特　集「AIとデータ―データに基づく意思決定と社会イノベーション創出―」

機械学習工学へのいざない

Introduction to the Machine Learning Engineering

丸山　宏　株式会社 Preferred Networks
Hiroshi Maruyama　Preferred Networks, Inc.
hm2@preferred.jp, https://www.preferred-networks.jp/ja/

城戸　隆　（同　上）
Takashi Kido　kido@preferred.jp, https://www.preferred-networks.jp/ja/

Keywords: machine learning engineering, software engineering, reuse, quality..

1. はじめに

人工知能という言葉は多様な文脈で使われる．コンピュータサイエンスの世界では，「人工知能」はその時代における最先端の情報技術を指す，という側面が強い．初期の人工知能研究においては，記号処理のための動的メモリ管理（ガーベージコレクションなど）や，探索アルゴリズム，構文解析アルゴリズムなどが盛んに研究された．これらの技術は現在では情報技術の根幹に取り込まれていて，人工知能の研究分野と考える人は少ない．第2の人工知能ブームにおいてはフレーム理論などの知識表現が研究されたが，これらの成果はオブジェクト指向やモデリング言語として，通常のプログラム開発に取り入れられている．現在の人工知能の中心的課題（の一つ）は深層学習を代表とする統計的機械学習であるが，これもまた，新たなプログラミングの方法論として，情報技術の体系の中に取り入れられていくであろう．

機械学習は，プログラミングの方法論という観点からすると，本特集のテーマであるデータに基づく帰納的なシステム開発手法と位置付けることができる．今までの演繹的なシステム開発，すなわちシステムの仕様を定義し，先験的な知識に基づいてそれをモデル化し，段階的に詳細化していくという方法論に対して，機械学習に基づく帰納的なシステム開発は，仕様を訓練データの形で表現し，実装は訓練（training）によって行うという形を取る．帰納的なシステム開発においては，仕様定義の方法，実装の方法，テストの方法，運用の方法などが今までのシステム開発と大きく異なり，そのため演繹的システム開発を前提としたソフトウェア工学の方法論の多くがそのままでは適用できない．

現在のところ，帰納的なシステム開発は一部の機械学習のエキスパートによって行われていて，このためAI人材が不足しているという議論がある．この状況は，1960年代半ばの「ソフトウェア危機」によく似ている．当時，大型汎用機が登場して，プログラムを書くエキスパートが不足すると予想された．ここで登場したのがソフトウェア工学であり，FORTRANなどの高級言語，ガーベージコレクション，仮想記憶，オブジェクト指向，開発・テストツールなどさまざまな技術が開発され，またそれらを利用して高品質なソフトウェアを効率的に開発する方法論が確立されていった．これらの知見を知識体系としたのが，ソフトウェア工学という学問領域である．

同様に，機械学習という新しい計算パラダイムの導入によって，これに適した新しい開発方法論が求められている．これを我々は機械学習工学と呼ぶ．

本稿では，以下次章において，機械学習を応用したシステムの開発工程がどのように今までのITシステムの開発工程と異なるかを概観する．3章では，機械学習工学における主要な課題を，訓練データの準備と管理，生成物の再利用，品質の担保，解釈可能性，要求工学とプロジェクト管理，計算資源，ソフトウェアアーキテクチャの7点に分けて考える．4章では，機械学習工学を取り巻く現状を俯瞰し，将来を展望する．

2. 機械学習システムとは

機械学習システムの典型的な構成例を図1に示す．機械学習システムはある入力を受け取り，何らかの計算を施したのち出力を出す，一つの関数とみなすことができる．入出力は多くの場合多次元であり，各変数は離散値であるか連続値であるかを問わない．また，N個の異なるラベルを取るカテゴリー変数は通常，バイナリ値を取るN個の変数として表現する．

この関数には，二つの異なる処理過程があり，それぞれを訓練パイプライン，推論パイプラインと呼ぶ．**訓練**

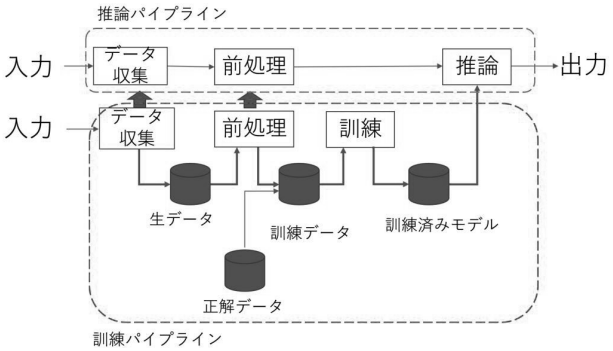

図1　機械学習システムの構成例

パイプラインは，訓練データセットから**訓練済みモデル**をつくる処理であり，主に開発時に処理が行われる．一方，**推論パイプライン**は実行時に通るパイプラインであり，訓練済みモデルを利用して入力を出力に変換する．

入力データは多くの場合，通常の ETL（Extract Transfer Load）などデータ収集処理を経てデータベースに格納される．これを**生データ**と呼ぶ．生データは，欠測値や外れ値を含んでいたり，そのままでは深層学習に入力する形になっていなかったりするので，**前処理**を行う．また，教師付き学習においては，訓練データのそれぞれの入力に対して望ましい出力を正解データとして与える必要がある．これら入力・出力ペアを**訓練データ**と呼ぶ．この訓練データを訓練アルゴリズムに与えて，訓練済みモデルを得る．

推論時には，入力データは訓練時と同じ ETL・前処理を経て，この訓練済みモデルに与えられ，出力に変換される．この出力は，訓練データの統計的性質を反映した近似予測値となる．

機械学習システムは関数であるが，すべての関数を正確に表現できるわけではない．しかし，すべての計算可能関数について，ある誤差範囲内で近似する深層ニューラルネットワークを構成できることが知られている [Cybenko 89]．このため，深層ニューラルネットワークは，擬似的にチューリング完全と考えることができる．

機械学習システムの開発は，多分に探索的になる．手に入るデータをもとに，要求される精度が出るかどうかは実際にやってみないとわからないからである．また，深層学習には多くのハイパーパラメータがあり，これらのチューニングもまた探索的となる．典型的な機械学習応用システムの開発プロセスの一例を図 2 に示す．

まずアセスメントフェーズでは顧客の要件定義を行い，ビジネス目標（KPI）を明確にする．同時に，機械学習システムは万能の人工知能ではなく，あくまでも帰納的なプログラミングの方法論であることを理解してもらうことも重要である．

要件とビジネス目標が明確になると，手に入るデータをもとに Proof of Concept（PoC）を行う．ここでの主眼は，機械学習コンポーネントの精度の見積もりである．現状の精度（ベースライン）を確認し，手に入るデータからいくつかの訓練済みモデルを構築し，最終的にこの機械学習コンポーネントが到達可能な精度の差の見積りを行うことである．もし，到達可能精度の推定値とベースラインの差が，ビジネス目標の達成に十分なものと判断されれば，パイロットフェーズに入る．

パイロットフェーズでは，実際のビジネスプロセスの一部を上記機械学習システムコンポーネントで置き換えて，実データを用いて実環境において期待される効果が出るかを確認する．多くの場合，既存システムとのインテグレーションを含み，既存のソフトウェア工学手法と連携しながらシステム開発が行われる．実環境における実データは，しばしば訓練データと違う振舞いをすることがあり，全社的な展開を行う前にこれを確認しておくことが重要である．このフェーズでは，アセスメントフェーズで決めたビジネス目標（KPI）が達成できることを確認する．

パイロット運用によって期待されるビジネス効果が出ることが確認されたら全社的に展開を行う．これ以降も，データの統計的性質を常に監視し，訓練済みモデルの精度が落ちてきた，あるいは入力データの統計的性質が大きく変わった場合には，モデルを再訓練することが必要となる．

3. 機械学習工学の主要な課題

機械学習システムの開発・運用では今までの演繹的なシステム開発とは異なる面が多い．我々が認識している主要な課題を 7 点，以下に述べる．

3·1 訓練データの準備と管理

データ分析の仕事の大部分はデータの前処理に費やされるといわれる．与えられたデータが仕様どおりであることはめったになく，外れ値や欠測値だけでなく，運用

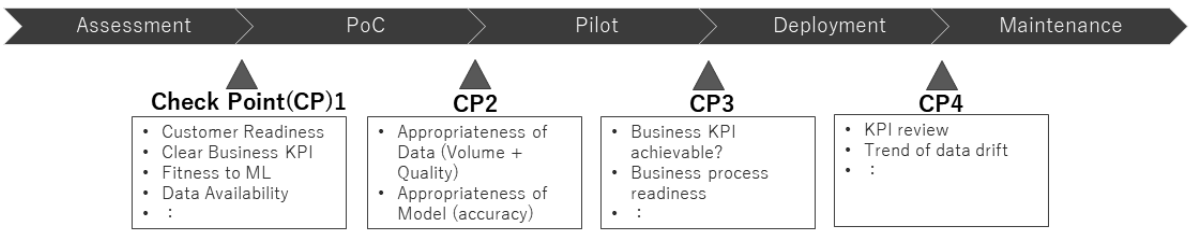

図2　機械学習システムの開発プロセスの一例

時に特別な意味をもたせた値を混入させることがよくあるからである．したがって，既存のシステムから得られるデータを利用したシステムの工数を見積もる場合，データクレンジングに多くの工数を想定しなければならない．

加えて，機械学習による帰納的プログラミングにおいては，各入力データに対してどのような出力が望まれるか，**教師信号**を付与しなければならない．これには主に四つのアプローチがある．

教師あり学習　多くの場合，教師信号は別のデータベースを参照することによって得られる．例えば，売上予測を行うシステムの場合，過去の実際の売上データが教師信号となる．人間が行っているタスクの置換えで帰納的プログラミングが使われる場合は，人手で教師信号を付与する（**アノテーション**と呼ばれる）ことも行われる．典型的なものは画像の判別問題などであり，自動運転などにおいては大きなコストをかけて人手で教師信号を作成することもある．大きな訓練データセットの場合，人海戦術になるので，クラウドソーシングを使ってアノテーションを行うことがよく行われる．

教師信号の与え方として，シミュレータを使う方法もある．画像から三次元モデルを認識する画像認識システムを考えよう．多くの画像に対して正しい三次元モデルをひも付けるためには，シミュレータ上で三次元モデルを生成し，それから得られる画像からコンピュータグラフィックスを生成すればよい．もう少し一般的には，ある関数の計算がよく知られているときに，ランダムに生成したデータ点に対して入出力ペアを計算して訓練データセットとすることにより，逆関数を計算する深層ニューラルネットワークを訓練することができる．

半教師あり学習　すべてのデータ点に対して個別にアノテーションを行うのはコストが高すぎる場合，多くのデータ点に対しては教師信号を与えずにその統計的な振舞いだけを学習させておき，比較的少数の教師付きデータを組み合わせる，半教師あり学習を利用することもできる．ただし，半教師あり学習はその振舞いがよくわかっていないこともあり，まだ広く利用されているわけではない．

教師なし学習　VAE（Variational Auto Encoder）や，GAN（Generative Adversarial Nets）などの生成モデルは，教師信号が全くなくても，機械学習アルゴリズムは入力データの統計的な性質を捉えることができる．これによって，例えば正常時の統計的性質を用いて，ある入力データが異常値であるかどうかの判定に使うことができる．

強化学習　教師信号はまた，各データ点に対して，得られた出力を見た後，その出力が望ましいものであったかをフィードバックすることによって，事後に与えることもできる．試行錯誤によって学習するロボットの制御などに使われることがある．

3·2 生成物の再利用

2章で議論したように，機械学習応用システムにおいては，通常のソフトウェア開発で得られるソースコードのほかに，訓練データセットと訓練済みモデルという二つの生成物がある．これらの生成物は，多くのコストと知見を結集したものであり，多大な価値をもつ知的財産と考えることができる．したがって，ソースコードや設計の再利用が今までのITの普及に極めて重要だったと同様に，訓練データセットや訓練済みモデルの再利用が，今後の機械学習応用システムの普及にとって鍵の一つになることは間違いない．

§1 訓練データセットの再利用

ビッグデータはしばしば「排気データ」と呼ばれる．ビジネスプロセスの副産物として生成されるデータが，本来の目的以外に使われることで価値を生むことが多いからである．その意味では，ビッグデータはすでに再利用されたデータということができる．

訓練データセットは，整形され正規化されたデータであり，さらに多くの場合正解データが付与されたものであり，極めて価値が高い．研究分野においては，訓練データセットの共有は広く行われているプラクティスである（例えば手書き文字認識のデータセットMNIST[*1]や，画像判別のデータセットであるImagenet [Deng 09]など）．また，自動運転の技術開発のために作成されたCityscap [Cordts 15]データセットは，自動車のカメラで取得した5 000枚の画像に，ピクセル単位でアノテーションをしたもので，自動運転の研究開発に多大な貢献をしている．

一方，訓練データセットの商業的な再利用については，まだ法律を始め一般的なルールが合意されておらず，なかなか再利用が進まない状況である．日本においては，著作権法の47条7において，著作権のあるデータに関して統計的処理を行った結果には，元の著作権が及ばないという明示的な規定があり[Ueno 17]，機械学習応用システムにおけるデータの再利用の促進が期待されている．

§2 訓練済みモデルの再利用

機械学習応用システムにおいて，もう一つの重要な再利用可能な生成物は訓練済みモデルである．訓練済みモデルは大まかにいって，ニューラルネットワークの構造と，各重みパラメータの組で表現される[*2]．訓練済みモデルの再利用には，図3に示すように，いくつかのパターンがある．

第1のパターンは，訓練済みモデルをそのままの形でコピーし，同じタスクに再利用することである．

第2のパターンは，訓練済みモデルに追加の訓練デー

---

*1 http://yann.lecun.com/exdb/mnist/

*2 Chainer [Tokui 15]などのフレームワークでは，ニューラルネットワークの構造が入力データによって動的に構成されることを許している．このような場合は，ニューラルネットワークの構造はプログラムコード片で表現され得る．

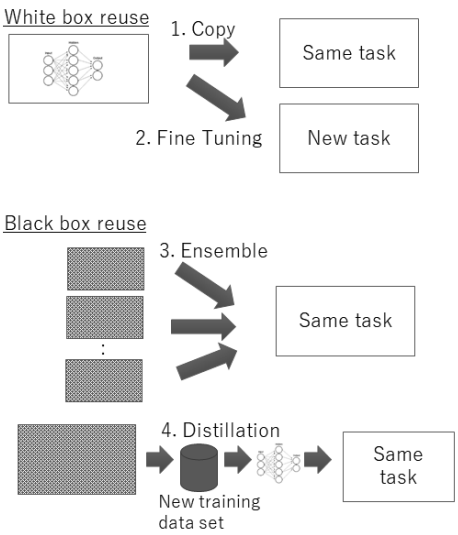

図3　訓練済みモデルの再利用パターン

タセットを加えて，似ているが異なる問題に適用することである．これを fine tuning と呼ぶ．特に画像処理などの場合，新しいモデルを一から訓練することは大きな計算量を必要とする．似た分野の訓練済みモデルから訓練をスタートさせることで，低コストで素早く訓練済みモデルをつくることができる．

上記二つのパターンは，訓練済みモデルの詳細までを知って再利用するパターンであり，ホワイトボックス再利用であるといえる．アカデミアにおいては，訓練済みモデルの共有は進んでいて，例えば Caffe Model Zoo[*3] においては画像処理を中心に多くの訓練済みモデルが公開されている．

訓練済みモデルの第 3 の再利用パターンは，アンサンブル（Ensemble）と呼ばれる．同じタスクを解く訓練済みモデルが複数ある場合，これらの訓練済みモデルを同じデータに対して適用し，それらの結果の平均を取ることで精度を向上できることが知られている [Dietterich 00]．このパターンにおいては，再利用の際にそれぞれの訓練済みモデルの詳細にアクセスする必要はない．入力を与えると出力を返すという API へのアクセスだけで，再利用が可能である．このため，アンサンブルはブラックボックス再利用であると考えられる．

第 4 の再利用パターンは，蒸留（Distillation）[Hinton 15] と呼ばれるものである．このパターンにおいては，元の訓練済みモデルは，新たな訓練データセットをつくり出すために用いられる．この新しい訓練データセットによって，全く別のニューラルネットワークを学習する．これも API 呼出しだけで行えるので，ブラックボックス再利用である．蒸留が法律的な意味でコピーに当たるのかどうかはまだ共通の理解がない．通常のソフトウェア開発においては，外部仕様だけから同機能のソフトウェアをつくり出すクリーンルーム開発が最も近い概念であるかもしれない．いずれにせよ，このあたりについては今後の議論が待たれる [Ueno 17]．

§3　練済みモデル交換フォーマット

訓練済みモデルの再利用には，システム間の相互運用性が欠かせない．訓練済みモデルの標準的な交換フォーマットとして，ONNX[*4]，NNEF[*5] が議論されている．

3·3 品質の担保

機械学習工学が従来のソフトウェア工学と大きく異なる第 2 の点が品質保証である．機械学習は本質的に統計的な性質をもつものであり，必然的にその品質も統計的に表現されることになる．機械学習応用システムの品質に関して問題となる点を下記に指摘する．

§1　バグと精度の交互作用

深層学習は非常に多次元の入力データに対して，次元間の交互作用をモデル化することができる．このことは同時に，一つの入力次元の値の変化が，すべての出力次元の値に影響を与える，ということを意味する．一般に，深層学習を使ったシステムは，システム中の任意の変更が他のすべての部位に影響を与える CACE（Change Anything Changes Everything）[Sculley 15] と呼ばれる性質をもつ．ソフトウェア工学における重要な価値観の一つである関心事の分離 [Parnas 72] とは反対の方向性である．

CACE は，次元間の交互作用だけでなく，深層学習のモデルと図 1 におけるパイプライン中の他のプロセスとの間でも発生する．例えばデータの前処理の内容が少しでも変われば，学習後のニューラルネットワークの重み全体が変化する．また，パイプラインのどこかに，精度に影響を与えるソースコードのバグがあったとしても，機械学習がその問題を隠すように学習を行ってしまうため，期待された精度が得られなかった場合，それが与えられた訓練データの品質によるものなのか，ハイパーパラメータの設定によるものなのか，それともソースコードのバグによるものなのかの切分けが困難である．

§2　テスト

機械学習応用システムをどのようにテストするかについても，まだ良いプラクティスが得られていない．一般のソフトウェア開発でよく知られている回帰テストという概念を考えてみよう．回帰テストとは，ソフトウェアに変更が加えられたときに，変更前に通っていたテストケースがすべて通ることを確認するためのテストである．機械学習応用システムにおいてはしかし，この概念はそのまま適用できない．モデルを更新して平均的により高い精度になったとしても，過去のテストケースの中には，今まで（たまたま）うまくいっていた入力が，新

*3 http://caffe.berkeleyvision.org/model_zoo.html

*4 http://onnx.ai/
*5 https://www.khronos.org/nnef

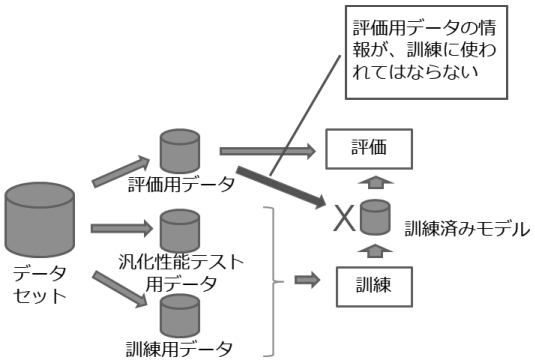

図4 機械学習システムのテストと情報の漏れ構成例

しいモデルでは正しい値を返さないことがあるからである．このほかにも，テストカバレッジはどのように定義されるべきか，機械学習応用システムのテストにおけるコーナケースとは何か，など考えるべきことは多い．

機械学習応用システムの精度はどのように評価すればよいだろうか．訓練時に，訓練データセットは，訓練用データ（training set）と汎化性能テスト用データ（validation set）に分割される．訓練データセットは，システムのパラメータを調整するのに用いることを述べたが，この汎化性能テスト用データは，出来上がった訓練済みモデルが未知のデータポイントに対してどの程度の誤差をもつか（これを**汎化誤差**と呼ぶ）を推定する．この汎化誤差はしかし，機械学習応用システム訓練済みモデルの最終的な精度を正しく表現しているとはいえない．汎化性能テスト用データによる精度に基づいてハイパーパラメータのチューニングを行うので，この汎化性能テスト用データの情報が訓練学習にフィードバックされてしまい，汎化性能テスト用データを含む訓練データセット全体に対して過学習してしまうからである．機械学習応用システムを正しく評価するためには，このような**情報の漏れ**がない，独立した評価用のデータセットを用いる必要がある [Wujek 16].

§3 セキュリティ

セキュリティに関して，機械学習応用システムには，新たなタイプの脅威がある．例えば [Evtimov 17] では，道路標識を見分ける判別器に対して，標識に物理的にステッカーを貼ることで一時停止標識をスピード制限標識に見間違えさせる，という攻撃が報告されている（図5）．

現在の機械学習アルゴリズムは基本的に，入力データの確率分布が，学習時と推論時で変化しないことを前提としている．このため，訓練データセットに本来は現れないようなデータを混入させたり，推論時に訓練データセットに現れないようなデータに加工したりして，機械学習応用システムに本来意図しない出力を出させるような攻撃が可能である．このような攻撃に対してどのように機械学習応用システムを守っていくかも，今後の課題といえよう．

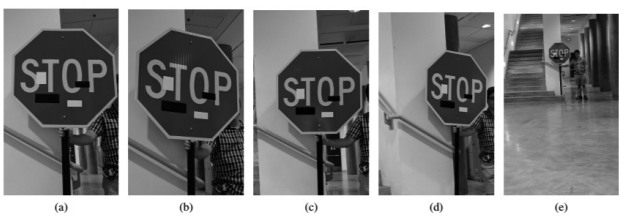

図5 交通標識を見分ける画像判別器への攻撃 [Evtimov 17]

§4 プロダクト品質とプロセス品質

機械学習応用システムの品質については上記のような難しさもあるが，一方で機械学習応用システムならではの可能性もある．通常のソフトウェア開発においては，出来上がったソフトウェア自体の品質（**プロダクト品質**と呼ぶ）を直接計測することは難しい．例えばプロダクト品質の指標として「残っているバグの数」を考えるとすると，そのようなものは正確にはわからないからである．このため，通常のソフトウェア開発で用いられる品質指標の多くは，そのソフトウェアを開発したプロセスの品質，例えば設計レビュー，コードレビューを正しく行ったか，などを評価したものになる．これを**プロセス品質**と呼ぶ．プロセス品質はしかし，出来上がったソフトウェアの品質をダイレクトに表現したものではない．

一方，機械学習応用システムにおいては，独立した評価用データセットによる客観的な精度をプロダクト品質の指標として用いることができる．その意味で，機械学習応用システムは，品質をより直接的に管理できるソフトウェア開発手法，ということもできるであろう．

3·4 解釈可能性（Interpretability）

機械学習工学に求められている説明性とは何であろうか？ 機械学習分野では，解釈可能性（Interpretability）という言葉がよく用いられるが，さまざまな文脈で多義的に用いられており，統一的な定義はなされていないのが現状である．また，「解釈可能性」がどのように定義されるかは多分に主観的な面もある．

Lipton [Lipton 16] は，機械学習モデルのどのような性質がモデルの解釈可能性（Model Interpretability）を高めるのかを論じている．Lipton は，解釈可能性を透明性（Transparency）と事後解釈可能性（Post-hoc Interpretability）に大別している．透明性は，ブラックボックス性と対になる概念で，透明性が高い（ブラックボックス性が低い）とは，機械学習モデルが動作するメカニズムがユーザに理解しやすいことを意味する．さらに透明性を（1）シミュレーション可能性（Simulatability）；人間が機械学習モデルの計算を追試できる，（2）分解可能性（Decomposability）；モデルを直観的に説明可能な各部分に分解できる，（3）アルゴリズムの透明性（Algorithmic Transparency）；学習アルゴリズムの挙動（収束性など）の妥当性を保証できる，に分類している．Deep Learning の結果はしばしば解釈不能であるといわれるが，高次元データにおいては，線

形モデルは Deep Learning よりも解釈可能性が高いとは必ずしもいえなくなる．高次元においては，線形モデルの透明性は，シミュレーション可能性，分解可能性において低下するからである．

事後解釈可能性は，結果を説明する付加情報の有益性を表す概念で，事後解釈可能性が高いとは，結果を説明する付加情報がユーザにとって有益であることを意味する．例えば，結果を説明する付加情報には，テキストによる説明 （ニューラルネットワークによるイメージキャプショニングなど），可視化 （t-SNE を用いたクラスタリングなど），局所データによる説明 （画像の一部をマスクして局所的な近傍領域の影響度を示すなど），例による説明 （どんな関係性が機械学習されたかを類似例を提示することによって示す）などがあげられる．

Ribeiro [Ribeiro 16] らは，複雑な分類器の判断基準を人間にも解釈可能になるように提示する手法として Local Interpretable Model-agnostic Explanations（LIME）を提案している．これは上述した局所データによる説明の一種で，説明したい事例 x の周辺から局所的にサンプリングした事例で解釈可能な分類器を訓練する．例えば，Newsgroup 文書の分類において，分類に影響した特徴とその影響度を示すことによって個別の予測についての説明を加える．利用者が説明を見て無関係と思う特徴を段階的に除去して分類器を再構築することにより，分類器の性能を向上させることができたことを報告している．

Koh らは，ブラックボックスと思われていた予測結果を理解するための計算モデルを提案している [Koh 17]．もしその訓練データを用いなかったら予測結果がどうなるかという発想で，統計学で古くから利用されていた影響関数（influence function）を用いて，個々の訓練データの有無や摂動が予測結果に与える影響を定式化した．この手法により訓練データにノイズを加える影響関数を利用することで，あるテストデータに対して予測を最も間違えるような訓練データを意図的に作成することが可能になる．またテスト誤差と訓練誤差に大きな乖離があった場合に，間違えたサンプルに対して影響関数が大きかったサンプルを探すことで，その乖離の原因を特定することも可能となる．訓練データに誤ったラベルが付与されたデータが混入されている場合，人手で誤ったラベルを探すのは大変だが，影響関数を用いることで楽に直せたという報告もなされている．

3·5 要 求 工 学

§1 要求発生のタイミング

初期のソフトウェア工学において，システム開発はウォータフォールモデルによって行われることが一般的であった．その背景には，一度下流まで開発が進んでしまうと，上流まで手戻りすることに極めて大きなコストがかかる，という「ものづくり」的な考え方があった．ソフトウェア工学の長い歴史の中で，ソフトウェアの仕様は事前には決めにくいということが徐々に理解され，近年ではアジャイル開発の手法が主流になってきている．

機械学習システムにおいては，要求は訓練データセットの形に翻訳され，実装は学習アルゴリズムによって（半）自動的に行われる．実装にかかる手間が相対的非常に小さいため，新たな要件，すなわち新たな訓練データセットに対応することが（訓練にかかる計算コストを無視すれば）相対的に低コストである．したがって，さまざまな要件を探索的に試すことができ，アジャイルな開発を加速することが期待できる．

さらに，強化学習に基づくシステムでは，そもそも要求はシステムが実際に走った結果に対するフィードバックとして報酬関数の形で与えられる．すなわち，要求はシステム開発の事後に与えられると考えてもよい．

§2 要求の厳密性

一方で，機械学習においては実装の過程に人が介在しないため，時として人が意図しない振舞いをすることがある．例えば，強化学習に基づく自動運転システムにおいて，安全性を高めるために衝突のペナルティを無限大にすると，全く動かない車になってしまう．動く車にするためには，動くという効用と，衝突のコストのバランスを，報酬関数の中に明示的に示さなければならない．報酬関数の決め方は，Reward Engineering と呼ばれ，その重要性と難しさが議論されている [Dewey 14]．

3·6 計算資源とその管理

深層学習の訓練には膨大な計算パワーが必要である．このために現在最も広く使われているのが，GPU と呼ばれる，元々グラフィックス処理用に設計されたプロセッサであり，深層学習における中心的な計算においては，汎用の CPU に比べて 1 桁以上高速である．一方，GPU は汎用プロセッサではないため，入力データの準備や出力の変換などには CPU が必要となり，深層学習を使ったシステムはヘテロジニアスな並列計算システムとなることが多い．残念ながら現状では，与えられた計算負荷を自動的に CPU と GPU に最適に振り分けるような仕組みがなく，プログラマは CPU の計算と GPU の計算を意識してプログラミングしなければならない．加えて，GPU の仮想化はまだ十分に普及していないので，少なくとも現状ではプログラマは例えば GPU の物理メモリを意識してプログラミングする必要がある．

また，深層学習のワークロードは訓練時と推論時とでは異なる．したがって，それぞれに最適なアーキテクチャがあり得る．例えば Google の TPU（第 1 世代）や，Microsoft の BrainWave[*6] は，推論に特化したコンピュータアーキテクチャである．

---

*6 https://www.microsoft.com/en-us/research/blog/microsoft-unveils-project-brainwave/

深層学習のワークロードはまだ十分に解明されていないこともあり，まだまだこれから新しいアーキテクチャが現れることも考えられる．これらのヘテロジニアスな計算環境を効率的に管理・運用することも機械学習工学の課題の一つといえよう．

3·7 ソフトウェアアーキテクチャパターン

§1 演繹的開発 vs 帰納的開発

深層学習は万能ではない．深層学習を含む統計的機械学習にはその本質的な限界もある．

1. 訓練時と推論時の確率分布が同じでなければならない．例えば販売予測のモデルをつくった後，市場に大きなインパクトのある新製品が投入された場合，モデルの統計的前提が崩れているかもしれない（これを**概念ドリフト**と呼ぶ）．
2. 訓練データに現れないまれな現象に対しては正しく推論できない．したがって巨大地震や飛行機事故などの希少事象のもとでの予測などは機械学習には困難である．
3. 訓練データに統計的バイアスが含まれることが避けられない．このため，深層ニューラルネットワークの出力には本質的に誤差があり，100％正確な値を得ることは不可能である．

加えて，深層学習は大量の訓練データと，膨大な計算資源を必要とする，という実用上のボトルネックもある．機械学習を用いてあるシステムを構築しようとしたとき，システム全体の中でどこに機械学習を用いるかについては慎重に検討する必要がある．

一般に，ある問題を機械学習によって帰納的に解くか，あるいは従前の演繹的なプログラミングによって解くかは，（1）先験的なモデル·解法の有無，（2）経験的なデータの有無，（3）誤差が許容されるか否か，の3点を考慮するとよい．

先験的なモデル，あるいは解法がわかっている問題に関しては演繹的に解くほうが一般的に高速であり精度も高い．例えば平方根の計算を x^2 の逆関数として機械学習によって解くことも可能であるが，よく知られているアルゴリズムで解いたほうが高速で精度も良い．また，銀行の勘定系システムのように，誤差が許されない計算も演繹的なシステム開発をすべきである．

一方，画像認識など先験的なモデルや解法がわからない，あるいは実世界を対象とするロボットなど環境が複雑すぎてモデル化が著しく困難であるような場合，経験的な入出力データが観測できるのであれば，帰納的なシステム開発をするべきである．

§2 演繹的開発と帰納的開発の組合せ

一つの大きな情報システムを開発する場合，システム全体をEnd-to-endで帰納的につくることも可能[Bojarski 17]であるが，一般には演繹的に開発するモジュールと帰納的に開発するモジュールの組合せとなる．GoogleやMicrosoftが提供する画像認識や音声認識APIは，演繹的なシステムから帰納的なシステムを呼び出す一例といえる．

もう少し詳細に見ると，既存のシステムの中でも，一部を帰納的開発に基づくモジュールに置き換えることも議論されている．例えば[Kraska 17]においては，データベースのインデックスを深層ニューラルネットワークで置き換えることで，インデックスのサイズを大幅に小さくできることが報告されている（図6）．

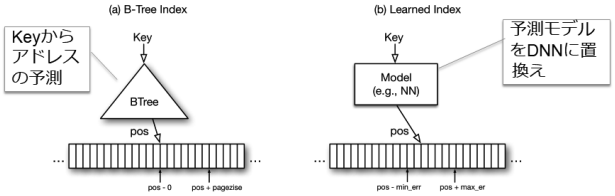

図6 B-Treeインデックスを深層学習に置き換える[Kraska 17]

演繹的プログラミングと帰納的プログラミングの組合せを前提とすると，また異なるアーキテクチャパターンも見えてくる．ゲーム木を探索するAlphaGo[Silver 17]，定理証明において推論木を探索するholophrasm[Whalen 16]はどちらも，問題を木の探索問題として定式化し，どの枝を優先的に探索するかの決定を深層ニューラルネットワークに任せている．これは，問題を非決定性アルゴリズムとして定式化し，DNNをオラクルとして用いる，ということにほかならない．これは，NP完全問題に対する新しい解法パターンといってよいだろう．

4. 機械学習工学の現状と展望

機械学習の実世界応用が進み，今までのソフトウェア開発のやり方との違いが認識されるにつれ，機械学習工学に関する関心が高まっている．最も多く機械学習工学の知見を蓄積しているのがGoogleやFacebookなどの企業であり，少しずつそれらの知見が公開されつつある．論文は，arXiv.orgに掲載されるほか，NIPSにおけるワークショップなどで発表されている[NIPS 17]．

日本においては，2017年8月に情報処理学会ソフトウェア工学シンポジウムでパネル討論「機械学習とソフトウェア工学」を行ったほか，2018年4月には「機械学習工学研究会」が日本ソフトウェア科学会の研究会として正式に発足する予定である．機械学習の研究者・実務家と，ソフトウェア工学の研究者・実務家がそれぞれの専門知識・経験を生かして，機械学習工学を推進していくことが期待されている．

今後，多くの情報システムが帰納的プログラミングに基づいて開発されるようになるのは間違いない．そのようなシステムの開発が，品質良くかつ効率的に行われるよう，機械学習工学の知識体系化を進めていきたい．

◇ 参 考 文 献 ◇

[Bojarski 17] Bojarski, M., Yeres, P., Choromanaska, A., Choromanski, K., Firner, B., Jackel. L. and Muller, U: Explaining how a deep neural network trained with end-to-end learning steers a car, arXiv preprint arXiv:1704.07911（2017）

[Cordts 15] Cordts, M., Omran, M., Ramos, S., Scharwächter, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S. and Schiele, B.: The cityscapes dataset, *CVPR Workshop on the Future of Datasets in Vision,* Vol. 1, No. 2, pp. 3（2015）

[Cybenko 89] Cybenko, G.: Approximations by superpositions of sigmoidal functions, *Mathematics of Control, Signals, and Systems,* Vol. 2, No. 4（1989）

[Deng 09] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. and Fei-Fei, L.: Imagenet: A large-scale hierarchical image database, *Computer Vision and Pattern Recognition*（*CVPR 2009*）, *IEEE Conf. on,* pp. 248-255, IEEE（2009）

[Dewey 14] Dewey, D.: Reinforcement learning and the reward engineering principle, AAAI Spring Symposium Series（25 March 2014）

[Dietterich, 00] Dietterich, T. G., et al.: Ensemble methods in machine learning, *Multiple Classifier Systems,* Vol. 1857, pp. 1-15（2000）

[Evtimov 17] Evtimov, I., Eykholt, K., Fernandes, E., Kohno, T., Li, B., Prakash, A., Rahmati, A. and Song, D.: Robust physical-world attacks on machine learning models, ArXiv e-prints（2017）

[Hinton 15] Hinton, G., Vinyals, O. and Dean, J.: Distilling the knowledge in a neural network, arXiv preprint arXiv: 1503.02531（2015）

[Koh 17] Pang W. Koh and Percy Liang, Understanding black-box predictions via influence functions, *ICML 2017*（2017）

[Kraska 17] Kraska, T., et al.: The case for learned index structures, arXiv preprint arXiv:1712.01208（2017）

[Lipton 16] Lipton, Z. C.: The mythos of model interpretability, *IEEE SMC Workshop, The Human Use of Machine Learning Venice*（2016）

[NIPS 17] https://nips.cc/Conferences/2017/

[Ribeiro 16] Ribeiro, M. T., Singh, S. and Guestrin, C.: Why should I trust you?, Explaining the predictions of any classifier, *Proc. 22nd ACM AIGKDD Int. Conf. on Knowledge Discovery and Data Mining*（*KDD 16*）, pp. 1135-1144, ACM（2016）

[Parnas, 72] Parnas, D. L.: On the criteria to be used in decomposing systems into modules, *Commun., ACM,* Vol. 15, No. 12, pp. 1053-1058（1972）

[Sculley 15] Sculley, D., et al.: Hidden technical debt in machine learning systems, *Advances in Neural Information Processing Systems,* pp. 2503-2511（2015）

[Silver 17] Silver, D., et al.: Mastering the game of go without human knowledge, *Nature,* Vol. 550, No. 7676, p. 354（2017）

[Tokui 15] Tokui, S., et al.: Chainer: A next-generation open source framework for deep learning, *Proc. Workshop on Machine Learning Systems*（*LearningSys*）*in the 28th Annual Conf. on Neural Information Processing Systems*（*NIPS*）（2015）

[Ueno 17] Ueno, T.: Copyright issues on artificial intelligence and machine learning, *1st Int. Workshop on Sharing and Reuse of AI Work Products*（2017）

[Whalen 16] Whalen, D.: Holophrasm: A neural automated theorem prover for higher-order logic, arXiv preprint arXiv: 1608.02644（2016）

[Wujek 16] Wujek, B., Hall, P. and Günes, F.: *Best Practices for Machine Learning Applications,* SAS Institute Inc.（2016）

2018 年 1 月 22 日　受理

著 者 紹 介

丸山　宏（正会員）

株式会社 Preferred Networks 最高戦略責任者．1983 年東京工業大学大学院理工学研究科修士課程修了．同年，日本アイ・ビー・エム株式会社入社．人工知能，自然言語処理，機械翻訳などの研究に従事．1995 年京都大学より博士（工学）授与．1996 年米 IBM 株式会社ソフトウェア事業部において，インターネット技術の評価，以降，XML や Web サービスの研究開発・標準化に従事．1997 ～ 2000 年東京工業大学情報理工学研究科客員助教授，2003 年アイ・ビー・エム・ビジネスコンサルティング・サービスへ出向．2006 ～ 09 年日本アイ・ビー・エム株式会社東京基礎研究所所長．2009 ～ 10 年キヤノン株式会社デジタルプラットフォーム開発本部副本部長．2011 ～ 16 年情報・システム研究機構統計数理研究所教授．2016 年 4 月より現職．

城戸　隆（正会員）

1996 年慶應義塾大学大学院理工学研究科計算機科学専攻にて遺伝的アルゴリズムを用いたハイブリッド探索に関する研究で博士（工学）号取得．1997 年から NTT 情報通信研究所知的エージェントグループ，1998 ～ 2001 年マレーシアのマルチメディアスーパーコリドー計画のもとで NTT 海外 R&D 拠点立上げのために赴任．2002 ～ 06 年遺伝子解析のベンチャー企業である HuBit Genomix 社にて疾患関連解析の研究開発に従事．2006 ～ 09 年，スタンフォード大学 Genetics Department の Stanford Microarray Database group にて客員研究員．2009 年より理研ジェネシス社で遺伝子解析研究に従事，2010 年より JST さきがけの大挑戦型プロジェクトとしてパーソナルゲノムに関する研究を遂行，2017 年より株式会社 Preferred Networks にて AI を用いたがん診断などの研究開発に従事．