3 AI実装技術

3-1 大規模深層学習のGPUへの実装技術

Design and Implementation of Deep Learning on GPU for Large Scale Neural Networks

根岸　康　今井晴基　Tung D. LE　河内谷清久仁

Abstract

深層学習（ディープラーニング）用システムでは，学習時間の短縮のためにGPU（Graphics Processing Unit）を使用することが多い．本稿では，最初に深層学習システムに要求される演算とその実装について解説し，次にGPUへの実装にあたっての技術課題とそれを克服するための実装技術について説明する．更に筆者らが主に取り組んでいるGPUメモリより大きい深層学習モデルをGPU上で実行するための大規模深層学習技術について，その必要性，実装方針，技術課題や実装技術について説明する．最後にMRIによる高解像度医療画像の大規模深層学習技術による解析事例について紹介する．

キーワード：深層学習，GPU，大規模深層学習，医療画像処理

1. 背　　景

深層学習は画像認識，音声認識など様々な分野において既存手法を圧倒する精度を達成し，その精度は年々向上している．例えば，画像認識の競技会ILSVRC[1]の大量の画像を1,000クラスに分類するタスクにおいて，2011年に従来手法により26%であったTop5エラー率（予測したトップ5クラス中に正解が存在しない率）が，2012年には深層学習に基づく手法により16%に改善され，更に2017年には2.3%に向上した（図1）．

深層学習では精度のみならず，学習や推論の実行速度も大きく向上している．例えば前述の画像分類タスクではGPUの導入以前は数週間の学習時間を必要としたが，多数のGPUの導入とそれを使いこなす深層学習技術の進歩により，現在では約1分にまで短縮されている（図2）[2]．学習時間の短縮はより多くのモデルやパラメータの探索を可能にし，結果として精度の向上にも貢献する．

このような深層学習技術や計算機技術の進歩により，産業界でも深層学習が大きな注目を集めており，深層学習を実際の業務で活用するための研究開発が進められて

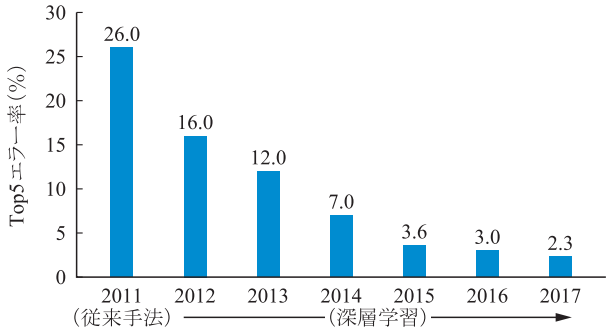


図1　ILSVRC画像分類タスクのエラー率　　従来手法で25%以上だったエラー率は深層学習により約2%まで削減された．

根岸　康　日本アイ・ビー・エム株式会社東京基礎研究所
E-mail　negishi@jp.ibm.com
今井晴基　日本アイ・ビー・エム株式会社東京基礎研究所
E-mail　imaihal@jp.ibm.com
Tung D. LE　日本アイ・ビー・エム株式会社東京基礎研究所
E-mail　tung@jp.ibm.com
河内谷清久仁　日本アイ・ビー・エム株式会社東京基礎研究所
E-mail　kawatiya@jp.ibm.com
Yasushi NEGISHI, Haruki IMAI, Tung D. LE, and Kiyokuni KAWACHIYA, Nonmembers (IBM Research-Tokyo, Tokyo, 103-8510 Japan).
電子情報通信学会誌　Vol.103 No.5 pp.495-500 2020年5月

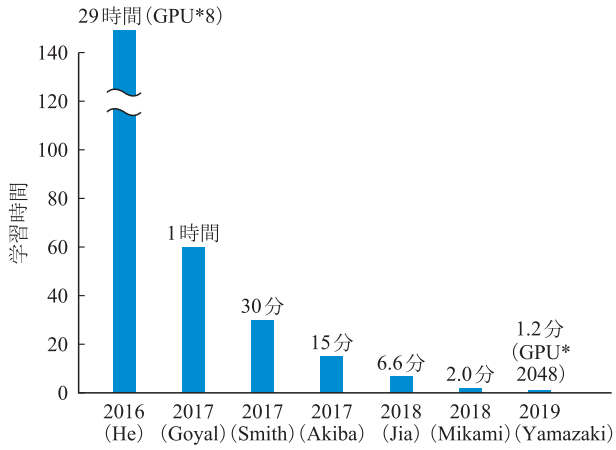


図2　ImageNet データセットの学習時間　GPU の導入以前数週間であった学習時間は多数の GPU の導入とそれを使いこなす深層学習技術の進歩により約1分まで短縮された．

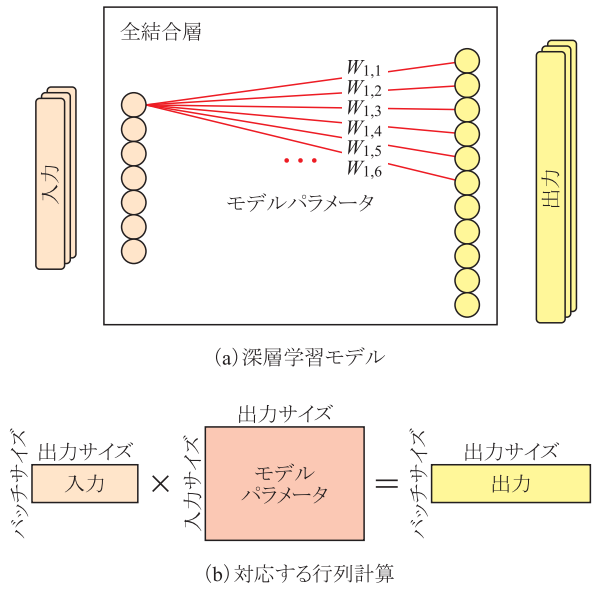


図3　深層学習モデルと演算　深層学習に必要な演算は行列積等の行列演算となる．

いる．GPU は学習フェーズの時間短縮に非常に有効であるので，研究開発に使用されるシステムの多くは GPU を使用している．

本稿では，最初に深層学習の GPU への実装技術について解説した後，その技術課題とそれを克服するための実装技術について説明する．最後に筆者らが主に取り組んでいる大規模深層学習を例にとって，その必要性，実装技術及び応用事例を紹介する．

2. GPU への深層学習実装技術

本章では深層学習に必要な演算と GPU への実装技術について述べる．

2.1　深層学習に必要な演算

深層学習処理は，学習用データを用いて深層学習モデルの最適なパラメータを学習する「学習フェーズ」と，学習したパラメータを使用して，実際のデータからその属性を推論する「推論フェーズ」の二つに分類できる．このうち特に学習フェーズの実行には膨大な計算が必要で，数週間から数か月の時間を要することもあった．深層学習には主に大きな行列に対する演算が必要となる．例えば，深層学習で使用される全結合層の順伝搬処理には入力データとモデルパラメータの行列積演算が必要となる（図3）．

深層学習の学習フェーズの処理は，①入力データからモデルの出力データ（以下中間データと呼ぶ）を生成する「順伝搬処理」，②モデルの出力データと正解を比較してモデルパラメータを更新する「逆伝搬処理」の二つで構成される．推論フェーズの計算は学習フェーズの順伝搬処理とほぼ同一である．この順伝搬処理及び推論フェーズの処理の各層の計算は，行列として表現される中間データを入力として，新たな中間データを生成する行列演算となる．生成された中間データは次の層の入力として使用され，最終的な出力を生成するまで順次各層での処理を行う．学習フェーズの逆伝搬処理では深層学習モデルの各層のモデルパラメータを層の逆順に更新する処理が行われるが，この逆伝搬処理も順伝搬処理と同じ大きさの行列に対する行列演算となる．

2.2　GPU の特徴

GPU は本来三次元のモデルからディスプレイ表示用のグラフィックス画像を生成するハードウェアであり，その主な用途は PC ゲームでの詳細なグラフィックス処理であった．このため，GPU はこのようなグラフィックス画像を生成することに特化した演算器及びメモリシステムを備えているが，近年 GPU の持つ高速な演算器とメモリをハイパフォーマンスコンピューティング等グラフィックス以外の用途に使用する General-Purpose computing on Graphics Processing Unit（GPGPU）技術が登場し，普及している．

2.3　深層学習フレームワークの GPU への実装

GPU 上で深層学習処理を行う場合，GPU 開発元が用意したランタイムライブラリを使用することが多い．NVIDIA 社製 GPU の場合，主に使用されるのは，深層学習固有処理用の cuDNN ライブラリ，より一般的な行列演算用の cuBLAS ライブラリの二つである[(3)]．深層学習フレームワークもこれらのライブラリを使用する．

深層学習の特に学習フェーズでは深層学習フレームワークを用いることが多い．主要なフレームワークに

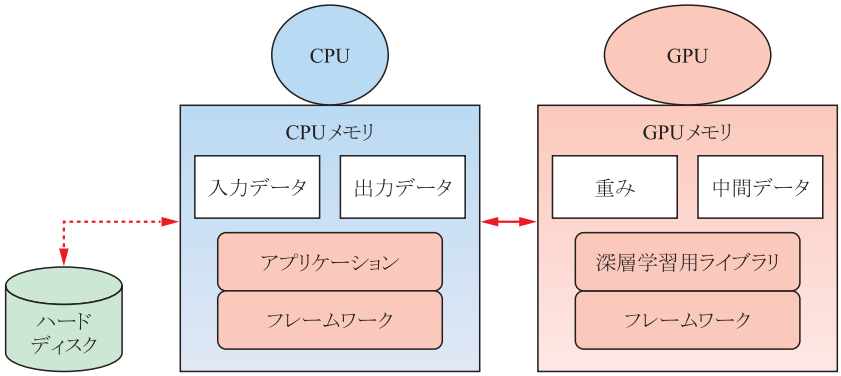


図 4　深層学習フレームワークの GPU への実装　深層学習フレームワーク及びアプリケーションの実装にあたっては CPU-GPU 間のデータ移動と深層学習ライブラリの利用方法に注意が必要となる．

は，TensorFlow，pyTorch，Keras，Chainer 等が挙げられる．主要なフレームワークはオープンソースソフトウェアとして公開されており，ダウンロードして無料で使用できる．その多くは Python 言語のライブラリとして実装され，利用者は深層学習フレームワーク用ライブラリを呼び出す Python プログラムを記述することで，深層学習の学習フェーズ処理，推論フェーズ処理を実行できる．

深層学習フレームワークは利用者が記述した Python プログラムから呼び出され，指定がなければ CPU 上で実行される．GPU を使用する場合，深層学習に必要な演算を直接行う部分が GPU 上で実行され，主プログラムである利用者プログラムは CPU 上で実行される．使用する演算器は実行時に指定する．例えば，TensorFlow では各層の処理に対応するオペレータとして CPU 用と GPU 用を用意し実行時に切り換える．Chainer では，Python 用の標準的な行列演算用モジュール numpy と同一の API を持つ GPU 用モジュール cupy を用意し，使用する演算器に応じて実行時に numpy 若しくは cupy を切り換えて使用する[4]．

2.4　GPU への実装時の課題と実装上の注意点

深層学習に必要な演算のほとんどは GPU 開発元が用意したライブラリによって実行されるため，フレームワークやアプリケーションの開発者が演算処理を直接記述する必要はない．

GPU 上での深層学習フレームワークの実装にあたっての課題として以下の三つが挙げられる．① GPU の計算能力と比較すると CPU-GPU 間のデータ転送速度が十分でないことが多い．② CPU 上で動作する Python から GPU 上で動作するモジュールの呼出しにはオーバヘッドがかかる．③深層学習ライブラリはより高速なアルゴリズムが存在しても使用 GPU メモリの少ないアルゴリズムを選択することがある．

上記三つの課題に対応する際に考慮すべき点は以下の 3 点である．①モデルパラメータと中間データは GPU 上に配置し CPU-GPU 間で不必要に移動しないこと．負荷の小さい処理であっても CPU 側で実行する処理が途中に存在すると中間データを CPU-GPU 間でデータ移動させる必要が生じて大きなオーバヘッドとなる（図 4）．②深層学習ライブラリ使用時に融合操作（fused operation）等を使用して，CPU-GPU 間の呼出し回数を最小化すること，③使用可能 GPU メモリ量に応じて深層学習ライブラリに適切なアルゴリズムを指定することが挙げられる．

3.　大規模深層学習

本章では，GPU メモリ以上の深層学習モデルを実行する大規模深層学習技術においてその必要性や実装方針を説明する．また，MRI 等高解像度医療画像の大規模深層学習技術による解析事例を紹介する．

3.1　大規模深層学習の必要性

ILSVRC 等の画像認識コンテストではより大規模な深層学習モデルを使用することで精度が向上している．例えば自然言語処理用深層学習モデル BERT でも大規模モデルほど精度が向上することが知られている[5]．また近年応用が進んでいる医療画像や衛星画像等の高解像度データに対応するためにもより大きなモデルが必要となる．一方，深層学習で使用される GPU のメモリ容量（16〜32 GByte 程度）は CPU メモリ（256 GByte〜1 TByte 程度）と比較すると小さい．これは GPU メモリに利用されている High Bandwidth Memory（HBM）が高価なためで，メモリ容量の今後の飛躍的な増加は困難と考えられている．

このように大規模化している深層学習モデルに見合った大きさの GPU メモリを使用することは困難になりつ

つある．このため，GPU メモリ容量以上の大規模深層学習モデルの実行を効率良く行う大規模深層学習モデルサポートは今後より重要な課題となると考えられる．

3.2 大規模モデルの深層学習手法

大規模モデルの学習手法には以下の方針が考えられる．

（1） データパラレル手法

深層学習の学習・推論フェーズでは通常複数の入力データをバッチと呼ばれる単位ごとでまとめて処理する．例えば画像処理でバッチ数 128 の場合，128 枚の画像データが 1 回でまとめて処理される．1 バッチの画像データを複数 GPU で処理することでより大きい深層学習モデルを利用可能になる．その際複数マシン上の GPU を利用することも可能である．ただしバッチ数 1 の場合にはこの手法は適用できない．

（2） モデルパラレル・パイプラインパラレル手法

一つの深層学習モデルを複数の GPU 上に配置する手法がモデルパラレルである．この手法では 1 枚の画像が複数 GPU で処理される．この手法の場合，現状プログラマが手動でモデルを複数 GPU に均等に分割する必要がある．また，1 枚の画像処理に GPU 間通信が必要となり，比較的多くのオーバヘッドが生じる．

（3） GPU メモリ仮想化手法

Unified Memory やアドレス変換サービス（ATS）[6] などの機構により，CPU メモリを GPU メモリとして利用することが可能になる．一般に CPU メモリの大きさは GPU メモリの数倍から数十倍なので大規模深層学習モデルの処理が可能になる．ただし CPU メモリは GPU メモリより低速で，現状の GPU の仮想記憶機構は GPU メモリ以上の CPU メモリ利用を想定していないため性能が十分ではなく，実用的な性能を得ることが難しい[7]．

（4） 再計算手法

学習フェーズの処理は入力データから出力データを生成する順伝搬処理と出力データと正解を比較して各層のモデルパラメータを更新する更新処理の二つで構成される．順伝搬処理の各層で生成される中間データは同じ層の逆伝搬処理で必要とされる．この手法では順伝搬処理で生成した中間データの一部を削除し，逆伝搬処理で必要となったときに再計算することで GPU メモリ使用量を削減する．通常深層学習では GPU 資源はほぼ 100% 利用されるので再計算時間がそのまま処理時間の増加となる．

（5） データスワッピング手法

データスワッピング手法は学習フェーズの順伝搬処理で生成した GPU 上の中間データを一時的に CPU メモリに転送し，逆伝搬処理時にそのデータが必要になった時点で GPU メモリに転送することによりメモリ使用量を削減する（図 5）．CPU-GPU 間のメモリ転送を GPU 計算時間とオーバラップできれば少ないオーバヘッドで実装可能である．この手法は，バッチ数 1 であっても適用可能である．また，単一 GPU に適用可能で，データパラレル手法等他の手法との組合せも比較的容易である．

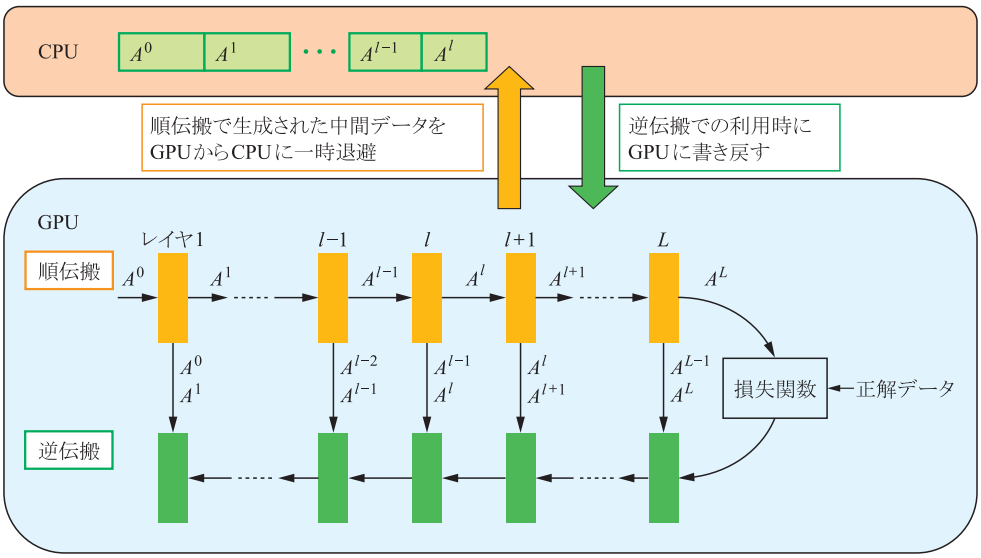


図 5 データスワッピング手法 順伝搬処理で生成された中間データを一時的に CPU メモリに転送し，逆伝搬処理で必要な時点で GPU メモリに書き戻すことにより GPU メモリ使用量を削減する．

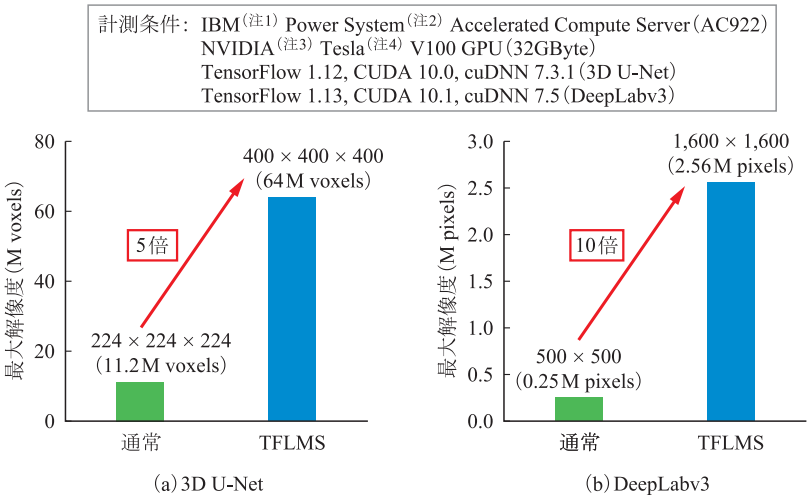
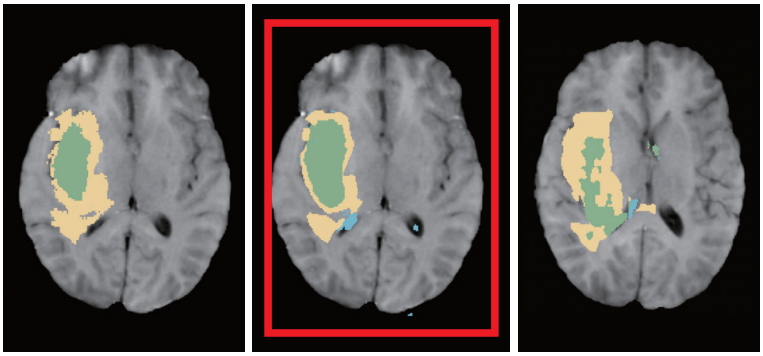


図 6　TFLMS による学習可能な最大解像度の向上　対象となる深層学習モデルによるが，TFLMS により学習可能な解像度が最大 10 倍向上する．



(a) 正解画像　(b) TFLMS(分割なし)　(c) 既存手法(分割あり)

図 7　3D U-Net を利用した脳腫瘍セグメンテーションの結果　TFLMS を利用して分割なしで画像解析を行うことにより，より精度の高い解析が可能になる．

3.3　高解像度医療画像への応用事例

IBM 東京基礎研究所で開発されたデータスワッピング手法を TensorFlow で実現するソフトウェア「TensorFlow Large Model Support（TFLMS）」[8]を利用した大規模深層学習モデルの実行事例として，高解像度画像を用いた画像のセグメンテーションの事例を示す[9]．セグメンテーションとは画像の各ピクセルをカテゴリー分類することで画像に含まれる物体の種類，位置，形状を認識する技術であり，医療画像における病変の認識や自動運転時の道路状況の認識など様々な分野で利用されている．

3D U-Net や DeepLabv3 は高性能なセグメンテーションを実現するために広く用いられているモデルである．図 6 のように，TFLMS を用いることで，それぞれ 5 倍，10 倍の解像度の画像が学習可能となる．このような高解像度の画像を用いることによって，より正確なセグメンテーションが実現可能となる．図 7 は 3D U-Net を用いた三次元の脳 MRI 画像の脳腫瘍のセグメンテーションの結果を示している．

従来手法では三次元 MRI 画像を分割することで GPU メモリを削減し高解像度画像の解析を行っているが，このような分割は精度の悪化を招く．TFLMS を用いて元の画像を分割することなしに学習することで，より正解画像に近い精度の高いセグメンテーションが可能となる．

4. ま　と　め

本稿では，深層学習システムの GPU への実装にあたっての技術課題とそれを克服するための実装技術につ

(注 1)　IBM は登録商標．
(注 2)　Power System は商標．
(注 3)　NVIDIA は登録商標．
(注 4)　Tesla は登録商標．

いて解説した．また，筆者らが主に取り組んでいるGPUメモリより大きい深層学習モデルをGPU上で実行するための大規模深層学習技術について，その必要性，実装方針，技術課題や実装技術について述べた．また，MRIによる高解像度医療画像の大規模深層学習技術による解析事例について紹介した．

文　　献

（1） “IMAGENET large scale visual recognition challenge（ILSVRC）2017,” 2017, http://image-net.org/challenges/LSVRC/2017/index

（2） M. Yamazaki, A. Kasagi, A. Tabuchi, T. Honda, M. Miwa, N. Fukumoto, T. Tabaru, A. Ike, and K. Nakashima, “Yet another accelerated SGD : ResNet-50 training on ImageNet in 74.7 seconds,” 2019, https://arxiv.org/pdf/1903.12650.pdf

（3） NVIDIA社，“CUDA Toolkit,” 2019, https://developer.nvidia.com/cuda-toolkit

（4） “Cupy : A NumPy-compatible matrix library accelerated by CUDA,” 2019, https://cupy.chainer.org/

（5） 杉山弘晃，成松宏美，東中竜一郎，“センター英語試験の意見要旨把握問題に対するBERTの適用方法の検討，”言語処理学会第25回年次大会，pp. 882-885, 2019.

（6） NVIDIA社，“NVIDIA TESLA V100 GPUアーキテクチャ，”2017, https://images.nvidia.com/content/pdf/tesla/Volta-Architecture-Whitepaper-v1.1-jp.pdf

（7） 根岸　康，今井晴基，土井　淳，河内谷清久仁，“Unified Memoryを用いた大規模ディープラーニングモデルの性能に関する考察，”日本ソフトウェア学会第34回大会，no. 一般1-2-L, 2017.

（8） T.D. Le, H. Imai, Y. Negishi, and K. Kawachiya, “Automatic GPU memory management for large neural models in TensorFlow,” the 2019 ACM SIGPLAN International Symposium on Memory Management（ISMM 2019）, pp. 1-13, 2019.

（9） H. Imai, S. Matzek, T.D. Le, Y. Negishi, and K. Kawachiya, “High resolution medical image segmentation using data-swapping method,” International Conference on Medical Image Computing and Computer-Assisted Intervention（MICCAI 2019）, pp. 238-246, 2019.

（2019年11月30日受付　2019年12月16日最終受付）

根岸　康（ねぎし　やすし）

平元東工大大学院理工学研究科情報科学専攻修士課程了．同年日本アイ・ビー・エム株式会社入社．以来，同社東京基礎研究所にてシステムソフトウェア・ハイパフォーマンスコンピューティングアプリケーション・深層学習フレームワークの性能最適化の研究に従事．ACM，情報処理学会各シニア会員．

今井　晴基（いまい　はるき）

平16筑波大大学院システム情報工学研究科知能機能システム専攻博士前期課程了．同年日本アイ・ビー・エム株式会社入社．以来，東京基礎研究所にてハイパフォーマンスコンピューティング分野等のアプリケーション高速化の研究開発等に従事．現在は，大規模深層学習の実行基盤の研究開発を行う．

Tung D. LE

平28総合研究大学院大複合科学研究科博士課程了．同年日本アイ・ビー・エム株式会社入社．以来，東京基礎研究所にてTensorFlow, Caffe, Chainer等の深層学習フレームワークの性能最適化の研究に従事．プログラム言語，AIシステムに興味を持つ．博士．ACM会員．

河内谷　清久仁（かわちや　きよくに）

昭62東大大学院理学系研究科情報科学専攻修士課程了．同年日本アイ・ビー・エム株式会社入社．以来，同社東京基礎研究所にてシステムソフトウェアの研究に従事．現在，同研究所システムズグループ部長．博士（政策・メディア）．情報処理学会シニア会員，ソフトウェア科学会会員，ACM Distinguished Member．