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

この記事は、Tech.Decoded に公開されている「Intel-Powered Deep Learning Frameworks」の日本語参考訳です。

さらなる洞察を提供

インテル コーポレーション、マシンラーニング & コンピューター・ビジョン、シニア・ソフトウェア・エンジニア、Pubudu Silva

人工知能 (AI) は、視覚認識、音声知覚、言語処理、意思決定のようなタスクを人間に代わって実行できる知的機械の概念であり、少なくともコンピューターが登場してから、次の大きな目標です。

マシンラーニングは、いくつかの重要な AI タスクを実行する際に非常に有効なことが分かっています。哺乳類大脳皮質のニューロン構造のルーズモデルである人工ニューラル・ネットワーク (ANN) は、その意欲的な設計とさまざまなタスクへの一般的適用性により、AI に有望な手法であると考えられていました。ANN の長所は、隠れた過渡状態 (隠れノード) を学習して維持するその能力にあります。この能力により、ANN は、いくつかの非線形関数をカスケードにして、入力から目的の出力まで、広範なマッピングを学習することができます。

学習した ANN では、隠れ層は階層的な段階のデータの内部抽象化を表し、深い層は高いレベルの抽象化を表します。哺乳類の脳も複数の階層的な処理層で情報を処理していると考えられています (例えば、霊長類の視覚系では、エッジ検出、初期形状検出の順で段階的に処理が行われ、徐々により複雑な視覚的形状に移行します)[1]。そのため、AI の研究には多層の「より深い」ANN が必然的に必要になります。

複数の深いカスケード層で段階的にデータを処理するネットワークは、一般に「ディープ・ネットワーク」と呼ばれます。サポート・ベクトル・マシン (SVM)、混合ガウス分布 (MoG)、k 近傍法 (kNN)、主成分分析 (PCA)、カーネル密度推定 (KDE) など、広く利用されているマシンラーニング・アルゴリズムの多くは 3 層を超える処理層を含んでいません。そのため、これらのアルゴリズムは「浅い」アーキテクチャーと考えることができます。層が 2 つから 3 つの ANN でも正常に訓練することはできますが、20 世紀の後半になると、より深い ANN を訓練する際に、いくつかの問題により訓練が失敗することがありました。直面した主な問題は次の 2 つです。

1. 勾配消失問題
2. 追加層により重みの数が増えたことによるオーバーフィッティング (過学習)

計算環境の進歩とともに、オーバーフィッティング問題は解消され、研究者は短時間に大量のデータサンプルでマシンラーニング・アルゴリズムを訓練できるようになりました。畳み込みニューラル・ネットワーク (CNN)

は複数の層の種類を備えたディープ・ネットワークですが、その重み共有手法により (同等の深さの全結合 ANN よりも) 多くの小さな重みを含みます。そのため、CNN は ANN よりも簡単に訓練できます。CNN の理論的な最良のパフォーマンスは、ANN よりもわずかに低いだけです。CNN は、教師あり画像学習タスクでは非常に一般的になりました。2006 年の Hinton ほかによる画期的な発見 [2] により考案された Deep Belief Network (DBN) やディープ・オートエンコーダーなどのディープ・ネットワークにより、教師なし学習も前進しました。

一般に、ディープ・ネットワークは、分類、回帰、画像キャプション生成、自然言語処理などの多くの AI 関連タスクで、ほぼすべてのほかのマシンラーニング・アルゴリズムよりもパフォーマンスが優れています。ディープ・ネットワークの大いなる成功は、特徴の階層を自立的に学習する方法によるものです。

ディープラーニングと従来の統計的学習法の大きな違いは、前者はローデータを自分で学習するのに対して、後者は人間が手を加えたデータの特徴を学習することです。ディープ・ネットワークは、DNN の学習が進むとともに、初期のレベルで与えられたタスクに最も適した特徴を自立的に生成します。その結果、オリジナルデータの学習タスク全体は費用関数ベースの最適化プロセスのままで、学習プロセスから当て推量や人による偏りが排除されます。

深い階層構造により、深い階層レベルではネットワークの前のレベルで学習した低レベルの特徴に基づいて高いレベルの特徴を学習できます。

図 1 は、どのように入力画像が徐々にディープ・ネットワークの高いレベルの表現に変換されるかを示しています。画像がネットワークで深くなるにつれ、より抽象的な表現になります。例えば、特徴の階層は、初期の層から順に、エッジ、形状、オブジェクトの一部、オブジェクト全体、シーンになることが分かります。しかし、実際には、「正しい」特徴ベクトルがこれらの抽象的な階層のどこに位置するか、学習なしで推測することは困難です。これは人間が手を加えた特徴を学習する場合の問題点です。一般に、深いネットワークでは出力層は高レベルの特徴を処理し、浅いネットワークよりも高いレベルの概念を学習できます。

さまざまなディープラーニング手法が開発者、データ・サイエンティスト、研究者の標準ツールになるとともに、ディープ・ネットワークの訓練およびスコア付けを支援する多くのディープラーニング・フレームワーク (Caffe、Tensorflow*、Theano*、Torch など) とライブラリー (MatConvNet、CNTK、Pylearn2、Deeplearning4j) が開発されています。これらのフレームワークとライブラリーは、退屈な定型作業の削減に大いに役立ちます。ユーザーは、個々のコンポーネントの実装ではなく、ディープラーニングの部分に労力をかけることができます。

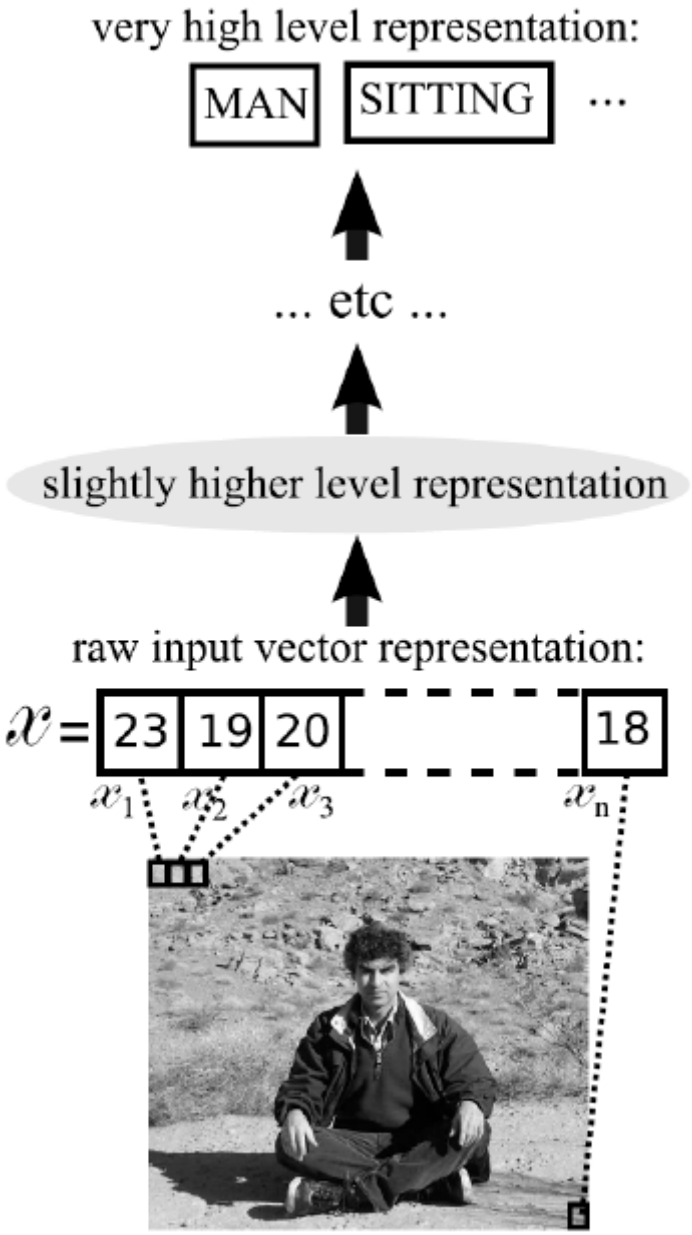

図 1 - 徐々に高いレベルの表現に変換される画像入力のディープ・ネットワーク処理
出典: Yoshua Bengio, Learning Deep Architectures for AI, 2009

多くのフレームワークとライブラリーはオープンソース・プロジェクトとして開発され、開発者コミュニティーによる積極的なサポートが行われているため、ユーザーはこれらのコードベースにアクセスできます。ディープラーニングは通常、処理に何日も (あるいは何週間も) かかる大規模なデータセットの訓練を含むため、一般的に使用されるディープラーニング・ソフトウェアのパフォーマンスの最適化は、テクノロジーの進歩にとって非常に重要です。

インテルは、特にインテル® アーキテクチャー向けにオープンソースのディープラーニング・フレームワークを最適化することにより、貢献を続けています。**マシンラーニング・サイト** (英語) には、インテルのマシンラーニ

ングおよびディープラーニングに対する取り組みの最新情報が含まれています。パフォーマンス最適化ツールと手法に関する詳細な情報も、このサイトから入手できます。いくつかの最適化作業は、ディープラーニング・アプリケーションを開発する際にソフトウェア開発者をガイドするケーススタディーとして、開発サイクルに使用できるフレームワークやライブラリーとともに公開されています。

例えば、Caffe の最適化で行ったプロセスは、ケーススタディー: **インテル® アーキテクチャー向けの Caffe の最適化: 最新のコード手法の適用** (英語) で紹介されています。**インテル® VTune™ Amplifier** は、CPU およびキャッシュの使用状況、CPU コアの使用状況、メモリーの使用状況、スレッドのロードバランス、スレッドロックなどのパフォーマンス最適化プロセスの初期ガイダンスとして使用できる、貴重な情報を提供する強力なプロファイル・ツールです。**インテル® マス・カーネル・ライブラリー (インテル® MKL)**、**インテル® スレッディング・ビルディング・ブロック (インテル® TBB)** 、OpenMP* などのライブラリーは、ディープラーニング・ソフトウェアの最適化に役立つことが分かっています。

ディープラーニング開発と研究を促進するため、インテルは、データ・サイエンティストおよびソフトウェア開発者がディープラーニング・ソリューションを開発、訓練、配備するための無料のツールセット、**インテル® Deep Learning SDK** (英語) を提供しています。SDK は Ubuntu*/CentOS* サーバーに接続する Web ベースのクライアントとして設計されています。インストール・ウィザードは、SDK とともに、サーバーのインテル® アーキテクチャー向けに最適化された一般的なディープラーニング・フレームワークをインストールします。SDK の訓練ツールは、GUI および高度な視覚化手法により、訓練データ、モデル設計、モデル選択の準備を単純化します。配備ツールは、モデル圧縮および重み量子化手法により特定のターゲットデバイス向けに訓練されたディープラーニング・モデルを最適化するために使用できます。

#DataScience (英語) #VisualComputing (英語)

参考文献 (英語)

1. Serre, T.; Kreiman, G.; Kouh, M.; Cadieu, C.; Knoblich, U.; and Poggio, T. 2007. “A quantitative theory of immediate visual recognition.” Progress in Brain Research, Computational Neuroscience: Theoretical Insights into Brain Function, 165, 33–56.

2. Hinton, G. E.; Osindero, S.; and Teh, Y. 2006. A fast learning algorithm for deep belief nets. Neural Computation, 18, 1527–1554. Your Path to Deeper Insight

インテルにより最適化されたディープラーニング・フレームワーク、ライブラリー、インテル® Deep Learning SDK などのディープラーニング・ツールの最新情報は、intel.com (英語) から入手できます。

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/performance (英語) 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。

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。