エッジ AI アルゴリズムの研究および事業応用動向

溝口 隼人
Edge AI Research Center 研究員
h.mizoguchi@edge-arc.jp

工藤 俊太郎
Edge AI Research Center 研究所長
kudo@edge-arc.jp

1 はじめに

近年、深層学習（ディープラーニング）を発端とする AI ブームにより、AI、とりわけ機械学習の実社会への適用が急速に進んでいる。これまで機械学習による学習/推論は、潤沢なコンピュータリソースが用意可能なクラウド環境などでの実行が主流であったが、最近では産業用ロボットや自動運転車、組み込み IoT 機器等のエッジデバイス上に機械学習モデルを組み込み、そのデバイス上で機械学習推論を行う “エッジ AI” にも注目が集まっている。デバイスで収集したデータをネットワーク経由でクラウドサーバにアップロードして推論を行い、その計算結果を再度受信してデバイス上で活用する “クラウド AI” と比較すると、エッジ AI には以下のような利点がある。

- ネットワークを経由することによるレイテンシが発生しないため、自動運転や産業用機械制御等の、超低レイテンシが求められる用途に対応可能
- エッジ側でデータ処理が行われる分、クラウド側に送信する必要があるデータ量が減り、運用時の通信量節約が可能
- ネットワークを経由させたくない機密情報を用いる場合でも機械学習推論が可能

一方、一般にエッジデバイス上で機械学習推論のために使用可能なリソースは非常に限られ、それがエッジ AI 普及の障害となっている。その対策の一つとして、Google の Edge TPU (Tensor Processing Unit) [1]、NVIDIA の Jetson [2] など、各社がエッジ側での深層学習処理を高速化するためのハードウェアを開発している。しかし、そのようなハードウェアを利用したとしても、クラウド AI で運用しているような機械学習モデルをエッジ側で運用でき、かつ十分な性能が出る例は、現状では多くないと推測される。そのため、ハード面のみならず機械学習アルゴリズムをエッジ側処理に最適化することもエッジ AI の普及には必要であり、そのための研究も数多く行われている。

本稿はエッジ AI に適したアルゴリズムに焦点を当て、1) 深層学習の軽量化、2) 深層学習以外の機械学習アルゴリズムの軽量化 という二つの大きな観点に基づいてアルゴリズムを調査し、それらの手法を分類・整理することで、エッジ AI 導入検討やアルゴリズム選定等の意思決定に寄与することを目的とする。

2 深層学習の軽量化によるエッジ AI アルゴリズム

深層学習モデルは人工ニューラルネットワークを多層化したものであり、狭義には 4 層以上の深層ニューラルネットワーク (DNN, Deep Neural Network) を指す。2012 年の画像認識コンペ ImageNet Large Scale Visual Recognition Competition (ILSVRC) において、畳み込みニューラルネットワーク (CNN, Convolutional Neural Networks) を用いた AlexNet [3] が従来手法に大きな差をつけて優勝したのをきっかけに、より大きな注目を集めることとなった。

画像認識、自然言語処理等のタスクにおいて優れた性能を発揮するが、その代償としてモデルのメモリ使用量や計算負荷が大きく、その特性がエッジデバイス上への展開を困難にしている。使用可能なリソースが限られたエッジデバイス上で深層学習を動作させる試みとして、

- 高速なモデルアーキテクチャの設計
- モデル圧縮
- 深層学習コンパイラによる最適化

等が盛んに研究されている。以下では上記 3 つについての研究や事業応用について紹介する。

2.1 高速なモデルアーキテクチャ

リソース制限のあるデバイス上で計算負荷やメモリ使用量を抑えた推論を行うことを目的とした、パラメータ数を抑えた深層学習モデルを設計する研究は数多く存在する。

文献 [4] では、CNN における高負荷な畳み込み演算を、2 ステップに分割することで計算量を削減するモデルである MobileNet を提案している。ImageNet の分類において、2014 年の ILSVRC で 2 位となったモデル VGG16 [5] と比較してサイズを 1/32、計算量 1/27 に削減しつつ、ほぼ同等の精度を達成している。また、同じく 2014 年の ILSVRC で優勝したモデルである GoogleNet [6] よりも良い精度を出しつつ、サイズを 2/3, 計算量を 2/5 以下に削減している。

MobileNet の発表以降も、改良版の MobileNetV2 [7], MobileNetV3 [8] や、SqueezeNet [9], ShuffleNet [10] など様々

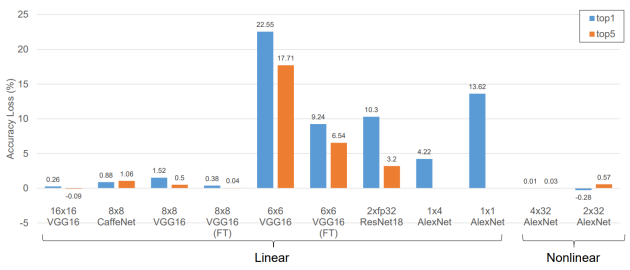

図 1　量子化ビット数と精度低下の関係 [12]

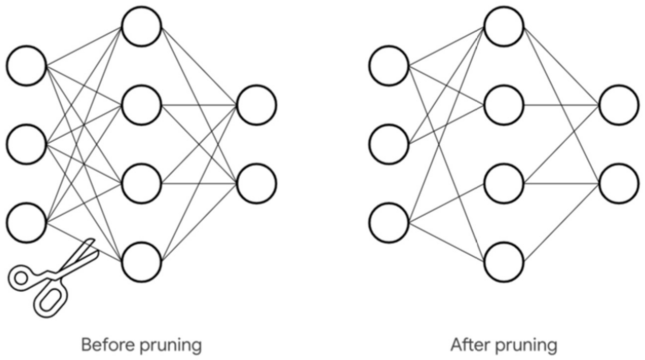

図 2　枝刈りの概念図 [16]

な手法が提案されており、高速・軽量なモデルの開発は進んでいる。

2.2　モデル圧縮

モデル圧縮は、既存の深層学習モデルを、なるべく精度を維持したまま軽量化する手法を指す。モデルのパラメータ数を減らすことによりメモリ使用量や計算負荷を削減出来るので、エッジデバイス上へのモデル組み込みに有効な手法である。モデル圧縮には様々なアプローチが存在するが、以下では主な手法として量子化、枝刈り、蒸留について述べる。

2.2.1　量子化 (quantization)

量子化は、ニューラルネットワークのパラメータなどを通常よりも小さいビット数で表現することによりモデルを軽量化する手法である。通常、ニューラルネットワークの演算は 32 ビットや 16 ビットの浮動小数点数により行われるが、これらをより小さいビットでの表現に置き換えることで、メモリ使用量を小さくすることが出来る。また、浮動小数点による演算を整数演算にすることが出来れば、処理に必要な回路を小さくすることも可能となる。

深層学習の算術演算においては、8 ビット固定小数点までは性能を大幅に低下させることなく符号化できることが知られている [11]。例えば 文献 [12] では、図 1 に示すように、量子化ビット数とアクティベーションのビット数による精度低下の関係がまとめられている。各モデルに作用させた量子化ビット数が、(量子化ビット数) × (アクティベーションのビット数) の形で表現されており、その時の精度ロスが縦軸である。図の結果から、8 ビットの量子化では精度が大きく低下していないことが読み取れる。

また、極端な例としては、量子化を 1 ビット（二値化）で行う研究も存在する。文献 [13] では、順伝搬/逆伝搬の重みを二値化する BinaryConnect を提案しており、特定のデータセットに対しては最先端の結果に近いエラー率が実現できたとしている。重みに加えてアクティベーションも二値化しているものとして文献 [14] の BinaryNet があり、メモリ使用量をより減らすとともに、数値演算をビット演算に置き換えられることによる高速化が図られている。またその後、文献 [15] において、2 値演算による畳み込み近似手法である XNOR-Net が提案されており、ImageNet 分類タスクにおいて BinaryConnect や BinaryNet と比較して 16% 以上の精度改善が達成できたとしている。

2.2.2　枝刈り (pruning)

ニューラルネットにおいては各ノードが結合しているが、ノード間の重みが小さく重要でないと思われるノードを削除することでパラメータ数を削減する手法を枝刈りという。図 2 に枝刈りの概念図を示す。

冗長な接続を削除する基準については、様々な方法が提案されている。例えば文献 [17] では、重要なノードを判断するためネットワークを学習させ、重要でない接続を刈りこみ、最後にネットワークを再調整するという手法が提案されており、ImageNet データセットにおいて AlexNext のパラメータ数を、精度を落とすことなく 1/9 にすることに成功している。

2.2.3　蒸留 (distillation)

蒸留は、事前に学習したサイズの大きいモデルを、より小さいモデルで模擬するように学習させる手法である。蒸留により、単純に小さいモデルを直接学習させるよりも良い精度のモデルを得ることが出来るとされる。図 3 に蒸留の概念図を示す。

文献 [18] においては、大きい深層ニューラルネットのモデルを小さいもので模擬し、その有効性を示している。また、文献 [19] においては、教師となる大きいモデルの出力だけでなく中間表現をヒントにして蒸留を行うことで、元の大きいモデルより精度を向上させることに成功している。

また、蒸留により深層ニューラルネットのモデルを、深層ニューラルネット以外のモデルで模擬することを試みている研究も存在する。例えば文献 [37] では、深層ニューラルネットの説明可能性が低いという欠点に対応するために、学習したニューラルネットを蒸留を用いて決定木で表現することを試みている。MNIST の分類において、深層ニューラルネットよりは若干悪化するが直接決定木を構築する場合よりは良い、という中間の精度を達成しつつ、決定木の特徴である説明可能性を一定程度得ることに成功している。

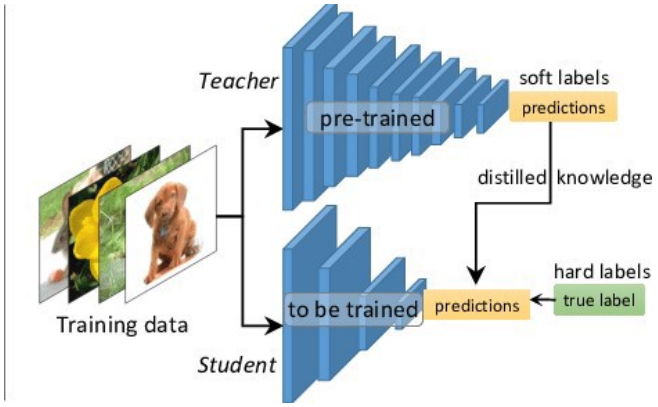

図 3　蒸留の概念図 [20]

2.2.4　モデル圧縮の事業への応用

上記のモデル圧縮技術は、国内外の企業により事業展開されている。LeapMind 株式会社は、「極小量子化技術」により、1 ビットの Weight (重み係数)、2 ビットの Activation (入力) という組合せでモデルを圧縮しても、性能をほとんど劣化させないことに成功していると表明している [21]. また、株式会社アラヤは、提供するエッジ AI ソフトウェア Pressai (プレッサイ) において、量子化、蒸留、枝刈りや独自の圧縮方式が利用可能としている [22]。

海外では、圧縮技術を活用したエッジ AI に強みを持っていた米シアトルのスタートアップ Xnor.ai が、2020 年に Apple により 2 億ドル台で買収されたと報道されている [23]。

3　深層学習コンパイラ

近年は TensorFlow [24], PyTorch [25], Caffe/Caffe2 [26] といった深層学習のフレームワークが数多く登場し、モデルの学習/推論を、抽象化されたインターフェースで実施出来るようになっている。しかしながら、様々な深層学習用ハードウェアが登場し多様化する中、各フレームワークにより作成されたモデルが、専用ハードの性能を効率的に利用できないことが懸念されている。その対策として、フレームワークを用いて作成された深層学習モデルを入力に取り、各ハードウェア向けに最適化されたコードを生成する深層学習コンパイラが開発されている。

一般的な深層学習コンパイラの処理の概念図を図 4 に示す。まず、深層学習モデルを中間表現 (IR, Intermediate Representation) に変換し、それに基づいて高レベル/低レベルでの最適化を施した上で、マシン依存の実行可能命令を出力するというのが基本的な概念となる。

XLA [28], TVM [29], Tensor Comprehension [30] など、産学から様々な深層学習コンパイラが提案されている。事業への応用という観点では、Idein 株式会社が、提供する IoT プラットフォーム Actcast において、安価なデバイスで Deep Learning 推論を高速化する最適化コンパイラ技術を活用している [31]。

4　深層学習以外の機械学習アルゴリズムのエッジ AI への活用

近年の深層学習のブームもあり、エッジ AI アルゴリズムについての研究は、深層学習を軽量化しエッジデバイス上で動かすことを目的とした研究が盛んである。

しかし、サポートベクターマシン (SVM), 木構造ベースアルゴリズムや k-近傍法といった従来からある機械学習アルゴリズムについても、マイコン等の非常にリソースが限られるエッジデバイスに組み込む場合には、同様に要求リソースや計算量が問題になり得るため、数多くの研究もされている。以下では、機械学習アルゴリズムとして木構造ベースのアルゴリズム、SVM, k-近傍法を取りあげ、それらをエッジデバイスに導入するための試みを概観する。

4.1　木構造ベースアルゴリズム

木構造ベースの機械学習アルゴリズムは、一般に学習データ数に対し対数時間オーダーでの推論が可能であり、速度の面ではエッジ側での処理に適しているといえる。しかしながら、モデルサイズは学習データに対し線形であり、またランダムフォレストのようなアンサンブル学習の手法を取る場合には、木が増える分モデルサイズ・計算負荷ともに増加する。容量節約のために単純に木の深さを浅くすると、精度が悪くなる可能性が高いという問題がある。

対策の 1 つとして、木のノードを剪定してサイズを小さくすることが考えられる。文献 [32] では、ランダムフォレストにおいて重みが小さいノードを剪定し、対象となる推論タスクによってはモデルサイズを縮小しつつ、精度の上昇も可能であることを示唆している。

また、別のアプローチとして文献 [33] では、モデルサイズの問題を改善した新しい木構造ベースアルゴリズムの Bonsai を発表している。Bonsai は Arduino UNO (8bit-16MHz ATmega328P マイクロコントローラ, 2KB RAM, 32KB フラッシュメモリ) 上に展開可能で、その上でミリ秒単位での予測が可能であり、他の資源効率の良い最新手法と比較して 30% も高い予測精度を達成したとしている。

木構造ベースのアルゴリズムへの事業応用として、株式会社エイシングでは、軽量/逐次学習可能/説明可能という性質を備えたエッジ AI アルゴリズムである AiiR シリーズを発表している [34]。

4.2　サポートベクターマシン

サポートベクターマシンはカーネル法と組み合わせて利用され、高精度のモデルを構築可能な手法だが、計算コストの問題から実用例は限定的となっている。そのため、計算量やメモリ使用量を減らす手法は数多く研究されており、例えば文献

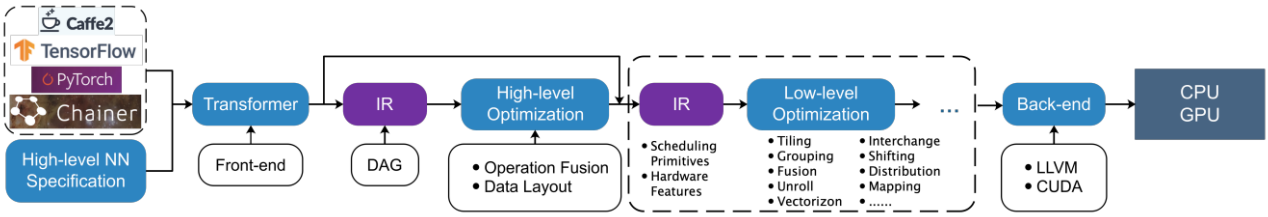

図 4　深層学習コンパイラの概念図 [27]

[35] では、近似法である Fastfood を提案している。これにより、展開次元 n, 入力次元 d としたとき通常はともに $O(nd)$ である計算量とメモリ使用量のオーダーを、$O(n \log d)$ の計算量と $O(n)$ のメモリ使用量に削減した上で、完全カーネル展開や Random kitchen sink といった他手法と同等の精度を達成している。

4.3　k-近傍法

k-近傍法は、対象となる点と特徴空間における距離が近い点群を利用して対象を分類する手法である。パターン認識等で広く用いられるが、ハードウェア制約の厳しいエッジ上に実装する際には、全学習データが推論時に必要であることに起因するモデルサイズの問題や、逐一学習データとの距離を計算する必要があることに起因する予測時間の大きさの問題が存在する。

それらの問題を解決するために k-近傍法を改良したアルゴリズムとして、文献 [36] では ProtoNN が提案されている。様々なデータセットにおいて、通常の k-近傍法, RBF-SVM や中間層が 1 層のニューラルネットのような一般的な手法と比較し、数桁低いモデルサイズを達成しつつ、ほぼ他の手法に劣らない精度を達成できるとしている。また、Arduino UNO (2KB RAM) 上に実装し、単純な線形モデルに近い推論速度、消費電力性能を達成できることを示している。

5　まとめ

リソースが限られたエッジデバイス上で機械学習推論を動かすためのエッジ AI 技術に関連するアルゴリズムを、深層学習とそれ以外の機械学習アルゴリズムに分けて概観した。本レポートが、エッジ AI の導入検討やエッジデバイス上で動くアルゴリズムの選定等の参考になれば幸いである。

参考文献

[1] Google Cloud, Edge TPU, https://cloud.google.com/edge-tpu/

[2] NVIDIA, "Buy the Latest Jetson Products" https://developer.nvidia.com/buy-jetson

[3] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

[4] Howard, Andrew G., et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." arXiv preprint arXiv:1704.04861 (2017).

[5] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[6] Szegedy, Christian, et al. "Going deeper with convolutions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[7] Sandler, Mark, et al. "Mobilenetv2: Inverted residuals and linear bottlenecks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[8] Howard, Andrew, et al. "Searching for mobilenetv3." Proceedings of the IEEE International Conference on Computer Vision. 2019.

[9] Iandola, Forrest N., et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size." arXiv preprint arXiv:1602.07360 (2016).

[10] Zhang, Xiangyu, et al. "Shufflenet: An extremely efficient convolutional neural network for mobile devices." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[11] Vanhoucke, Vincent, Andrew Senior, and Mark Z. Mao. "Improving the speed of neural networks on CPUs." (2011).

[12] Guo, Kaiyuan, et al. "A survey of FPGA-based neural network accelerator." arXiv preprint arXiv:1712.08934 (2017).

[13] Courbariaux, Matthieu, Yoshua Bengio, and Jean-Pierre David. "Binaryconnect: Training deep neural networks with binary weights during propagations." Advances in neural information processing systems. 2015.

[14] Courbariaux, Matthieu, et al. "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1." arXiv preprint arXiv:1602.02830 (2016).

[15] Rastegari, Mohammad, et al. "Xnor-net: Imagenet classification using binary convolutional neural networks." European conference on computer vision. Springer, Cham, 2016.

[16] Merenda, Massimo, Carlo Porcaro, and Demetrio Iero. "Edge Machine Learning for AI-Enabled IoT Devices: A Review." Sensors 20.9 (2020): 2533.

[17] Han, Song, et al. "Learning both weights and connections for efficient neural network." Advances in neural information processing systems. 2015.

[18] Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean. "Distilling the knowledge in a neural network." arXiv preprint arXiv:1503.02531 (2015).

[19] Romero, Adriana, et al. "Fitnets: Hints for thin deep nets." arXiv preprint arXiv:1412.6550 (2014).

[20] twords data science, "Knowledge Distillation : Simplified", https://towardsdatascience.com/knowledge-distillation-simplified-dd4973dbc764
[21] LeapMind, "Technology", https://leapmind.io/technology/
[22] ARAYA, "Pressai", https://www.araya.org/pressai/main/
[23] GeekWire, "Exclusive: Apple acquires Xnor.ai, edge AI spin-out from Paul Allen' s AI2, for price in $200M range." https://www.geekwire.com/2020/exclusive-apple-acquires-xnor-ai-edge-ai-spin-paul-allens-ai2-price-200m-range/
[24] Abadi, Martín, et al. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." arXiv preprint arXiv:1603.04467 (2016).
[25] Paszke, Adam, et al. "Pytorch: An imperative style, high-performance deep learning library." Advances in neural information processing systems. 2019.
[26] Jia, Yangqing, et al. "Caffe: Convolutional architecture for fast feature embedding." Proceedings of the 22nd ACM international conference on Multimedia. 2014.
[27] Xing, Yu, et al. "An In-depth Comparison of Compilers for Deep Neural Networks on Hardware." 2019 IEEE International Conference on Embedded Software and Systems (ICESS). IEEE, 2019.
[28] Leary, Chris, and Todd Wang. "XLA: TensorFlow, compiled." TensorFlow Dev Summit (2017).
[29] Chen, Tianqi, et al. "TVM: An automated end-to-end optimizing compiler for deep learning." 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18). 2018.
[30] Vasilache, Nicolas, et al. "Tensor comprehensions: Framework-agnostic high-performance machine learning abstractions." arXiv preprint arXiv:1802.04730 (2018).
[31] Idein Inc. "Product" "https://idein.jp/ja/#product
[32] Ren, Shaoqing, et al. "Global refinement of random forest." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.
[33] Kumar, Ashish, Saurabh Goyal, and Manik Varma. "Resource-efficient machine learning in 2 KB RAM for the internet of things." International Conference on Machine Learning. 2017.
[34] AISing, https://aising.jp/
[35] Le, Quoc, Tamás Sarlós, and Alex Smola. "Fastfood-approximating kernel expansions in loglinear time." Proceedings of the international conference on machine learning. Vol. 85. 2013.
[36] Gupta, Chirag, et al. "Protonn: Compressed and accurate knn for resource-scarce devices." International Conference on Machine Learning. 2017.
[37] Frosst, Nicholas, and Geoffrey Hinton. "Distilling a neural network into a soft decision tree." arXiv preprint arXiv:1711.09784 (2017).