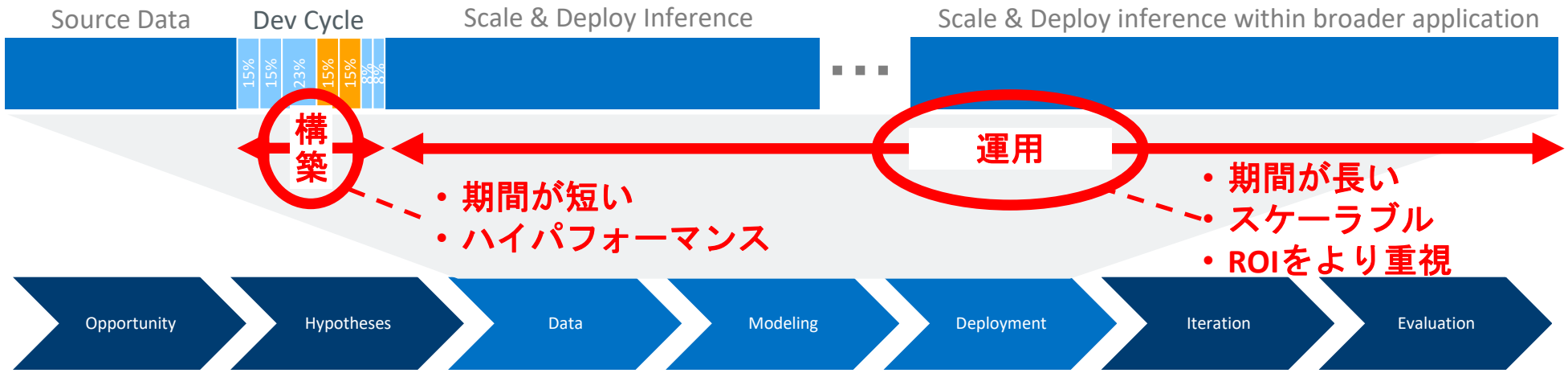
アクセラレータ

モデルの構築
（学習メイン、推論少々）
目標の精度を目指し、限られた期間内にひたすらモデルを作り続けるフェーズ

モデルの運用
（推論メイン、学習少々）
モデルを本番環境へ展開し新規ビジネスデータをひたすら推論するフェーズ

CPU

Build, Deploy & Scale TIME-TO-SOLUTION



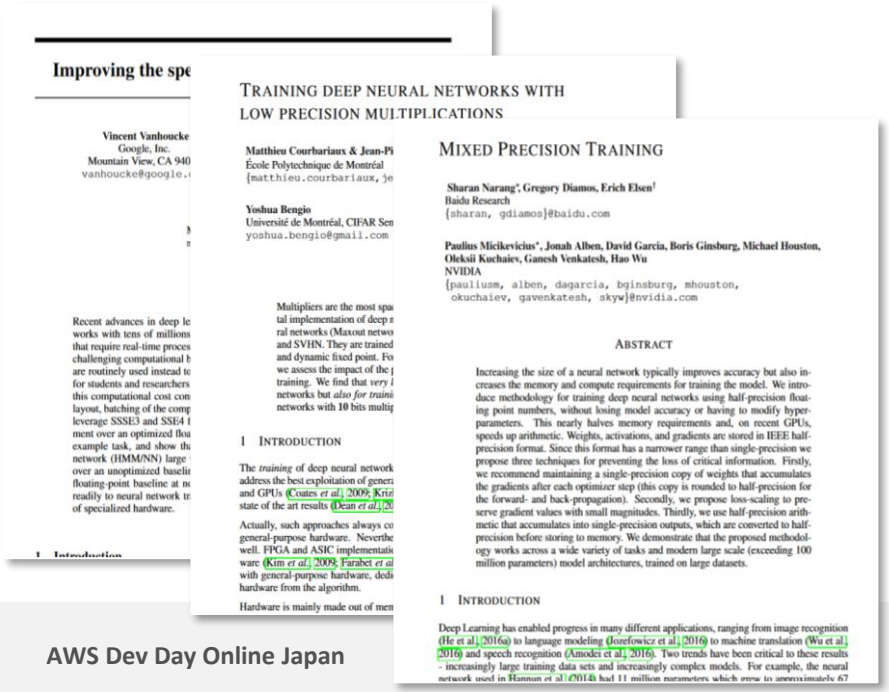
インテルAI
概要

モデルの量子化とは

DLブースト/デモ

ディープラーニングにおける量子化の歴史

- さまざまな研究者がさまざまなモデルで正確度の損失を最小限またはゼロに抑えながら、低い数値精度でのディープラーニングを実証
  - 16ビット乗算器を使用した学習
  - 8ビット以下の乗算器を使用した推論



2011 年以降の論文…

Vanhoucke、その他著 (2011 年) CPU における量子化されたアクティベーションと 8 ビットへの重み (https://storage.googleapis.com/pub-tools-public-publication-data/pdf/37631.pdf (英語))

Courbariaux、その他著 (2015 年) 低数値精度乗算器と高精度累算器 (https://arxiv.org/pdf/1412.7024.pdf (英語))

Micikevicius、その他著 (2017 年) 16 ビット浮動小数点 ( ) 乗算器と全精度累算器を使用して学習処理を行い、全精度重みを更新 (https://arxiv.org/abs/1710.03740 (英語))

モデルの量子化とは？ (INT8 編)

- 学習済みモデル内部で数値として表現されている「重み」、「アクティベーション (レイヤーへの入力値)」をデフォルトの FP32 から INT8 へ変換すること

なぜモデルの量子化が必要なのか？

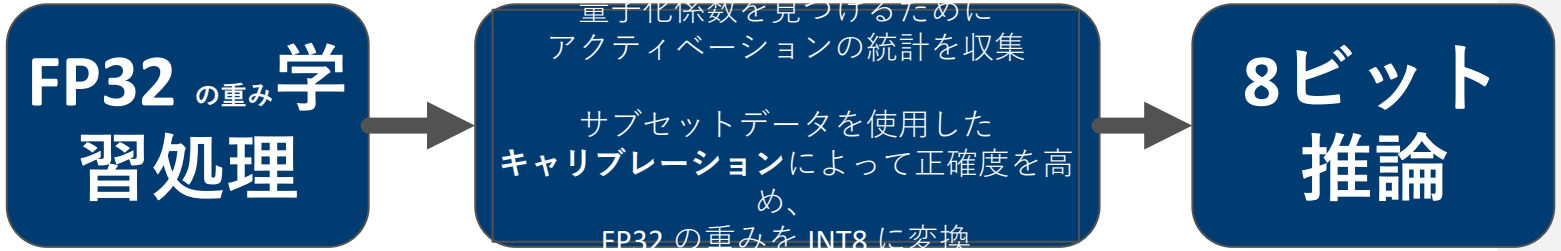
同一ハードウェア上にて
AIモデルの実行性能
（レイテンシーor スループット）
を向上させるため
↓
ROI の向上、TCO の削減

量子化：2種類の実現方法

- 「学習処理後の量子化」が主流ですが「QAT」も成熟してきています。

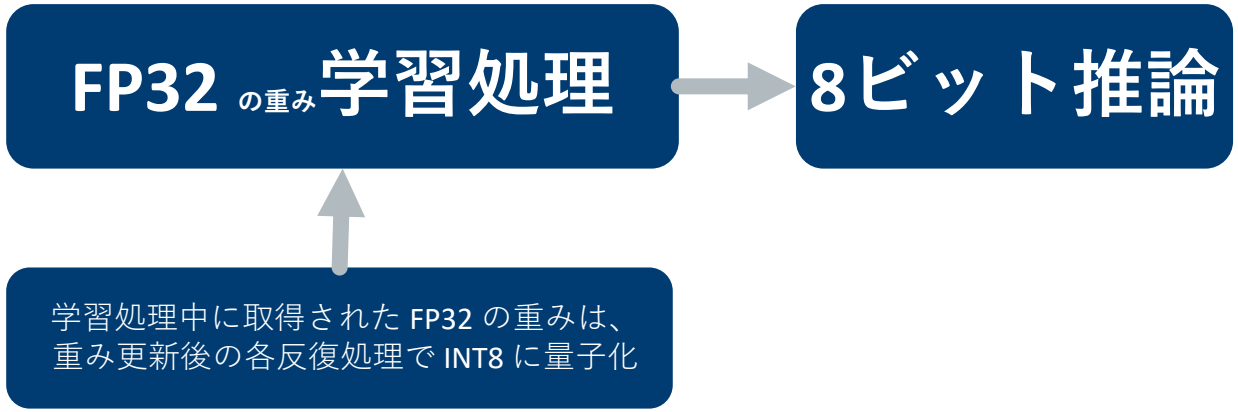
学習処理後の量子化

通常どおりの学習処理を行い、FP32 の重みを取得。推論の実行前に低い数値精度に変換し、キャリブレーションによって正確度を向上



QAT
(Quantized Awareness Training)

「偽」の量子化を使用して、フォワードパスとバックワード・パスで量子化の効果をシミュレート



ディープラーニング・フレームワークごとの量子化対応状況

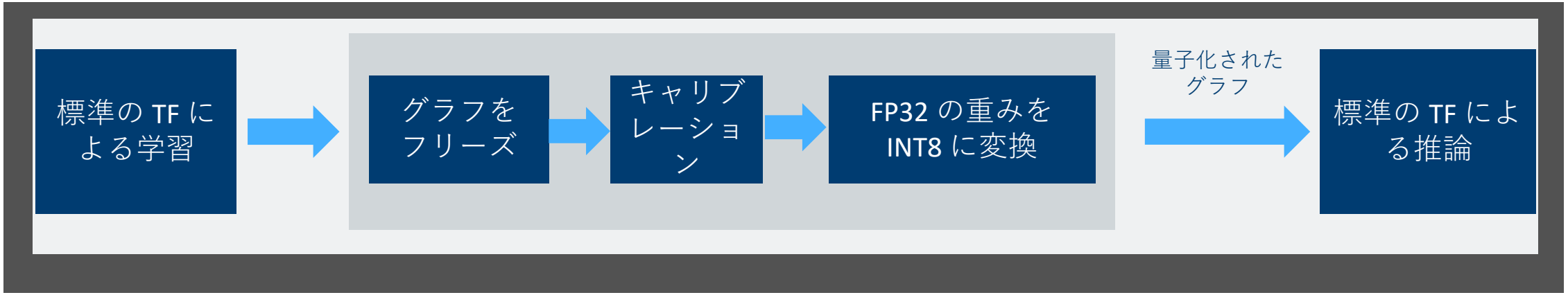
	学習後の量子化	QAT
TensorFlow	○	○
PyTorch	○	○
OpenVINO™	○	○*

＊NNCF というPreview機能で対応。

ディープラーニング・フレームワークごとの量子化ツール

- TensorFlow*
  - 学習後の量子化：https://github.com/IntelAI/tools　(インテルの量子化ツール)
  - QAT：https://www.tensorflow.org/model_optimization/guide/quantization/training
- PyTorch*
  - 学習後の量子化 / QAT ともに下記 URL を参照 (v1.3 から導入)
  - https://pytorch.org/docs/stable/quantization.html
  - https://pytorch.org/blog/introduction-to-quantization-on-pytorch/
- OpenVINO™ ツールキット
  - 学習後の量子化： https://docs.openvinotoolkit.org/latest/_README.html

TensorFlow* の学習処理後の量子化ステップ



- グラフのフリーズと最適化
  - protobuf 形式での推論グラフの保存と、冗長ノードの除去 (ID、CheckNumerics など)、定数の畳み込み、バッチ正規化の畳み込みのためのグラフ変換の適用が含まれる。
- キャリブレーション
  - このステップでは、学習データ (またはキャリブレーション・データ) の小さなサブセットでグラフを実行し、アクティベーションの範囲をフリーズ。生成されるグラフは、Requanitze ops を融合することによってさらに最適化される。
- FP32 グラフを量子化されたグラフに変換
  - このステップでは、fp32-ops を、融合および量子化可能な ops に置き換え、アクティベーションのために必要な変換 ops (QuantizeV2、Requanitze など) を追加。このステップで重みの量子化も行われる。

Intel Low Precision Optimization Tool (iLiT) for INT8 acceleration

NEW

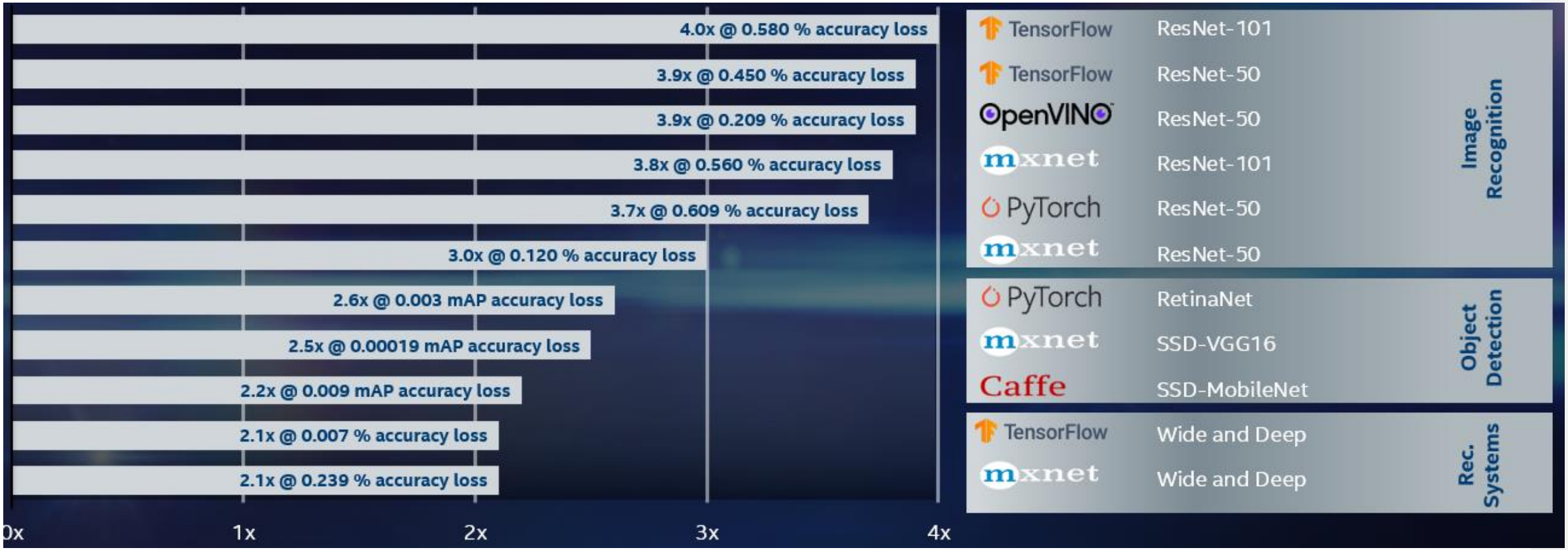
- TensorFlow、PyTorch、MXNet用の低精度最適化ツールを提供
- 自動精度駆動およびメモリフットプリントチューニングをサポート
- 新しいバックエンド（PDPD、ONNX RTなど）と新しいチューニング戦略/メトリクス（UCBのHAWQなど）のための拡張機能を提供

https://github.com/intel/lp-opt-tool

量子化における留意事項: Performance + Accuracy

性能向上が期待できる反面、精度（Accuracy）の低下（最大1%）も起こりえる

第 2 世代インテル® Xeon® プロセッサー上の DL ブースト (VNNI) と第 1 世代インテル® Xeon® プロセッサー上の FP32 の比較



ベースライン:
インテル® Xeon® Platinum 8180 プロセッサーの FP32 性能

推論の相対スループット性能 [高いほど高性能]

インテルAI
概要

モデルの
量子化とは

DLブース
ト/デモ

インテル® ディープラーニング・ブースト(VNNI)

インテル® Xeon® スケーラブル・プロセッサー (Skylake)

FP32

Sign	Exponent								Mantissa																			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00

Typical AVX-512 instruction to perform convolutions: **vfmadd231ps**

1st gen Intel® Xeon® Scalable processor with Intel® DL Boost

×最大4倍

INT8

Sign	Mantissa							
07	06	05	04	03	02	01	00

Typical AVX-512 instructions to perform INT8 convolutions: **vpmaddubsw, vpmaddwd, vpaddd**

1st gen Intel® Xeon® Scalable processor with Intel® DL Boost

×最大3倍

第 2 世代インテル® Xeon® スケーラブル・プロセッサー (Cascade Lake)

INT8

Sign	Mantissa							
07	06	05	04	03	02	01	00

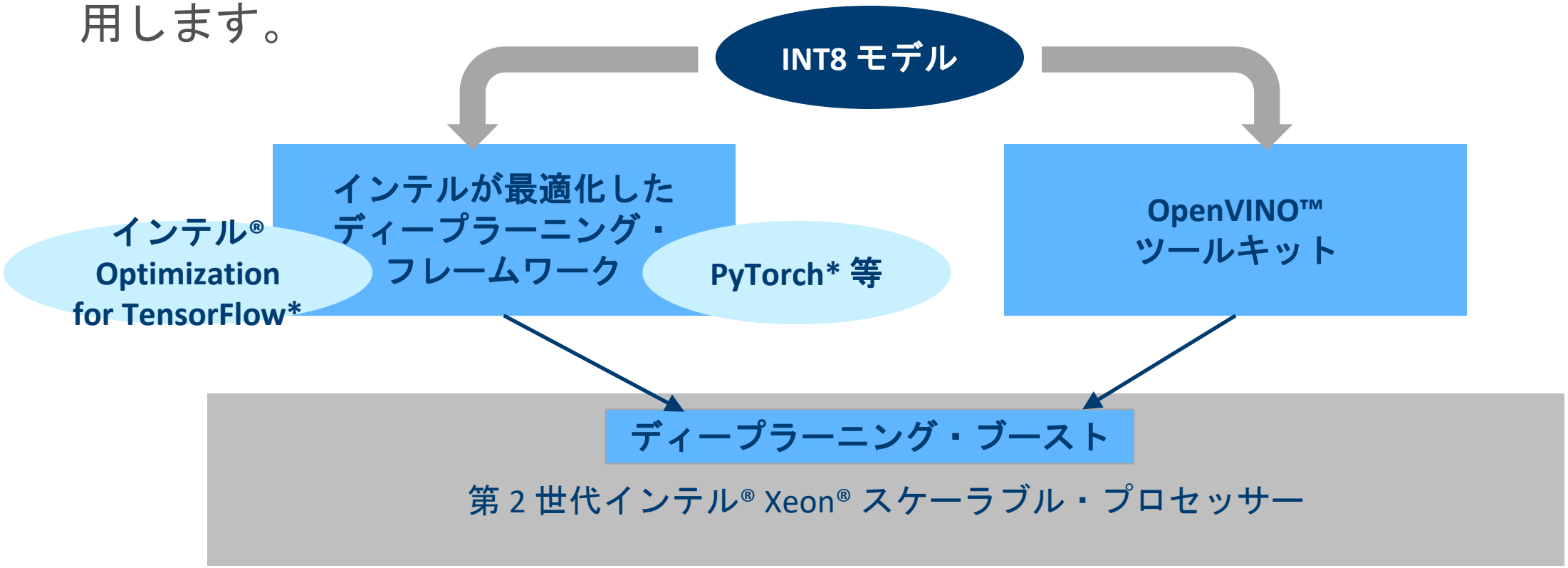
Typical AVX-512 instructions to perform INT8 convolutions: **vpdpbusd**

2nd gen Intel® Xeon® Scalable processor with Intel® DL Boost

Vector Neural Network Instructions (VNNI) 対応

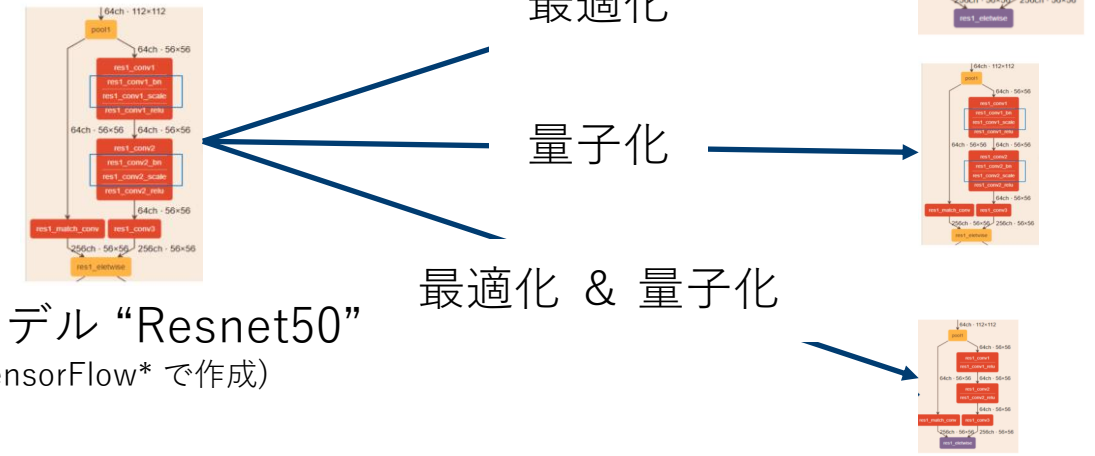
INT8 モデルをディープラーニング・ブースト上で動かすには…

- インテルが最適化したディープラーニング・フレームワークか OpenVINO™ ツールキットをご利用いただくと自動的にディープラーニング・ブーストを使用します。



デモ：TensorFlow* で作成した Resnet50 を量子化

- TensorFlow* および OpenVINO™ ツールキットの量子化ツールを使用して、“学習後の量子化” を適用し、事前 / 事後の性能を計測します。



元のモデル “Resnet50”
(TensorFlow* で作成)

最適化

量子化

最適化 & 量子化

by OpenVINO™

by TensorFlow

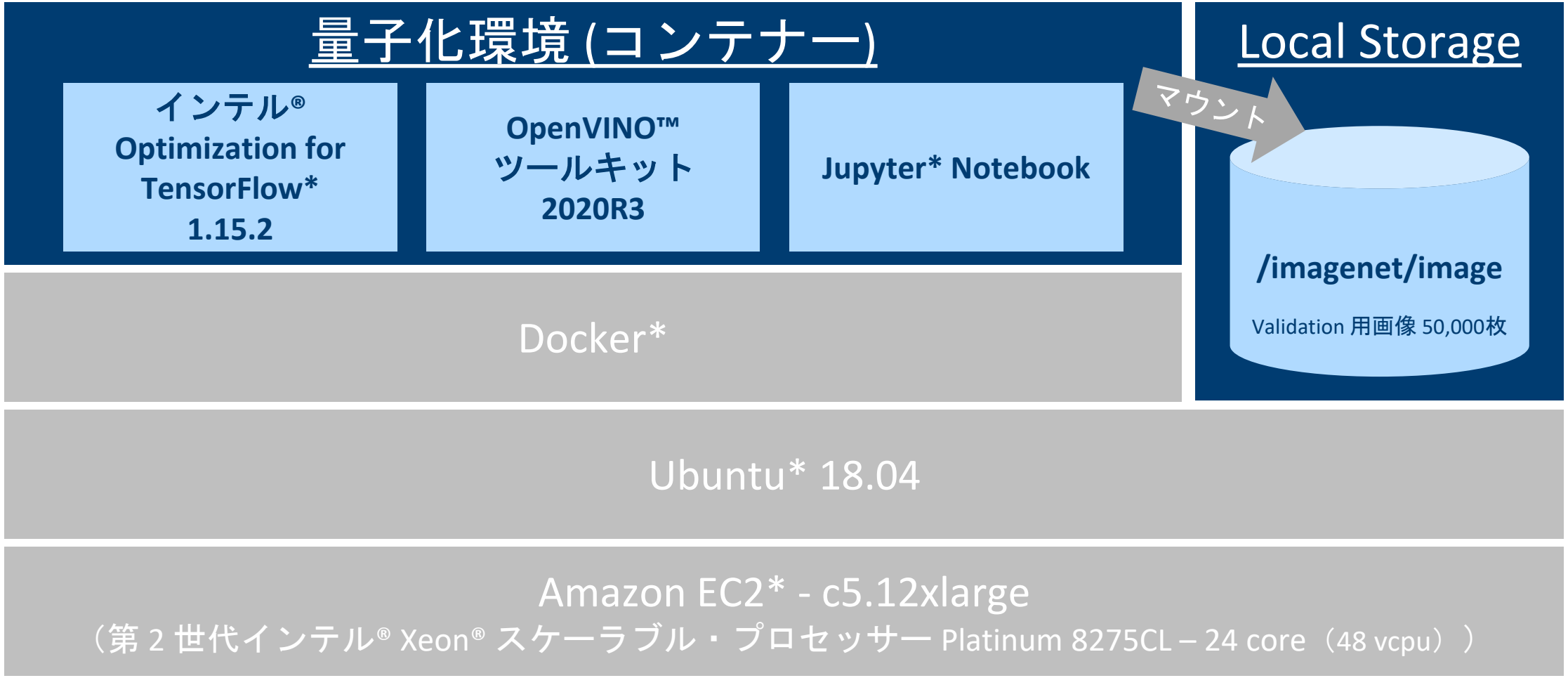
※量子化ツールとして「iLiT」を利用

by OpenVINO™

* 2020年5月現在、インテル® ディープラーニング・ブースト (VNNI) が搭載された 第 2 世代インテル® Xeon® スケーラブル・プロセッサー、第 10 世代 インテル® Core™ プロセッサー・ファミリー (Ice Lake† のみ) にてより効力を発揮する

† 開発コード名

デモ環境の説明


量子化環境 (コンテナー)
インテル®
Optimization for
TensorFlow*
1.15.2
OpenVINO™
ツールキット
2020R3
Jupyter* Notebook
Local Storage
マウント
/imagenet/image
Validation 用画像 50,000枚
Docker*
Ubuntu* 18.04
Amazon EC2* - c5.12xlarge
（第 2 世代インテル® Xeon® スケーラブル・プロセッサー Platinum 8275CL – 24 core（48 vcpu））

今回のデモ環境構築および手順はこちらで確認できます

https://github.com/hiouchiy/IntelAI/blob/master/tensorflow_quantization/demo_ilit.md

まとめ

- モデルの量子化は、モデルの重みやアクティベーションの数値表現形式を FP32 から INT8 など低精度な形式に変換することでモデルの実行性能を向上させる技術
- 新量子化ツール「iLit」およびインテル® OpenVINO™ ツールキットを使用して容易にモデルの量子化が可能
- インテル® ディープラーニング・ブースト(VNNI)は INT8 のモデルをより少ないクロック数で処理するためにCascade Lakeから搭載された拡張命令
- インテルが提供するディープラーニングフレームワークを使用して量子化されたモデルを実行すると、自動的にインテル® ディープラーニング・ブーストが使用される

免責条項

インテル® テクノロジーの機能と利点はシステム構成によって異なり、対応するハードウェアやソフトウェア、またはサービスの有効化が必要となる場合があります。実際の性能はシステム構成によって異なります。

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。

テストでは、特定のシステムでの個々のテストにおけるコンポーネントの性能を文書化しています。ハードウェア、ソフトウェア、システム構成などの違いにより、実際の性能は掲載された性能テストや評価とは異なる場合があります。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。詳細については、http://www.intel.com/benchmarks/ (英語) を参照してください。

インテル® アドバンスト・ベクトル・エクステンション (インテル® AVX) は、特定のプロセッサー演算で高いスループットを示します。プロセッサーの電力特性の変動により、AVX 命令を利用すると、a) 一部の部品が定格周波数未満で動作する、b) インテル® ターボ・ブースト・テクノロジー 2.0 を使用する一部の部品が任意または最大のターボ周波数に達しない可能性があります。実際の性能はハードウェア、ソフトウェア、システム構成によって異なります。詳細については、http://www.intel.co.jp/jp/technology/turboboost/ を参照してください。

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。

記載されているコスト削減シナリオは、指定の状況と構成で、特定のインテル® プロセッサー搭載製品が今後のコストに及ぼす影響と、その製品によって実現される可能性のあるコスト削減の例を示すことを目的としています。状況はさまざまであると考えられます。インテルは、いかなるコストもコスト削減も保証いたしません。

インテルは、本資料で参照しているサードパーティーのベンチマーク・データまたはウェブサイトについて管理や監査を行っていません。本資料で参照しているウェブサイトにアクセスし、本資料で参照しているデータが正確かどうかを確認してください。

aws DEV DAY
ONLINE JAPAN

Thank you!

インテル株式会社
APJデータセンター・グループ・セールス
AIテクニカル・ソリューション・スペシャリスト
大内山　浩

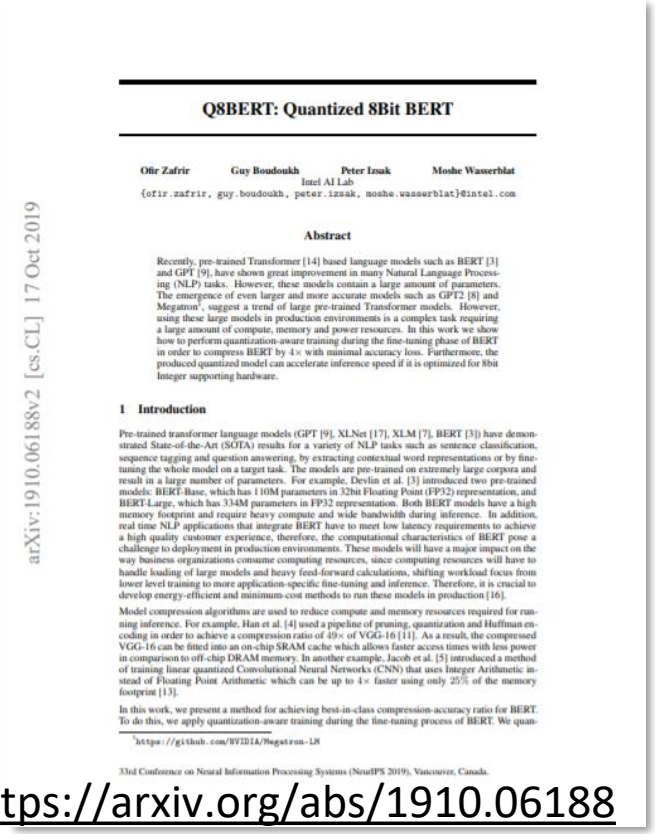
intel

In Partnership with intel

BACKUP

量子化に関する今後の展開
～多様性の拡大 / 生産性の向上～

Q8BERT (Quantized 8-bit BERT)
～より多くのモデルへ対応～



arXiv:1910.06188v2 [cs.CL] 17 Oct 2019

Q8BERT: Quantized 8Bit BERT

Ofir Zafrir Guy Boudoukh Peter Izsak Moshe Wasserblat
Intel AI Lab
{ofir.zafrir, guy.boudoukh, peter.izsak, moshe.wasserblat}@intel.com

Abstract

Recently, pre-trained Transformer [14] based language models such as BERT [3] and GPT [9], have shown great improvement in many Natural Language Processing (NLP) tasks. However, these models contain a large amount of parameters. The emergence of even larger and more accurate models such as GPT2 [8] and Megatron[1], suggest a trend of large pre-trained Transformer models. However, using these large models in production environments is a complex task requiring a large amount of compute, memory and power resources. In this work we show how to perform quantization-aware training during the fine-tuning phase of BERT in order to compress BERT by 4× with minimal accuracy loss. Furthermore, the produced quantized model can accelerate inference speed if it is optimized for 8bit Integer supporting hardware.

1 Introduction

Pre-trained transformer language models (GPT [9], XLNet [17], XLM [7], BERT [3]) have demonstrated State-of-the-Art (SOTA) results for a variety of NLP tasks such as sentence classification, sequence tagging and question answering, by extracting contextual word representations or by fine-tuning the whole model on a target task. The models are pre-trained on extremely large corpora and result in a large number of parameters. For example, Devlin et al. [3] introduced two pre-trained models: BERT-Base, which has 110M parameters in 32bit Floating Point (FP32) representation, and BERT-Large, which has 334M parameters in FP32 representation. Both BERT models have a high memory footprint and require heavy compute and wide bandwidth during inference. In addition, real time NLP applications that integrate BERT have to meet low latency requirements to achieve a high quality customer experience, therefore, the computational characteristics of BERT pose a challenge to deployment in production environments. These models will have a major impact on the way business organizations consume computing resources, since computing resources will have to handle loading of large models and heavy feed-forward calculations, shifting workload focus from lower level training to more application-specific fine-tuning and inference. Therefore, it is crucial to develop energy-efficient and minimum-cost methods to run these models in production [16].

Model compression algorithms are used to reduce compute and memory resources required for running inference. For example, Han et al. [4] used a pipeline of pruning, quantization and Huffman encoding in order to achieve a compression ratio of 49× of VGG-16 [11]. As a result, the compressed VGG-16 can be fitted into an on-chip SRAM cache which allows faster access times with less power in comparison to off-chip DRAM memory. In another example, Jacob et al. [5] introduced a method of training linear quantized Convolutional Neural Networks (CNN) that uses Integer Arithmetic instead of Floating Point Arithmetic which can be up to 4× faster using only 25% of the memory footprint [13].

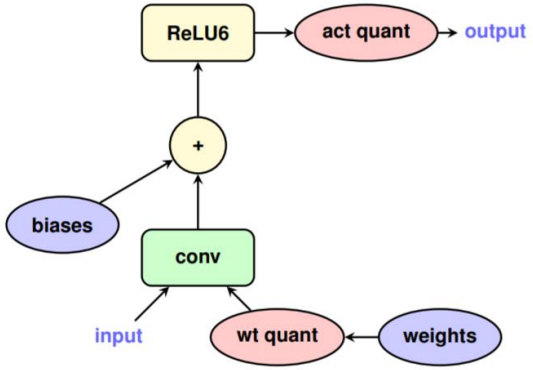
In this work, we present a method for achieving best-in-class compression-accuracy ratio for BERT. To do this, we apply quantization-aware training during the fine-tuning process of BERT. We quan-

[1] https://github.com/NVIDIA/Megatron-LM

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

https://arxiv.org/abs/1910.06188

QAT (Quantization Aware Training)
～より簡単に量子化を実現～



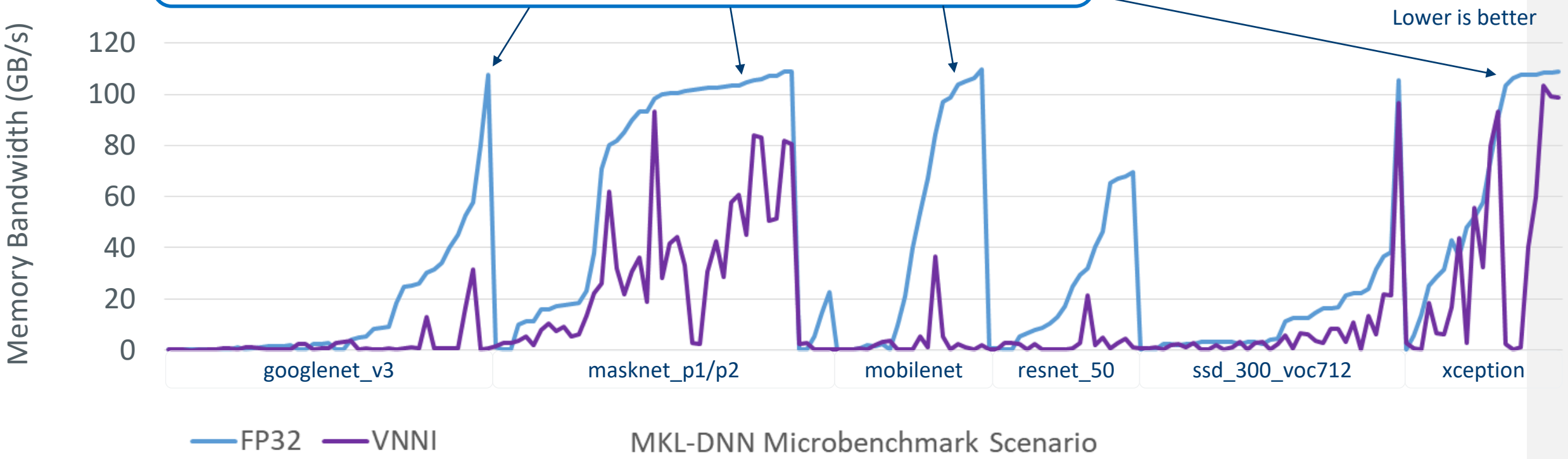
Model	Floating-point baseline model	QAT model	Delta	Post-training full integer quantized model
MobileNet v1 1.0 224	71.03%	71.06%	0.04%	69.57%
MobileNet v2 1.0 224	70.77%	70.01%	-1.07%	70.2%
ResNet v1 50	76.3%	76.1%	-0.26%	75.95%

https://blog.tensorflow.org/2020/04/quantization-aware-training-with-tensorflow-model-optimization-toolkit.html

Deconstructing Intel® DL Boost with VNNI: Memory Bandwidth

Intel® MKL-DNN benchdnn micros run on
1 socket, 28 cores at fixed frequency

High Bandwidth scenarios w/ FP32 transitioned into Low Bandwidth w/ Intel® DL Boost



Significant Reduction in Memory Bandwidth Constrained Scenarios