製薬協

AI ってなに？

日本製薬工業協会
データサイエンス部会
2018 年 タスクフォース 6
2019 年 5 月

目次

はじめに

日本製薬工業協会 医薬品評価委員会 データサイエンス部会のタスクフォース 6 では，昨年加盟会社を対象に AI への取り組み状況についてアンケート調査を行い，その結果を 2018 年 12 月に報告書として公表した．アンケートでは，AI に関する人材不足（開発するエンジニアだけでなく，AI を用いる企画立案できる人材の不足も含む）に加えて，そもそも AI で何ができるかわからないといった回答も少なくなかった．

本書では，AI を開発するエンジニアではなく，主に製薬業界で AI をビジネスに導入しようとする担当者や AI を用いたサービスを活用したいと考えるユーザ向けに，AI とはどういったものか，AI を開発・導入するにはどういった情報や環境が必要となるのか，また現時点あるいは近い将来に AI を用いると具体的に何ができるのかを説明・紹介する．また，AI の実装例として，胸部 X 線画像から胸水を診断する AI の開発について，実行できることを確認したプログラムと共に掲載した．AI の開発に興味を持ち，開発にチャレンジしてみよう，という方にとっても参考にしていただけると考える．

本書が製薬業界，特に臨床開発から市販後関連での AI 導入，活用の一助となれば幸いである．

なお，本書で引用等行っている URL の最終参照日は全て 2019 年 3 月 20 日である．

1. AIってなに？

人工知能（Artificial Intelligence；AI）は様々な分野・場面で用いられ，その効果・有用性が大きな話題になり，製薬企業においても新薬開発を含めた様々な場面における AI の利用可能性が大いに期待されている．一方で，AI について複数の定義が提案されており，共通の定義として広く認識されたものは未だ確立していないのが現状である．この状況では，プロジェクトチームで AI を活用することになった際，チームメンバー間で「AI」の認識が異なり，AI を活用できない結果，プロジェクトの進捗に支障をきたすことが危惧される．

本章では，まずこれまでに人工知能に関わる研究者により提案されてきた AI の定義及び一般的に捉えられている AI のイメージをまとめた上で，本報告書内での「AI とは何か？」に対する考えを示す．これまでに提案されてきた定義や一般の捉え方全てを網羅している訳ではないが，これが今後製薬業界における「AI とは何か？」の認識を共通化するための一助につながれば幸いである．

1.1. これまでに人工知能に関わる研究者により提案されてきた AI の定義

1956 年に開催されたダートマス会議にて，「人工知能」という言葉がジョン・マッカーシーによって始めて使われて以降，多くの AI 研究者によって AI の定義が提案されてきた[1]．それらのうち代表的なものを**表 1-1** に示す．AI は，大まかには「知的な機械，特に，知的なコンピュータプログラムを作る科学と技術」と説明されているものの，その定義は研究者により異なっている状況である [2].

AI という研究分野は，一般に考えられているよりも，もう少し学術的な色合い，あるいは真理の追究の色合いが濃いと言える[3]．AI の研究は①人間の知能そのものをもつ機械を作ろうとする立場（汎用型人工知能，強い AI とも表現される[4]）と②人間が知能を使ってすることを機械にさせようとする立場（特化型人工知能，弱い AI とも表現される[4]）があり，実際の研究のほとんどは②の立場に立っている[5]．即ち，人間の脳と同じ機能・プロセスにする必要はなく，知能の原理を見つけ，違う方法であっても人間の脳と同じ判断をコンピュータで実現できればよいと考えられている[3,5].

表 1-1 国内の主な専門家による AI の定義[3]

研究者	所属	定義
中島　秀之	公立はこだて未来大学	人工的につくられた，知能を持つ実体．あるいはそれをつくろうとすることによって知能自体を研究する分野である
西田　豊明	京都大学	「知能を持つメカ」ないしは「心を持つメカ」である
溝口　理一郎	北陸先端科学技術大学院大学	人工的につくった知的な振る舞いをするもの（システム）である
長尾　真	京都大学	人間の頭脳活動を極限までシミュレートするシステムである
堀　浩一	東京大学	人工的につくる新しい知能の世界である
浅田　稔	大阪大学	知能の定義が明確でないので，人工知能を明確に定義できない
松原　仁	公立はこだて未来大学	究極には人間と区別がつかない人工的な知能のこと
武田　英明	国立情報学研究所	人工的につくられた，知能を持つ実体．あるいはそれをつくろうとすることによって知能自体を研究する分野である（中島氏と同じ）
池上　高志	東京大学	自然にわれわれがペットや人に接触するような，情動と冗談に満ちた相互作用を，物理法則に関係なく，あるいは逆らって，人工的につくり出せるシステム
山口　高平	慶應義塾大学	人の知的な振る舞いを模倣・支援・超越するための構成的システム
栗原　聡	電気通信大学	工学的につくられる知能であるが，その知能のレベルは人を超えているものを想像している
山川　宏	ドワンゴ人工知能研究所	計算機知能のうちで，人間が直接・間接に設計する場合を人工知能と呼んでよいのではないかと思う
松尾　豊	東京大学	人工的につくられた人間のような知能，ないしはそれをつくる技術

1.2.　現在一般的な AI の捉え方

インターネットの検索エンジンや Google の音声検索・翻訳機能等，AI を利用した多くの商品・サービスが既に世の中に溢れている．そのような状況もあり，最近，様々な場面で「人工知能」「AI」という言葉が聞かれるが，これらは，特化型人工知能という意味で使われていると言える．東京大学准教授　松尾豊先生は，著書「人工知能は人間を超えるか」で，世間一般の AI の捉え方について以下のように述べている[3]：

“「ある製品に知能がある」というときに，最もイメージしやすいのが，「その製品が何か考えているように見える」ことであろう．掃除ロボットの「ルンバ」であれば，部屋の形やゴミの状況によって動きが変わる．人工知能内蔵の洗濯機であれば，洗濯物の量や温度，湿度等によって洗濯のしかたが変わる．状況に応じて，どのように動作すればよいかと考え，より「賢い」振る舞いをする．つまり，入力（人間の五感に相当する「センサー」により観測した周囲の環境や状況）に応じて，出力（運動器官に相当する「アクチュエーター」による動作）が変わるということである．（中略）人工知能をエージェント（ソフトウェアのプログラム）と考え，その入力と出力の関係から考えると，世の中で語られている人工知能も理解しやすい．”

松尾准教授はまた，世の中で AI と呼ばれるものを，以下のレベル 1 から 4 の 4 段階で整理した[3]：

- レベル 1 :単純な制御プログラムを「人工知能」と称している
- レベル 2：古典的な人工知能（ルールベース）
- レベル 3：機械学習（マシンラーニング）を取り入れた人工知能
- レベル 4：ディープラーニング（深層学習）[1]を取り入れた人工知能

この切り口は，これまでに AI と呼ばれてきた技術や手法が時代により変化してきたことを示すと同時に，AI と機械学習・深層学習との関係を理解するためにも有用と考えられる．AI と機械学習・深層学習との関係については，後の 1.3 項でもう少し詳細に紹介する．

1.3. AI と機械学習・深層学習の関係

最近では AI という言葉とともによく目にする用語として，機械学習や深層学習がある．本項では，それぞれの概略を紹介する（詳細は第 3 章を参照のこと）．後述の**図 1-1** のとおり，深層学習は機械学習の 1 つのカテゴリだが，それぞれ別に取り扱っている書籍も多く説明もしやすいことから（例えば[3，6]），本報告書でも単に「機械学習」と表記した場合は深層学習を含めず，含む場合は節，項の初めに「深層学習を含む機械学習」と明示する．

1: [3]では「ディープラーニング」を用いているが，本報告書では「深層学習」を用いる．

1.3.1. AI と呼ばれるものの変遷

コンピュータ技術や社会インフラ設備の発展とともに AI と呼ばれるものは変化してきた．1.2 項で紹介した 4 段階の AI の変化とともに AI，機械学習，深層学習の関係を**図 1-1** に示す[6, 7].

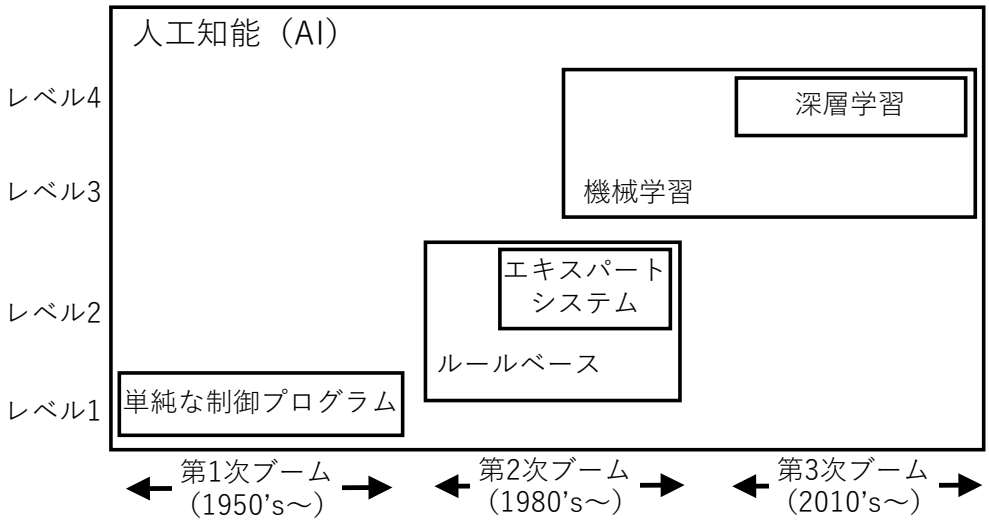


図 1-1 AI の変遷

1950 年代に入り，コンピュータの登場に伴い，簡単な条件を記述した単純な制御プログラムからなる AI（レベル 1）が生まれた．コンピュータによる「推論」や「探索」の研究が進み，特定の問題に対して解を提示できるようになったことから，第 1 次 AI ブームが発生した．しかし，迷路や数学の定理の証明のような簡単な問題（トイ・プロブレム）は解けても複雑な現実の問題は解けないことが明らかになった結果，ブームは急速に冷めた[6].

1980 年代に入ると，家庭用コンピュータの普及や多くのデータを蓄積するデータベースの開発に伴い，第 2 次 AI ブームが発生した．この頃には規則を記述し，より複雑な状況に対応できるルールベースの AI（レベル 2）が使われた．データベースに大量の専門知識を取り込み，その分野の専門家のように振る舞う AI をエキスパートシステム [2]と言う[6]．エキスパートシステムはある程度成功したものの，専門家が持つ知識の多くは経験的，暗黙的であるためそれらの知識獲得はとても困難であること，獲得した知識が非常に多くなると互いに矛盾していたり一貫していないものが出てきて知識を適切に維持管理するのが困難であること，またより広い範囲の知識を表現するのはとても困難である [3]ことが明らかになってくると，AI は再び冬の時代に入った．

---

2 代表的なものとして，MYCIN や DENDRAL がある[6].
3 この問題を解決するために「知識表現」の研究が進められ，オントロジー研究につながった．詳細は[3，6]参照.

ルールベースの AI の課題（プログラムに記述されていない例外や矛盾したルールには対応できない）に対し，第 2 次ブームの時点でコンピュータが自ら学ぶ機械学習も研究されていたが，当時はコンピュータの性能も含め大規模データを利用できる環境が整っておらず実用化は困難であった．しかし，2000 年代に入り，コンピュータ性能の向上とインターネット環境の整備により，後述する機械学習（レベル 3）や深層学習（レベル 4）が実現可能となり，現在の第 3 次 AI ブームが花開いた．

1.3.2. 機械学習

機械学習とは，AI のプログラム自身が学習する仕組みで[6]，パターン認識による分類や予測等多くの場面で利用されている．ここで「学習」とは「分ける」という処理を指し，機械学習は，既存データ（訓練データ）から特徴量（対象を認識する際に注目すべき特徴を量的に表したもの）及び用いるアルゴリズムを人間が決め，コンピュータが訓練データを処理しながら「分け方」を自動的に習得する[3]．訓練データが多いほど望ましい（正しく分類できる，予測精度が上がる）学習結果が得られる．それは一方で，実用的な精度をもつ機械学習を構築するためには大量の訓練データが必要になることを意味する．

機械学習はアルゴリズムに統計学に用いられる数学モデルや手法を利用することにより，あらゆる状況を明示・設定しなければならなかったルールベースから脱却し，入力に対して柔軟な対応が可能となった．一方，機械学習の精度を上げるためには何を特徴量とするかが肝であるのに，それは（少なくともその選択肢は）人間が決めないといけないという問題（「特徴量設計」問題）がある[3]．これに対して，コンピュータが既存データから重要な特徴量を生成する手法ができつつある[3]．次項では，その方法の一つである深層学習を紹介する．

1.3.3. 深層学習

コンピュータ自身が訓練データから特徴量を生成し，複数のアルゴリズムを連続的に利用した機械学習を深層学習という．深層学習は機械学習の手法の 1 つであるニューラルネットワークをより多くの層に拡張して取り入れた AI であり（詳細は 3.1.3 項を参照），出力結果の精度が他の機械学習手法より高い（分類であれば正解率等の評価指標が良い，予測であれば通常の統計モデルよりも誤差の少ないモデルを作成する）ことが知られている[8, 9]．

深層学習が従来の機械学習と大きく異なる点として，1 層ずつ階層ごとに学習していく点，及び自己符号化器（オートエンコーダー）[4]という「情報圧縮器」を用いることが挙げられる[3].

深層学習には大量のデータと高性能なコンピュータの演算能力が必要となることから，これまで利用は限定的なものであった．しかし，最近ではデータの蓄積やインフラ整備，コンピュータの処理能力の向上[5]に伴い，個人でも深層学習が利用できるようになった．一方，入力データは層を通るごとにコンピュータより情報を要約されるため，複数の層を用いる深層学習では最終的に得られたモデルにおける特徴量の意味を解釈することが難しく，判断根拠が分からないという問題（ブラックボックス問題）が指摘されている[10].この問題に関する解決策として可視化などの技術も開発されている[11，12]．その他，人工知能分野における難問として，フレーム問題[6]やシンボルグラウンディング問題[7]が挙げられる[3，6].

1.4. 本報告書での「AI」とは？

これまで見てきたとおり，一言に「人工知能」「AI」と言っても，一般的な認識だけでなく専門家が提案する定義も複数存在するのが実情である．その背景として，そもそも「知性」や「知能」自体の定義がないことから，人工的な知能を定義することもまた困難である事情が指摘されている [2]．また，「AI」という言葉が使われる対象の技術や手法が時代により大きく変わってきているため，AI とは何かということをわざわざ定義するよりも，年代ごとに「これが AI を代表する存在です」と言っていくしかない，という意見もある[13]．一方，AI とは何かについて一つの意味を提示することは，今後社内やプロジェクトチームで AI について議論する際に共通理解につながることが期待される．

そこで，AI 研究のほとんどが特化型人工知能の立場で行われていること，また現在特化型人工知能を AI と表現されることが多いことから，本報告書では特化型人工知能（弱い AI）に焦点を当て，情報科学の意味を含め，AI を「データから機械的にルールを見つけ出して情報処理をするもの，またはその技術」と捉えることとする．

---

4 入力層，隠れ層，出力層からなり，入力層と出力層が同じニューラルネットワーク．この学習により，隠れ層には入力の情報が圧縮されたものが反映される．詳細は[3, 6]参照.

5 とくに GPU（Graphics Processing Unit）の登場により，深層学習が大きく発展した[6].

6 「今しようとしていることに関係のあることがらだけを選び出すことが、実は非常に難しい」ことを指す．詳細は[3，6]参照.

7 記号（文字列，言葉）をそれが意味するものといかにして結びつくかという問題．コンピュータは記号の「意味」が分かっていないので，記号をその意味するものと結び付けることができない．詳細は[3，6]参照.

2. AIにおけるデータ

AIはデータから抽出した特徴から目的にあったアルゴリズムをコンピュータに作らせるため，データへの依存が大きくなる．AIが性能を正しく発揮するためには，それなりのボリュームの，それも質の良いデータが必要である．そのため，目的を正確に理解した上で，それに適したデータを選択することが重要である．

本章では，AIに利用するデータの種類及びデータを準備する際の留意点を紹介する．

2.1. 目的に適したデータ

AIでは，新たに一からデータを収集するよりも，既存のデータから目的に適したデータを選択し利用するのが一般的である．目的に適したデータを選択するためには，そのデータの特性を把握することが重要である．把握する方法は様々だが，一つの例として，5W2Hに沿ってデータを把握する方法がある．臨床試験データにおける5W2Hの例を**表 2-1**に示す[10]．

表 2-1 臨床試験データにおける5W2Hの例

5W2H	内容
Who	対象となった被験者の背景情報（疾患，選択基準，性別，年齢等）
What	収集されているデータ項目及びその内容
When	収集された時期
Where	地域及び施設数等
Why	臨床試験の目的
How many How much	症例数，来院頻度，収集期間等

[10]を臨床試験データの場合に合わせて改変

また，データを選択する際に考慮すべきポイントを**表 2-2**に示す．目的に適したデータがない場合は，既存のデータを加工，統合して，目的に適したデータを作成する（前処理．詳細は**2.3 章**を参照）ことや，新たにデータを作成・収集することを検討すべきである．適切なデータ量が分からない場合や大量のデータの購入が必要な場合は，まずはある程度のデータ量で一度学習・評価を行い，その結果をみながらデータを追加していく方法を検討すべきである．

表 2-2 データ選択のポイント

項目	内容
量	十分なデータ量であるか（訓練データと検証データに分ける場合は，それも考慮する）
属性	数値，カテゴリ，テキスト，画像，音声等のデータ形式
期間	データの収集された期間が，目的に対し十分であるか
入手先	自社，商用，オープンデータ等 入手に必要な費用，利用の制限やデータの信頼性を確認する
精度	データの正確性・信頼性・網羅性等が，目的に対し十分であるか
法令・契約	著作権法，不正競争防止法，個人情報保護法，電気通信事業法，越境，個別の契約等の利用条件

2.2. データの種類

最近では様々な形式のデータを AI で取り扱っているが，それらは大きくは「構造化データ」と「非構造化データ」の 2 つに分類することができる [14]．**表 2-3** に具体例を含めて整理した．

表 2-3 構造化データ / 非構造化データ

形式	内容	例
構造化データ	表（テーブル）形式のデータ	RDB（Relational database），Excel, CSV 等
非構造化データ	表（テーブル）形式で表せないデータ	文章（テキストデータ），音声，画像，動画，センサーデータ等

非構造化データを AI で取り扱う際，そのまま分析する方法（例．画像データ）と，一度構造化データに変換してから分析する方法（例．テキストデータ）の 2 種類がある．

また，データの種類を識別する上で「目的変数の有無」も重要な要素となる．目的変数の有無により，学習方法やアルゴリズムの選択が大きく異なる．

データの分類（構造化/非構造化）と目的変数の有無を正確に把握し，これらに適したアルゴリズムを選択することが，AI の活用において重要である（詳細は 3 章を参照）．

2.3. データの前処理

データのフォーマットを合わせるだけでなく，精度の向上や予測結果が不安定になるのを防ぐため，AI にデータを学習させる前にデータを加工する．この処理を，学習よりも前に行う処理という意味で「前処理」と呼ぶ[10]．データの前処理には，目的変数や説明変数と言ったデ

ータの「まとまり」を加工するだけでなく，異常値やデータ数と言ったデータの性質を加工することも含まれる．

2.3.1. 目的変数の加工

目的変数を分析に合わせた粒度[8]で集計する．グループ化とカテゴリ化がよく用いられる：

- グループ化

  複数の目的変数を統合する処理をグループ化と呼ぶ．例えば商品の需要予測をする際，商品別の売上金額が目的変数になるが，商品点数が非常に多いときは，商品タイプ等でグループ化して合算した売上金額を目的変数にすることがある．

- カテゴリ化

  連続変数を「1」「0」等のカテゴリデータに変換する処理をカテゴリ化と呼ぶ．目的変数が数値で推定が目的の場合，通常は回帰を用いて学習するが，回帰は目的変数中の非常に大きな値に左右されやすいため，例えば規定値以上は「1」,規定値未満は「0」とカテゴリ化したデータを用いて推定した方がよいことがある．

2.3.2. 説明変数の加工

データ量・種類やアルゴリズムによるが，説明変数を無加工のまま学習させると適当な精度が得られないことがある．その際は，説明変数を加工する必要がある（「説明変数の二次加工」「特徴量作成」「属性作成」等と呼ばれる）．深層学習等，説明変数の加工を学習処理の中で自動的に行う学習も増えてきたが，多くのケースでは人力で適切な説明変数を作成する必要がある．代表的な説明変数の加工方法を以下に示す：

- 自己回帰変数の作成
  目的変数が時系列データの場合，目的変数の過去の値を説明変数として追加することがある．これを自己回帰変数と呼ぶ．自己回帰変数は，次に説明する平滑化等を行う他，差分をとる等の処理（例：月の売上値が累計値しかない場合の先々月と先月の差）をすることが多い．

- 平滑化
  説明変数が時系列データのとき，時点のばらつきが大きく経時プロットがギザギザしていると，AI がノイズを過度に学習してしまうことで挙動が安定しないことがある．その場合は，ノイズをあらかじめ除去（平滑化）することが必要である．平滑化の方法は多数あるが，代表的な手法として移動平均法，ローパスフィルタ，ハイパスフィルタがある．

8 データをまとめる単位.

移動平均法は，前後の n 点の平均値を用いて値を置き換える方法である．移動平均法を用いると，**図 2-1** のように，経時プロットのギザギザを消すことができる．「前後」にすると，最新の測定時点に近い時点の平均を算出する際に未来のデータを含んでしまうため，「前 n 点の平均値」にすることもある．

- カテゴリデータのグループ化
  説明変数に選択肢の非常に多いカテゴリデータがあるときは，グループ化する（例：都道府県を地方にグループ化する）ことが多い．あるカテゴリに該当する訓練データが 1 件しかない場合，その 1 件を過度に評価してしまう状態（過学習．詳細は 3 章を参照）を引き起こしやすくなるが，グループ化により回避することができる．
- 複数の説明変数の合成・次元圧縮
  多くの変数を用いると，変数間の相関による多重共線性の問題が生じやすくなり，過学習が生じやすくなるなど，不都合が生じやすい．これを避けるために，説明変数同士の演算により合成変数を作成することがある．説明変数の数は次元数とも呼ばれ，変数を統合することで次元数が減ることから，この作業を「次元圧縮」と呼ぶ．最もよく行われる次元圧縮の方法が主成分分析である．複数の説明変数を基に主成分分析を実行し，主成分を説明変数に選択する．特にこの手法は主成分間が独立であることから多重共線性の問題の回避に有用である．
  深層学習等のアルゴリズムでは，このような合成変数作成や次元圧縮を内部で実行するため，処理が不要になることが多い（深層学習の利点の一つ）．一方で，線形回帰分析やロジスティック回帰分析等のアルゴリズムの場合，精度を上げるためにこの処理が必要である．

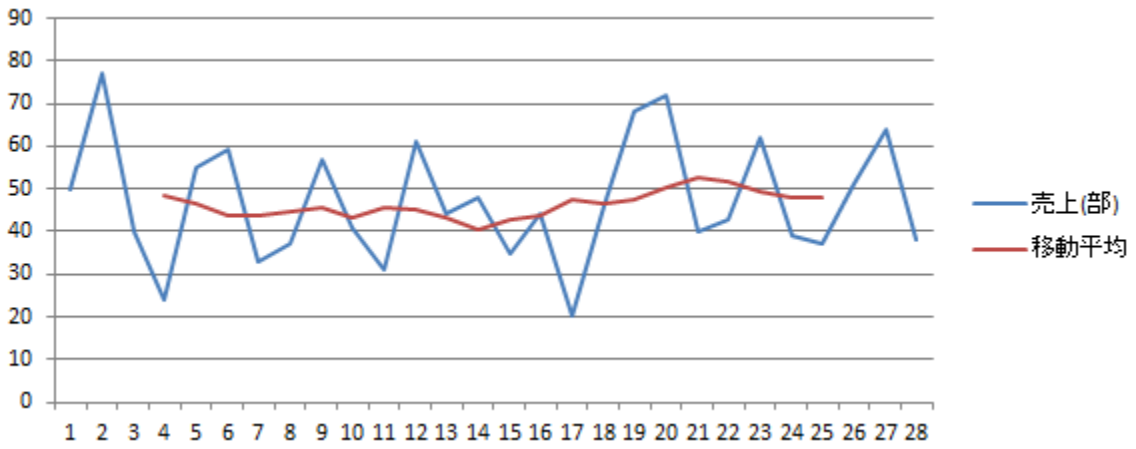


図 2-1 平滑化：移動平均のイメージ

2.3.3. 異常値の処理

目的変数，説明変数のどちらかで異常値が多いとき，うまく学習されないことがある．そのため，学習前に異常値を削除あるいは編集する必要がある．数値やカテゴリデータにおける主な異常値のタイプを以下に示す：

- 欠損（例：値が空欄である，または null のような欠損を示す値が入っている）
- 現実的にあり得ない値（例：年齢として 200 を超える値がある）
- 異常に大きい，または小さい数値（例：臨床検査値の異常値）
- あり得ないカテゴリデータ（例：都道府県のカテゴリに都道府県以外のデータがある）

これらを検出するために，異常判定をするルールを作る必要がある．判断ルールを検討する際，一度データを目視確認して，どんな異常値が存在するのか及びその量を確認することが有用である．

異常値を検出したら，異常値を処理する．異常値の処理方法は，主に以下の二つである：

- 異常値を含んだレコードを削除する
- 異常値を，別の正常なデータに置換する

異常値の処理方法は，異常値の割合や値の置換のしやすさから判断する．例えば，時系列の繰り返し測定中の稀な異常値は平均と標準偏差による推定など置換しやすい．逆に，連続して数十ものデータが異常値といったケースでは置換は困難である．異常値の置換方法は，主に以下の三つである：

- 前後の値の平均値等，推定値で置換する
- 他の説明変数の値から推定するルールを作り，そのルールに基づいて置換する
- 「その他」「異常」等，他とは区別可能な値で置換する

2.3.4. データ数の加工

例えば，目的変数（Yes/No のカテゴリデータ）10000 データのうち正解のラベルは 9970 が Yes，30 が No のように，ラベルごとのデータ数に著しく偏りがある場合，このデータをそのまま学習させたモデルでは，どのような入力データでも Yes という結果を出してしまう．これは No のデータが極端に少ないため，「予測精度を高くするなら全て Yes」と考えてしまうためである．

この様なケースでは「リサンプリング」という処理を行う．訓練データをそのまま使うのではなく，少数ラベルのデータ数に合わせて多数ラベルのデータから一定の比率でランダムに抽出する（アンダーサンプリング），あるいは多数ラベルのデータを基に一定の比率で少数ラベ

ルを複製しながら抽出する（オーバーサンプリング）ことで，訓練データのラベル間のバランスを取る．

しかしながら，これらの単純なリサンプリング手法には問題がある．オーバーサンプリングでは少数ラベルのデータが重複して学習に使用されるため，モデルの過学習につながる可能性があり， 同様に，多数ラベルのデータを アンダーサンプリング すると，2 つのラベル間で重要な違いをもたらすデータが学習から除外されてしまう可能性がある．オーバーサンプリングの問題を考慮しながら不均衡データのラベル間のバランスを取る方法として SMOTE（Synthetic Minority Over-sampling Technique）[15]がある．SMOTE は少数ラベルのデータとその k-最近傍点との間の点をランダムに選ぶことで少数ラベルのデータを人工的に作成し，ラベル間のバランスを取る．

2.3.5. 画像データの加工

これまではデータの種類に依らず検討すべき加工について述べてきた．ここからは，画像データ特有の懸念事項・処理方法について述べる．

2.3.5.1. 異常データの処理

画像データにおける主な異常データのタイプを以下に示す：

- サイズが異常に大きい，または小さい
- 解像度が異常に低い
- 意味をなさない画像（例：写真データなのに，全面が黒一色等）

画像データも数値・カテゴリデータと同様，まずは目視で確認し，そのあとに異常データの判断ルールを作って異常データを抽出する．しかし，画像データは数値やカテゴリデータと異なり置換が難しいため，抽出した異常データは削除することが多い．また，深層学習は，画像データが非常に多いと異常な画像が多少あっても問題なく学習できることも多いため [9]，数値やカテゴリデータほど異常データに対して敏感にならなくてもよいことがある．

9画像データは数値やカテゴリデータに比べて多様であるため異常データをノイズとして無視されやすいこと，また画像を取り扱う場合大量のデータに対して深層学習を利用することが多いことから，数値やカテゴリデータほど異常データに対して敏感にならなくてもよいことがある．

2.3.5.2. データの水増し

少ないデータで学習する方法として，元の訓練データに変換を加えてデータ量を増やす処理を行う．画像データでよく使われるデータの増幅方法の一つにデータオーグメンテーションがある．データオーグメンテーションは一つの画像から異なる画像を生成する方法である．主な処理方法を以下に示す：

- 反転・回転
- 色の変更（明度・彩度を変更する）
- 拡大・縮小
- 別の画像との合成（例：様々な背景画像を合成する）
- 部分画像抽出（画像の中の一部だけを取り出す）

上記の処理を行うことで，元のデータ数の 100 倍以上のデータを作成できる．深層学習の精度はデータ数に依存するため，学習する前にデータオーグメンテーションを用いることが多い．

2.3.6. 自然言語データの加工

続いて，自然言語データ（人が日常的使っている英語や日本語等の言語）特有の処理方法について述べる．

テキスト中には括弧，句読点，あるいは JavaScript のコードや HTML タグ等のノイズが含まれていることがあるため，これらを削除する必要がある．機械はアルファベットの大文字小文字（A と a）を別の文字と判定するため，これらを統一する必要がある．また，単語抽出の際，文章中には頻出するがそれ自身は意味を持たない単語（is，です等の助詞・助動詞や the 等の冠詞）を除去する必要がある．除去の方法には，辞書を用いる方法や高頻度に出現する単語を「意味を持たない単語」と見なして除去する方法等がある．

文字数が非常に少ない（例：文書ではなく単語レベルの記載），あるいは誤字・脱字といった問題もある．これらは判定が難しいため，完全に削除・修正しきれないと考えた方がよく，異常データが含まれることを前提に評価を行う．異常データがあっても，頻度が少なく，かつ，訓練データ量が多ければ，学習はうまくいく．

また，日本語入力特有の処理として，以下のものがある：

- 機械は全角半角（ｱとア）を別の文字と判定するため，これらを統一する必要がある．
- 日本語は英語のように単語間に空白区切りがないため，名詞や動詞などの品詞に分類しながら単語を切り出す処理（形態素解析と呼ばれる）が必要になる．

- 日本語では同じ意味の単語もひらがな・カタカナ・漢字・英語で表記することができ（りんご・リンゴ・林檎・apple），テキスト内に混合していることがあるため，辞書を用いて 1 つの代表的な表現に置き換える必要がある．
- 形態素解析には名詞（固有名詞，専門用語）などの辞書が用いられる．診療録など医療専門家が作成した文章を扱う場合は医学専門用語（疾患名，薬剤名，略号などを含む）などを用いて精度を高める工夫がされる．

2.3.7. アノテーション

アノテーションはデータに意味付けを与えるプロセスであり，データへの正解のラベル付けによる教師データの作成，画像のマーキング，あるいは特徴量の付与など様々な方法がある．アノテーションの例としては以下のものがある．

- 画像中の対象物を枠で囲む，人の顔であれば目鼻口角などをマークする
- 画像に写っている対象物の種類，数などのラベルを付ける
- 花の画像に対して種類名のラベルを付ける
- SNS の文章にポジティブ/ネガティブ/ニュートラルなどのラベルを付ける

このように，データに対して具体的に意味付けをすることにより，データから読み取るべき情報を後工程となるモデルの学習に明確に伝えられる．ただし，アノテーションに間違いが多いと，モデルの学習や評価を含めた，後のプロセスに致命的な影響を与えてしまう可能性もあるため，機械学習においてアノテーションは重要なプロセスである．

2.4. 訓練データ量について

深層学習を含む機械学習において必要な訓練データ量（サンプル数）を予め見積もる明確な基準はない。対象となるデータの種類やアノテーションを含む品質，用いる手法・モデルによって変わり，結果となる分類数あるいは特徴量（説明変数）が多いといった複雑な問題ほど多くの訓練データが必要となる．データ量の目安となる経験則の一つに「モデルのパラメータ数の 10 倍のデータ数が必要」という「バーニーおじさんのルール」がある[6]．これは経験則であり，説明変数の合成・次元圧縮（2.3.2 項参照）により用いる特徴量を減らす，あるいはオリジナルデータを加工してデータ量を増やす（2.3.4 項参照）などの対処方法もあるため，必ず満たさなければならないあるいは十分な条件ではない。しかしながらシグモイド関数を用いた単層ニューラルネットワークと等価である多変量ロジスティック回帰分析ではシミュレーションなどにより説明変数の 10 倍程度のデータ量が必要と言われている[16,17,18]ことからも 1 つの目安になるであろう．ただし，ここでのデータ量は訓練データ全体ではなく分類であれば最小

のグループのデータ数を意味している[10]。したがって例えば二値分類で共に 5 割ずつの場合でも必要な訓練データ量はその 2 倍となる。さらにパラメータ数は項目数（例えば「病名」「薬剤」であれば 2 つ）ではなく，その中に含まれる層の数（病名数，薬剤数）であることおよび多層のニューラルネットワーク（深層学習含む）では，パラメータ数（およそ各層のユニット数[11]を掛け合わせた数）であることに注意が必要である．

手元にある以上のデータを用意することや，多量のアノテーション作業には多くのコストがかかるため，十分と見積もられたデータ量を事前に用意するのは現実的ではないかもしれない．そのため，例えば，まずは POC（Proof of Concept 概念実証）[12]あるいは開発初期には用意しやすいデータ[13]を用いて分類を一部に限定（分類しやすい場合に絞って種類を減らす）した学習を試し，その結果に基づき必要なデータ量を再度検討する，また，それを用意することが困難な場合は目的とするモデルの精度を含め戦略を見直すことは合理的であろう．

10 Peduzzi P（1996）[15]においても変数あたりのイベント数（events per variable ：EPV）として定義されている．

11 3.1.3 参照

12 経済産業省のガイドライン[17]では AI 技術を利用したソフトウェアの開発をアセスメント，PoC，開発，追加学習の 4 段階に分けている．

13 手元にあり，アノテーションとして利用できる情報があらかじめ含まれているデータ

3. 機械学習の手法

本章では，現在の AI の具体的な手法である機械学習及び深層学習について，広く知られている分類別に紹介する.また，用いる手法を選択する際の留意点及び学習結果の評価方法を紹介する．

3.1. 機械学習の分類

本項では，機械学習及び深層学習の分類, 及び各分類で使用される代表的な手法を紹介する．

機械学習及び深層学習は，目的に応じて**図 3-1** のように分類されることが多い．それぞれの簡単な説明と特徴を以下で紹介する．詳しい解説はインターネットや書籍等豊富にあるので参照されたい（例えば[3，6]）．

図 3-1 機械学習及び深層学習の分類

3.1.1. 教師あり学習

教師あり学習は，入力データとあらかじめ人が与えた分類等の正解がセットになったデータを用いる学習方法で，例えば，犬と猫の区別といった画像認識に利用される．既知のデータの入力と出力の関係性（モデル）を得る学習フェーズ，入力データからモデルを通じて未知の出力を予測する予測フェーズの 2 つのフェーズからなる[20]．機械学習では，モデル獲得のために用いるアルゴリズムを人間が事前に選択する．教師あり学習の代表的なアルゴリズムに，回帰及び分類がある．

3.1.1.1. 回帰

回帰は，売上予測や人口予測等，連続値を予測する際に用いられるアルゴリズムの 1 つである．

最も単純な手法として線形回帰が挙げられる．線形回帰は，目的変数と説明変数（予測因子）の関係を線形関数で表現したものである．例えば，ある施設の来場者数を予測する場合，ある施設の来場者数が目的変数，曜日，天気，施設内でのイベントの有無等が説明変数にな

る．教師あり学習では，過去のデータ（訓練データ）から得られる目的変数及び説明変数を用いて，来場者数の予測モデルを構築する．そして，得られた予測モデルの説明変数に興味ある値を入力することで，その条件の下での来場者数を予測することができる．

線形回帰で説明変数と目的変数の関係を表現することが難しい場合は，2 次曲線や 3 次曲線等の非線形関数を用いた多項式回帰を用いることで表現が可能になる（**図 3-2**）．しかし，次数を大きくしすぎると，訓練データのもつノイズまで考慮した，訓練データの結果に過度に適合したモデルを構築してしまうことがある（過適合，過剰適合，過学習）[21]．このようなモデルは，訓練データと異なるデータでは推定精度が低くなってしまう（モデルの汎化性を失う）[21]．過学習を防ぐため，損失関数[14]に正則化項を追加する Lasso 回帰，Ridge 回帰[21]を用いることが多い．

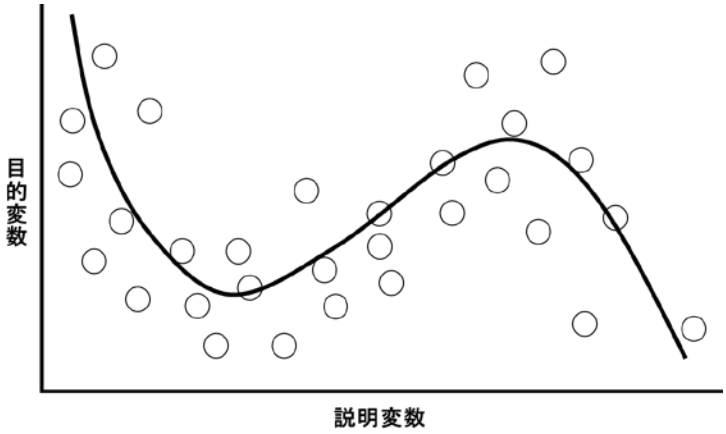


図 3-2 多項式回帰

3.1.1.2. 分類

分類は，迷惑メールの判別や画像診断等，カテゴリを予測する際に用いられるアルゴリズムの一つである．

分類の代表的な手法の 1 つに分類木が挙げられる．分類木は，分岐処理[15]をツリー状に形成したもので，トップから再帰的に対象データを分岐させて，当該データが属するカテゴリ（クラス）を決定する手法である[22]．分類木の特徴の一つとして，学習したモデル構造が比較的単純な形で可視化されるため，解釈しやすい点がある（**図 3-3** 左側）．一方，データを条件分岐で分けていくという性質上，木が深くなる（分岐数が増える）と学習に使えるデータ数が少なくなるため過学習になりやすい傾向がある．これについては，剪定して木の深さを制限する

14 予測結果と正解データとの誤差を評価する関数
15 ジニ係数などの不純度と呼ばれる基準を用いて，できるだけ同じクラスがまとまるように条件分岐する。

ことで過学習をある程度防ぐことが可能である．

分類木を応用した手法の一つにランダムフォレストがある（**図 3-3** 右側）．ランダムフォレストは，元の訓練データから複数のサンプルデータセットをランダム抽出し，各データセットを用いて作成された複数の分類木それぞれで予測された結果の多数決によりクラスを判定する手法である．予測性能は分類木より高く，ハイパーパラメータ[16]が少ないためチューニング[17]も比較的容易にできる．

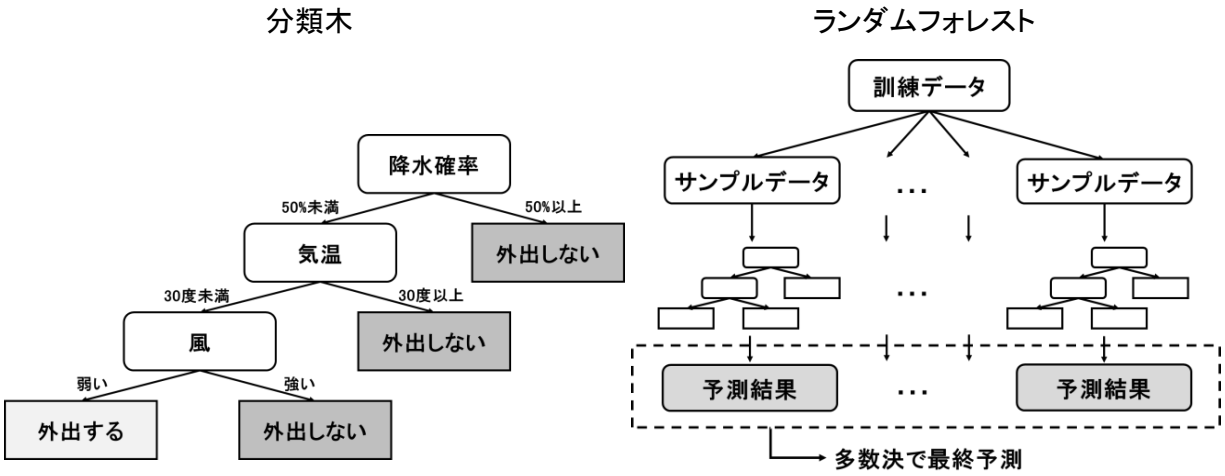


図 3-3 分類木とランダムフォレスト

その他，非線形の 2 値分類の標準的なアルゴリズムとしてサポートベクターマシン（support vector machine ；SVM ）およびニューラルネットワークが挙げられる（ニューラルネットワークは **3.1.3 節**で説明する）．ニューラルネットワークはシグモイド関数などを用いるが，SVM は sign 関数（sign(u) = -1 (u≤0); 1(u>0)）を用いるニューロンモデルにより非線形分離可能な空間を高次元の線形分離可能な境界（超平面）にマッピングすることで，**図 3-4** に示すような線形分離不可能な場合においても分類することが可能となる．

16 モデルを作成するために人があらかじめ決めておく値。（決定木の階層数など）
17 機械学習のアルゴリズムを調整するパラメータを試行錯誤しながら変えてみて，より良い結果ができるパラメータを探索すること[19]

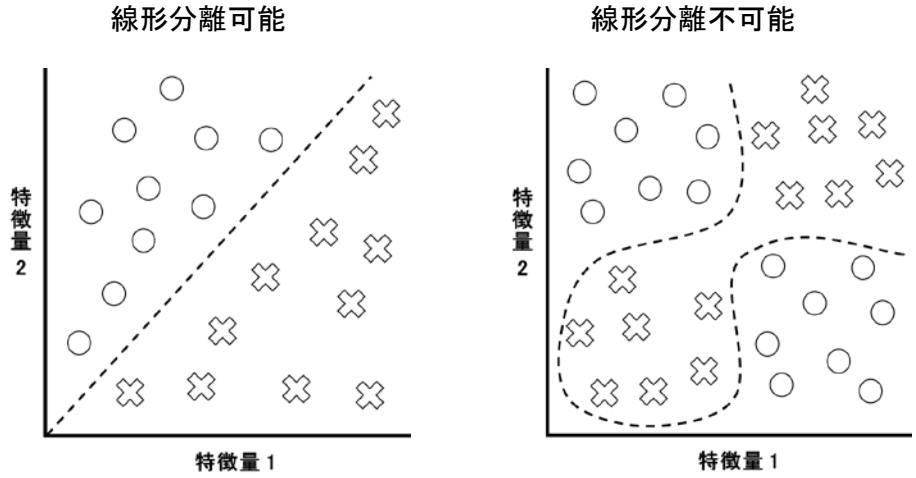


図 3-4 線形分離可能，線形分離不可能の例

3.1.2. 教師なし学習

教師なし学習では教師あり学習における「正解」に当たる「出力すべきもの」が決まっておらず，コンピュータがデータの中から傾向を自ら見出す学習方法である．まだ見出されていない問題の解決策や，答えが判明していない分類（クラスタリング）等を行いたいときに利用される．教師なし学習の代表的なアルゴリズムに，クラスタリング及び次元削減がある．

3.1.2.1. クラスタリング

クラスタリングは，顧客の嗜好によるセグメンテーション等，データを何かしらの基準でグルーピングする際に用いられるアルゴリズムの 1 つである．

最も似ている組み合わせから順番にまとめていく方法（階層あり）と，事前にいくつかのクラスタに分けるかを決め，決めた数のクラスタに分割する方法（階層なし）がある．階層なしのクラスタリングの代表的な手法の 1 つに k 平均法（k-means）がある．k 平均法は，**図 3-5** に示すようにクラスタの重心を用いて，事前に定めた k 個（**図 3-5** の例では 3 個）のクラスタに分割する方法である[20]．k 平均法は単純なアルゴリズムであり，広く用いられている．

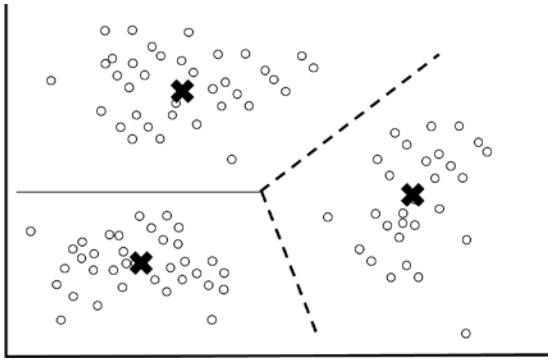


図 3-5 k 平均法(k-means)の例

3.1.2.2. 次元削減

次元削減は，意味のありそうな代表的な変数を選ぶ（特徴選択）場合や，変数間の相関による多重共線性の問題を避けるために相関のある変数をまとめたり，可視化や計算量削減のために高次元のデータを低次元に変換する（特徴抽出）場合に用いられるアルゴリズムの 1 つである[20, 23]．**図 3-7** で「応答がある？」の質問の前に矢印が戻っているのを見てもわかるように，次元削減したデータを他のアルゴリズムに利用することも可能である．

次元削減の代表的な手法の一つに主成分分析がある．最近は，主成分分析より分かりやすく関係性が可視化できる t-SNE[tsne]も人気である [20]．

3.1.3. 深層学習

1.3 節で示した通り，深層学習は機械学習の手法の 1 つであるニューラルネットワークを，より多くの層に拡張して取り入れた AI である．本節では，深層学習の基礎となるニューラルネットワークの概略を説明する．

ニューラルネットワークは脳の振る舞いを模倣した神経回路網モデルを，数式を利用して表したネットワークモデルのことである[7]．ニューラルネットワークの起源はパーセプトロンというアルゴリズムである．ニューラルネットワークを説明するために，まずはパーセプトロンについて説明する．

パーセプトロンは**図** 3-6 に示すように，複数の特徴量（入力）を受け取り，1 つの出力（0 か 1 の二値）を行うアルゴリズムである．具体的には，要因となる入力値（x_1，x_2）に重み（w_1，w_2）を付けて合算したものにバイアス（b）を加え，活性化関数[18]で処理した結果（$y = h(w_1x_1 + w_1x_1 + b)$）を出力する．活性化関数はステップ関数であり，0（$y \leq 0$）もしくは 1（$y>0$）が出力される．パラメータは重みとバイアスであり，これらを適切に調整することで適切なモデルを作成する．

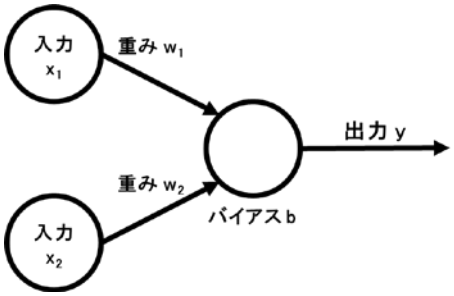
ニューラルネットワークは**図** 3-6 に示すように，パーセプトロンを階層型に構築したモデルで活性化関数にシグモイド関数や ReLU 関数などを使用したネットワークである．一番左の列（入口）を入力層，一番右の列（出口）を出力層，中間の列を中間層もしくは隠れ層と呼ぶ[23]．一つ一つの丸はユニットと呼ばれ，複数の入力を受け取り処理したのち，値を出力するものである．中間層を増やして，計 4 層以上に階層を深くしたものを深層学習と呼ぶ[7]．入力層は入力データを受け取り，出力層は中間層からの出力を元に最終的な予測を出力する[22]．出力層では，入力データがどのクラスに属するか確率で表される（例えば画像判定の場合，入力し

18 出力値を非線形変換する関数

た画像が「犬」である確率は 0.65,「猫」である確率は 0.26,…という結果が得られる）．訓練データを用いて学習されたニューラルネットワークの出力結果と正解データの誤差を損失関数を用いて比較し，それが小さくなるように重みや閾値をコンピュータが自動的に調整する．

パーセプトロンは線形分離可能なデータは分離できるが，線形分離不可能なデータは分離できない（**図 3-4, 図 3-6**）．それに対し，ニューラルネットワークは線形分離不可能なデータも分離が可能である．

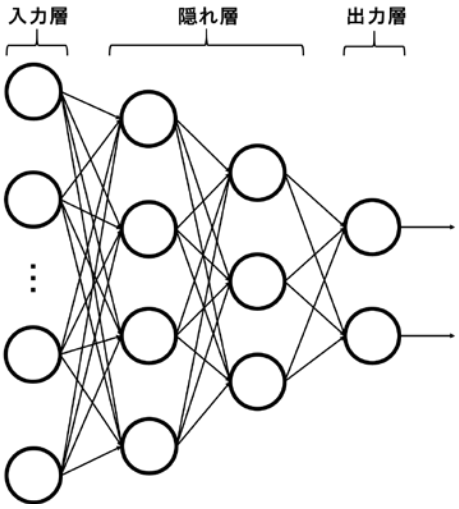
パーセプトロン



活性化関数

ステップ関数：$h(x) = \begin{cases} 0\ (x \leq 0) \\ 1\ (x > 0) \end{cases}$

ニューラルネットワーク



主な活性化関数

シグモイド関数：$\mathrm{h}(x) = 1/(1 + exp(-x))$

ReLU 関数：$\mathrm{h}(x) = \begin{cases} 0\ (x \leq 0) \\ x\ (x > 0) \end{cases}$

図 3-6 パーセプトロンとニューラルネットワーク

3.1.4. 強化学習

強化学習は教師データのある学習ではないが，ニューラルネットワークの出力（行動）の良し悪しに対して報酬（スコア）を与える先生があり，全体的な期待点数が最大になるよう途中の過程を最適化する（報酬の最大化）ように学習する手法である．例えば囲碁の AlphaGo の場合，「ゲームに勝つ」という戦略目的に向かって何かしらの行動をとり，その結果（勝敗）をもとに最終的に勝つ行動を学習させていく．[20]．すべての打つ手のパターンを網羅した訓練データを準備することは現実的には不可能である[24]．コンピュータは取るべきアクションの

選択肢を事前に教えられるのではなく，どのようなアクションが最大の報酬（例：ゲームのスコア）を生み出すかを発見するために，さまざまなシナリオを試行する．

3.1.5. 半教師あり学習

先に挙げた 3 つの学習方法の他，「半教師あり学習」がある．教師あり学習は高い精度を得られる一方，すべての訓練データに期待する結果を示したラベルを付与する必要があり，膨大なデータを準備するためには多くの時間と労力を要する．これを解決するために半教師あり学習が考え出された．様々な手法が提案されており，例えば，①少数の入力と正解がセットになったデータセットに教師あり学習を用いてモデルを構築し，②入力のみの正解がないデータセットに①で構築したモデルを用いてラベルを付し，③②でラベルが付されたデータセットも用いて再度教師あり学習を用いてそのモデルを再学習する方法等がある．

3.2. どのアルゴリズムを使えば良いのか？

深層学習を含む機械学習では，データをアルゴリズムで処理することでコンピュータに学習させるため，アルゴリズムを適切に選択することが精度良い機械学習を行う上で肝になる．一方，機械学習のアルゴリズムは非常に多くの種類があり，この分野について経験豊富な人でない限り「どのアルゴリズムを使えば良いのか？」という疑問を抱くことが想定される．この疑問への答えは，どのようなデータが利用できるのか（規模，品質，性質），目的は何か（データを使って何がしたいのか），どれくらい計算時間がかかるのか等によって異なる．多くの Web サイトでアルゴリズムの選択に関するチートシートが提案されている[25，26，27]，本報告書では，3.1 節で示した「教師あり学習」，「教師なし学習」に対応する形で整理されている，SAS 社のサイトで紹介されている機械学習（深層学習を含まない）アルゴリズム選択チートシート（**図 3-7**）[25]を紹介する．

このチートシートを用いると，「次元削減が必要？」，「応答（「正解」と同意義）がある？」，「数値を予測？」の 3 つの質問に対して順に Yes/No を選択することで，大きく「教師あり学習：回帰」，「教師あり学習：分類」，「教師なし学習：クラスタリング」，「教師なし学習：次元削減」の 4 つのカテゴリに分類できる．さらに続けて各分類内の質問（楕円形）に回答していくと，推奨アルゴリズム（長方形）に到達できる[19]．このシートは，幅広い機械学習アルゴリズムの中から特定の課題に最適なアルゴリズムを見つけ出すために役立つ

19 「教師あり学習：回帰」のディシジョンツリーは回帰木，「教師あり学習：分類」のそれは分類木と呼ばれる

が，あくまでも基本的な推奨アルゴリズムに到達することを意図しており，到達した推奨アルゴリズムが必ずしも最適なアルゴリズムでない場合もあることに注意が必要である[25].

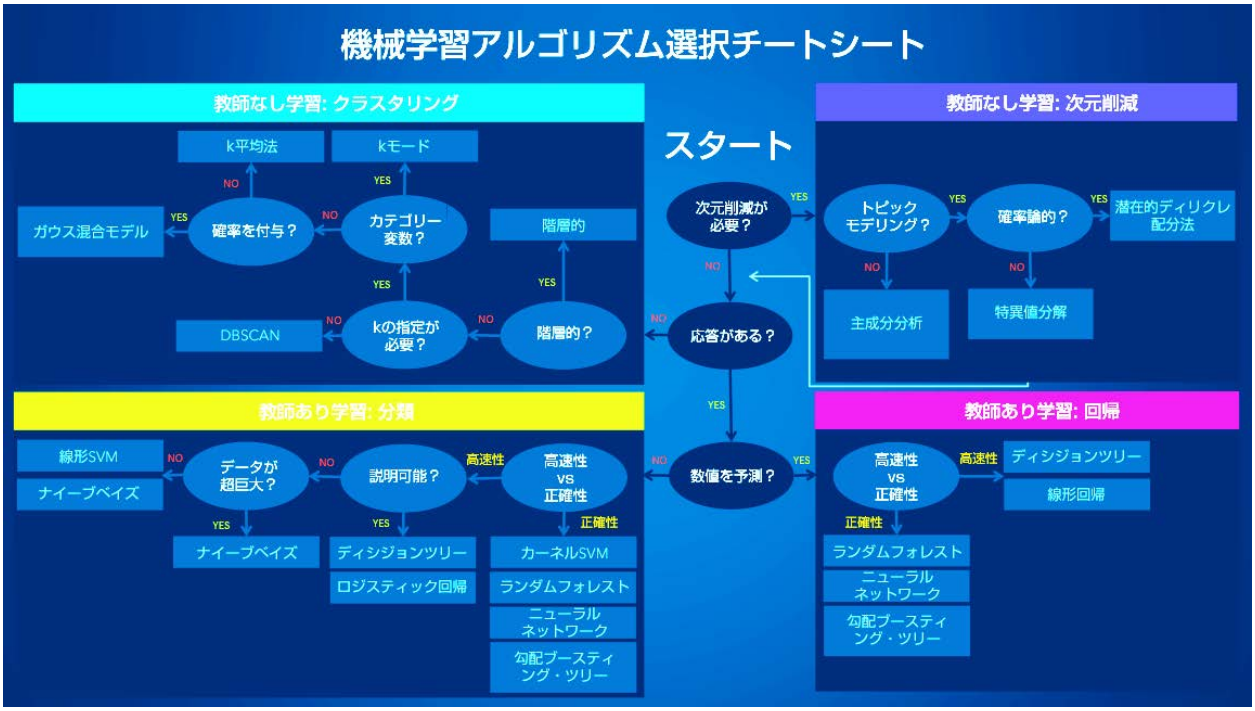


図 3-7 機械学習アルゴリズム選択チートシート [20]

3.3. 学習結果の評価

前項で紹介したチートシートを利用して，目的やデータを基にアルゴリズムを選択したとしても，それが本当に実用に耐えられるものかどうか，性能評価する必要がある．また，複数のアルゴリズムが使える場合，それらの性能評価を基に，実際に利用するアルゴリズムを選択するかもしれない．このような意味で，機械学習の性能評価は非常に重要である．

機械学習の性能評価は，大きく分けて学習時間，判定時間及び判定精度の 3 つがある[7]．学習時間は訓練データからモデルを作るまでの時間，判定時間はモデルを使って新たなデータから判定結果を得るまでの時間で，早いほど処理速度に関する性能が高いことを意味する．本項では 3 つ目の判定精度について詳細に紹介する．

機械学習を利用することで，新しい入力データから未知の出力結果（分類，予測等）を得ることができる．しかし，その出力が正しいかを真に判断することはできない．そこで，通常は，事前に手元にある全データを①学習フェーズに用いる訓練データ，②モデルの評価及びパ

20 SAS Institute Japan 株式会社 に掲載許諾をいただいている．

ラメータのチューニングのために用いる検証データ，及び③モデルの予測結果を評価するテストデータに分割し，モデルの性能を評価する．ここでは，4 章で利用している評価指標（正解率，適合率，再現率，F 値）および陰性的中率，特異度について説明する[28].

4 章では，胸水（Effusion）の有無という 2 値データを対象にしており，このとき既存データのラベル（正解を意味する）と AI による判定結果（分類結果）の関係は**表 3-1** のような 2×2 表に整理できる．このとき，各評価指標は以下の式で表すことができる：

表 3-1 既存データのラベルと AI による判定結果の関係

AI による判定結果 / 既存データのラベル	胸水あり	胸水なし
胸水あり	a 人	b 人
胸水なし	c 人	d 人

- 正解率（正確度）：すべてのデータのうち，AI による判定結果と既存データのラベルが一致（ともに「胸水あり」又は「胸水なし」）した割合.

$$\text{正解率} = \frac{a+d}{a+b+c+d}$$

- 適合率（陽性的中率，精度）：AI により「胸水あり」と判定されたデータのうち，既存データのラベルも「胸水あり」であったデータの割合

$$\text{適合率} = \frac{a}{a+c}$$

- 再現率（感度）：既存データのラベルが「胸水あり」であったデータのうち，AI により「胸水あり」と判定されたデータの割合

$$\text{再現率} = \frac{a}{a+b}$$

- 陰性的中率：陽性的中率と対になる指標．AI により「胸水なし」と判定されたデータのうち，既存データのラベルも「胸水なし」であったデータの割合

$$\text{適合率} = \frac{d}{b+d}$$

- 特異度：感度と対になる指標．既存データのラベルが「胸水なし」であったデータのうち，AI により「胸水なし」と判定されたデータの割合

$$\text{特異度} = \frac{d}{c+d}$$

AI の判定が緩くなり「胸水あり」と判定されやすくなると，AI により正しく「胸水あり」と判定されたデータ数（a）が増えるため再現率は高くなるが，同時に誤って「胸水あり」と判定されるデータ数（c）も増えるため適合率は低くなる．逆に，AI の判定がどんどん厳しくなり「胸水あり」と判定されにくくなると，c が小さくなるため適合率は高くなるが，同時に誤って「胸水なし」と判定されるデータ数（b）が増えるため再現率は低くなる．つまり，適合率と再現率にはトレードオフの関係が成立する．これら二つの指標を総合的に評価する指標の一つに F 値がある．

- F 値：適合率及び再現率の調和平均．

$$\text{F 値} = \frac{2 \times \left(\text{適合率} \times \text{再現率}\right)}{\text{適合率} + \text{再現率}}$$

このように，推定精度に関する評価指標は様々あり，これらの指標を全て用いるのではなく，目的に合った指標を用いる必要がある．例えば健康診断等のスクリーニング的な検査の場合，再現率をより重視する．これは，後に精密な検査が控えているため，正確性より見逃すリスクの方が重要視されるためである．また，正確性に関する指標として正解率が最初に頭に浮かぶが，判定結果が陽性又は陰性に極端に偏る場合は学習とは関係なく正解率が増す（分子の a 又は d が増加する）ため，学習の評価として不適切といえる．

3.4. 推定精度が低い場合の要因の分析

推定精度が低い場合は，一歩踏み込んでモデルの確認を試してみる．確認といっても，詳細な数学的理論にアプローチを求める訳ではない．例えば画像分類の場合，どんな情報をどんな順序で利用した結果どのような出力が得られるかを確認できれば，画像分類の判断を誤る要因が推測できる．その推測に基づき，例えば，訓練データに不備（欠損や反転，関係のない画像の混入等）がないか確認・修正したり，誤り易い画像でもより特徴のあるものを追加することで，モデルを改善できる．ただし，扱うデータは大量・多変数なので値をひとつひとつ確かめるのは困難である．モデルやデータの確認には，データの可視化が有用である．以下は，深層学習により画像を分類する学習済モデルである VGG16[21] によるピザの画像の識別過程を，公開されている可視化ツール[22]を用いて確認した例である．

a）画像下側にピザがあり，上側には人物も写っている．深層学習により，この画像が何かか

21 VGG16 はオックスフォード大学の Visual Geometry Group が ImageNet と呼ばれる大規模画像データセットを学習して作成された有名な多層ニューラルネットで 1000 のクラスに画像を分類できる．[29]

22 可視化手法の一つである Grad-CAM[29]の著者が作成したプログラムを keras ように修正したものが MIT ライセンスで公開されている．https://github.com/vense/keras-grad-cam/

を判断する．

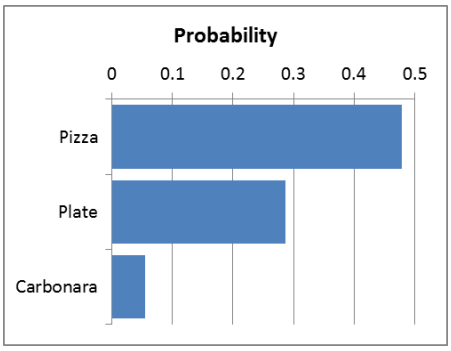
b）深層学習による判断結果を棒グラフで示したもの．上段が「ピザ」，中段は「皿」，下段は「カルボナーラ」である確率を示す（これ以降の確率の低い分類は省略）．グラフから，この画像が「ピザ」である確率は約 0.48，「皿」が 0.29 であることが分かる．

c）Gradient-weighted Class Activation Mapping（Grad-CAM）[29]による画像 a）のヒートマップ（Grad-CAM に関する詳細は Demonstration Links[23]を参照）．Grad-CAM は深層学習が判断に用いた特徴箇所を可視化する技術であり，注目度が高い（判断の寄与が大きい）箇所ほど赤色を呈す．

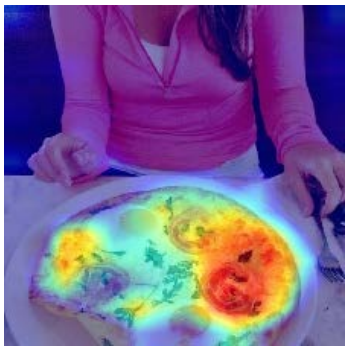
d）Grad-CAM では注目度の高い特徴箇所を特定できるが，その内容の詳細はわからない．そこで Grad-CAM を拡張した Guided Grad-CAM により特徴箇所を高解像度で出力した．

c）及び d）の出力から，まずはベーコン，次いで生地の形に注目していることがわかる．以上より，深層学習では，赤いベーコン，次いでピザの丸い形が「ピザ」と識別する判断に使われていることが推定された．

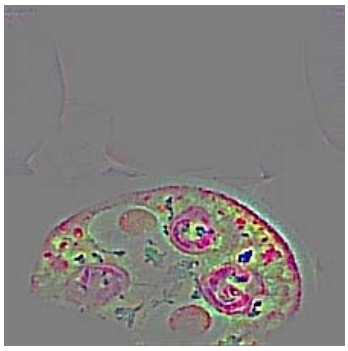
a）元画像[24] テーブル上のピザ



b）確率スコア棒グラフ



c）Grad-CAM 出力



d）Guided Grad-CAM 出力

図 3-8 AI の画像判断過程の可視化[29]

23 http://gradcam.cloudcv.org/

24 元画像出典：https://storage.googleapis.com/openimages/web/index.html
CC BY 2.0（https://creativecommons.org/licenses/by/2.0/legalcode）

3.5. 同じデータ，同じ目的で複数のアルゴリズムが使える場合，どうしたらよいか？

同じデータ，同じ目的（ここでは画像分類）で，ロジスティック回帰とニューラルネットワークを比べた報告がある[31]．多くの場合，深層学習により正確性が向上すると言われているが，中には手書き文字認識に深層学習を使うと逆に正確性が損なわれるという報告もある[32]．この矛盾は，訓練データの内容に起因する．深層学習を含む機械学習は大量のデータを必要とするが，これが少なかったり，データの質（画素数，欠損を含めたデータの異常値，教師あり学習を利用する場合ラベルの正確度等）が悪かったりすると，深層学習はその性能を正しく発揮できない．また，項目数があまり多くなく，変数間の関係が事前に特定できれば，それらを考慮した統計アルゴリズムを選択した方が早く適切なモデルを構築できると思われるが，項目数が豊富であれば共変量かは不明であっても深層学習を選択した方がより早く性能の良いモデルに到達できるだろう．現時点では具体的にどの手順が最適かは示せないが，複数のアルゴリズムが使える場合はデータの量や質を勘案してアルゴリズムを選択する，使用可能なアルゴリズムをそれぞれ適用した後に性能評価し，推定精度の高いものを選択する，あらかじめ採用するアルゴリズムの順序を決めておき，最初に推定精度などの基準を満たしたものを選択するなどの方法が考えられる．

4. 実装してみよう

本章では，AI の実装例として，2017 年に National Institutes of Health（NIH）が公開した X 線画像データ[33]に基づき特定の疾患の有無を診断するためにニューラルネットワークを利用した AI を作成する．このデータは誰でもアクセス可能であり，同一のデータを性能評価に用いることでモデルの性能比較を容易にし，コンピュータによる疾患の検出，診断の能力を高め，最終的には臨床医のより正確な診断につながることが期待されている．画像データの解析入門では，サンプルデータ MNIST（0 から 9 の手書き文字）がよく用いられるが，今回は製薬業界により関連がある本データを解析対象とした．

今回の実装には Python 及びその深層学習用ライブラリの Keras[34]を用いた．Python は AI の実装でしばしば見かけること，Keras は「ニューラルネットワークで迅速な実験を可能とすることに重点をおいて開発」されたライブラリであり，単純な 2 値分類タスクに対しては容易に実装できることからそれぞれを選択した．実装に当たっては，Keras の開発者である Francois Chollet の本[35]を参考にしている．Python 及び Keras に関するその他の詳しい情報は，インターネットや書籍等が豊富にあるので参照されたい．

本章を参考に自ら手を動かして実装することで，AI についての理解を深めていただきたい．

4.1. 本実装で用いる X 線画像

以下でデータが公表されている[33][25]（https://nihcc.app.box.com/v/ChestXray-NIHCC）．データの概要を以下に示す．

- データ数：112,120 レコード（30,805 人）の正貌 X 線画像
- 解像度：1024*1024 ピクセル
- ファイル形式：PNG
- 疾患ラベルの種類：各 X 線画像に対して，Atelectasis（肺拡張不全），Consolidation（浸潤影），Infiltration（浸潤），Pneumothorax（気胸），Edema（水腫），Emphysema（気腫），Fibrosis（線維症），Effusion（胸水），Pneumonia（肺炎），Pleural_thickening（胸膜肥厚），Cardiomegaly（心肥大），Nodule（結節），Mass（腫瘤），Hernia（ヘルニア）の疾患の有無が判定されている．一つの X 線画像に対して複数の疾患が「あり」と判定されているものもある．このようなデータは多ラベルデータと呼ばれる．全データに対して付与されている疾患ラベルの分布を**図 4-1** に示す（複数疾患が「あり」と判定された場合，

25 利用にあたっては https://nihcc.app.box.com/v/ChestXray-NIHCC の FAQ_CHESTXRAY.pdf に「The usage of the data set is unrestricted. But you should provide the link to our original download site, acknowledge the NIH Clinical Center and provide a citation to our CVPR 2017 paper.」と記載されている

それぞれ 1 件としてカウント）．No Finding は 14 疾患いずれにも該当しないデータを示す．分布を見ると分かる通り，少なくとも一つの疾患が「あり」と判定されているデータは，14 疾患いずれにも該当しないデータよりも少なく，不均衡なデータとなっている．この特徴はモデル作成時に考慮する必要がある．

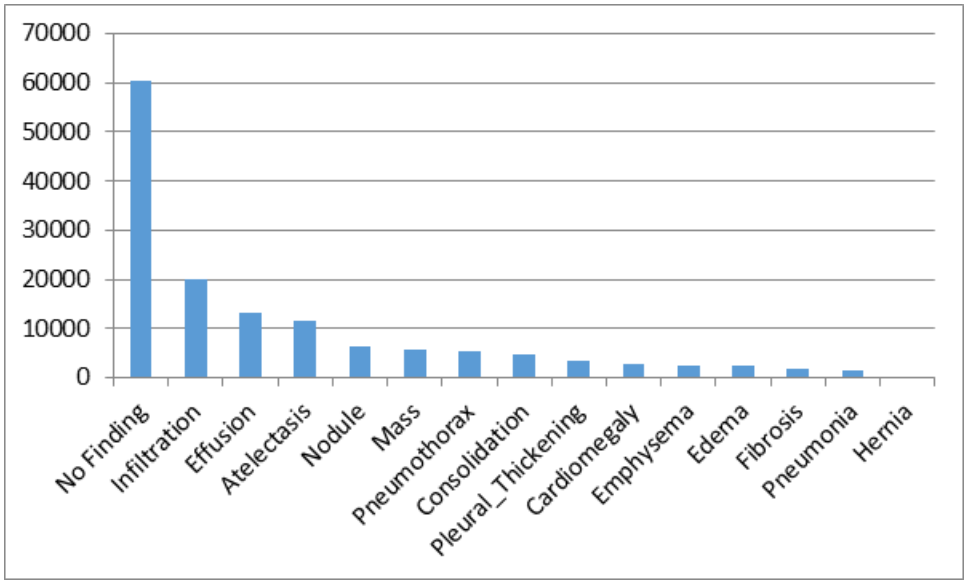


図 4-1 解析対象データにおける該当疾患の有無の分布
［縦軸：該当疾患を含む X 線画像数］

- 疾患ラベルの付け方：読影レポート（未公表）を自然言語処理することにより，各疾患の有無ラベルが付与された．他のデータベースを用いて，放射線科医のラベル付けと同様の自然言語処理によるラベルを比較した結果，適合率，再現率は共に 0.9 以上であった（適合率，再現率に関する詳細は 3.3 項を参照）．もし自然言語処理では系統的にラベルが付与できていないパターンがある場合，そのようなデータを利用して作成した AI は，適切にラベル付できていないパターンに引っ張られる形で系統的な間違いを起こすため注意が必要だが，ここでは AI を実装することに注目し，詳細な評価は行わない．

4.2. 課題設定

4.1 項で述べた通り，各 X 線画像は複数の疾患ラベルを持つ多ラベルデータであるが，ここでは胸水（Effusion）の有無の判定のみに着目する．

4.2.1. モデル評価の準備

X 線画像分類を行うモデルを学習・評価するために，既存データを訓練データ，検証データ，テストデータに分割する．検証データでハイパーパラメータ[26]を調整しながら訓練データを用いてモデルを学習させ，出来上がった最終モデルはテストデータを用いて性能評価を行う（性能評価に関する詳細は 3.3 項を参照）．性能評価用のテストデータをモデルの学習で利用する訓練データ及び検証データと区別することにより，まだ得られていない新しいデータに対する汎化性能の評価をより妥当なものとする．モデルを評価する際に，モデルを作成するために利用したデータを用いる，予測したい時点よりも未来のデータを用いるなど，本来，利用すべきでない情報を用いることは Leakage と呼ばれ過学習を引き起こす[36]．ハイパーパラメータの調整も含めて，学習に用いたデータは，モデルにその情報が含まれているため，汎化性能の評価に用いることはできない．

今回用いるデータにはテストデータのリストも X 線画像と一緒に提供されていたため，それを利用してデータを分割した．各患者の X 線画像はテストデータまたはそれ以外のいずれか一方にのみリストされている．つまり同一患者の X 線画像がテストデータとそれ以外の両方に含まれることはない．そのため，X 線画像データとそのラベルのみを用いている場合，Leakage は起きにくいと考えられる．訓練データ及び検証データを区別するリストは提供されていなかったため，テストデータを除いたデータをランダムに 8 対 2 に分割し，それぞれ訓練データと検証データとした．この分割には，各データが同一患者のものか否かは考慮しておらず，訓練データと検証データに同一患者のデータが含まれる可能性がある．本来はそのような状況にならないよう分割した方が適切であるが，ここではプログラムを簡単にするため対処しなかった．

4.2.2. 実装例

4.2.2.1. 実装環境

付録にプログラムを示す．プログラムは以下の環境（**表 4-1**）で実行できることを確認した．GPU を用いて全てのデータを利用して学習した場合，12 時間程度で実行が完了した．GPU が無くても実行は可能だが，その場合は 10 日程度かかると想定される．

---

26 モデルを作成するために人があらかじめ決めておく値。（例えばモデル更新時の更新の程度を調整する学習率や学習を繰り返す回数などがある）

なお，GPU 版の Tensorflow を利用するためには CUDA や cuDNN[27]が必要になるが，対応しているハードウェアやソフトウェアのバージョンが決まっているため注意が必要である．Tensorflow の公式 Web サイトでも，GPU 版の Tensorflow を利用する際は Docker[28]を用いるのが最も簡単であると述べられている[37]ので，環境構築に困った際には Docker の利用を検討すると良いだろう．

表 4-1 実行環境

OS	Ubuntu 16.04 LTS
Python	3.6.6
Tensorflow	1.10.0
Keras	2.2.4
CPU	Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz コア数：4
メモリ	32 GB
GPU	GeForce GTX1080Ti

4.2.2.2. ニューラルネットワークの設計

本報告書で用いたニューラルネットワークの設計を**表 4-2** にまとめた．本データを用いた解析が先行研究で実施されており[38]，その結果から，「実装をする前」からある程度の好ましい性能が得られることは分かっていた．そのため，期待していた結果が得られなかった場合，プログラムミス等を疑うことができた．しかし，通常，実装する前に好ましい学習済みモデルが得られるかは分からない．そのような状況において期待していた効果が得られない場合，プログラムミスかそれともデータの量・質に問題があるのか，といった点を見極める必要がある．またデータの量・質に問題があるのであれば，どの程度データを増やせばあるいは質を高めれば良いのか判断するのは非常に困難である．これは，臨床試験において必要被験者数を算出するために効果の大きさやばらつきを検討・判断するのとは異なる．

27 Tensorflow は機械学習を行うフレームワーク， CUDA，cuDNN は GPU を使用するミドルウェア．第 5 章参照
28 5.8 コンテナ 参照

表 4-2 ニューラルネットワークの設計

項目	設定	理由
ニューラルネットワークの構造	ImageNet で学習済みの DenseNet-121 [39]をパラメータの初期値として与えて，最後の分類を行う層を本タスク用に入れ替えたものを用いた．	他にも利用可能なモデルはあるが，モデルのパラメータ数がそれほど多くなかったことと，ChexNet [38]でも利用されていたため選択した．
入力画像	224×224 ピクセルのカラー画像として読み込んだ．	DenseNet 等，Keras で再利用可能なモデルの入力形式は全て縦画素数×横画素数×3 チャネルである．そのため，X 線画像はグレー画像だが，カラー画像（RGB）として読み込んだ．
データ拡張	ランダムに水平方向に画像を反転した．	訓練データを水増しするために様々な方法があるが，X 線画像は正面からの画像で，画像サイズもそろっているため，水平方向の反転のみ行った．
損失関数	2 値交差エントロピーを用いた．	2 値分類問題の一般的関数で，Keras でも実装されているため利用した．
評価関数	AUC	疾患の有無を決めるカットオフ値は状況に応じて適宜変更されることも考えられるため，そのような状況に柔軟に対応できる AUC を用いた．なお AUC は微分不可能なため，直接最適化を行うには微分可能な関数で近似する必要があるが [40]，ここでは簡単に損失関数と独立に評価関数を設定した．
エポック数 [29]	50 回	充分大きくした上で評価するのが適切と考えられる．50 回でもまだプラトーに達していないためより多くの学習が必要と考えられるが実行時間を考慮して 50 回とした．

4.2.2.3. X 線画像の判定結果

検証データに対して AUC が最大となるパラメータを学習済みモデルに用いることとした．**図 4-2** の各図は以下の指標（縦軸）とエポック数（横軸）の推移を示す．点は訓練データに対する値，実線は検証データに対する結果を示す．また，テストデータを用いて最終モデルを性能評価した結果を**表 4-3** に示す．まだ 50 エポックの時点でプラトーに達していないためより時間をかけて学習させる必要があるように見受けられる．一方で，F 値，適合率，再現率については 20 エポック程度から訓練データと検証データに対する結果が乖離し始めており過学習が始まっているようでもある．実際，検証データの AUC が最大になる 50 エポック時点の各検証

29全ての訓練データを学習に利用した回数．全てのデータを一度にメモリ上に載せることが出来ない場合など，バッチと呼ばれる単位にデータを分割してパラメータ更新することがある．例えば訓練データが 100 個あり，20 個ずつ 5 つのバッチに分割してニューラルネットワークを学習させる場合，5 つのバッチ全てを利用すると 1 エポックとなる．

データに対する結果とテストデータに対する結果も大きく乖離し過学習を示唆されるため対応を検討する必要がある.

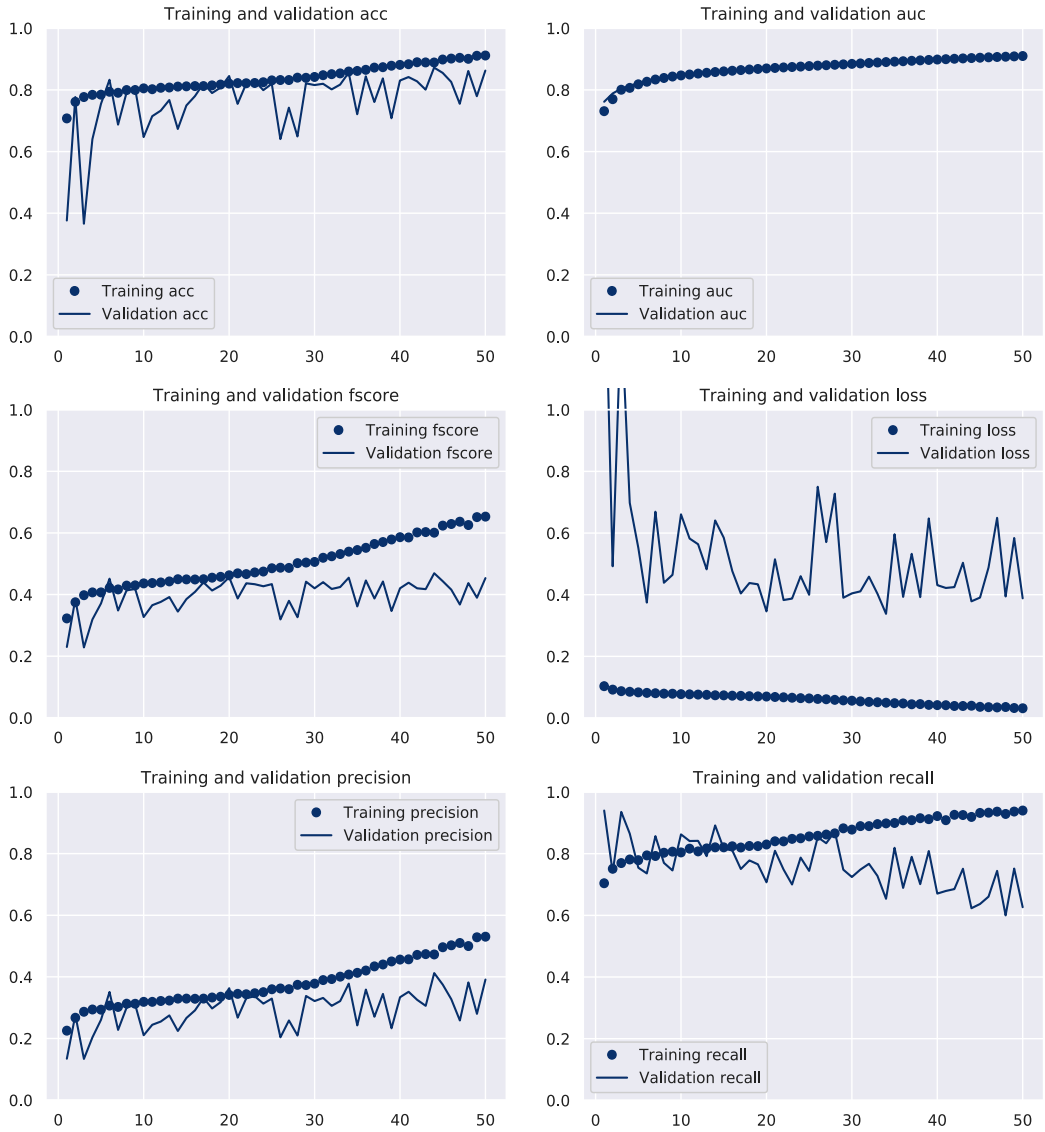


図 4-2 トレーニングデータ及び検証データに対する各評価関数

[縦軸:正解率(左上), AUC(右上), F 値(左中), 損失(右中), 適合率(左下), 再現率(右下);横軸:エポック数]

表 4-3 最終モデルの性能評価

正解率	AUC	F 値	損失	適合率	再現率
0.76	0.75	0.34	0.68	0.31	0.45

ここで学習済みモデルを用いて，胸水の判定を行ってみる．図 4-3 の左の画像は，テストデータかつ放射線科医が疾患の場所を特定している X 線画像のうち，今回利用した学習済みモデルで胸水である確率が最も高い X 線画像を選択した．下の画像は Grad-CAM [29]により作成したヒートマップを示す．学習済みモデルが X 線画像のどの部分に着目しているかを示しており，胸水ありの判定に影響が大きい箇所ほど明るく示されている．このヒートマップを元の X 線画像に重ねることにより右の画像が得られる．緑色の四角形は放射線科医が疾患ありと判定している場所を示している．**図 4-4** に胸水の確率が高く判定された 6 枚の X 線画像の評価結果を示す．おおむね疾患がある場所に応じて評価しているが 1 枚は全く異なる場所に反応している．このような情報はニューラルネットワークを評価する上で有用な情報になると考えられる．このように AI の判定根拠を人間にも理解しやすい形で提供する方法も研究されている．

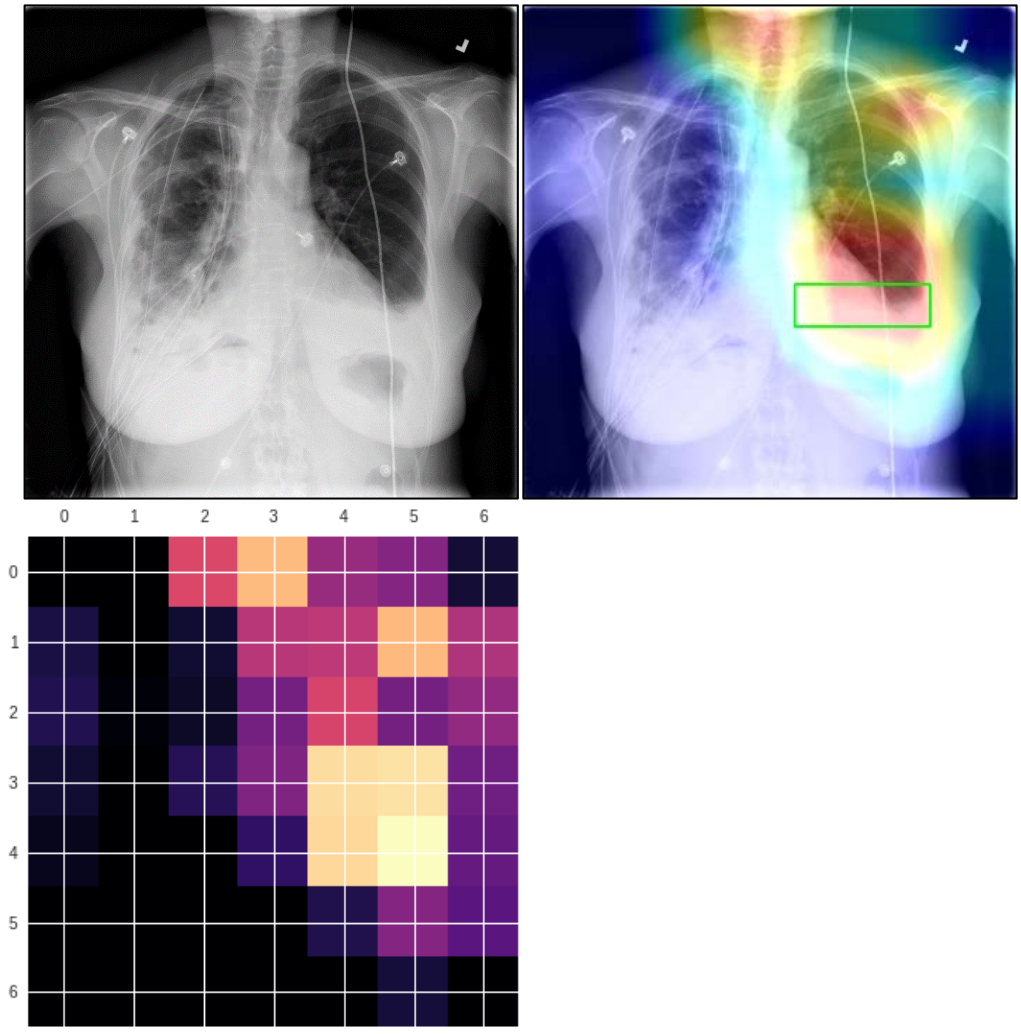


図 4-3 Grad-CAM による X 線画像評価

（左：X 線画像（胸水あり），右：重ね合わせ画像，下：ヒートマップ）

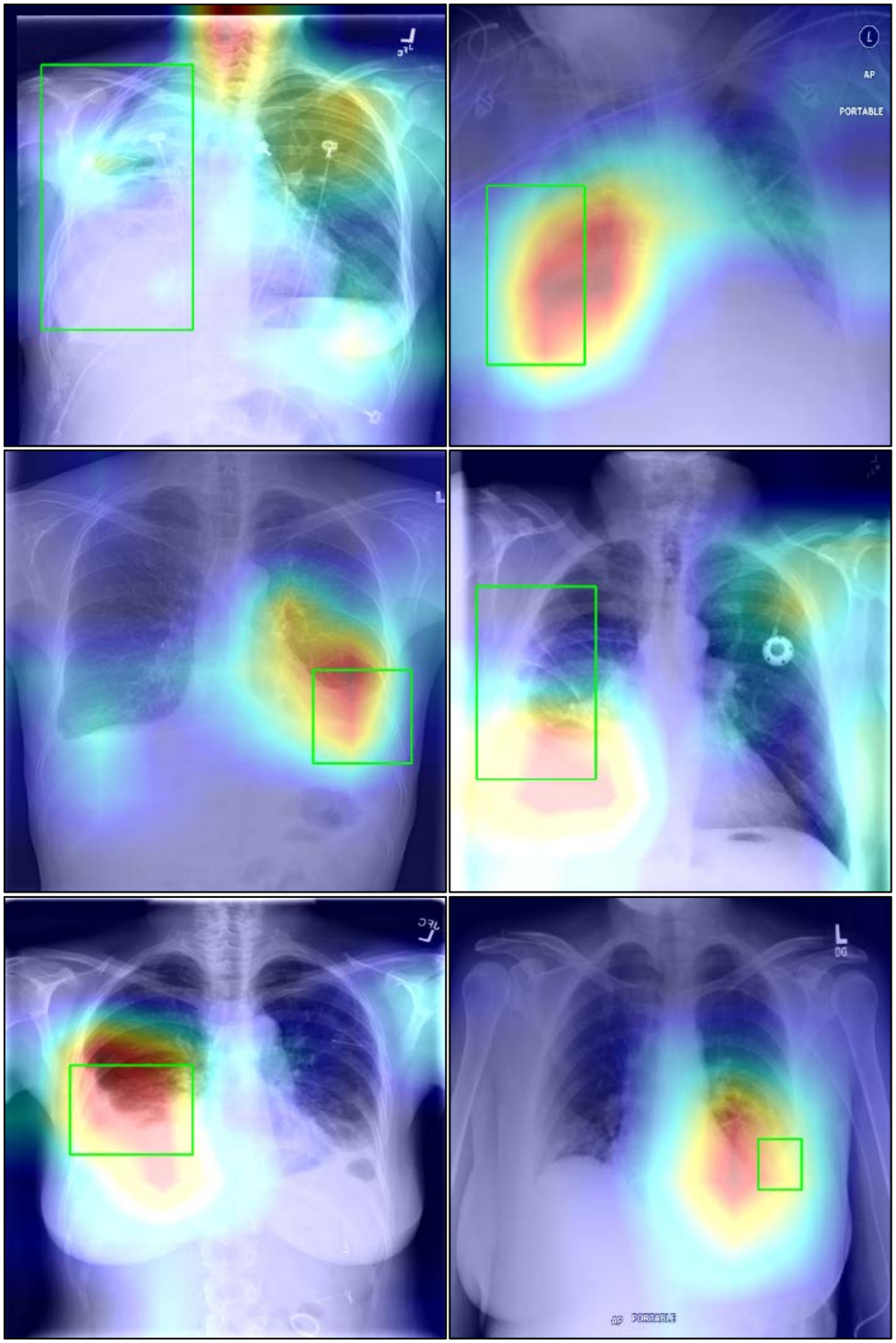


図 4-4 重ね合わせ画像

4.2.3. データ数が性能に与える影響

性能に影響を与える因子として，入力データ数，ニューラルネットワークの構造（隠れ層の数，ユニット数，ネットワークのつなぎ方など），学習方法（学習率，エポック数，バッチサイズ，最適化方法など）等が考えられるが，ここでは入力データの数について評価する．**図 4-5** に，学習・評価に用いる訓練データ及び検証データをそれぞれの全体に対する割合を 1（**図 4-2** と同じ），0.1，0.01，0.001 と変えた場合の各評価関数の学習経過を示す．変動が多く傾向の評価が困難であったため，各時点の値を指数移動平均で置き換えたものを**図 4-6** に示す．凡例は[各データの全体に対する割合：評価項目]の形式で表示した．本来は利用可能なデータ数に応じてハイパーパラメータを調整する必要があると考えられるが，今回は行っていないため，結果の解釈には注意が必要である．全般的にデータ数が少ないと過学習する傾向が見られる．

図 4-7 に訓練データ及び検証データをそれぞれの全体に対する割合を 1（表 4-3 と同じ），0.1，0.01，0.001 と変えた場合に，テストデータに対する各評価関数の値を示す．データの割合が 0.1，0.01，0.001 についてはそれぞれ 10 回ずつランダムに抽出し，それらの平均値及び標準偏差を図示した．

正解率はデータ数の増加に伴う改善があまり見られない上に標準偏差は大きくモデル評価として適切ではないように見受けられる．一方，適合率や再現率はデータ数の増加に伴い改善が見られる．胸水有りとラベル付けされたデータ数が少ないため，AI が全てのデータに対して「胸水無し」と判定するモデルを選択してしまうと，正解率は簡単に高くなってしまう．AI で「胸水有り」と判定されたデータに対する「胸水有り」とラベル付けされたデータの割合に当たる適合率及び「胸水有り」とラベル付けされたデータに対する AI が適切に「胸水有り」と判定したデータの割合に当たる再現率は，データの増加が適切に画像判定の性能改善に影響したかどうかを反映していると考えられる．ここでの検討内容はデータのカテゴリ分布などにより結果は異なってくる．学習済みモデルの性能評価の際には実際のデータに合わせた適切な評価が必要である．

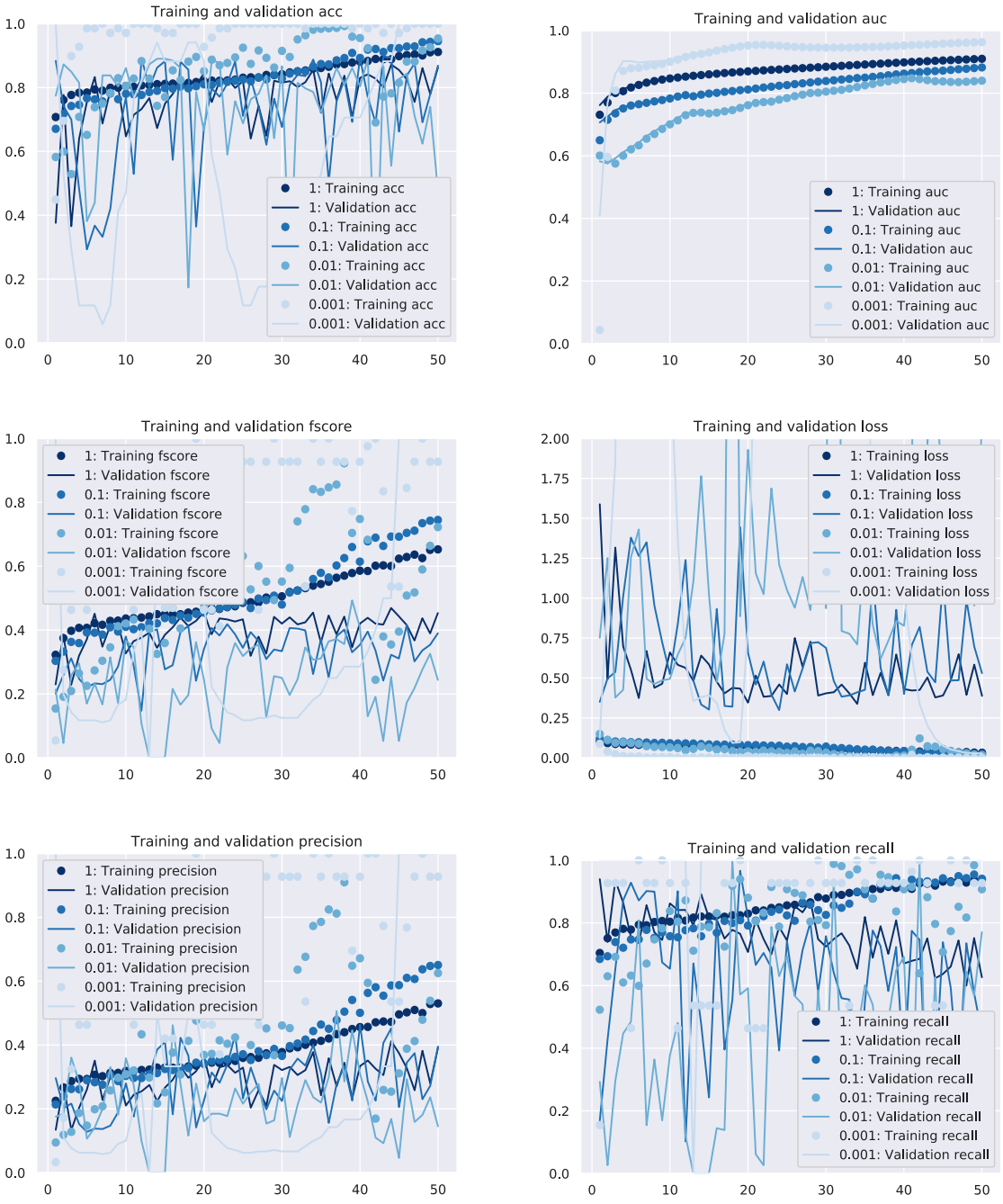


図 4-5 訓練データ及び検証データ数を変えた際の各評価関数の推移

[縦軸:正解率(左上), AUC(右上), F 値(左中), 損失(右中), 適合率(左下), 再現率(右下);横軸:エポック数]

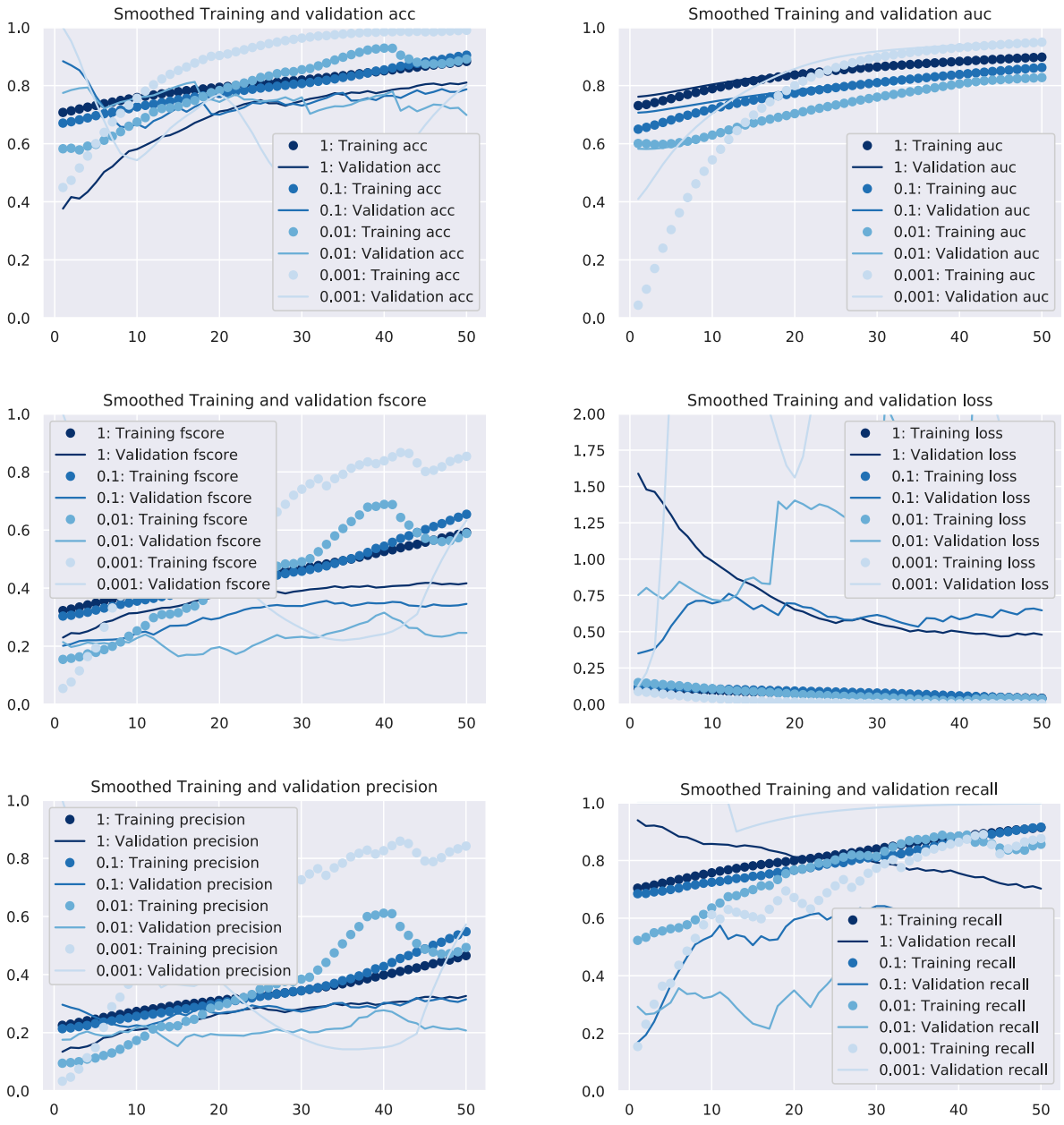


図 4-6 指数移動平均で平滑化した図 4-5 の各評価関数

[縦軸:正解率(左上), AUC(右上), F 値(左中), 損失(右中), 適合率(左下), 再現率(右下);横軸:エポック数]

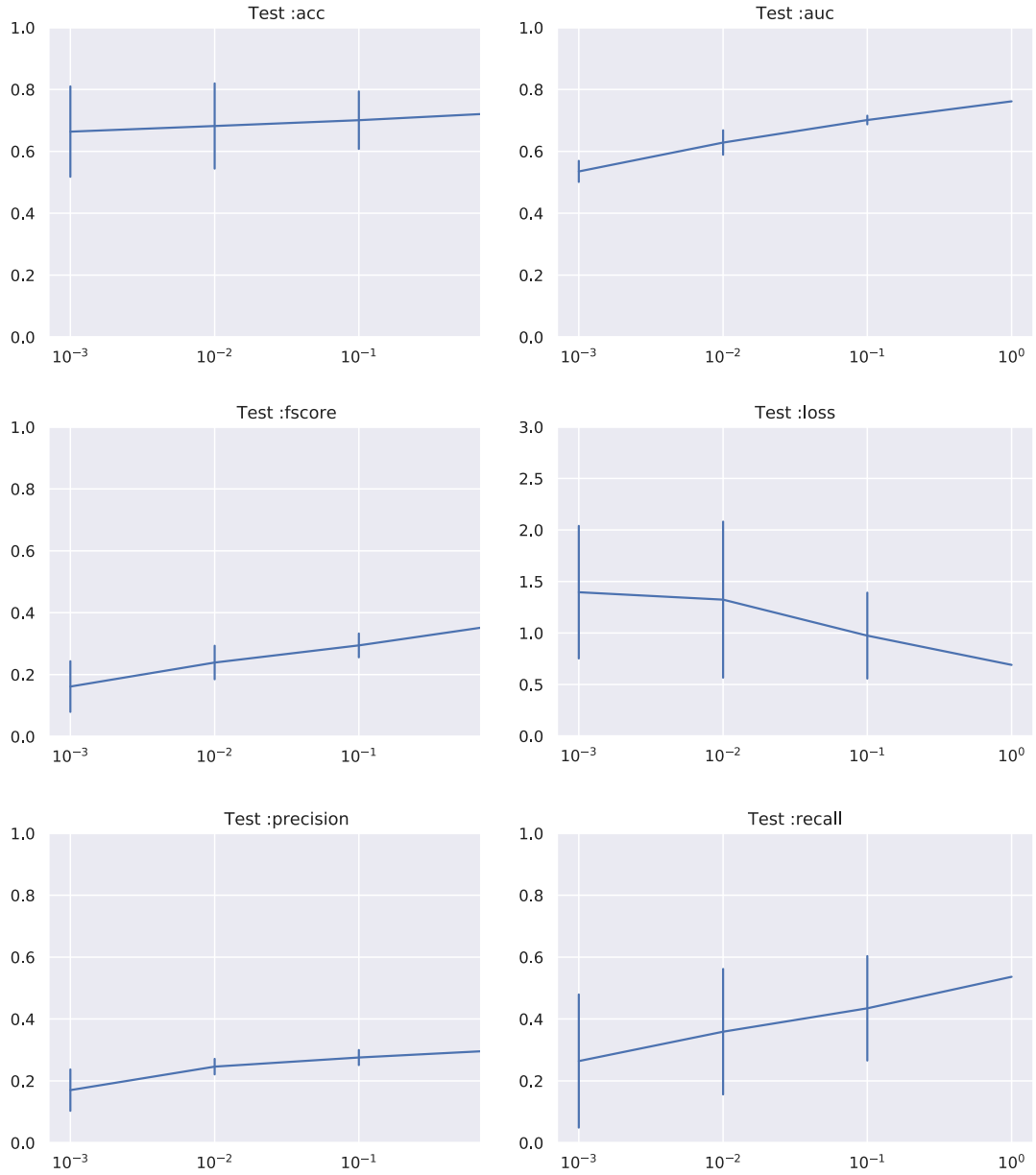


図 4-7 訓練データ及び検証データの使用割合とテストデータに対する各評価関数（平均±標準偏差）
[縦軸:正解率(左上), AUC(右上), F 値(左中), 損失(右中), 適合率(左下), 再現率(右下);横軸:データ割合]

4.3. まとめ

ここでは AI の実装例として，X 線画像データの分類タスクを行った．また，学習・評価に利用可能なデータ量と様々な評価関数の関係を示した．一方で，今回利用した深層学習は広い範囲で研究が活発に行われており検討できていない内容も含まれる．例えば学習率を学習の途中で増減させて学習を促進させる方法なども提案されている．また，X 線画像を自ら読むことが出きず，評価指標（正解率，再現率等）に対する目標値を設定しなかったため，モデル性能に関する妥当性は評価できなかった．

深層学習に必要なオープンソースソフトウェアのライブラリが充実していること，コンピュータ自身が訓練データから特徴量を生成できるという深層学習独特の特性（1.3.3 項）から，画像処理に関する専門知識を必要とせず AI を実装できた．一方，X 線画像に関する臨床的な知識があれば，入力データとして用いるパラメータを適切に事前に選択し，モデル性能を更に改善できたかもしれない．

AI 作成は手元にデータがあればすぐに始められる．更に公表データを利用することで，豊富な量のデータに基づき自作の AI の性能を高めることも可能になる．場合によっては手元の少数データだけで必要な性能は達成できるかもしれない．

技術が誰の手にも届くところにあるため，今後も AI を作成・利用する人がますます増加することが想定される．それに合わせて，結果の解釈やアルゴリズムの選択ミスを含めた誤用が増えるかもしれない．まだ AI は自ら考えることはできず，人が与えたデータに応じた答えを返すだけで，間違ったことを教えればその通りに間違える．実装の経験を通じ，改めてデータそのものに対する理解，大量のデータ収集，公表データの取り込み等，データ及びそのリテラシーに対する重要性を感じた．

5. AI の環境（ハードウェア，ソフトウェア）について

AI の環境は大きく開発（学習）環境と運用環境に分けられるが，ここでは開発（学習）環境について紹介する．環境には大きくクラウドとオンプレミス（クラウドではない物理的なワークステーションやサーバ）がある．クラウドはサービスごとにソフトウェアの自由度に制限があったり，ハードウエアは従量制で必要時に柔軟に変更できるため，ここでは基本的なオンプレミスの環境について紹介する．オンプレミスのハードウェアからプログラミングインターフェースまでの大まかな環境を**表 5-1** に整理した．特徴的なのはハードウェアに多量のメモリや演算装置として GPU を用いることである（GPU は主にディープニューラルネットワークを用いる場合）．ライブラリは深層学習を含む機械学習を行うプログラム（データ加工，数値計算，モデル，関数，視覚化等）を目的，種類ごとにパッケージにしたもので，Application Programming Interface(API)[30]を利用して開発環境やフレームワーク，他のパッケージから機能を呼び出して利用される．フレームワークも機械学習を行うプログラムを目的，種類ごとにパッケージにしたものであるが，ライブラリに比べて関数の種類や機能の範囲が広く，汎用的/基礎的な命令を担っている．フレームワークを用いず Python などの言語とライブラリだけで機械学習を行ったり，フレームワークから別のフレームワークをライブラリとして呼び出して用いられることもある（このため**表 5-1** の境目の線を破線とした）[31]．フレームワークにも画像処理が得意，計算機化学が得意，といった特徴がある．フレームワークとライブラリ及び言語の間には可能な組合せと不可能な組み合わせがあり，開発したい機械学習の種類や用いるモデルに合わせて適切なものをそれぞれ選択する必要がある．主なフレームワークは機械学習を行う基礎的な機能のみを提供しており，ラベル付け（アノテーション）や画像の解像度の統一といった前処理，学習過程の確認・評価，学習結果の分析といった機能は別のツールやライブラリを用いる必要がある．

30 アプリケーション（ソフトウェア）同士が互いに機能を利用したり操作するインターフェース（仕様）

31 表 6-1 ではエディタ，開発環境（IDE），言語，ツール，フレームワーク，ライブラリと区別を行っているが，明確に線引きがされるのではなく，例えば言語の R は単体で IDE 機能を有しており，エディタは実行・デバッグ機能と結果の表示機能を有しているものが多く，やはり開発環境 IDE とも言える．また，ツール，フレームワーク，ライブラリも扱い方によって相互に位置づけが変わり，フレームワークも別のフレームワークやライブラリから参照すればライブラリやツールとなる．本章では比較的よく目にする分類を用いて紹介する．

表 5-1 大まかなオンプレミスな深層学習を含む機械学習の環境

区分	例
エディタ	Emacs，Atom，Sublime Text，Visual Studio Code，Jupyter Notebook
開発環境（IDE）[32]	PyCharm，Neural Network Console，Rstudio，Eclipse，Python IDLE，（Anaconda）
言語	Python2.x，Python3.x，Julia，R，Java，Scala，Perl，Mathematica，MATLAB，C++，SAS
ツール	Mlflow，TensorFlow Extended，NNVM/TVM，Intel Nervana Graph，TensorRT，TensorFlow Fold，Tensor2Tensor，TensorBoard，Facets，NVIDIA DIGITS，Julius，HTK，MeCab，KNP，Labelbox，LabelMe，RectLabel，matplotlib，TensorSpace
フレームワーク	TensorFlow，Chainer，Caffe，MXNet，PyTorch，The Microsoft Cognitive Toolkit (CNTK)，SAS Viya
ライブラリ	Keras，Gluon，scikit-learn，Chainer Chemistry，caret，NumPy，Word2Vec
OS	64bit Windows，Linux（Ubntu）
並列コンピューティングプラットフォーム	CUDA[#1]，cuDNN（CUDA Deep Neural Network library），OpenCL[#2]，NVIDIA SLI（Scalable Link Interface），
ハードウェア	CPU（1-2 個[#3]），GPU（0-10 個[#3]），TPU（0-数個[#3]） メモリ（16GB-4TB DDR4）， ストレージ（256GB-数 TB SDD，1TB-数十 TBHD），これらに対応したマザーボード

#1：NVIDIA が提供している GPU による並列計算処理のための開発環境
#2：CPU と GPU が混在した環境で並列計算を行うフレームワーク
#3：物理的な演算装置数

5.1. 演算装置

通常のソフトウェア，統計解析やデータベース，WEB アプリケーションの多くは様々で複雑な命令を一つずつあるいは数個をセットで順に処理するため，コンピュータ内で情報を処理（演算）する中央演算処理装置（Central Processing Unit；CPU）はこういった処理に最適化するよう作られている．データベースや WEB サーバなど多くの同時アクセスの処理が必要なサーバでは，高性能（同時に 28 個の命令を処理できるなど）な CPU を複数個搭載している場合もある．ディープニューラルネットワーク（DNN）以外の機械学習は主に CPU を利用するため，高性能な CPU を用いることで処理時間を短縮でき，その結果，より多くのパターンを試みることができる．対して，DNN は個々の演算は比較的単純であるがとてつもなく多量の

32 Integrated Development Environment（統合開発環境）の略

行列計算を処理する必要がある．このため，高性能な CPU を用いても機械学習には長時間を要し，数日，数週間かかる場合もある．コンピューターグラフィックによる動画の表示・処理は画面の 100 万個を超えるドット[33]の色彩・輝度を 1 秒間に 20 回以上計算する必要があるため，ゲーム用のパーソナルコンピューターでは専用の演算装置である GPU が用いられてきた．GPU は比較的単純な数千の演算処理を同時に行うことができるため，GPU を多量の演算が必要な地震や気象のシミュレーションの演算装置として利用する General-purpose computing on graphics processing units（GPGPU）が行われるようになった．2018 年末時点で上位に位置付けられる CPU と GPU の計算能力は**表 5-2** のように DNN で必要な単精度以下の計算は大きな差がある．デスクトップ PC としては高性能の Core i9-7960X は 1 秒間に単精度の浮動小数点計算を 963 億回行えるが，ゲーム用 GPU の上位機種である GeForce RTX2080Ti は 14.2 兆回行えるにもかかわらず，価格はあまり変わらない．こういった背景から，深層学習を含む機械学習，特に高性能な CPU でも実行が困難（数日単位で時間を要する）な大きな行列計算を多く行う DNN[34]では GPU が必須となっている．更に高性能な DNN 環境では複数の GPU が用いられる．近年は DNN 向けに行列計算を効率よく処理する仕組みを持った GPU[35]が提供されている．更に，DNN 専用の演算装置として application specific IC（ASIC)も開発・利用されており，Google は行列演算のみ処理するテンソル演算装置（Tensor Processing Unit；TPU）を開発して使用している．

高性能な CPU は発熱が大きく冷却が重要であることはよく知られているが，GPU も発熱量が大きく，大きな冷却能力を要する．DNN の学習では数時間連続して計算することも少なくないため，ファンの増設や水冷機能を有する GPU のパッケージもある．

手元にあるノートパソコンを用いて手軽に DNN を用いた推論を行いたいという場合，USB で接続できる DNN 用 AI アクセラレーター「Movidius Neural Compute Stick2」（Intel 社）は，フレームワークを TensorFlow または Caffe に限定されるが DNN の演算能力は 1 秒間に 4 兆回[36]あるので，目的によっては有用である．

---

33 正確には座標ベクトルで定義されたポリゴン

34 DNN で行っている主な計算である行列の積和演算（重みの行列に別の行列をかけて足し合わせ）は三次元のグラフィックスでポリゴンを用いるときの行列演算と同じである．

35 1 世代前の GeForce GTX 1080Ti は行列演算用のコアがないが，GeForce RTX 2080 Ti は行列演算用のコアがある．

36 ビジネスで用いられる Intel Corei7 の CPU の浮動小数点演算は 1 秒間に 100 億から～400 億回程度である．

表 5-2 CPU，GPU の比較 [37]

	Core i9-7960X	Xeon Platinum 8180	GeForce RTX 2080 Ti	Tesla V100
種類	Intel の PC 用 CPU の上位機種	Intel CPU の最高峰	ゲーム用 GPU の上位機種	ディープラーニング用 GPU
コア数	16	28	4352	5120
スレッド数	32	56	-	-
倍精度(64bit)	0.0963TFLOPS	1.12 TFLOPS	-	6.38TFLOPS
単精度(32bit)	0.0963TFLOPS	2.24 TFLOPS	14.2TFLOPS	12.75TFLOPS
半精度(16bit)	0.0963TFLOPS	2.24 TFLOPS	28.5TFLOPS	25.5TFLOPS
消費電力	165W	205W	250W	250W
コスト	約 16 万円	約 120 万円	約 18 万円	約 140 万円

5.2. OS（operating system）

深層学習を含む機械学習には扱えるメモリ量が多く，多くの並列処理が可能な OS が必要となる．そのため 64bit 版の Linux，Windows，または Unix が用いられる．ワークステーション，サーバではなくパーソナルコンピューターであれば（ハードウェアを拡張できないが）MacOSX でも可能である．Linux は種々のパッケージ・ディストリビューションがあるが，Ubuntu が多く用いられる [38]．

5.3. フレームワーク/ライブラリ

主なフレームワークについて，対応する OS，言語，ライセンス，GPU 対応の有無などを**表 5-3** に整理した．また，深層学習を含む機械学習でよく用いられる言語である Python から使用でき，よく用いられるライブラリを**表 5-4** に示した．

表 5-4 の「Anaconda」は Python 本体と Jupyter Notebook，各種ライブラリ，パッケージマネージャーや各種ユーティリティをひとまとめにした Python ディストリビューションのひとつである．つまり，Anaconda をインストールすれば，Python を用いて機械学習を行う基本的な実行環境が構築できる．ライブラリは機械学習の開発を容易にしてくれるが，内部で別のライブラリ（NumPy など）やフレームワークを呼び出していることが多く依存関係があるた

37 FLOPS は 1 秒間に実行できる浮動小数点計算の回数（理論値）．T は単位 tera（1 兆）を表す．理論上の能力はコア数×クロック(コアの動作速度)×IPC(クロック当たりの命令実行数)により計算できる．

38 **表 5-3** 主なフレームワーク一覧の対応 OS 欄参照

め，バージョンの違いによってエラーが生じることがあり，管理に注意が必要である．Python に並び機械学習に多く用いられる R 言語には多数の機械学習アルゴリズムが利用可能な caret というパッケージ（R 言語では「ライブラリ」でなく「パッケージ」と呼ぶ）があり，それを用いることでデータセットの作成，前処理，特徴選択を行える．

フレームワークの TensorFlow は，書籍を含む情報量の多さなどからも，深層学習を含めた機械学習で最も有名で最も多く用いられていると思われるが，あらゆることができるわけではない．TensorFlow を含めた各フレームワーク及びライブラリには特徴・得手不得手があるので，目的に応じて使い分ける必要がある．日本製で DNN 専用の Chainer は学習中に動的にモデルを変更できる define by run[39]に初めから対応しており，有向グラフ（モデル）[40]を扱いやすく構造モデルの畳み込みを行いやすいという特徴があり，分散学習（ChainerMN），強化学習（ChainerRL），画像認識（ChainerCV）など目的別にパッケージが作られている．更に創薬分野で行われる分子構造を用いた化学的性質の予測を行う深層学習向けのライブラリ Chainer Chemistry も提供されている．**表 5-3** に示したフレームワークはほとんどがオープンソースであるが，SAS Viya のように有料で年間ライセンス形式のものもある．

表 5-3 主なフレームワーク一覧

名称/開発元	対応 OS	言語	ライセンス	GPU	MEMO/公式サイト
ELKI /Erich Schubert, Arthur Zimek	Ubuntu(Linux)[41], Windows, MacOSX	Java	AGPLv3		データマイニング用ライブラリ https://elki-project.github.io/
H2O /H2O.ai	Ubuntu(Linux), Windows, MacOSX	Scala, R, Python	Apache 2.0	○	分散処理やインメモリ処理などをサポート．Web GUI なども用意されいている． https://www.h2o.ai/
Mallet /Andrew McCallum	Ubuntu(Linux), Windows, MacOSX	Java	CPL1.0		統計的自然言語処理，文章分類，トピックモデリングに特化している http://mallet.cs.umass.edu/index.php

39データを流し（学習し）ながらニューラルネットの構造の構築を行う方法．構築してから（固定されたニューラルネットで）データを流す方法は define and Run.

40 ノード（人，物，化合物，情報など）の関係をエッジ（矢線）でつないで表したもの．「有効グラフを用いてモデル化する」という意味から「有効モデル」とも用いられる．

41 「Ubuntu(Linux)」は Linux のディストリビューションである Ubuntu が推奨されているが，他の Linux が排除されているのではないことを表している．

名称/開発元	対応 OS	言語	ライセンス	GPU	MEMO/公式サイト
Orange /University of Ljubljana	Ubuntu(Linux), Windows, MacOSX	C++, Python	GPL3.0		機械学習・データマイニングおよびデータ視覚化用の Python ライブラリ https://orange.biolab.si/
scikit-learn /INRIA, TelcomParisTech, Google	Ubuntu(Linux), Windows, MacOSX	Python	BSD		Python 用機械学習（データマイニング）ライブラリ http://scikit-learn.org/stable/
Shogun / Soeren Sonnenburg, Gunnar Raetsch	Ubuntu(Linux), MacOSX, Windows	Python, Octave, R, Java/Scala, Lua, C#, Ruby	GPL3.0		・サポートベクタマシン（SVM）に重点を置いている機械学習ライブラリ http://www.shogun-toolbox.org/
TensorFlow / Google	Ubuntu(Linux), Windows, MacOSX	Python3.x, C++（C#, Haskell, Julia, Rust, Ruby, Scala, R）[42]	Apache 2.0	○	おそらくもっとも使われている．対応する領域も広い． https://www.tensorflow.org/
Chainer / Preferred Networks	Ubuntu, CentOS, Windows, MacOSX	Python3.x	MIT	○	日本製，define by run の先駆け https://chainer.org/
Caffe /Yangqing Jia, Berkeley AI Research	Ubuntu, CentOS, Windows, MacOSX	C++, Python, MATLAB	BSD 2-Clause license	○	Deep learning framework，画像処理に強い http://caffe.berkeleyvision.org/
MXNet /University of Washington, CMU, AWS	Ubuntu(Linux), Windows, MacOSX	C++, JavaScript, Python3.x, R, Matlab, Julia, Scala, Clojure, Perl	Apache 2.0	○	メモリ効率が良い．豊富な学習モデルが使え，画像認識，自然言語処理からレコメンド生成まで行える． https://mxnet.apache.org/
PyTorch / Adam Paszke etc., NYU, Facebook	Ubuntu(Linux), Windows, MacOSX	Python3.x, C++	BSD-style licensed	○	Python のパッケージ．GPU を用いたテンソル計算・自動微分を強化される． http://pytorch.org/

42 サードパーティーの言語バインディングを用いることで可能.

名称/開発元	対応 OS	言語	ライセンス	GPU	MEMO/公式サイト
The Microsoft Cognitive Toolkit (CNTK) /Microsoft	Ubuntu(Linux), Windows,	BrainScript, Python3.x, C#	MIT	○	有向グラフを利用してニューラルネットワークを一連の演算的ステップとして記述できる．幅広い領域に対応しているが，特に音声認識/画像認識/検索適合性評価に強い．強化学習にも対応． https://www.microsoft.com/en-us/cognitive-toolkit/
DSSTNE /Amazon	Ubuntu(Linux)	Python，R，Scala，Julia，C++	Apache 2.0	○	スパースな行列データに強い． https://github.com/amzn/amazon-dsstne
Neural Network Library(NNabla) / SONY	Ubuntu(Linux), Windows,	C++, Python3.x	Apache 2.0	○	Neural Network Console という GUI 環境によりコードを書かなくても DNN を設計できる． https://nnabla.org/
SAS Viya /SAS	Ubuntu(Linux),	SAS，C++, Java, Python3.x, R，Matlab, Perl		○	https://www.sas.com/ja_jp/software/viya.html
Dopamine /Google	Ubuntu(Linux), MacOSX	Python3.x, C++	Apache 2.0	○	TensorFlow ベースの強化学習フレームワーク．容易さ，柔軟性，安定性，再現性が高められている． https://github.com/google/dopamine

表 5-4 Python から使用でき，よく用いられるライブラリー覧

名称	分類	機能・特徴，公式サイト	Anaconda 標準
NumPy	数値計算	数値計算全般，配列（テンソル）処理を行える．計算処理が速い．http://www.numpy.org/	○
pandas	データ分析	Python を使って Excel や SQL，R 言語のような感じでデータを取り扱えるようにしてくれる便利なライブラリ．R 言語のような統計解析，ベクトル処理，グラフ表示（簡易な）も行うことができる．http://pandas.pydata.org/	○
SciPy	数値計算	NumPy ではコードが長くなる高度な統計，クラスタリング，積分，線形代数，フーリエ変換，信号処理，画像処理，画像配列操作，遺伝的アルゴリズム，特殊関数等の数値計算を簡単に行える．https://www.scipy.org/	○
matplotlib	グラフ作成	データを pandas よりも複雑なグラフや画像データ，表として表示できる．https://matplotlib.org/	○

scikit-learn	機械学習	分類回帰クラスタ分析（教師あり：ニューラルネットワーク，サポートベクターマシン，ランダムフォレスト，ナイーブベイズ，k 近傍法，Ada Boost，教師なし：k 平均法，主成分分析など)を手軽に実装できる．http://scikit-learn.org/	○
NLTK（Natural Language ToolKit	自然言語処理	単語の切り出し，テキスト中に連続して現れる単語や頻繁に表れる一連の単語列の切り出し，統計処理（学習）を行える．http://www.nltk.org/	○
Word2Vec	自然言語処理	ニューラルネットワークのモデルの一つである Skip-Gram などを用いて隠れ層の重みを用いて単語の意味をベクトルに縮約する．https://radimrehurek.com/gensim/models/word2vec.html	
Gensim	自然言語処理	トピックモデルに特化しており TF-IDF（Term Frequency-Inverse Document Frequency），LSA（Latent Semantic Analysis），LDA（Linear Discriminant Analysis），word2vec などの典型的なアルゴリズム，テキスト処理のツールが利用できる．https://radimrehurek.com/gensim/	
MeCab	自然言語処理	最も有名な日本語の自然言語処理器（形態素解析エンジン）．平均的に ChaSen, Juman, KAKASI より高速に動作する．http://taku910.github.io/mecab/	
Cabocha	自然言語処理	機械学習(SVM)を利用した構文解析器．MeCab と組み合わせて使用する．http://taku910.github.io/cabocha/	
OpenCV	画像処理	代表的な画像処理・認識ライブラリ．オープンソースであるが，一部アルゴリズムは非商用．画像認識に加えて，画像のノイズ除去，3 次元画像処理，AR/VR 対応等幅広い画像処理を行うことができる．http://opencv.org/	
scikit-image	画像処理	特徴量や認識に関するアルゴリズムに強い．OpenCV にないアルゴリズムもある．https://scikit-image.org/	

5.4. 言語

深層学習を含む機械学習専用の言語があるのではなく，行う機械学習に適したフレームワーク，ライブラリが対応している言語を用いることになる．Python はほとんどのフレームワーク，ライブラリが対応しており，ユーザが多く，WEB 及び書籍の情報が豊富である．また，Python のコードはシンプル（他の言語より短いなど）で読みやすいことも利点である．

Python には Python2.x と Python3.x 系のバージョンがあるが，両者には一部互換性がない．Python2.x は主に WEB アプリケーション開発で用いられ，機械学習のフレームワークは対応していない場合があり，2020 年にはサポートされなくなるため注意が必要である．統計解析，データの可視化で多く用いられる R 言語も機械学習に用いられることが多い．その他，C++,

Java，Scala なども用いることができる．主なフレームワークを利用できる言語も**表 5-3** に整理した．

5.5. エディタ・開発環境（IDE）

プログラムコードを書くにはエディタが必要であるが，その入力支援機能（関数の入力補完，引数のリスト表示，変数の色分け，ブロックでの畳み込みなど）並びに開発中のコードのデバッグ，ステップ・バイ・ステップでの実効，結果の確認が必要で，この一連の流れは開発効率に大きく影響する．エディタを含む開発環境は各言語専用のものもあるが，多くは汎用で，各言語用のプラグインをインストールすることで利用できる．Java の IDE（エディタ含む）として有名な Eclipse は PyDev というプラグインを用いることで Python を扱うことができる．PyCharm は Python 専用の高機能な IDE として有名で，有償版と無償版がある（有償版はインスペクションが追加され，プラグインで機能を追加でき，クラス図が生成でき，データベースツールがあるなど）．Anaconda は Python と深層学習を含む機械学習を行う基本的なライブラリ，エディタ・開発環境（Jupyter Notebook）をまとめたディストリビューションであり，Python のインストーラーとしてよく用いられる．Python を使用可能なエディタ・IDE の一覧が Python のサイト（https://wiki.python.org/moin/PythonEditors）に掲載されている．

5.6. ツール

データの視覚化・確認，アノテーション（ラベル付け），前処理，学習モデルの視覚化，ハイパーパラメータの自動設定，モデルのデプロイメント[43]など，深層学習を含む機械学習に関する色々な便利ツールである．複数のツールをまとめて「プラットフォーム」として提供されているものもある．

5.6.1. アノテーションツール

アノテーションとは訓練データに教師情報を付与する作業であるが，例えば画像データに分類の情報（ラベル）を付けるだけでなく，画像中の分類の対象物（人の顔と目など）の範囲をマークしたり，自然言語処理では文書中の目的とする単語を分類するタグで挟んで単語同士の関係付けをするなど，煩雑な作業を多量に行う必要があり，複数のアノテーター[44]によりバリデーションを行う場合もある．こういったアノテーション作業を行うためのソフトがアノテーションツールである．例えば Microsoft からは無償の VoTT（Visual Object Tagging Tool

43 作成したプログラム，システムなどを開発環境から運用環境に配置して利用可能にすること．
44 アノテーション作業をする人のこと．

https://github.com/Microsoft/VoTT）が公開されている．VoTT は対象物を四角形で囲むだけであるが，Google からは AI を用いて画像内の境界識別（領域抽出）をサポートする Fluid Annotation が発表されている[41]．自然言語処理（テキストデータ）では brat（MIT ライセンス http://brat.nlplab.org/index.html ）doccano（MIT ライセンス https://github.com/chakki-works/doccano）などがある [45].

5.6.2. プラットフォーム

総合開発環境と重なる部分があるが，深層学習を含む機械学習のデータの準備や結果の管理などを行うツールをセットにしたものが提供されている．代表的と思われる 2 件に加えて医療分野の画像に特化した 1 件を紹介する．

- TensorFlow Extended (TFX)：https://www.tensorflow.org/tfx/

  TensorFlow Extended はその名前の通り，Google が提供する深層学習を含む機械学習システム構築時に必要となる機能を提供するプラットフォームであり，TensorFlow Data Validation，TensorFlow Transform，TensorFlow Model Analysis，TensorFlow Serving の各ライブラリで構成されている．

  - TensorFlow Data Validation：データの統計量の可視化，スキーマ推定，異常検出などデータの確認（検証）を行うためのライブラリ．
  - TensorFlow Transform：前処理を行うためのライブラリ．例えばデータの正規化，テキストデータをインデックスに変換する，あるいはデータ分布に基づいて浮動小数点の値を整数に変換するなど．TensorFlow Transform は代表的な前処理方法を提供することで，モデルと前処理を近づけて学習時と推論時の前処理適用が統一できるようにしている．
  - TensorFlow Model Analysis：モデル評価を様々な切り口で手軽に行える．これにより，モデル性能改善のヒントを得られることが期待される．

- Machine Learning Data Flow（MLFlow）：https://mlflow.org/

  深層学習を含む機械学習の開発において複雑になりがちな実行環境，モデル，パラメータ，評価結果などの管理を行うプラットフォーム．作成モデルのデプロイについても簡単な API を提供する機能でカバーされている．MLFlow Tracking，MLflow Projects，MLflow Models の 3 つの機能で構成されてる．

  - MLFlow Tracking：機械学習のモデル作成時の情報を保存，管理する機能．モデルご

45アノテーションツールは https://github.com/topics/annotation 及び https://github.com/topics/image-annotation に色々リストされている．

とに学習に用いるデータ，前処理・学習モデルの方法・ハイパーパラメータ，テストデータに対する評価結果等が保存される．

- MLflow Projects：作成した機械学習モデル作成コードをパッケージ化して誰でも利用できるようにする機能．パッケージにはローカル上のコードだけでなく Git Hub 上のコードも含まれる．実行時の環境をファイルに記述しておくことで再現できる．
- MLflow Models：MLflow Tracking で保存したモデルを簡単にデプロイする機能.

- NVIDIA Clara Platform（Clara）https://developer.nvidia.com/clara

  NVIDIA 社から提供される CT，MRI 等の医療画像のディープラーニング，分析をサポートする医療用の AI プラットフォームである．医療画像を効率よく学習させるには，画像中の対象となる部分を正確にマーキングするアノテーションが必要となるため，とても負担が大きい．Clara はアノテーションをサポートする AI と，医療画像を分析する学習済モデル，転移学習を行う Tool，学習させたモデルを運用する Tool が含まれる．アノテーション Tool は，例えば肝臓の正面，右側面，断面の 3 つの画像がある場合，1 人目の正面と側面の画像中の識別したい境界を数か所クリックするとそこから境界線を自動で抽出し 3 次元上のモデルに境界が認識される．これにより，肝臓は 8 倍，脾臓は 10 倍，すい臓は 4 倍ほど早くアノテーションが行えるとされている．

5.7. 深層学習を含む機械学習用ワークステーション・サーバ

オンプレミスで深層学習を含む機械学習の環境を構築することを前提に紹介してきたが，ハードウェアの選定，フレームワーク，ライブラリのバージョン合わせを含め，環境構築には躓くことが少なくなく手間を要する．また，ハードウェアも専用のものが必要で組合せの問題もあるため，あらかじめ一通りの動作検証された環境をインストールした機械学習用のワークステーション，サーバが販売されており，購入した日から開発を行うことが可能である．これらは将来的に必要となる GPU やメモリのソケットの規格，数などハード的な上限から選定するが，価格は搭載する GPU の機種と数の影響が大きく，初期は例えば GPU は 1 個とするなど，コストを抑えてスタートすることができる．

5.8. コンテナ

5.7 節で説明した通り，深層学習を含む機械学習のフレームワーク，ライブラリにはバージョン合わせが必要であるため，1 つの機械学習環境で複数の機械学習の開発や運用を行うと，ライブラリの参照エラーなど障害が生じることがある．そのため，オンプレミスのワークステーション・サーバを用いる場合でも機械学習ごとに仮想環境を構築して他の機械学習と切り離

すことが行われる場合がある．従来，仮想環境（ハイパーバイザ型やホスト型）では，ハードウェアホスト OS 上に仮想マシン環境を構築してその中でゲスト OS を含む仮想マシンを稼働させているが（**図 5-1** 左側），機械学習を独立にするだけであれば，仮想環境ごとの OS は必要ない．そのため，Unix/Linux OS が提供する「コンテナ」というアプリケーションプロセスを独立させる仕組みが用いられている．コンテナは名前空間やリソースが他のプロセスやコンテナからは隔離され，おのおの固有の設定を持て，コンテナ内のアプリケーションとは独立したコンピュータ上で動作しているように見える．コンテナは OS 上の 1 プロセスであるため管理コスト・リソースは通常のプロセスとほとんど変わらず，仮想マシンと比較すると OS が複数稼働する必要がない分軽い．コンテナの利用にはコンテナ内で利用するアプリケーションなどを 1 つのパッケージとしてデプロイする必要がある．コンテナを Windows や Amazon Web Services（AWS）等のクラウド環境でも利用可能にするコンテナ型仮想化ソフトとして Docker（**図 5-1** 右側）がオープンソースのプロジェクトとして公開され，広く利用されている．

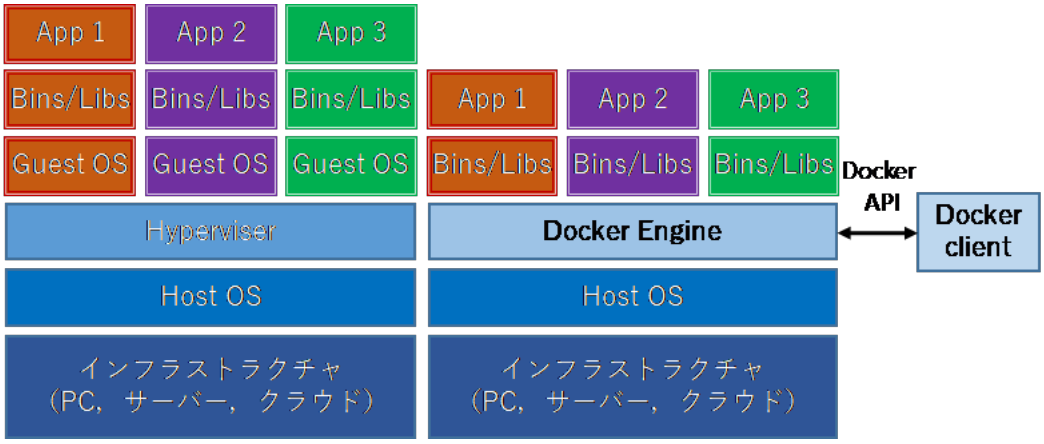


図 5-1 通常の仮想化と Docker による仮想化

5.9. クラウド（パブリッククラウド）環境

企業や個人など不特定多数のユーザにインターネットを通じてサーバやストレージ，データベース，ソフトウェアなどのクラウドコンピューティング環境を提供するサービスは特定ユーザにのみ向けたサービスと区別してパブリッククラウドと呼ばれる．有名なものに Google Cloud Platform（GCP），Microsoft Azure，IBM Cloud，Amazon Web Services（AWS）などがあり，AI 環境も提供されている（**表 5-5**）．パブリッククラウドでは，高額のハードウェアや通信回線を自社で購入・保守する必要がなく，必要なときに必要な量・性能のクラウド環境を素早く利用することが可能である．パブリッククラウドは AI 環境も多くの場合は従量制の

契約であるため，ストレージやデータベース，演算装置を追加したい場合でもサーバ構成などを意識することなく（費用は掛かるが）容易に拡張できる．少量/短時間の使用ならば，比較的安価に利用できることも大きな魅力である．加えて，クラウドの AI 環境では画像や音声，自然言語処理など学習済のモデル（AI）が提供されており，API を用いて簡単にサービスに組み込むこともできる．

表 5-5 主なパブリッククラウド AI の概要

名称 （提供元）	概要 オフィシャルサイト	学習済 AI の種類， アルゴリズム
Google Cloud Machine Learning （Google）	事前学習済みのモデルを活用して独自のカスタムモデルを生成可能なサービス． https://cloud.google.com/products/machine-learning/?hl=ja	• 動画分析 • 画像分析 • 音声認識 • テキスト分析 • 翻訳 （学習済モデルでの使用アルゴリズムは非公開）
Amazon Machine Learning （Amazon）	複雑な機械学習アルゴリズムやテクノロジーを習得する必要なく，機械学習を利用できる． https://aws.amazon.com/jp/aml/	• 回帰分析 • 二項分類 • 多項分類
Azure Machine Learning （Microsoft）	クラウドの予測分析サービスであり，そのまま使用可能なアルゴリズムのライブラリを用いてインターネットに接続した PC でモデルを作成し，迅速にデプロイできる． https://azure.microsoft.com/ja-jp/overview/machine-learning/	• スケーラブルブーストディシジョンツリー • ベイシアンレコメンダーシステム • ディープニューラルネットワーク • ディシジョンジャングル • 分類 • 多項分類と二項分類 • 回帰クラスタリング
Watson Machine Learning （IBM）	IBM の実績あるアナリティクスプラットフォーム上に構築されている．REST API コネクターを利用して，任意の言語で独自のアルゴリズムを構築できる．IBM Watson を中核とするコグニティブ・ソリューションである． https://www.ibm.com/analytics/jp/ja/technology/machine-learning/	非公開

6. 教育・情報源

6.1. 教育

AI をビジネスに導入する場合，深層学習を含む機械学習により構築されたモデル（AI）を一部に用いた販売されているサービスを導入する，自社（外注含む）で社内利用するサービスに用いる AI を開発する，自社で社外（医療従事者，患者等）に提供するサービスに用いる AI を開発するといった場合と，機械学習を用いてデータをマイニングすることで新たな知識を発見するといった場合がある．前者では必ずしも利用者自身が機械学習の手法などを理解し，プログラミングできる必要はないが，後者では必須となる．また，前者でも自社で開発する場合のプロジェクト担当者には開発エンジニアに訓練データの解釈やモデルの選択を行うのに必要なドメイン知識を説明する力，機械学習の基本的理論や訓練データの品質の影響，学習結果のモデルの評価方法などはスキルとして重要である．つまり，AI をビジネスに用いるにあたって，全ての場合に“自力で開発できるスキル”が求められるのではないが，外注する場合でも何もわからなくていいというわけではない．

一般社団法人 日本ディープラーニング協会[46]では，ディープラーニングの基礎知識を有し，適切な活用方針を決定して事業応用する能力をもつ人材（ジェネラリスト）と，ディープラーニングの理論を理解し，適切な方法を選択して実装する能力を持つ人材（エンジニア）に分けて，前者は「G 検定」（**表 6-1**），後者は「E 資格」のスキルセットの定義，トレーニングと試験が実施されている[42]．また，Danysz K（2018）では，市販後安全監視に AI を用いる場合に担当者に求められるスキルを King E.（2011）が提案するモデルに従って**表 6-2** の 3 段階に分けて定義しているが[43]，その内容は G 検定とほぼ重なっている．G 検定は分かりやすい公式テキストが販売されており[6]，「ディープラーニング」と銘打っているが，内容は機械学習全般をカバーされているので，企業で AI を導入・運用・活用する担当者は検定を受けなくても自習に役立つであろう．自ら開発する「エンジニア」の場合，開発する分野はある程度特定されるので，ベンダーのエンジニアのように機械学習について幅広い知識，スキルは不要かもしれないが，**表 6-3** にあるような基礎的スキル・知識が必要となる．企業のデータサイエンス（生物統計など）担当者であれば，このうち数学，統計，SQL などいくつかのスキル・知識は有しているであろうから，機械学習特有のパートに絞ることができる．そういった場合，入門として Python の習得，TensorFlow，Keras，scikit-learn といった有名なフレームワーク，ライブラリを用いたテキスト，チュートリアルコースは多くあり，有用である．例えば，我々

46 一般社団法人 日本ディープラーニング協会ホームページ https://www.jdla.org/

製薬業界に近いものとして日本メディカル AI 学会にはメディカル AI 専門コースと資格制度がある[47]．講義資料は協会の WEB サイトに，チュートリアルのデータ・コードは GitHub にいずれも無料で公開されており，同じく無料で利用できる Google Colaboratory で実際にコーディングと機械学習を行うことができる[48]．この講義資料は**表 6-4** のように MRI 画像を扱うもの，血液の顕微鏡画像からの細胞検出など，医療分野の実践的な内容となっている．学習用環境を用意する必要がなくコードを書き写し実行して確かめながら学習できること，フレームワークに分子構造，受容体，Pathway，時系列の自然言語など生物化学分野で扱うことが多いグラフ構造に適した Chainer を用いることも魅力である．

これ以外にも，AI ブームを反映して無料／有料含め，提供されるクラウドの学習環境を用いて Python などでコードを書き，確かめながら学習する e-Learning が多く提供されている．経済産業省では第四次産業革命スキル習得講座認定制度[49]を 2018 年から行っており，AI の開発者教育講座を多く認定されているので，教育講座を検討する際には参考になるであろう．

表 6-1 JDLA Deep Learning for GENERAL 2019 **のシラバス**[50]

· 人工知能（AI）とは（人工知能の定義）
· 人工知能をめぐる動向 探索・推論，知識表現，機械学習，深層学習
· 人工知能分野の問題 トイプロブレム，フレーム問題，弱い AI，強い AI，身体性，シンボルグラウンディング問題，特徴量設計，チューリングテスト，シンギュラリティ
· 機械学習の具体的手法 代表的な手法，データの扱い，応用
· ディープラーニングの概要例題 ニューラルネットワークとディープラーニング，既存のニューラルネットワークにおける問題，ディープラーニングのアプローチ，CPU と GPU，ディープラーニングにおけるデータ量
· ディープラーニングの手法 活性化関数，学習率の最適化，更なるテクニック，CNN，RNN，深層強化学習，深層生成モデル
· ディープラーニングの研究分野 画像認識，自然言語処理，音声処理，ロボティクス（強化学習），マルチモーダル
· ディープラーニングの応用に向けて 産業への応用，法律，倫理，現行の議論

47 日本メディカル AI 学会公認資格（メディカル AI 専門コース）：
https://japan-medical-ai2019.org/qualification.html
48 講義資料ページ: https://japan-medical-ai.github.io/medical-ai-course-materials/
GitHub リポジトリ: https://github.com/japan-medical-ai/medical-ai-course-materials
49 http://www.meti.go.jp/policy/economy/jinzai/reskillprograms/index.html
50 https://www.jdla.org/business/certificate/?id=certificate_No03

表 6-2 Proposed drug safety/pharmacovigilance core competencies[43]

Skillsets: analytical/assessment skills	
Level 1	Ability to understand concepts of artificial intelligence, natural language processing, machine learning and deep learning. Ability to interact with and identify issues with artificial intel- ligence, natural language processing, machine learning and deep-learning outputs in user interface. Understanding of how acceptance or overwriting machine learning outputs in user interface relates to deep learning.
Level 2	Ability to distinguish and describe concepts of artificial intel- ligence, natural language processing, machine learning and deep learning. Ability to troubleshoot with artificial intelligence, natural language processing, machine learning and deep-learning outputs in user interface. Troubleshooting issues with acceptance or overwriting machine learning outputs in user interface related to deep learning.
Level 3	Ability to describe and train on concepts of artificial intel- ligence, natural language processing, machine learning and deep learning. Ability to modify artificial intelligence, natural language pro- cessing, machine learning and deep-learning outputs in user interface, by initiating retraining of algorithm. Resolving issues with acceptance or overwriting machine learn- ing outputs in user interface related to deep learning.

表 6-3 深層学習を含む機械学習エンジニアに必要なスキル・知識

①	プログラミングスキル：Python など機械学習で用いる言語，言語で汎用的に用いる NumPy，pandas などのライブラリ及び機械学習で用いる scikit-learn，TensorFlow，keras などのライブラリ及びフレームワークのプログラムを作成するスキル.
②	数学：微分，線形代数，行列，ベクトル，組合せ，確率など
③	統計の知識：分散・標準偏差，確率分布，推定，検定，抽出など
④	機械学習の基礎知識：前処理，教師あり学習と教師なし学習，特徴量設計，データセット作成，手法（単回帰，重回帰，最小二乗法，パーセプトロン，ロジスティック回帰，決定木，ランダムフォレスト，サポートベクトルマシン，ディープラーニングなど），ライブラリ・フレームワーク（scikit-learn，TensorFlow，Keras など），モデル評価（交差検定，混合行列など）のスキル・知識
⑤	SQL を使ってデータベースを操作する知識
⑥	クラウドの知識（用いる場合）
⑦	学習済モデルを組込むアプリケーションの知識
⑧	開発する分野・データに関する知識

表 6-4 メディカル AI 専門コース講義資料目次 [48]

1 章：機械学習に必要な数学の基礎
2 章：機械学習ライブラリの基礎
3 章：ニューラルネットワークの基礎
4 章：Deep Learning フレームワークの基礎
5 章：実践編: MRI 画像のセグメンテーション
6 章：実践編: 血液の顕微鏡画像からの細胞検出
7 章：実践編: ディープラーニングを使った配列解析
8 章：実践編: ディープラーニングを使ったモニタリングデータの時系列解析

6.2. 情報源

プログラミングやデータの加工を含め，深層学習を含む機械学習・AI とその周辺の知識，技術情報は書籍や論文よりも WEB のコミュニティやフォーラムが充実している．有名と思われるものを**表 6-5** に整理した．また，AI の技術，サービス及び開発ベンダーを調べるには，特許申請情報 [51]が役立つ．

表 6-5 深層学習を含む機械学習のコミュニティなど

名称	紹介・キャッチコピー・特徴/URL
GitHub	プログラム（ソースコード）のバージョン管理ソフトである Git を用いて作成したプログラムやデータを公開，共有するサービス．機械学習で用いる多くのフレームワーク，ライブラリ，言語などのリポジトリ，配布にも用いられている．機械学習のチュートリアル資料，サンプルコードも多くある． https://github.com/
Stack Overflow	プログラマーに対する Q&A サイト．オープンソースソフトウェアを利用する場合，環境構築やバグの問題は避けて通れないが，そのような問題も含め広く議論されている． https://stackoverflow.com/
Stack Exchange	データサイエンスの専門家，機械学習のスペシャリスト，およびこの分野についてもっと知りたい人のための質疑応答サイト． https://datascience.stackexchange.com/
Kaggle	The Home of Data Science & Machine Learning Kaggle が運営するデータサイエンスコンペティションを通じてデータを有する企業や政府などの組織とデータサイエンティスト/機械学習エンジニアを繋げるプラットフォーム． https://www.kaggle.com/

51 特許庁 J-PlatPat https://www.j-platpat.inpit.go.jp/web/all/top/BTmTopPage,
Google Patents https://patents.google.com/advanced など

Google Cloud Platform のドキュメント	Google Cloud Platform（GCP）の使用方法，サンプルなどのドキュメントのサイト．Python 等の言語毎に GCP の使い方の説明，チュートリアル，GCP コミュニティの入り口などがある． https://cloud.google.com/docs/?hl=ja
TFUG	TensorFlow User Group Tokyo TensorFlow User Group（TFUG）は有志による TensorFlow のコミュニティです． TFUG は以下のような人を歓迎します: •TensorFlow を使っている人 •他のフレームワークを使っているけれど TensorFlow にも興味がある人 •現在 TensorFlow を使っていないけれど興味がある人 https://tfug-tokyo.connpass.com/ https://www.facebook.com/groups/178559235921208/
Chainer Colab Notebooks	Chainer User Group によるフレームワーク Chainer を用いた Deep Learning 実践ノートブック集．入門者用及び中級者用の Hands-on，公式事例集などがある．ボタンを押せばコードを Google Colaboratory で実行できる． https://chainer-colab-notebook.readthedocs.io/ja/latest/index.html#
qiita	Hello hackers ! Qiita は，エンジニアリングに関する知識を記録・共有するためのサービスです．コードを書いていて気づいたことや，自分がハマったあの仕様について，他のエンジニアと知見を共有しましょう https://qiita.com/
AINOW	AINOW は，人工知能を知り・学び・役立てることができる国内最大級の AI・人工知能専門メディアです．2016 年の 7 月に創設されました．「バイトル」「はたらこねっと」「ナースではたらこ」などを展開するディップ株式会社に設置された dip AI. Lab が運営しています．2 万件以上の AI 系ニュース掲載．毎日 AI に関する記事を収集する日本初のメディアです． http://ainow.ai/
Team AI	日本最大級 6,000 人の機械学習コミュニティ，A.I.研究会&ハッカソンを毎週渋谷で開催，シリコンバレーの雰囲気を味わってみませんか？ A.I. Research Community & Hackathon 徹底的に実務に根ざした，米国流で議論中心の研究会を無料で体験できます．データ分析ハッカソンはかなり実際の仕事に近い環境なので，直近の業務に活かしていただける知識と経験が手に入ります． https://www.team-ai.com/
JHC	Japan H2OCommunity 当コミュニティは，オープンソースである H2O 3，商用版 Driverless AI 及びディープラーニングを含む機械学習の利用促進のための情報共有を行うための任意団体です． https://jhc.h2o.jp/
SPJ	SPJ 社による AI に関する技術，事例などをわかりやすく解説されている記事が掲載されている．カテゴリーは機械学習，チャットボット，自然言語処理，画像分類・生成，音声認識，海外最新 AI 事情 がある． https://spjai.com/category/article/

7. AIを用いて実現できそうなサービス，システムの例（アイデア）

本章では，他分野を含めて既に実用化されているあるいはほぼ確立された技術要素を用いて，「AI技術を応用すればできる（できそうな）」製薬企業特有の用例，いわばアイデアを紹介する．既存の特定商品や，ある企業が開発又は導入した内容の紹介は，業界内のネットワークや業界紙から入手可能であることから，本報告書では取り扱わない．

（ア）電子カルテの画面から記載された情報をEDCにキャプチャ

① 何をするAIか？：

治験，製造販売後調査等では臨床データの収集にElectronic Data Capture（EDC）が用いられている．実態は名称と異なり，医師が紙に手書きする代わりにコンピュータ画面に手入力が行われている．このAIは電子カルテ画面のCapture（写真画像）から光学的文字認識（Optimal Character Recognition/Reader：OCR）処理によりElectronic Data Captureシステム（EDC）に取り込めるデータを出力することでデータ入力の手間を大きく改善できる．

② 特徴：

- 電子カルテの外部へのネットワーク接続やテンプレートの設定など行うことなくEDCにデータを取り込める．
- 読み取る範囲や項目は「医薬品」，「臨床検査」など項目を表すキーワードによりAIが識別する．
- 解像度の課題はあるが，X線画像にも対応可能である．
- 治験，製造販売後調査，副作用調査を問わず対応可能である．
- 画像も保存すれば，原資料の確認も可能である．
- 画像を保存する場合は，保存前に患者を特定可能な情報（氏名，生年月日，医療機関内IDなど）をAIで識別してマスクする．
- 医師が見ている電子カルテの画面からデータを取得するので，電子カルテの複雑なデータベース（テーブル・項目）と出力するデータ項目のマッピングは不要である．

③ 用いるAIの技術：

中心となるAIの技術は画像解析によるテキストデータ抽出と自然言語処理である．ディープラーニングを用いて，定型又は半定型帳票の印刷された紙の画像やPDFファイルからテキストデータを抽出し，CSVあるいはXMLなど構造化（規則的）データ

に出力する OCR ソフトは複数商品化されている（「AI-OCR」などと呼ばれる）．この AI は取り込むデータ項目に関連する複数のキーワードを設定して学習させることで，電子カルテの画像上のキーワードから取り込むべきデータ範囲を判断する．加えて，傷病名や薬剤名などのマスターデータと深層学習を含む機械学習を用いることで，各項目の読み取り精度を高められる．基本技術は既に実用化されているので新規の AI 開発は必要なく，既存学習モデルの転移学習または Fine Turning [52]で開発は可能であろう．

④ 学習：

既に商品化されている請求書や調査票から OCR により構造化データを作成する学習済 AI（モデル）を用いて転移学習などを行う．マスキングすべき患者の個人識別情報（氏名，生年月日，医療機関内 ID など），薬剤名や臨床検査など，電子カルテの画面から項目を特定するためのマスタデータと，電子カルテまたはそれに似せた画面を用いて，画面に表示しているデータを教師データとして OCR 及び画面から構造化データを取り出す学習を行う．加えて，医薬品，傷病名，臨床検査項目，診療行為等のマスタデータと，製造販売後調査，副作用自発報告や実際の診療情報を用いて汎用的な文字データのテキストデータ化の転移学習などを行う．

⑤ 開発されるアプリの動作イメージ

スマートフォンのアプリには[患者]，[薬剤]，[臨床検査]，[画像]，[病歴]などのメニューボタンがある．例えば[臨床検査]のボタンを押して電子カルテの検体検査結果（臨床検査）の画面の写真を撮ると，写真上に取り込む範囲の矩形が表示される．範囲を確認（ずれていればドラッグして修正）して[読取り]ボタンを押すと，読み取ったデータがテーブルとして表示される．この時，予め必要な検査項目が設定されている場合，それ以外の検査項目はグレーで表示される．必要なデータのみテーブルに含まれていることを確認したら[登録]ボタンを押す．アプリは読み取り範囲などの情報を GPS の位置情報とセットで記憶することで，2 回目以降は（他のユーザでも）識別精度が改善される．データと画像はサーバーに送信されるが，読み取った画像のうち

52 学習済の AI に新たにデータを追加して学習させることは広く「追加学習」と言われるが，学習済の多層のニューラルネットワークで，入力層から出力層に向かって多くの層（の重み）を固定して出力層側のいくつかの層のみを異なる分野のデータで学習させる方法は「転移学習」と呼ばれる．例えば，哺乳類の画像から犬，牛，ウサギなどを分類する学習済モデルに魚のデータを学習さる転移学習により，少ないデータで効率よく魚を分類する AI を作ることができる．逆に，開発したい AI に必要な訓練データが用意できない場合に訓練データを入手しやすい分野で AI を開発して転移学習により目的とする AI を開発することが可能である．

個人情報部分はアプリ内の AI が識別して画像加工（マスキング）することで，外部に送信されない．

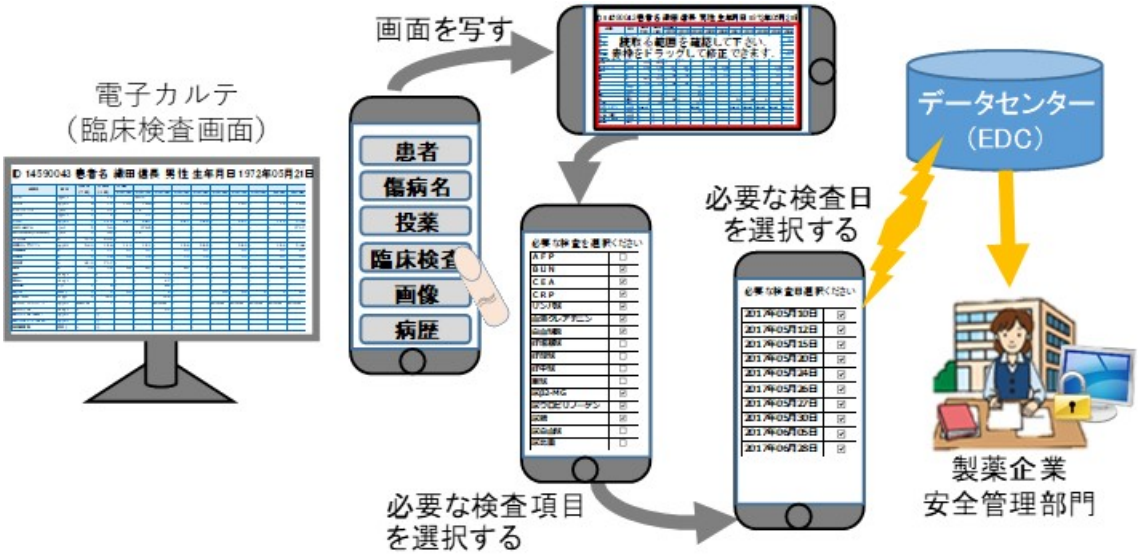


図 7-1 電子カルテからのデータキャプチャイメージ（臨床検査の場合）

⑥　残る課題など：

多くの場合，電子カルテに入力される処方データは 1 日量と回数（日数）あるいは日付と使用量で記録されており，治験や調査などで通常用いられている 1 日量と使用期間という形式となっていないため，予め定義したルールに従って自動変換する，あるいはデータ登録時に変換結果を医師が確認するといった処理が必要となる．あらかじめ電子カルテの画面をフォーム登録しないため，チェックボックスへの対応が難しい．また，取込み画面数が多くなる，あるいは症例数が多い場合，作業が煩雑となる．

⑦　AI 技術がすでに用いられている又は類似のサービス等：

・　FlexiCapture/ ABBYY

文書を理解・分解・転送する作業は最新の機械学習メソッドを使用して自動訓練された AI ベースの分類ソフトウェアによって自動化されている．構造化・半構造化された文書（請求書，税務書類，要望書，添付文書など），あるいは通信文や契約書といった完全に構造化されていない文書であっても処理・分類する．

https://www.abbyy.com/ja-jp/flexicapture/

- DX Suite/ AI inside
  独自の AI／ディープラーニングを使用することで手書き文章に対応．出力ファイルが CSV に対応し，様々な RPA，BPO ソフトとの連携も可能．クラウド型の対応も可能． https://inside.ai/dx-suite/
- Prexifort-OCR/ NTT データ
  黒枠帳票の読み取りにも対応．傾き・伸縮は自動で補正，姓名辞書・郵便番号辞書・ユーザ辞書と単語照合させることで，単語の認識精度を向上させている．
  http://prexi.jp/
- Tegaki/コージェントラボ
  罫線なしの手書き文字だけでなく，チェックボックスや文字を囲っている〇印なども読み取れる．API を用いてテンプレートデータと読み取りイメージデータを Tegaki のエンドポイントに送信することで利用可能． https://www.tegaki.ai/
- Flax Scanner/シナモン
  請求書や契約書等，あらゆる非構造なドキュメントから情報を正確に抜き出し整理する AI エンジン．フォーマットがバラバラな帳票へも対応可能で，さまざまな業務場面に活用できる． http://cinnamon.is/#product
- FormOCR/エヌジェーケー活文
  病院・学校事務，人事給与管理，顧客管理，販売管理，生産管理，マーケティング，経理・会計業務など幅広い業務・業種で利用される定型帳票のデータ入力を効率的に実現する OCR ソフトウェア．帳票に記入された手書き文字/活字文字を OCR 処理し，CSV・TEXT データへの出力を行なう．
  https://mediadrive.jp/products/formocr/index.html
- Intelligent Data Extractor/日立ソリューションズ
  スキャンした書類の PDF データをフォルダに格納するだけで書類の種別で仕分けし，データを抽出する．複数ページにわたる書類もページごとに種別を自動判別する．チェック画面では書類の種別ごとにタブを切り替えて抽出結果を確認できる．
  https://www.hitachi-solutions.co.jp/katsubun/sp/ide/
- Amazon Comprehend Medical と Amazon Rekognition
  画像からテキストを抽出するサービス Amazon Rekognition とテキスト内にある保護対象医療情報の検出と識別をサポートする Amazon Comprehend Medical を使用して医療画像の匿名化を行える．
  https://aws.amazon.com/jp/blogs/news/de-identify-medical-images-with-the-help-of-amazon-comprehend-medical-and-amazon-rekognition/

（イ）診療記録，レセプトデータから副作用シグナル（イベント）の検出

① 何をするAIか？：

曝露とイベントの不均衡に基づくSignal Detection手法は多くの場合，3件を超えるイベント数が必要でcase毎の関連の強さを考慮していない．また，電子カルテなどの診療記録やレセプトデータ（以下「診療データ」）を用いる場合，副作用情報の多くが記載されると考えられる経過記録は自由記載のテキストデータであるため利用されていない．AIを用いて自由記載を含む診療データから，医薬品による副作用の可能性のあるイベント（症例）を抽出する．集積したデータに用いることで，既知の副作用のみでなく，医薬品成分，薬効以外に部分構造，ターゲット（レセプター，タンパクなど）を含むパスウエイ，トランスポーターの情報（データベース）を用いたデータマイニングを行えば，単純な不均衡手法では困難な未知の副作用のシグナルを検出することも可能である．患者毎に検査や受診の都度用いることで患者に生じた副作用を早期に検出することにも利用できる．

② 特徴：

- イベントを有害事象名（副作用名）とするのではなく，有害事象に伴って生じる入院，受診，治療中止などをイベントとし，イベントの原因となった医薬品と傷病名を類推する．これにより患者・時点毎にイベントが生じたとする識別確率とこの原因となる医薬品の確率および傷病名（有害事象名）の確率が出力される．
- 医薬品リスク監視に用いるSignal Detectionとしてはイベント識別確率，原因薬剤確率，原因傷病名確率を重みとして用いることで個々のSignalの確からしさをその統計量に反映できる．
- 医薬品と傷病名（事象名）の組合せだけでなく，被疑医薬品の投与中止（次回以降の処方データがない），入院，非規則的な受診，新たな医療機関・診療科の受診，新たな医薬品の投与，その他診療行為を用いて総合的な判定を行う．
- 自由記載（テキストデータ）中の有害事象記述を自然言語処理により抽出することで傷病名の付与がされにくい有害事象も扱う．
- 初期の訓練データは副作用自発報告，製造販売後調査のデータを用いるが，自社のデータでは医薬品の偏りがあり，データ量も不足する．JADER[53]はデータ項目が少な

53 独立行政法人医薬品医療機器総合機構「医薬品副作用データベース」（Japanese Adverse Drug Event Report database）https://www.pmda.go.jp/safety/info-services/drugs/adr-info/suspected-adr/0003.html

いため，FAERS[54]データを翻訳，コード変換して用いる．

・ 医薬品の薬効成分の構造を KEGG の SIMCOMP[55]の手法により算出される構造類似性，化合物タンパク質相互作用の評価，トランスポーター，ターゲット及びパスウエイを共有する医薬品の情報を用いて医薬品とイベントの原因となった傷病との関連性の強さをデータマイニングにより算出する．

③ 用いる AI の技術：

ルールベース，教師あり学習，半教師あり学習，ディープラーニングによる自然言語処理（NLP：Natural Language Processing）と自然言語理解（NLU：Natural Language Understanding）の組合せ．

④ 前処理及び訓練データの作成

知識モデルの作成，ルールベースでのデータ抽出，医薬品の分類データ（ラベル）付与などを行う．テキストデータからのイベント抽出にはルールベースと機械学習を併用する．

・ 知識モデル作成
  - 入院や死亡など，構造化されたデータにその用語があれば分類可能なものを識別するイベント辞書を作成する．
  - 有害事象名，医薬品名をマスタデータに変換・統合するシノニムデータを用いる．
  - 臨床試験，製造販売後調査，FAERS データ，文献データを用いて有害事象（副作用，疾病）毎の各種イベントの発生確率，発生までの期間，持続期間（入院の場合）及び医薬品ごとの各種イベントの発生確率，発生までの期間，持続期間（入院の場合），各有害事象（副作用，疾病）の発生確率を年齢区分，性別，糖尿病，心血管疾患，脳血管疾患，悪性腫瘍，免疫疾患などの基礎疾患の有無を含む計算モデル（ロジスティックモデル及び樹木モデル）を作成する．
・ 医薬品ごとの適応症データベースおよび使用条件（入院して使用する，○○を中止して使用するなど）データベースを作成する．
・ テキストからイベント，医薬品，傷病名等の抽出

54 FDA's Adverse Event Reporting System https://www.fda.gov/Drugs/GuidanceComplianceRegulatoryInformation/Surveillance/AdverseDrugEffects/default.htm

55 About SIMCOMP https://www.genome.jp/tools/simcomp/simcomp_help.html

「○○は□□□によるものと疑ったが，△△△によるものであった」「○○○は認めなかった」など，存在を肯定/否定/原因/結果とする用語との組み合わせによって異なる．ラベル付き訓練データとして副作用自発報告及び製造販売後調査を用い，イベントを構成する症状，病名については，レセプト電算マスタ[56]，MedDRA，万病辞書[57]などの辞書を，医薬品は HOT コードマスタ等の辞書を基準とした機械学習を行う．その上で形態素解析により単語抽出，肯定/否定等の識別を教師あり機械学習により行う．この学習モデルを用いて電子カルテのテキストデータのサンプルを処理し，結果を人が確認した結果をラベルとして再度学習させる．

- 受診日の設定：レセプトの場合，受診日の項目がないため，日計表の初診や再診の診療報酬の算定日などから設定する．
- 入/退院日の設定：レセプトでは，再入院の場合，初回の入院日が記録されている．また，退院日はコメントコードとなっているため，診療行為レコードから入院料算定が毎日なされているかを検索し，入院料の算定が始まる日を入院日，入院料の算定が途切れる日を退院日とみなす．
- 医薬品の開始日/終了日（中止日）：医科レセプトでは処方日がないため日計表データから医薬品ごとの算定回数を検索し，算定が始まる日を開始日，終わる日を終了日とする．外来では，算定日に処方全ての算定回数が記録されているため，算定日に算定回数を加えた日数を終了日とする．
- 訓練データの作成
  比較的きれいな（簡潔，明確な）電子カルテ等のデータを用いて辞書，シノニム，オントロジーなどによるルールベース及び単純なモデルまたは既存の深層学習を含む機械学習モデル[58]によるラベル付け（アノテーション）を行い，訓練データを作成する（**図 7-2**）．付与できないあるいは誤ったラベルが付与されるデータは除くか手作業によりアノテーションを行う．

56 診療報酬情報提供サービス http://www.iryohoken.go.jp/shinryohoshu/downloadMenu/
57 MEDNLP 医療言語処理グループ 万病辞書 http://sociocom.jp/~data/2018-manbyo/index.html
58 1（イ）⑧AI 技術がすでに用いられている又は類似のサービス等：参照

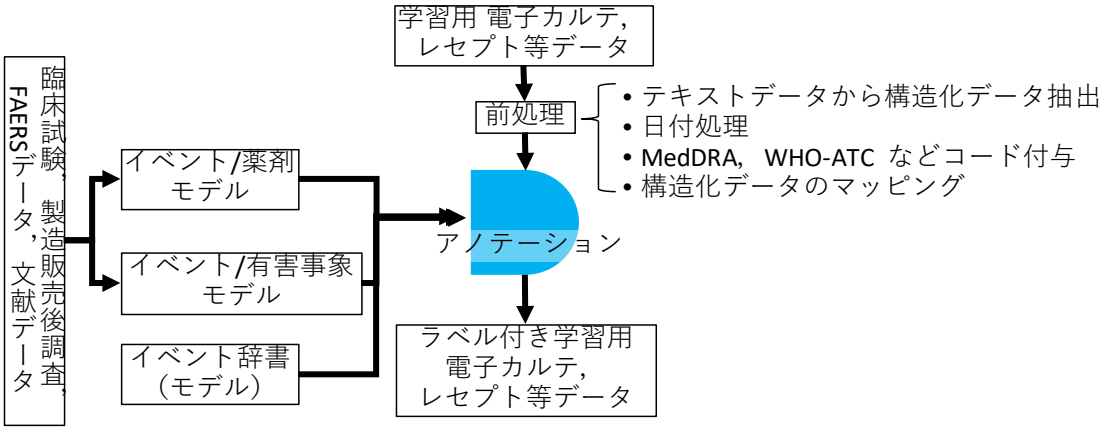


図 7-2 訓練データの作成の概要

⑤ 学習

・ 全体のフローを**図 7-3** に整理した．学習はイベントの抽出と原因となった有害事象名（傷病名）及び原因となった医薬品名の抽出の大きく 3 つに分けられる．

・ イベント

死亡，計画的でない入院，治療の中断・中止，新規の罹患，疾患の増悪，新たな医療機関・診療科の受診，臨床検査値の異常（電子カルテのみ）をイベントとしてイベントの種類ごとに特徴量を設定した教師あり学習により識別モデルを構築する．このモデルを用いて学習用の診療記録，レセプトのデータセットの識別を行い，結果の答え合わせを行う．この答え合わせの結果をラベルとして転移学習により診療記録用，レセプト用の各々の識別モデルを構築する．

・ 原因有害事象

イベントの学習で用いた臨床試験，製造販売後調査，FAERS データ，ヒト疾患ネットワークデータ[59]を用い有害事象名毎に教師あり学習により各イベント種類の識別モデルを構築する．このモデルを用いてイベントの学習で作成したイベントのラベル付診療記録，レセプトのデータセットの識別を行い，結果の答え合わせを行う．この答え合わせの結果をラベルとして転移学習により診療記録用，レセプト用の各々の識別モデルを構築する．

・ 原因医薬品

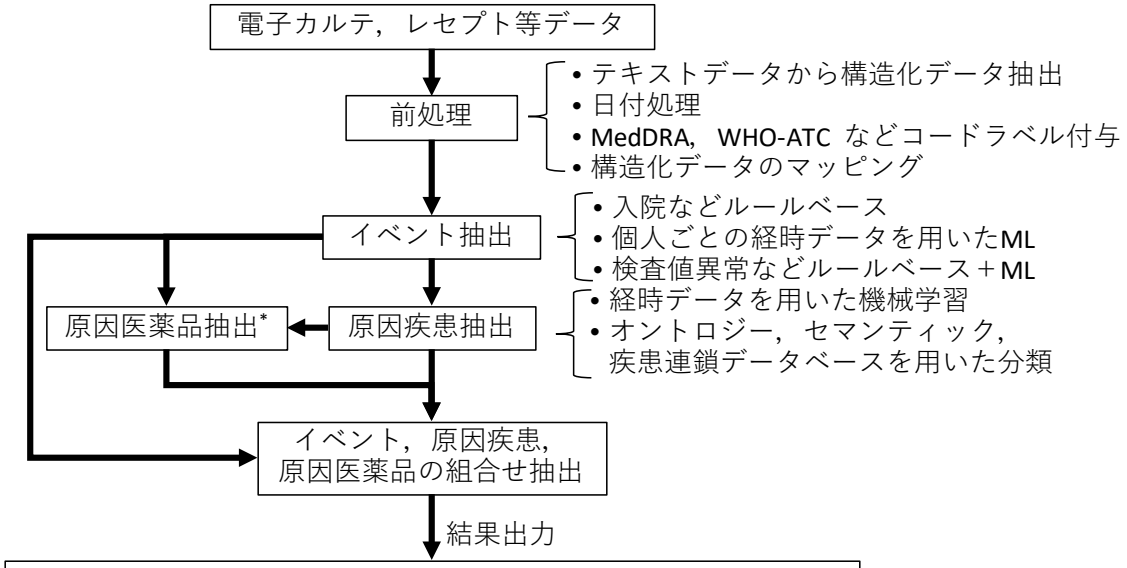
原因有害事象と同様にイベント種類ごとに原因となる医薬品識別のモデルを 2 段階

59 例えば，Kwang-Il Goh, Michael E. Cusick, David Valle, Barton Childs, Marc Vidal, and Albert-László Barabási, The human disease network. PNAS May 22, 2007 104 (21) 8685-8690,
KEGG DISEASE Database https://www.kegg.jp/kegg/disease/disease_ja.html

の教師あり学習により構築する．

⑥ 開発されるシステムの動作イメージ

システムにソースの種類を選んでデータを読み込むと，前処理によりテキストデータの構造化，構造化データのマッピング及びコーディングが行われる．次いでイベント，原因疾患（有害事象名），原因医薬品の抽出及びその組合せの抽出が行われる．抽出データから患者毎及び原因医薬品と原因疾患の組み合わせ毎のイベント発生確率が算出される．



* • 治験，製造販売後調査，副作用自発報告を知識ベースとして学習
• 医薬品の開始，継続・中止情報使用
ML：machine learning

図 7-3 イベント検出のフロー

⑦ AI 技術がすでに用いられている又は類似のサービス等：

・ 診療報酬明細書のデータを用いたデータマイニングに関する技術的検討業務 2010 年度報告書医薬品医療機器総合機構の「診療報酬明細書のデータを用いたデータマイニングによるシグナル検出に関する検討報告書」[60].

60 MIHARI Archive （https://www.pmda.go.jp/safety/surveillance-analysis/0007.html）内のレセプトデータを用いたデータマイニングによるシグナル検出に関する検討報告書（https://www.pmda.go.jp/files/000148495.pdf）

- 自然言語処理技術による副作用関係抽出[61]
  用語抽出，関係抽出，表記ゆれ解消，辞書拡張など複数の異なる言語処理の要素技術を統合して医薬品における副作用出現状況の調査を支援するシステム．退院時サマリーから副作用に関して記述されている箇所を特定し，医薬品や副作用症状ごとに集計できる．
- 日本語病名抽出システム MedEX/J[62]及び患者症状抽出器 AEX (Adverse Effect eXtractor)[63]
  MedEX/J は科研費[62]を用いて開発された．形態素ではなく，病名用語抽出に特化しており，後処理として，ICD10 への標準化，事実性判定など，臨床的に重要な処理も組まれている．AEX は AMED の事業[63]により作成されたソフトウエア．患者が記述した文章から症状表現を抽出し，病名コード（ICD コードまたは，MedDRA）を付与する
- Concept Encoder
  Concept Encoder は，ヘルスケア向上に向け，これまでに蓄積された医療・ヘルスケア関連ビッグデータと，この先追加され続けるデータを，自然文で記述された部分も含めて，エビデンスに基づいて有効に解析・活用することができる AI とされれている．
  https://www.fronteo-healthcare.com/conceptencoder
- TXP Medical 株式会社，きりんカルテ DX
  医学用語変換技術を用いて表記が統一されていない病名，薬剤名を正規化・標準化．SOAP（subjective, objective, assess- ment, plan）で記載されたカルテの記述を構造化する．
  https://txpmedical.jp/service/index.html
  https://xirapha.jp/news201809030001/
- Innovation in Pharmacovigilance: Use of Artificial Intelligence in Adverse Event Case Processing[64]

---

61 FUJI XEROX テクニカルレポート No.21 2012 年 自然言語処理技術による副作用関係抽出
https://www.fujixerox.co.jp/company/technical/tr/2012/t_03.html

62 厚生労働科学研究費補助金（臨床研究等 ICT 基盤構築研究事業），2016 年～2017 年，「カルテ情報の自動構造化システムと疾患数理モデルの逐次的構築，及び，自動構造化機能を有した入力機構の開発」厚生労働科学研究費助成金 臨床研究等 ICT 基盤構築研究事業 医療言語処理グループ MedNLP http://sociocom.jp/~data/2017-MEDEX/index.html

63 国立研究開発法人日本医療研究開発機構（AMED），2016 年～2018 年，医薬品等規制調和・評価研究事業「患者の自覚症状により副作用の早期発見を可能とする方策に関する研究」厚生労働科学研究費助成金 臨床研究等 ICT 基盤構築研究事業 医療言語処理グループ MedNLP http://sociocom.jp/~data/2017-AEX/index.html#abst

64 Schmider J, Kumar K, LaForest C, Swankoski B, Naim K, Caubel PM. Clin Pharmacol Ther. 2018 Oct 10

ファイザー社が自社の安全性データベースに登録されているナラティブソースから有害事象名，被疑薬，併用薬，患者生年（年齢）・性別，報告者などの抽出を 3 社のベンダーの深層学習を含む機械学習で比較した報告

- DRAR-CPI[65,66]
  薬剤の分子構造と結合可能なヒトタンパク質の代表的な組合せのデータベースを用いて，化合物の分子構造データから相互作用する傾向があるオフターゲット候補が提案され，入力された分子とライブラリ薬との間の正または負の関連性スコアが計算されるサービス．

- Amazon Comprehend Medical
  深層学習を含む機械学習(ML)を用いて高精度で医療情報を抽出する，HIPAA の対象となっている新サービス [67]．これは大量の構造化されていない医療テキストを処理するためのコスト，時間，手間が低減される．薬，診断，投与量のような実体や関係を抽出することができ，また PHI(保護医療情報)も抽出することがでる．Amazon Comprehend Medical を利用することで，エンドユーザは解析が困難な為ほとんど利用されることのない生の臨床メモから価値を得ることが可能になる．これらのメモから情報を抽出して電子カルテ(EHR)や治験管理システム(CTMS)のような他の医療システムと統合できる．
  https://aws.amazon.com/jp/blogs/news/extract-and-visualize-clinical-entities-using-amazon-comprehend-medical/
  https://aws.amazon.com/jp/blogs/news/amazon-comprehend-medical-jp/

65 Luo H, Chen J, Shi L, Mikailov M, Zhu H, Wang K, He L, Yang L . DRAR-CPI: a server for identifying drug repositioning potential and adverse drug reactions via the chemical-protein interactome.Nucleic Acids Res. 2011 Jul;39(Web Server issue):W492-8.

66 DRAR-CPI, a server for predicting Drug Repositioning and Adverse Reaction via Chemical-Protein Interactome http://202.120.31.160/drar/?page=home

67 HIPAA は米国の保険医療情報の電子化の推進とプライバシー保護やセキュリティ確保について定めた法律であり，Amazon Comprehend Medical はその対象となるデータを扱っているという意味

（ウ）感情認識を用いた ePRO

① 何をする AI か？：

スマートフォンを用いた ePRO に患者の感情情報，認知機能情報を付加する．何らかのスコアを用いて医師あるいは患者自信による気分や精神状態の評価は情報の棄損が少なくなく，変化の検出に不十分な場合が少なくない．表情と音声を用いて計測することでスコアに現れない感情情報，認知機能情報を得ることが可能になる．

② 特徴：

設問に患者が音声により答えることで，回答の内容，音声及び画像を総合して扱うマルチモーダル学習により患者の感情，回答の正確さ（虚偽）を数値化する．感情認識は 27 種類の基本感情[68]（音声は 24[69]）のベクトルとすることで，この組み合わせによる 2000 種対以上の派生した感情を測定できるため，ePRO の目的に合った感情評価が可能である．

③ 用いる AI の技術：

自然言語処理，音声による感情認識，動画中の顔の抽出（Facial Landmark Detection），画像による微細表情読取り（Improved Dense Trajectory），マルチモーダル学習，画像（顔の動画）による脈拍数測定[70]

④ 学習

ePRO プラットフォームとして学習・モデル構築を行う．学習及び動作は**図 7-4** のようになる．音声からテキスト変換及び質問に対する回答の抽出は個別の ePRO 毎に追加学習することで精度が高められる．

- 表情：FACS[71]の有資格者によりアノテーションを行ったビデオデータを用いて教師あり学習を行う．既に開発されている表情認識モデルを用いて転移学習を行う．
- 音声からテキストデータへの変換：質問と想定される答えのナレッジデータと既に種々開発されている音声認識モデルを利用する．

---

68 Cowen AS, Keltner D, Self-report captures 27 distinct categories of emotion bridged by continuous gradients. Proc Natl Acad Sci U S A. 2017 Sep 19;114(38):E7900-E7909

69 Cowen AS, Elfenbein HA, Laukka P, Keltner D, Mapping 24 emotions conveyed by brief human vocalization. Am Psychol. 2018 Dec 20.

70 非接触の脈波検出技術を用いてスマートフォンのカメラで脈拍数を推定するアプリが複数提供されており，新たな開発は必要ない．

71顔の筋肉を 40 以上のユニットに分類してそれぞれの動きの大きさで表情を定義して感情と表情筋の連動性を解析することで表情から感情を読み取る技術（FACS）を利用する。

・ 音声から感情認識：開発されている声の抑揚，強さ，言葉の間隔などから喜怒哀楽，不安，迷い等の感情を識別する音声感情認識モデルとヒトによりアノテーション付けした音声データを用いて教師あり学習を行う．既に開発されている音声感情認識モデルを用いて転移学習を行う．

・ 画像，テキスト，音声の各モデルの統合
複数のデータソース各々のディープニューラルネットワークの深い層（最下層）を共有するマルチモーダル学習を行う．

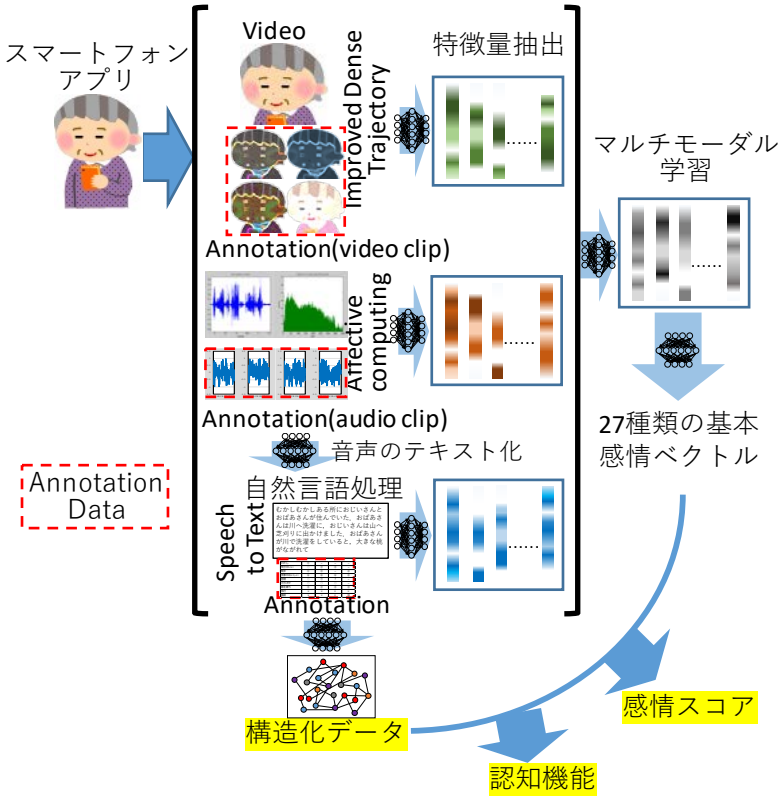


図 7-4 学習及び動作イメージ

⑤ 開発されるシステムの動作イメージ

ePRO の質問を作成してシステムに登録し，対象となる患者にアプリを配布する．患者はアプリの質問に音声で答えることで，質問に対する回答がテキストで画面に表示され，確認できる．ePRO システム内では感情認識により，患者の答えの正しさ（例えば，服薬状況，運動量など回答がアップコーディングなど歪められていないか），感情（落ち込んでいないかなど）が評価され，回答と共に記録される．

⑥ 残る課題など

表情のアノテーション（教師データ作成）には FACS の有資格者が必要であるが，多量の教師データを作成するため，その確保が困難である．

⑦ AI 技術がすでに用いられている又は類似のサービス：

・ Systems for speech-based assessment of a patient's state-of-mind および Speech-Based Automated Cognitive Status Assessment
米国 SRI International 社が開発した AI 技術．前者は会話（スピーチ）の音声から精神状態のメンタルヘルス判別モジュールと鬱病や心的外傷後ストレス障害（ＰＴＳＤ）のような特定の疾患に苦しんでいる可能性を判別するモジュールからなる．後者は Story Retelling，絵の説明など認知機能テストを AI で行う．
https://www.sri.com/

・ Deception Analysis and Reasoning Engine（DARE）
アメリカの Maryland 大学と Dartmouth 大学の研究者が開発．顔の微細な表情（眉や口角の動きなど）から対象者が嘘をついているかどうかを判定する．嘘を見抜く確率（AUC）は 90%程度とされている．
https://doubaibai.github.io/DARE/
https://github.com/Doubaibai/DARE

・ 声ダケノ感情認識テスト Empath
音声等の物理的な特徴量から気分の状態を独自のアルゴリズムで判定するため，言語を問わず喜怒哀楽や気分の浮き沈みを判定できる．https://webempath.net/lp-jpn/

・ PST（Psychoanalysis System Technology:医療用音声精神分析技術）
科学検証できる高い性能で，人間の主観による感情認識の限界に迫った記録を持ち，国立研究所と共同で行った fMRI の実験において，脳の情動活動と ST[72]の分析結果が連動していることが確認されている（$p<0.01$）．また，金融機関と共同で行った利用者の行動予測分析においては，無作為に選択された電話音声から，8 割以上の精度で発話者の行動を予測することに成功している．
https://www.agi-web.co.jp/technology/

・ フェイスリーダー
自動的に 7 つの基本的な表情（喜び・悲しみ・嫌悪・怒り・恐怖・驚き．軽蔑）を分析し，また，注視方向，頭部の向き，人物の特徴も計算される．20 のアクションユニットも加えられている．消費者行動研究，ユーザビリティ研究，心理学，教育研究，市場調査など，500 以上の大学，研究機関および世界の多くの企業で使用されているとされている．
http://www.sophia-scientific.co.jp/human/products/software/facereader/

72 ST（Sensibility Technology）は声から感情を認識する技術

（エ）動画中の顔を匿名化

① 何をするAIか？：

スマートフォンなどで撮影した動画中の顔を微細な表情を維持したまま架空の顔（平均顔）に置き換えて匿名化することで，取得する情報に顔画像を含まず，表情の画像（動画）を非個人情報として扱うことが可能になる．また，情報の取得（調査）に対する同意を得やすくなる．対象者の顔の表情を動画として記録して利用したいが，それが誰であるかはわかる必要がない，個人情報として扱いたくない場合などに利用可能である．

② 特徴：

動画中の顔の置き換えはフェイクニュースの作成などで用いられているAIである．ePROで患者の顔の動画を扱うことに対する患者の抵抗感を軽減できる．また，電磁的に記録された顔データは個人情報の保護に関する法律（個情法）の個人識別符号に当たるため，取扱いに制限があるが，画像を保存して利用したい場合等に用いることで個情法の規制を受けないデータに変換できる．加えて，異なる患者を同じ平均顔に置き換えることで，表情の比較などが行いやすくなる．反対に，感情認識により抽出された感情を標準顔に再現して比較することで感情認識の結果を視覚的に確認できる．

③ 用いるAIの技術

動画中の顔の抽出（Facial Landmark Detection），顔の入れ替え（Face Swap），条件付き敵対的生成ネットワーク（cGAN：Conditional Generative Adversarial Networks）

④ 学習と開発されるAIの動作イメージ

・ 置き換えの学習

顔を置き換えるには，まず画像中の顔の位置を識別し，顔の中の目，鼻，口などの位置を特定する必要がある．まず，ビデオをフレームに分けてFacial Landmark Detectionにより顔の位置を抽出し，Delaunay Triangulationにより得られる顔のパーツの配置情報から顔の大きさ，向きの情報を得る．また，表情を維持して顔を置き換えるには，顔の特徴と表情の情報を分離する必要がある．これには画像の情報を圧縮して復元するAutoencoder（自己符号化器 **図 7-5**）[73]という手法を用いる教師なし学習を用いることができる．学習は，generator（生成器）ネットワークにより元画

73 元画像をそのまま復元するのではなく，新しい画像を作るため，Variational Autoencoderと呼ばれる．

像の情報を encode（削減・圧縮）し，decode（復元）した画像を discriminator（判別器）ネットワークを用いて真偽を判定することにより行う．このニューラルネットワークには「笑っている男性－男性＋女性＝笑っている女性」というように画像の加減算を行える GAN（Generative Adversarial Networks，敵対的生成ネットワーク）から派生した手法が用いられる．Autoencoder に同じ顔の様々な表情の画像を用いて学習させることで，表情のない平均的顔は特徴表現モデルとなり，表情はノイズとして分離される．そして特徴表現モデルを入れ替えることで他人の顔（平均顔）を出力できる．画像の入れ替えはエンターテイメント分野，フェイクニュースなどで多く用いられており，Swap 処理（顔の入れ替え）はオープンソースで提供されているライブラリを利用すれば学習は不要にすることも可能である．また，画像に性別，年齢，人種といったラベルを付けて discriminator に用いることで学習効率が良くなる．加えて，作業量は多くなるが，「(ウ)感情認識を用いた ePRO」の表情からの感情抽出モデルと感情情報毎の平均顔を用いることでより自然な表情を期待できる．

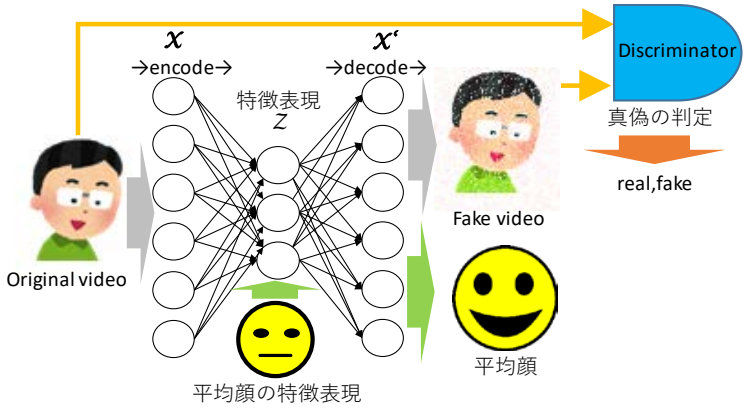


図 7-5 Autoencoder と GAN の概念

・ 平均顔の作成：

置き換え後の平均顔のデータを作成する．平均顔は，複数の顔の平均（目・鼻・口の間隔，大きさ，フェイスラインなど）から作成できる．この場合も Autoencoder を用いることができる．また，微細な表情を含めて置き換えるため，平均顔にも様々な表情が必要であるため，複数（例えば 20 人）の様々な表情を含むビデオ画像から感情認識により抽出した感情ベクトルと対応する静止画像を切り出し，感情ベクトルが似た画像をモーフィング処理して平均顔と感情ベクトル（平均）のデータを作成する．モーフィングを行う場合，顔の各パーツの輪郭の特定（抽出）が甘いと出来上がった平均化の各パーツの輪郭がぼやけるため，輪郭抽出は AI と人手を併用する必要がある．顔全体で平均顔のデータセットを作成すると多くのデータ作成が必要になるため（2000 パターン以上），目，口など部位ごとに作成して合成することも可能であ

る．平均顔は男女性別・年齢を問わず共通で作成することも，作業量が大きくなるが，男女別，世代別に作成することもできる，

⑤ 課題

・ オリジナルの顔と平均顔の差が大きいと精度が低下し，置き換え後の顔がぼやけたり，おかしな顔になりやすい．この対策には，性別，年代，人種別の平均顔を用意する．

・ 加工する対象の顔を用いた学習が 1 度は必要であるため，初めての撮影で同時に加工してオリジナルの動画を破棄して加工済動画のみ残すことはできない．

⑥ AI 技術がすでに用いられている又は類似のサービス：

・ OpenCV（Open Source Computer Vision Library）
コンピューターで画像や動画を処理するのに必要なさまざま機能が実装されたオープンソースのライブラリ．BSD ライセンスで配布されているため学術用途だけでなく商用目的でも利用可能．また，マルチプラットフォーム対応されているため，様々な環境で利用可能である．
https://opencv.org/

・ Deep Video
スタンフォード大学の Michael Zollhöfer 准教授らが開発し公開した，音声と口の動きを他の動画にインポートできる AI．顔の向きの変化・動きにも対応．
https://web.stanford.edu/~zollhoef/papers/SG2018_DeepVideo/page.html

・ Xpression
EmbodyMe 社が開発したスマートフォンのフェイクビデオアプリ．動画の中の人の表情を自分の表情に置き換えてリアルタイムで動かせる．
https://company.embodyme.com/ja/news/xpression/

・ StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation．C.Yunjey et al. CVPR, 2018
単一のモデルで複数のドメイン変換を可能とした GAN．単一の generator と discriminator のみを使用して入力イメージを任意の目的ドメインに柔軟に翻訳できる他，様々な学習の工夫が論文には記されている．
https://arxiv.org/abs/1711.09020

・ pgGAN（Progressive Growing of GANs）
NVIDIA 社が開発した GAN．GAN の学習においてオリジナル画像を低解像度（4×4 ピクセルなど）から段階的に高解像度を用いて学習するように，学習の過程を工夫することで 1024×1024 ピクセルという高解像度な画像を生成できる．
https://research.nvidia.com/publication/2017-10_Progressive-Growing-of

（オ）ドラックリポジショニングへの利用

① 何をするAIか？：

疾患，薬剤のデータベースまたは情報ネットワークから疾患のリスク，因子を分析・評価し，ドラックリポジショニングへ活用する．新薬開発は長期の期間と多額の資金を要するにも関わらず，成功確率はきわめて低く，特に近年，新薬の創出は困難になっている．そうしたなか，既存医薬品から新たな薬効を有する部分構造（シーズ）の探索を含むドラッグリポジショニングは，臨床レベルにおける安全性と体内動態に関する情報が得られているまたは推定が容易であることから，「開発コストの大幅な削減」，「開発期間の短縮化」という利点をもつ一方，膨大なデータの解析と探索が必要であり，シーズから派生する化合物を創出するためのスクリーニングは難易度が高く，個々の研究者等の知識や経験値による能力で行われている．このAIは，疾患および生体内分子ネットワークと薬剤および副作用データベースを連結した情報ネットワークにより，疾患の原因因子，副作用を含む薬剤の作用機序，薬剤による副作用リスクなどを知識ベースで特定および共起性等のデータマイニングによる関連性の情報を深層学習を含む機械学習により推定する．さらに，立体構造などから推定される薬剤とタンパク質の相互作用と情報ネットワークから副作用および未知の作用を予測することにより，ドラックリポジショニングへ活用する．また，情報ネットワークから間接的に，治験において発生したごく少数の副作用事例が偶発的なものなのか，潜在的なリスクとなるのかの情報が得られる場合がある．また，疾患の早期発見やある薬剤の高い有効性・安全性が期待される患者層別バイオマーカーを探すことにも活用できる．

② 特徴：

- ドラックリポジショニングを多面的なデータに基づき発見・評価できる．
- 動物や試料などを用いるウエットな実験を要しない．
- 副作用の発生要因について探索できる[74]．
- 既存医薬品と標的たんぱく質の立体構造などを用いたシミュレーションを行うことで，有効な治療法がない病気や希少疾患に対して有効性が期待できる薬剤を探索できる．

74 例えば，Yuki YAMAGATA，et alia. Ontology-based Knowledge Systemization for Interpreting the Mechanism of Liver Toxicity. The 32nd Annual Conference of the Japanese Society for Artificial Intelligence, 2018

・ 創薬をはじめ，治験，製造販売後調査，副作用調査を問わず対応可能である．

また，情報ネットワークから間接的に，下記 2 点についても効果をもつ．

・ 治験において発生したごく少数の副作用事例が偶発的なものなのか，潜在的なリスクとなるのかを機序のシミュレーションの情報を用いた演繹的な評価が可能となる．
・ 情報ネットワークから間接的に疾患の早期発見やある薬剤の高い有効性／安全性が期待される患者層別バイオマーカーを探すことができる．

③ 用いる AI の技術：

薬（化合物）の構造とその標的および反応を教師データとして深層学習によりモデルを構築し，未知の化合物の作用（毒性等）を予測する RASAR（read-across structure activity relationships）[44]と呼ばれる手法および Structure-Based Drug Design (SBDD)の手法である化合物の立体構造と標的タンパク質の分子動力学シミュレーション，ドッキングシミュレーション，定量的システム薬理学（Quantitative Systems Pharmacology：QSP）がベースである．従来，これらには support vector machines (SVM)，Random Forests (RF)，Bayesian learning などが用いられてきたが，近年，精度の改善が報告されている様々な deep neural networks（DNN）を用いる[46].

薬が作用する未知のタンパク質の同定のために，薬の立体的化学構造からその薬が相互作用するタンパク質を予測する手法（3D-QSAR：3D- Quantitative Structure-activity Relationships），既知の薬理作用情報を用いた手法，既知の遺伝子発現情報を用いた手法，複数の酵素反応を扱う手法（mass action stoichiometric simulation models[47]）などにより，それぞれ深層学習のモデルを作成し，数百万の化合物・タンパク質間相互作用のデータや様々な病気の分子機序のデータを使って予測モデルを学習させる．

疾患間の関係，受容体タンパク質-生体反応パスウェイ等のネットワークの構築においては，知識ベースに加えて，グラフニューラルネットワーク技術やセマンティックモデル化技術などの人工知能技術[48,49,50,51]を実装することで，多岐にわたるデータを収集し，疾患に関連する因子を判断する技術が利用可能である．

④ 学習

表 7-1 にあるような既知の薬剤－受容体・酵素－疾患（生体反応，副作用）の関係（薬剤の標的タンパク質，適応疾患など）を教師ありデータとして学習を行う．薬剤分子の量子化学計算データ，その他実測データとタンパク質のフラグメント分子軌道計算（FMO：Fragment Molecular Orbital）のデータを用いて推定した薬剤分子と受容体や酵素との相互作用の大きさ，結合する薬剤分子の部分構造（リガンド）に対して既知の標的タンパク情報を正解として薬剤－タンパク質相互作用モデルを学習する．また分子構造に共通して存在する部分構造を見出し，疾患関連分子への相互作用から有効性や毒性を予測する．

表 7-1 学習などに用いる各種の情報

ネットワーク名	情報源
薬剤ネットワーク	既知の薬効分類（WHO-ATC），KEGG[75]，3DPSD[76]　など
疾患ネットワーク	FAERS[54]，臨床医学オントロジー[77]　など
生体内分子ネットワーク	FMO データベース [78]，RCSB PDB(蛋白質構造データバンク)[79]，CMap(コネクティビティマップ)[80]，KEGG，GenBank[81]，NCBI[82]など

その他，文献データベースや特許情報プラットフォーム JPlatPat[83]　　など

⑤　動作イメージ

既知又は新規の薬剤について，その薬剤の標的となる分子の構造から，生体内分子ネットワーク内のデータベースにおいてスクリーニングを行うことで，疾患や副作用に関連する分子を見つけ出し，既存薬剤の新たな対象疾患の可能性，新規薬剤の対象疾患の絞り込み，または副作用となりうる事象について探索する（**図 7-6**）．

75 KEGG DRUG Database https://www.genome.jp/kegg/drug/，KEGG PATHWAY Database https://www.genome.jp/kegg/pathway.html
76 5 大学連携プロジェクト 3 次元医薬品構造データベース http://3dpsd.toyaku.ac.jp/3dpsd/
77 臨床医学オントロジー http://lodc.med-ontology.jp/
78 FMO 創薬コンソーシアム http://eniac.scitec.kobe-u.ac.jp/fmodd/members_only/manual/5.html
79 RCSB; Research Collaboratory for Structural Bioinformatics https://www.wwpdb.org/
80 Connectopedia https://clue.io/connectopedia/
81 GenBank Home https://www.ncbi.nlm.nih.gov/genbank/
82 Welcome to NCBI https://www.ncbi.nlm.nih.gov/
83 特許情報プラットフォーム https://www.j-platpat.inpit.go.jp/web/all/top/BTmTopPage

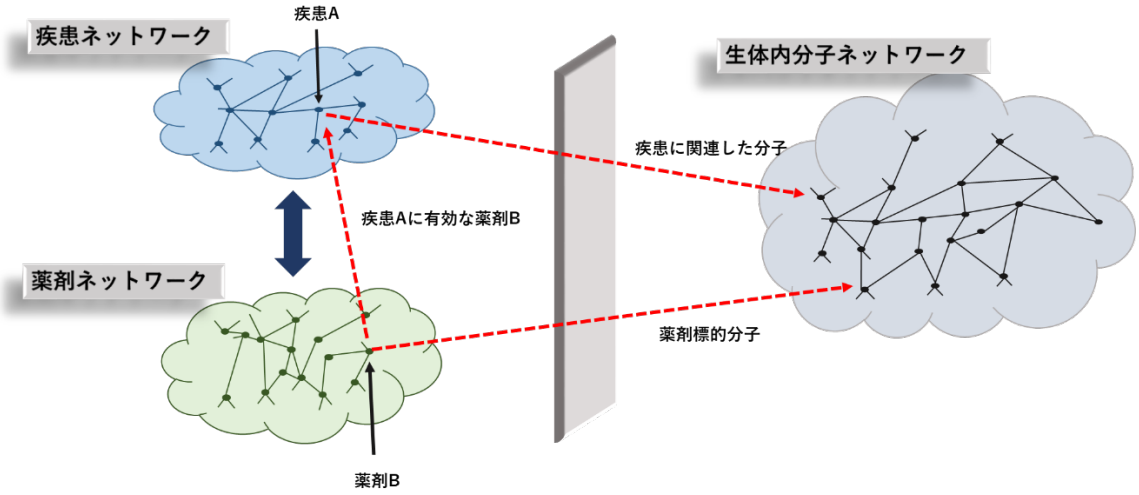


図 7-6 各ネットワークの参照イメージ

具体的には，深層学習を含む機械学習により疾患，薬剤のデータベースまたは情報ネットワークから疾患の治療に転用可能な副作用（例えば血管新生阻害など）を有する複数の既存医薬品を知識ベースで特定し，関連性の強さを機械学習により推定する．これらの情報と医薬品の分子の立体構造から共通の部分構造を見出し，治療目的とする疾患および原因レベルで共通性のある疾患群に関連する生体内分子およびそのネットワーク上の分子と見出した部分構造の相互作用（結合）をシミュレーションにより算出する．

⑥ 課題など

- 同一疾患でも人によって薬の効果が異なったり，発症メカニズムや進行度合いが変わったりするような疾患においては，個人差情報の公開データが少なく，ネットワークの構築が困難である．
- 高分子化合物においてネットワークの構築が困難である．
- SBDD は対象となるタンパク質が硬い場合に限られる．
- 疾患の原因タンパクおよびそのネットワークが不明な場合は予測できない．
- 薬剤およびそのターゲットデータに社内のデータのみを用いる場合，偏ったデータとなる場合があると考えられる．その場合，学習により得られるモデルも偏ったものとなってしまう．
- 精度の高い分子間相互作用の計算にはアニーリングなど組合せ最適化問題を効率よく解くアルゴリズムと演算装置が必要となる．

⑦ AI 技術がすでに用いられている又は類似のサービス：

- Trusting Social Score/ Numerate
  低分子薬に特化したインシリコ創薬プラットフォームを提供．従来型の結晶構造分析からの創薬ではなく，SAR， patents，phenotypic data 等のデータから候補物質を探索
  http://www.numerate.com/platform/
- Watson for Drug Discovery/ IBM
  薬剤標的や既存薬の新たな効能を特定できる
  https://content.mkt.watson-health.ibm.com/wdd-2018-adwords-request-wdd-free-trial.html
- DrugClust[52]/ University of Cambridge
  医薬品の構造，タンパク質相互作用，副作用などのプロファイルと PubChem Compound database，DrugBank，KEGG，Matador などのデータベースを用いてクラスター分析を行い，ベイジアンアプローチを使用して副作用予測のスコアを取得できる．R のプログラム（パッケージ）が公開されている．
  https://cran.r-project.org/web/packages/DrugClust/index.html.
- ドラッグリポジショニングサービス/ Excelra
  SAR 情報のデータベース GOSTAR とバイオマーカーのデータベース GOBIOM を中心に構築，8 つのアプローチからドラッグリポジショニングの基盤を構築
  https://www.excelra.com/discovery/#data-science-driven
- GoSTAR/ Excelra
  Global Online Structure Activity Relationship Database 低分子化合物のデータベース
  https://www.gostardb.com/gostar/
- myPresto/次世代天然物化学技術研究組合
  医薬品開発支援のために作成された分子シミュレーション計算のプログラム集
  https://www.mypresto5.jp/
- D-iOrgans/ Karydo TherapeutiX
  疾患モデルにおけるラットと人の全身網羅的多器官遺伝子発現地図を用い，化合物をラットに与え，20 の臓器の遺伝子発現を計測し，ヒトの場合にマッピングを行う．ウエットの実験が必要であるが，化合物のターゲットの特定が難しい副作用などでも高い精度が得られる．
  https://karydo-tx.com/our-technology/d-iorgans/
  https://www.cell.com/iscience/fulltext/S2589-0042(18)30023-3

（カ）ウェアラブルデバイスを用いたデータ収集と行動情報のリンク

① 何をする AI か？：

ウェアラブルデバイスで収集したデータ（血圧，脈拍数，体温のバイタルや，ECG，加速度情報など）の補足情報として，映像や音声情報を元に，対象者がいつどのような体勢をとったか，また，どのような表情であるかをリアルタイムで分析し，ウェアラブルデバイスの収集データと補足情報をリンクさせることにより，収集されたバイタルの意味づけ（血圧が立位，仰臥位のものなのか，脈拍数の上昇が，感情（怒りに伴う興奮）によるものなのか，イベント（転倒等）によるものなのか）を可能にする．

② 特徴：

- 画像や音声情報から AI が体勢，表情を識別する．
- 活動量計等の加速度情報からのみでは推測しにくい体勢情報を画像情報と組み合わせて学習することで推測精度を改善できる．

③ 用いる AI の技術：

動画中の顔の抽出（Facial Landmark Detection），加速度情報からの体勢推定，動画から対象の体勢の推定，画像による微細表情読取り（Improved Dense Trajectory）．

④ 学習：

- 動画情報から体勢の推定：
  公開されている 3D ResNet[84]あるいは OpenPose[85]の学習済みモデルを用いて画像情報から体勢を推定できるため，特に学習は不要である．
- 加速度情報からの体勢の推定：
  体勢を推定する機能がある多軸の加速度センサーを組み込まれた活動量計等のウェアラブルデバイスを用いることで，学習を行う必要はない．しかし，動画情報を用いる場合に比べて精度は低下するため，事後的に動画により推定された体勢情報を教師デ

84 GitHub 上で公開されている https://github.com/kenshohara/3D-ResNets-PyTorch
Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet, In CVPR, 2018
Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Learning Spatio-Temporal Features with 3D Residual Networks for Action Recognition, In CVPR, 2017

85 GitHub 上で公開されている
https://github.com/CMU-Perceptual-Computing-Lab/openpose
Zhe Cao and Tomas Simon and Shih-En Wei and Yaser Sheikh. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. In CVPR, 2017.

ータとして学習させることで精度の向上が期待できる．学習方法はまず，大勢の対象者屋内記録データの画像と加速度情報を元にディープニューラルネットワークを用いて加速度センサーから体勢を推定する学習をさせたのち，個人のデータを用いて出力層と出力層の直前の隠れ層のみを学習の対象にする（他の層の重みは大勢での学習結果で固定）転移学習を行うことで個人ごとに最適化した学習済みモデルを構築することでセンサーのみでの計測精度を向上できる．

・ 動画情報から表情の推定：
Amazon や Microsoft 社が提供している API[86]を用いて，動画情報から顔の認識，表情の推定が可能．対象者とその顔の位置を抽出する AI と併用することで，本書の 7.（ウ）で用いる感情認識 AI を流用することも可能．

⑤ 動作イメージ

図 7-7 のように，ウェアラブルデバイスで収集可能な情報（体温，脈拍数，血圧，加速度）に加え，屋内に設置されたビデオカメラで収集された動画情報から，表情や姿勢情報を収集する．

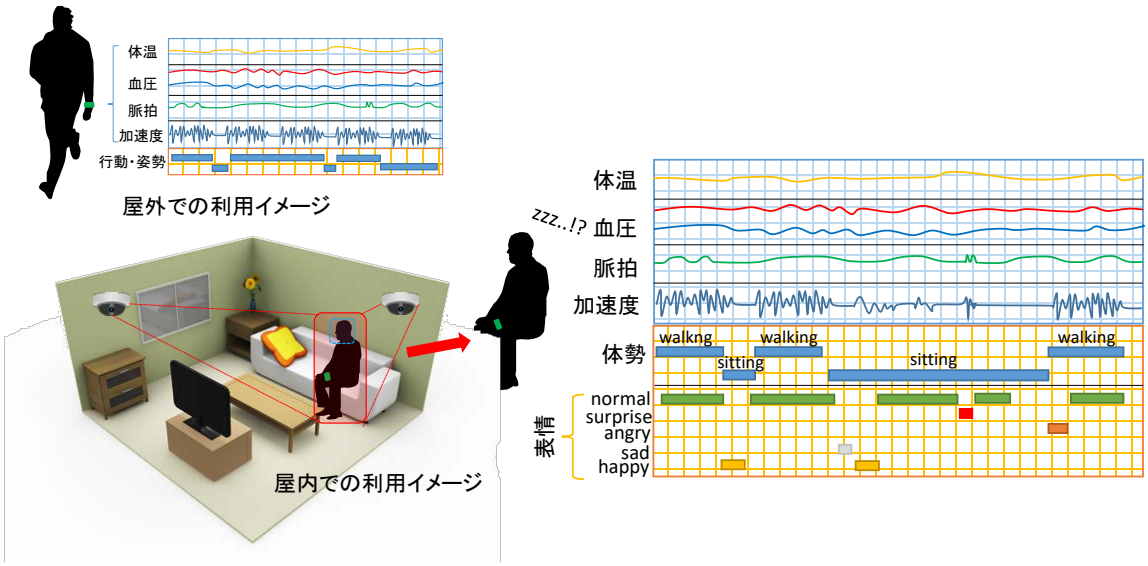


図 7-7 各種データの取得イメージ

86 Amazon Rekognition https://aws.amazon.com/jp/rekognition/
Microsoft Azure Face API https://azure.microsoft.com/ja-jp/services/cognitive-services/face/

屋内：

ビデオカメラで収集された動画情報を元に，被験者の体勢（立っている，座っている，横たわっている），表情（通常状態，悲しい，驚き，怒り，楽しい等）を分析・コード化し，ウェアラブルデバイスから得られた情報と同期させ，ウェアラブルデバイスで収集されたバイタルデータの変動の意味づけができるような時系列データとして保管する．

屋外：

ウェアラブルデバイスで収集された加速度情報から体勢（立っている，座っている，横たわっている）を分析・コード化し，ウェアラブルデバイスから得られた情報と同期させ，ウェアラブルデバイスで収集されたバイタルデータの変動の意味づけができるような時系列データとして保管する．

⑥　課題など

動画情報が収集できない屋外では，体勢の特定は加速度情報からのみ推定される．ウェアラブルデバイスで収集された加速度情報からの体勢推定の質は向上してきているものの，屋内で収集可能な動画情報からの補足情報に比べて推定精度が落ちることが予想される．また，加速度情報を収集するための加速度センサーで厳密に測定するには，手，足，腰，頭部に計 4 か所センサーが必要となるが，被験者への負荷を考えると，常時これらのセンサーを全て着用する事は難しい．

この対策として，「④学習」の「加速度情報からの動作・姿勢の推定」に記載した大勢の対象者の手または腰に装着した加速度センサーデータと，動画情報を用いた学習済みモデルから，被験者データを使った転移学習を用いる．

⑦　AI 技術がすでに用いられている又は類似のサービス：

・　Observer XT
画像情報から対象者の行動をコード[87]化し，行動データとウェアラブルデバイスのような外部データとを同期させることができる．
http://www.sophia-scientific.co.jp/human/products/software/the-observer-xt/

・　フェイスリーダー
自動的に 7 つの基本的な表情（喜び・悲しみ・嫌悪・怒り・恐怖・驚き．軽蔑）を分

87 https://patents.google.com/patent/US20120265104A1/en
行動のコード化は OWAS (the Ovako Working Analyzing System)，RULA (the Rapid Upper Limb Assessment)，REBA (the Rapid Entire Body Assessment)などの人間工学の分析方法が採用されている．

析し，また，注視方向，頭部の向き，人物の特徴も計算される．20 のアクションユニットも加えられている．消費者行動研究，ユーザビリティ研究，心理学，教育研究，市場調査など，500 以上の大学，研究機関および世界の多くの企業で使用されているとされている．
http://www.sophia-scientific.co.jp/human/products/software/facereader/

- OpenPose
  カーネギーメロン大学で開発された「Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields」の論文[85]で発表した手法が実装されたライブラリ[85]で，動画情報からその人物の姿勢をリアルタイムで分析する事ができる．

- 歩容認証
  人の歩幅，歩行速度，姿勢や，動きの左右非対称性等の情報を元に，個人を特定する認証方法．解像度は頭から足先まで 30 画素あれば，顔が画像に移っていない状況下でも個人を特定できる．
  https://www.jst.go.jp/pr/jst-news/pdf/2018_02/2018_02_p08-11.pdf

- Posture Visualizer
  加速度情報からの姿勢判定プログラム．3 軸加速度計内蔵の活動量計から得られた情報を元に姿勢を判定する．
  https://www.kicnet.co.jp/solutions/biosignal/sleep/posture-visualizer/

(キ) 治験総括報告書の自動生成

① 何をするAIか？：

治験実施計画書（Protocol）や解析計画書（SAP），解析結果等から治験総括報告書（CSR）を自動的に生成する．多くの製薬企業は治験症例のデータ固定からCSRを完成させるまでに統計解析を除いても数ヶ月の時間を要している．また，その作成プロセスの中ではProtocol、SAP、解析結果等など他のドキュメントから多くを書き直しており，作成やその確認にかかる時間は，承認申請を控える製薬企業にとって大きな時間である．AI導入により，このような時間・手間の短縮や，メディカルライティング担当者のリソース不足における作業遅延の解消が期待される．

② 特徴：

- ProtocolやSAPの文書，解析結果及びデータセット等から，AIによってCSRのDraftを作成させる．
- Inputは，Protocol，SAP，Clinical Trial Management System（CTMS），Blank CRF，解析結果，データセットが想定される．
- Protocolからは臨床試験のデザイン，SAPからは統計解析の手法を読み込む．自然言語処理でキーワードを識別し，それを基にCSRを自動生成する．キーワードはマスタ化させることで，より識別が容易かつ正確になることが期待できる．また，CSRのテンプレートを用意し，キーワードとテンプレートのマッピングをパターン化させる．

③ 用いるAIの技術：

CSRのテンプレートを用いて情報（Protocol，SAP，解析結果等）とそれに対応する表現をパターン化して差し込み印刷のように利用する場合はルールベースが大半を占めると考えられ，深層学習を含む機械学習によるいわゆる“AI”を用いる部分は多くはない．

治験結果の影響を受けにくい「5．倫理」から「9．治験の計画」及び「16．付録」は定型的（Protocol等の既存のドキュメントからの引き写しという意味を含む）に扱える場合が少なくないが，要約を行ったり，その組合せは複雑で曖昧さがある場合は機械学習による自然言語処理が必要となる．

治験結果を利用しながら文章作成する必要がある「10．治験対象患者」から「13．考察と全般的結論」は，深層学習を含む機械学習による文章作成が必要になる．ま

ず，訓練データから治験薬や病名，効果指標名，統計解析結果の数値などのなどの固有表現を特定してタグ付けを行い，その前後の文章のつながりを学習し，文書生成モデルを構築する．

患者背景や安全性等，試験薬や対象疾患に依らず共通化できる箇所については汎用化が可能である．一方，有効性評価項目等の試験固有の箇所については，より多様な文章に対応できる深層学習モデルが必要となる．いずれの学習モデルも過去の CSR 等を学習させて構築するが，試験固有の多様な文書となる場合はより多くの多様な訓練データに試験デザインや帰無仮説の多重性等のラベルを用いた条件付きの深層学習を行うなど工夫が必要となる．

CSR の文章は複数の一連の文が前の文を受け継ぎながら続くため，AI にも前の文を記憶して引き継ぐ仕組みが必要となるため例えば LSTM（Long short-term memory）などを用いる．

また，日本語の場合は，まず単語を切り出す形態素解析が必要になり煩雑であるので，学習及び出力は英語により行い，日本語が必要な場合は AI を用いた翻訳ソフトを移転学習により精度を高めて用いる．

CSR の各章に対して必要と考えられる AI 技術を，**表 7-2** に示す．

表 7-2 CSR の各章におけるデータソースと用いる技術

章	ソース（Input）	用いる技術
1.標題ページ	Protocol, CTMS 等	ルールベース
2.概要	Protocol, SAP, 解析結果等	ルールベース，文書要約，自然言語処理
3.目次	-	ルールベース
4.略号及び用語の定義一覧	Protocol 等	ルールベース，自然言語処理
5.倫理	CTMS, データセット等	ルールベース
6.治験責任医師等及び治験管理組織	CTMS 等	ルールベース
7.緒言	Protocol 等	ルールベース
8.治験の目的	Protocol 等	ルールベース
9.治験の計画	Protocol, CTMS 等	ルールベース，自然言語処理

章	ソース（Input）	用いる技術
１０.治験対象患者	Protocol, SAP, CTMS, 解析結果, データセット等	ルールベース，文書要約，自然言語処理
１１.有効性の評価	Protocol, SAP, CTMS, 解析結果, データセット等	ルールベース，自然言語処理
１２．安全性の評価	Protocol, SAP, 解析結果, データセット等	ルールベース，自然言語処理
１３.考察と全般的結論	Protocol, SAP, 解析結果等	自然言語処理
１４.本文中には含めないが，引用する表，図及びグラフ	解析結果, データセット等	ルールベース
１５.引用文献の一覧表	-	ルールベース，自然言語処理
１６.付録	Protocol, BlankCRF, CTMS, 解析結果, データセット等	ルールベース

④　学習

出力物の精度をあげるために，過去の CSR を学習させる際は，ラベルとして CSR のパート（章及び段落）の情報，治験薬名（記号），対象疾患，試験デザイン（試験の相，種類，比較群の数，統計手法，指標（オッズ比，ハザード比等）等）を与えて，Protocol，SAP，解析結果等の CSR の元となるドキュメントやデータと共に読み込ませて CSR の文章中の固有表現のタグ付けを行い，ルール（ナレッジ）ベースと深層学習（LSTM-RNN：Long short-term memory - Recurrent Neural Networks, cGAN：Conditional Generative Adversarial Network，cVAE：Conditional Variational Autoencoder 等）を組み合わせて臨床試験結果など，CSR の本文を自動作成する学習モデルを構築する.

十分な精度を持つ学習モデルを構築するためには多くの CSR 等のデータが必要となるが，１社で全てを用意するのは困難であろう．訓練データ不足を補う１つの方法は 3.3.5.2 で説明している１つの画像データを加工してデータを増幅する手法の適用が考えられる．つまり，Protocol，SAP，CSR の治験薬名，対象疾患名，解析結果の数値などを様々に入れ替えたダミーの訓練データを作成することで，治験デザインなどに

よりパターン化できる部分と治験ごとに異なる部分を効率よく学習できるであろう．そのほかの方法として，CSR ではないが，文章が似ており，多量に入手が可能な臨床試験の論文と臨床試験登録システムデータを用いて深層学習を含む機械学習を行い，これをもとに転移学習を行うことで，比較的少ない治験数のデータから効率よく学習が可能となることが期待される．また，更に，学習方法においても訓練データを効率よく利用する手法（boosting など）を用いることで，精度を向上させる．

用いる深層学習の方法で特徴的な LSTM のイメージを**図 7-8** に示す．このなかで tanh は tanh 関数[88]を，σ はシグモイド関数を活性化関数に用いる単一ニューロンである．LSTM-RNN は**図 7-8** の薄緑色のブロック（LSTM ブロック）を**図 7-9** のように RNN の内部に複数数珠繋ぎにして用いられる．**図 7-9** は 5 つの単語（w_1 ～w_5）のつながりを入力として Embedding 層で埋め込みベクトル（w_t）に変換し，LSTM 層で出力ベクトル（y_t）と新しい内部状態ベクトル（h_t）を生成する．y_tから全結合層で softmax[89]活性化関数を用いて次の単語w_{t+1} の確率を推定する．実際にはこれだけではなく，Protocol，SAP，解析結果などの固有データを与えて文章を作成するため，固有データを**図 7-9** のw_1 に与えたり，下層で埋め込んだり，固有データの前後の文章を**図 7-9** により生成させる，**図 7-9** の図で言えば上側に$w_6 \rightarrow w_2$という逆向きの処理を追加する BidirectionalRNN（双方向 RNN）を用いるなど，より複雑なモデルが必要である．

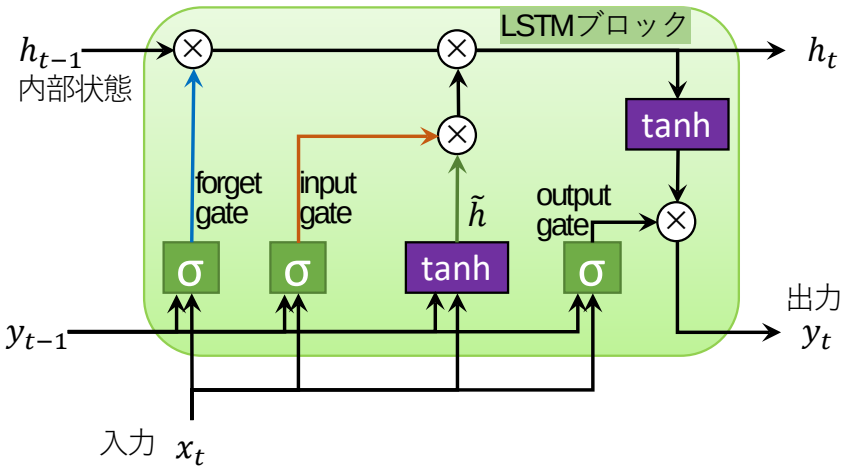


図 7-8 LSTM の計算イメージ

88 tanh 関数：双曲線正接関数, Hyperbolic Tangent Function

89 3 分類以上の各々の確率を出力する関数.

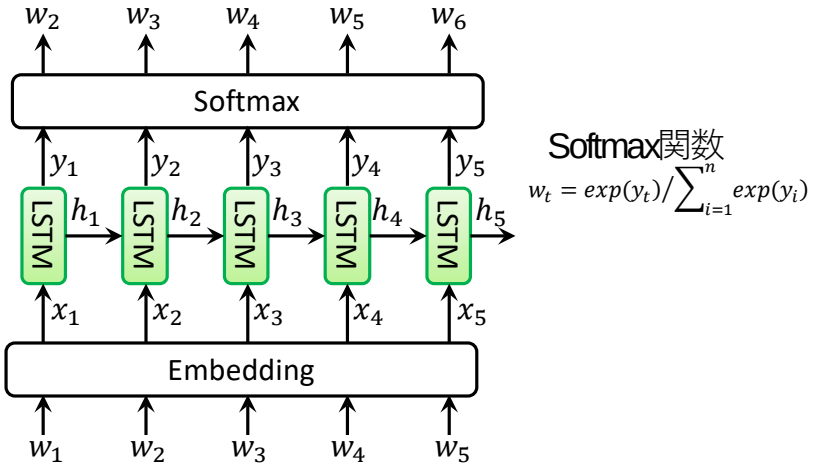


図 7-9 LSTM-RNN による言語モデルネットワーク

⑤ 動作イメージ

図 7-10 のように，INPUT として，Protocol や SAP 等の文書と，解析結果やデータセットの“データ”と共にデータに意味付けするアノテーションを準備し，AI にて CSR のテンプレートに必要事項を差し込むことで，OUTPUT として治験総括報告書を作成する．どの程度 AI に Draft させるかによって，INPUT は適宜増減すると考えられる．

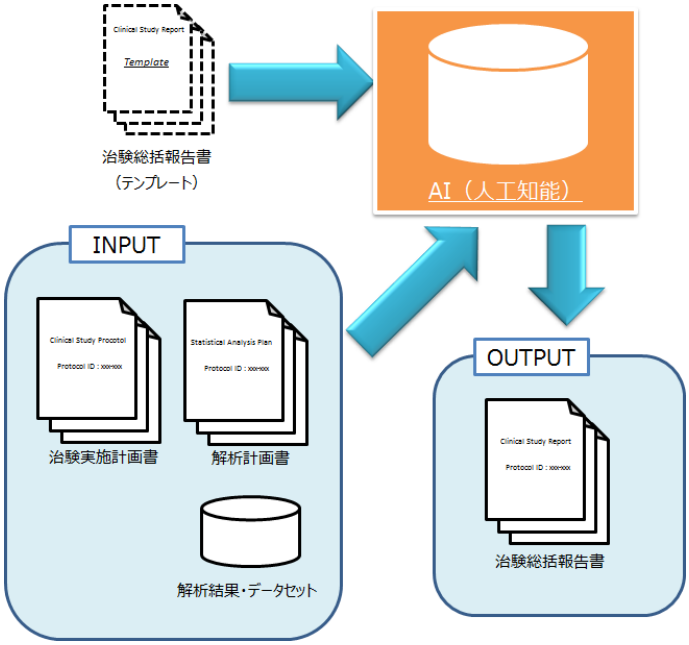


図 7-10 INPUT と OUTPUT のイメージ

⑥　課題など

大きな課題の一つとして，十分な精度を持つ学習モデルを構築するために必要な訓練データ量（CSR 等の数）の確保が挙げられる．また，訓練データに含まれていない新しい試験デザインや新規疾患分野の場合は精度が低下するため，利用する際には注意が必要である．複雑な試験デザイン等，ルールベース（テンプレートへの差込み）で対処できる箇所が少ない場合，深層学習を含む機械学習による対応が増える．そのため，開発パイプラインが多岐にわたる場合も，ルールベースで対応できるように Protocol（例えば，CeSHarP[90]）や SAP も標準化すべきである．

GxP 等のレギュレーションの下では，データを取り扱う上で，バリデーションの問題を切り離すことはできないため，あくまで作成支援という位置づけになると考えられる．特に試験固有な場合が多い有効性の評価などは出力される文章の品質が低下しやすいが，重要な部分であるため，AI を用いないことも選択肢となる．いずれにしても，どこまでを AI の技術を用いるか，何を人間の目で QC すべきか，各社で考えやポリシーの違いから対応が異なることが想定される．

⑦　AI 技術がすでに用いられている又は類似のサービス：

・　Neural Architectures for Named Entity Recognition[53]
Named Entity Recognition(固有表現抽出)と BidirectionalRNN を用いて大規模な辞書や多量の教師付きデータを用いずに効率よく文書を生成する方法．GitHub に論文著者によるプログラムが公開されている．

https://github.com/glample/tagger

https://sites.google.com/site/ermasoftware/getting-started/ne-tagging-conll2003-data

・　完全自動決算サマリー 日本経済新聞社
日本経済新聞社が株式会社 言語理解研究所の動文書要約・生成エンジン P-laei を用いて開発した決算公告の PDF ファイルから公表後すぐに AI が売上や利益などの数字とその背景などの要点をまとめた「決算サマリー」文書を生成して配信されている．配信するまでは完全に自動化し，人によるチェックや修正などは一切行なわれていない．

https://pr.nikkei.com/qreports-ai/

90 CeSHarP：Clinical electronic Structured Harmonized Protocol，ICH M11 臨床試験プロトコル文書のフォーマットの調和と電子化で Protocol を電子的に扱えるよう構造化の規格策定が進められている，

・ 文章要約サービス

文章の要約は文章中から LexRank[54]などによる重要文（センテンス）を抽出し，文の短縮を行う抽出型の方法と，TensorFlow textsum[55]など，文章の意味を汲み取ったうえで適切な要約を作成する抽象化型の手法がある．前者を用いて，徳島県では定例記者会見，審議会の議事録を AI による自動要約を提供するサービス「徳島発！「AI 要約サービス」」[#1]が開始されている．これ以外にも WEB 上で入力した文章を要約するサービスなどが提供されている[#2]．

#1：https://tokushima.smartshoki.jp/
#2：https://a3rt.recruit-tech.co.jp/product/TextSummarizationAPI/, https://text-summary.userlocal.jp/，https://www.paper-digest.com/

・ Articoolo

イスラエルの Articoolo Inc.が開発した AI による自動記事作成ツール．自然言語処理の方法論で，文章を書く人間の考え方を模範し，ゼロから文章を作り上げるアルゴリズムになっている．いくつかのキーワードだけで文章を作り出すことができる．また，日本語対応のため，日本の企業も様々な場面で活用できる．

https://articoolo.com/

・ Wordsmith

Automated Insights, Inc.が開発したデータを文章に変換するツール．様々なデータを分析してその特徴から自然言語処理で，文章を生成する．データの性質からデータと正しく理解して文章が生成されているかを人間が判断する必要もあるが，人間が記載したかのような文章が作成できる．Wordsmith も Articoolo と同様に日本語対応である．

https://automatedinsights.com/wordsmith/

・ Quill

Narrative Science が開発したデータから自然言語を自動生成する Automated Narrative Generation Platform．データを入力すると AI により自動的にそのデータを分析し，データから読み取れる内容から文章を生成してレポートを出力する．レポートの内容はそれぞれのデータの増加率や全体に対する割合，推奨するアクションなど様々なものをデータの特徴に合わせて用いられ，人間が記載したかのように特徴のある文章が作成できる．

https://narrativescience.com/products/quill

本章で取り上げた 7 つの例のうち，(ア)，(ウ)，(エ)，(カ)は実用化されている既存の学習済モデルの転移学習など，既存技術の流用により，(イ)，(キ)は既存技術の応用・発展実現可能と考えられるが，(オ)は既存技術では性能が不十分でる，訓練データや計算リソースの不足など実現までの課題は少なくない．

臨床開発周辺を中心に例示したが，機械学習，特に深層学習は汎用性の高い技術であり，これら以外にも多くの可能性が考えられる．また，実際に AI を取り入れたい業務・ビジネスや解決したい課題がある場合，まず利用可能な技術や，技術・経験のあるベンダーの情報を得ることから始めることになる．インターネットで検索できるが，関係のないノイズも多いので，特許申請情報[51]の検索は比較的ノイズも少なく有用である．

おわりに

AI とはなにか？ その概念，歴史から学習に用いるデータの準備，基本的な手法とアルゴリズム，実装の実例，AI のシステム等の環境，教育ならびに AI を用いる事例のアイデアまで紹介した．意外に簡単に感じられた方もあれば，やはり難しく感じられた方もあるかもしれない．本書を作成したタスクフォースメンバーも手探りで調べ，確認しながら書いた部分が少なくないため，十分にかみ砕けていない説明があったかもしれない．“もう少し知りたい”，あるいは“具体的にサービスを導入したい”という方にとっては，**6 章**あるいは各事例の「AI 技術がすでに用いられている又は類似のサービス等」が手掛かりになると思われる．

日本製薬工業協会 医薬品評価委員会 データサイエンス部会タスクフォース 6 のアンケート調査から，多くの会社で既に AI を導入済あるいは導入を検討していることが示唆される[45]．また，導入済あるいは導入を予定している AI の機能も非常に多岐にわたっており，製薬業界における AI への期待は非常に大きいことが想定される．その一方，AI を過信して「どんな問題であっても機械学習や深層学習の複雑な AI さえ使っておけば解決できる」と短絡的に考えるのもまた問題である．**1.3.2** 項で述べた通り，深層学習を含む機械学習はルールベースではなく，データに潜む経験や暗黙知を機械学習のプログラム自身が学習して分類や予測のためのモデルを構築する仕組みである．逆に言えば，暗黙知などがない明示的なルールで構築可能なモデルであれば，機械学習を用いる必要はないし用いるべきではない．例えば，単純な作業のみで構成されたデータ処理作業を人に代わって処理するシステムを開発する場合，人が内容を読んで判断した結果を入力したりデータ加工する部分は多くはないであろう．その場合，ルールベースの AI を利用すれば十分かもしれない．AI を利用する際は，まず目的（解決すべき課題）を明確にし，そもそも目的を達成するためにはどんな方法が最適かを考えることが有効であろう．

AI の開発には大量のデータとデータを理解し意味付けるドメイン知識が必要であり， IT ベンダー単独でサービスを開発・提供することはできない．同様に，製薬企業 1 社で AI を利用しようとしても，利用可能なデータの量や多様性が不足しているがために妥当なモデル性能を得られないかもしれない．あるいは，他業界（例えば診療情報，ゲノムデータ）のデータ利用が必要になることも少なくない．以上を勘案すると，効率的に AI を活用するためには，同業者・他業者含めたコラボレーションが非常に有効な策の一つになると考えられる．まずはニーズやアイデアをオープンに共有するなどして，コラボレーションのチャンスを増やしていってはいかがだろうか．

参考文献

1 人工知能学会．“What’s AI：人口知能の話題”．人工知能学会．https://www.ai-gakkai.or.jp/whatsai/AItopics5.html.

2 総務省．“平成 28 年度 情報通信白書 第 1 部 第 4 章 第 2 節：人工知能（AI）の現状と未来”．総務省．http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h28/pdf/n4200000.pdf.

3 松尾豊．人工知能は人間を超えるか ディープラーニングの先にあるもの．株式会社 KADOKAWA，2015，43-45p, 49-53p, 103-107p, 116-117p, 136-141p, 149p.

4 侍エンジニア塾ブログ編集部．“【5 分でわかる】人工知能とは？概要や種類をわかりやすく解説”．Samurai．http://www.sejuku.net/blog/7290#AI-6.

5 人工知能学会．“What’s AI：人工知能って何？”．人工知能学会．https://www.ai-gakkai.or.jp/whatsai/AIwhats.html.

6 日本ディープラーニング協会（監修）．ディープラーニング G 検定．株式会社 翔泳社，2018，4p, 8-10p, 49p, 67p, 74p, 94-105p, 130p, 36p, 121p, 128p.

7 AI 人工知能テクノロジー．人工知能 AI ブームの歴史 https://newtechnologylifestyle.net/人工知能 AI ブームの歴史

8 総務省 ICT スキル総合習得教材 [コース 3] データ分析 3-5：人工知能と機械学習，5p，18p http://www.soumu.go.jp/ict_skill/pdf/ict_skill_3_5.pdf

9 ディープラーニング（Deep Learning）とは？【入門編】 https://leapmind.io/blog/2017/06/16/ディープラーニング（deep-learning）とは？【入門編】/

10 本橋洋介．人工知能システムのプロジェクトがわかる本．株式会社 翔泳社，2018，70-74p，114-126p.

11 斉藤壮司．“ニュース＆レポート AI のブラックボックス問題解消へ，IBM や富士通が技術開発”．日経 XTECH．https://tech.nikkeibp.co.jp/atcl/nxt/mag/nc/18/020800017/100200137/.

12 齊藤由希．“自動運転技術：AI をブラックボックスにしないために，”判断の根拠“の解析を”．MONOist．http://monoist.atmarkit.co.jp/mn/articles/1809/21/news050.html.

13 田中潤，松本健太郎．誤解だらけの人工知能　ディープラーニングの限界と可能性．光文社，2018，30p.

14 梅田 弘之．“ビジネスに活用するための AI を学ぶ：機械学習の仕組みと学習データ”．Think It. https://thinkit.co.jp/series/6729 .

15 N.V.Chawla, K.W.Bowyer, L.O.Hall, W.P.Kegelmeyer．SMOTE: Synthetic Minority Over-sampling Technique．JAIR 16(2002) 321–357

16 Peduzzi P, Concato J, Kemper E, Holford TR, Feinstein AR.: A simulation study of the number of events per variable in logistic regression analysis. J Clin Epidemiol. 1996 Dec;49(12):1373-9.

17 Peduzzi P, Concato J, Feinstein AR. Holford TR.:Importance of events per independent variable in proportional hazards regression analysis II. Accuracy and precision of regression estimates．J Clin Epidemiology 1995 Dec;48(12): 1503-10

18 新谷 歩．多変量解析－説明変数の選び方．今日から使える医療統計．東京：医学書院；2015：101-113

19 経済産業省 AI・データの利用に関する契約ガイドライン 平成 30 年 6 月

20 有賀 康顕・中山 心太・西林 孝．仕事ではじめる機械学習．オライリージャパン，2018，4p, 60-63p.

21 瀧 雅人. “これならわかる深層学習入門”. 講談社サイエンティフィック, 2017, 27p, 28p.

22 @ishizakiiii. “機械学習の情報を手法を中心にざっくり整理” . Qiita. https://qiita.com/ishizakiiii/items/f6909696c616fd6294ca.

23 @aya_taka. “30 分でわかる機械学習用語「次元削減(Dimensionality Reduction)」”. Qiita. https://qiita.com/aya_taka/items/4d3996b3f15aa712a54f

24 Machine Learning Explained: Understanding Supervised, Unsupervised, and Reinforcement Learning https://www.datasciencecentral.com/profiles/blogs/machine-learning-explained-understanding-supervised-unsupervised

25 SAS Japan. “機械学習アルゴリズム選択ガイド”. https://blogs.sas.com/content/sasjapan/2017/11/21/machine-learning-algorithm-use/.

26 Microsoft Azure. “機械学習アルゴリズム チート シート”. https://docs.microsoft.com/ja-jp/azure/machine-learning/studio/algorithm-cheat-sheet.

27 パーソン. “機械学習の手法，チートシートの解説”. Hatena Blog. http://person.hatenablog.jp/entry/2018/03/17/115050.

28 CUBE SUGAR CONTAINER，Python: 機械学習で分類問題のモデルを評価する指標について，https://blog.amedama.jp/entry/2017/12/18/005311

29 R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In arXiv: 1610.02391v3, 2017.

30 Simonyan K, ZissermanVery A.　Deep Convolutional Networks for Large-Scale Image Recognition．arXiv:1409.1556v6 10 Apr 2015

31 Python-Qiita，畳み込みニューラルネットワーク すごさを従来の機械学習のアルゴリズムと比較する https://qiita.com/koshian2/items/f9f20f3e0eee711b1505

32 Aidemy Blog，深層学習を使うべきで「ない」手書き文字認識【ロジスティック回帰と CNN の比較】https://blog.aidemy.net/entry/2017/07/25/175933

33 Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, Ronald M. Summers ChestX-ray8: Hospital-scale Chest X-ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases, IEEE CVPR, pp. 3462-3471,2017 [https://arxiv.org/pdf/1705.02315.pdf]

34 Keras Documentation https://keras.io/ja/

35 Francois Chollet (著), 株式会社クイープ (翻訳), 巣籠 悠輔 (監訳)．Python と Keras によるディープラーニング．マイナビ，2018.

36 Shachar Kaufman , Saharon Rosset , Claudia Perlich, Leakage in data mining: formulation, detection, and avoidance, ACM Transactions on Knowledge Discovery from Data (TKDD) - Special Issue on the Best of SIGKDD 2011 TKDD Homepage archiveVolume 6 Issue 4, December 2012 Article No. 15 [doi>10.1145/2020408.2020496]

37 Tensorflow https://tensorflow.org/

38 P. Rajpurkar, J. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya et al., “Chexnet: Radiologistlevel pneumonia detection on chest x-rays with deep learning,” arXiv preprint arXiv:1711.05225, 2017.

39 Gao Huang, Zhuang Liu, and Kilian Q Weinberger. Densely connected convolutional networks. arXiv preprint arXiv:1608.06993, 2016.

40 Lian Yan, Robert Dodier, Michael C. Mozer and Richard Wolniewicz. Optimizing Classifier Performance via an Approximation to the Wilcoxon-Mann-Whitney Statistics. Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), 2003.

41 Mykhaylo A, Jasper R. R. U, Vittorio F, Fluid Annotation: A Human-Machine Collaboration Interface for Full Image Annotation. https://arxiv.org/abs/1806.07527 (Submitted on 20 Jun 2018 (v1), last revised 20 Dec 2018 (this version, v5))

42 一般社団法人 日本ディープラーニング協会 「当協会が実施する資格試験について」 https://www.jdla.org/business/certificate/?id=certificate_No03

43 Danysz K, Cicirello S, Mingle E, Assuncao B, Tetarenko N, Mockute R, Abatemarco D, Widdowson M, Desai S. Artificial Intelligence and the Future of the Drug Safety Professional. Drug Saf. 2018 Oct 20

44 Luechtefeld T, Marsh D, Rowlands C, Hartung T. Machine Learning of Toxicological Big Data Enables Read-Across Structure Activity Relationships (RASAR) Outperforming Animal Test Reproducibility. Toxicol Sci. 2018 Sep 1;165(1):198-212.

45 医薬品評価委員会　データサイエンス部会　タスクフォース 1. “国内における製薬企業での人工知能（AI）の導入状況の課題”. JPMA NEWS Letter, 2019 年 1 月号 No. 189. http://www.jpma.or.jp/about/issue/gratis/newsletter/archive_after2014/89cm.pdf

46 Hessler G1, Baringhaus KH. Artificial Intelligence in Drug Design. Molecules 2018, 23(10), 2520.

47 Jamshidi N, Palsson B Ø. Mass Action Stoichiometric Simulation Models: Incorporating Kinetics and Regulation into Stoichiometric Models. Biophys J. 2010 Jan 20; 98(2): 175–185.

48 Carpenter, Kristy A.; Huang, Xudong. Machine Learning-based Virtual Screening and Its Applications to Alzheimer's Drug Discovery: A Review. Current Pharmaceutical Design, Volume 24, Number 28, 2018, pp. 3347-3358

49 Zheng X et al. DTI-RCNN: New Efficient Hybrid Neural Network Model to Predict Drug–Target Interactions. ANNML-ICANN 2018pp 104-114

50 Abdeddaïm S, Vimard S, Soualmia L.F. The MeSH-gram Neural Network Model: Extending Word Embedding Vectors with MeSH Concepts for UMLS Semantic Similarity and Relatedness in the Biomedical Domain. arXiv preprint arXiv:1812.02309 (2018)

51 Zheng et al. edge2vec: Representation learning using edge semantics for biomedical knowledge discovery. arXiv preprint arXiv:1809.02269 (2019)

52 Dimitri GM, Lió P. DrugClust: A machine learning approach for drugs side effects prediction. Comput Biol Chem. 2017 Jun;68:204-210.

53 Lample G, Ballesteros M, Subramanian S, Kawakami K, Dyer C. Neural Architectures for Named Entity Recognition. arXiv preprint arXiv: 603.01360v3 (2016)

54 G. Erkan, D.R.Radev, LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. JAIR 22 (2004) 457-479

55 Google AI Blog Text summarization with TensorFlow Wednesday, August 24, 2016 https://ai.googleblog.com/2016/08/text-summarization-with-tensorflow.html

日本製薬工業協会 医薬品評価委員会 データサイエンス部会

2018 年度タスクフォース 6 「AI ってなに？」

執筆者・タスクフォースメンバー		担当
アステラス製薬株式会社	中島 吉弘	第 4 章，付録
エーザイ株式会社	石川 秀一	第 2 章，第 4 章，付録
グラクソ・スミスクライン株式会社	若松 美弥	第 3 章，第 4 章，付録
ノバルティスファーマ株式会社	柿爪 智行#	第 1 章
小野薬品工業株式会社	吉川 俊博（2018 年 10 月まで）	―
千寿製薬株式会社	鳥本 円	第 7 章オ
大日本住友製薬株式会社	兼山 達也#	第 5 章，第 6 章，第 7 章ア～エ
大日本住友製薬株式会社	大橋 規之	第 4 章，付録
中外製薬株式会社	坂上 拓	第 7 章カ
帝人ファーマ株式会社	眞野 博貴	第 1 章，第 2 章，第 3 章
富士フイルム富山化学株式会社	田口 賢治	第 1 章，第 3 章
武田 PRA 開発センター株式会社	大江 仁美（2018 年 9 月まで）	―
武田 PRA 開発センター株式会社	竹内 大介（2018 年 10 月から）	第 7 章キ

担当副部会長

中外製薬株式会社　山本 英晴

#：タスクフォースリーダー

レビュアー

エーザイ株式会社	酒井 弘憲
サノフィ株式会社	加藤 智子
サノフィ株式会社	月田 あづさ
ファイザー株式会社	土綿 慎一
塩野義製薬株式会社	藤原 正和
小野薬品工業株式会社	冨金原 悟

A. 付録

本報告書で用いたプログラム[1]およびその実行方法について以下に示す.

A.1 実行方法

A.1.1 ローカル環境での実行

X 線画像データ[2]およびプログラム（.py ファイル，.ipybn ファイル）を以下のフォルダ構成で保存する．プログラムは GitHub から **Clone or download** ボタンをクリックして.zip ファイルにしてからダウンロードすることができる．その後 CNN_x-ray.ipynb が実行可能となる.

```
.
├── BBox_List_2017.csv
├── CNN_x-ray.ipynb
├── Data_Entry_2017.csv
├── myGenerator.py
├── myMetrics.py
├── myTrainer.py
├── myUtility.py
├── test_list.txt
├── train_list.txt
├── images
│   ├── 00000001_000.png
│   └── ...
└── logs
```

A.1.2 Colaboratory での実行

Colaboratory[3]とは Google が提供している，機械学習の教育，研究を目的とした研究用ツールである．完全にクラウドで実行される Jupyter ノートブック環境が，設定不要かつ無料で利用することができる．FAQ[4]にあるように PC 版の Chrome と Firefox では完全に動作するよう検証済みである．なお，起動から 12 時間を超える，または 90 分を超えるアイドル時間があると仮想マシンが破棄されるため注意が必要となる.

Google ドライブに x-ray フォルダを作成し，以下のようにファイルを保存しておく．その後 CNN_x-ray.ipynb が実行可能となる．ファイルのコピーや.tar.gz の展開などはプログラムで実行している.

1 https://github.com/nakashimagif/x-ray
2 https://nihcc.app.box.com/v/ChestXray-NIHCC
3 https://colab.research.google.com/notebooks/welcome.ipynb?hl=ja
4 https://research.google.com/colaboratory/faq.html

```
x-ray
├── BBox_List_2017.csv
├── Data_Entry_2017.csv
├── myGenerator.py
├── myMetrics.py
├── myTrainer.py
├── myUtility.py
├── test_list.txt
├── train_list.txt
└── images
    ├── images_001.tar.gz
    ├── images_002.tar.gz
    ├── ...
    └── images_012.tar.gz
```

ファイルのコピーでは authorization code を利用して Google ドライブを Coraboratory にマウントする必要がある．不明な点については Google ドライブの読み込み [5]と保存を参照されたい．なお，直接ローカル PC から Colaboratory にファイルをアップロードすることも可能であるが，X 線画像データのサイズが非常に大きく時間がかかる．繰り返し実行を行うような場合は Google ドライブにデータを予めアップロードしておいてそれをコピーした方が早い．

A.2 プログラム

A.2.1　CNN_x-ray.ipynb（Python Script に変換して表示）

```
# # Colaboratory で実行する場合のみ実行
# - [Google ドライブの読み込みと保存](https://colab.research.google.com/notebooks/io.ipynb)

# In[ ]:


import glob
import tarfile
from google.colab import drive
drive.mount('/content/gdrive')

get_ipython().system('mkdir images')
get_ipython().system('mkdir logs')

get_ipython().system('cp gdrive/My\\ Drive/x-ray/* .')
get_ipython().system('cp gdrive/My\\ Drive/x-ray/images/* images')


# In[ ]:
```

5 https://colab.research.google.com/notebooks/io.ipynb

```
gz_list = glob.glob("images/*.tar.gz")

for gz_file in gz_list:
    tar = tarfile.open(gz_file)
    tar.extractall()
    tar.close()

get_ipython().system('pip install Keras==2.2.4')
get_ipython().system('pip install Keras-Applications==1.0.6')
get_ipython().system('pip install Keras-Preprocessing==1.0.5')


# # 前処理

# In[ ]:


import os
import glob
import shutil
import numpy as np
import pandas as pd
import random
import pathlib
import keras
from keras_preprocessing.image import (array_to_img, img_to_array, load_img)
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
from IPython.display import display

#auto relord modules
get_ipython().magic('load_ext autoreload')
get_ipython().magic('autoreload 2')


# In[ ]:


random.seed(2018)
df = pd.read_csv('Data_Entry_2017.csv', encoding='us-ascii')

# Keep df where the file exist in images folder.
file_list = glob.glob("images/*")
file_list = [os.path.basename(file) for file in file_list]
file_list = pd.DataFrame(file_list, columns=['Image Index'])
df = pd.merge(df, file_list, on='Image Index', how='inner')

Findings = df["Finding Labels"].str.split('|')
Findings_list = [word for word_inner in Findings for word in word_inner]
Findings_list = pd.DataFrame(Findings_list, columns=['text'])
print(Findings_list['text'].value_counts())
Findings_list = list(Findings_list.drop_duplicates()['text'])
Findings_list.remove('No Finding')
for finding in Findings_list:
    df[finding] = Findings.apply(lambda x: 1 if finding in x else 0)


# In[ ]:
```

```
# Create train_list.txt and validation_list.txt

random.seed(2018)
train_val_list = set([])
with open('train_val_list.txt', 'r') as f:
    for row in f:
        train_val_list.add(row.rsplit()[0])
train_list = random.sample(train_val_list, int(0.8 * len(train_val_list)))
train_list = sorted(train_list)
validation_list = train_val_list - set(train_list)
validation_list = sorted(validation_list)
with open('train_list.txt', 'w') as f:
    f.write("\n".join(train_list))
with open('validation_list.txt', 'w') as f:
    f.write("\n".join(validation_list))


# In[ ]:


for tt in ['train', 'validation', 'test']:
    tmp_list = []
    with open(tt + '_list.txt', 'r') as f:
        for row in f:
            tmp_list.append(row.rsplit()[0])
    df[tt] = df['Image Index'].isin(tmp_list).apply(
        lambda x: 'Y' if x == True else 'N')


def data_split(x):
    if x.test == "Y":
        return "test"
    elif x.train == "Y":
        return "train"
    else:
        return "validation"


df['split'] = df.apply(lambda x: data_split(x), axis=1)


# # ニューラルネットワークの訓練

# In[ ]:


import myTrainer

y_col = ['Effusion']


# In[ ]:


# trainingx = myTrainer.trainer(
#     trainer_name='densenet_p3',
#     dataframe=df,
#     data_proportion=10**-2,
```

```
#     y_col=y_col,
#     lr=10**-7,
#     batch_size=32,
#     initial_epoch=0,
#     epochs=50)
# trainingx.lr_finder('densenet_p3')

# trainingx = myTrainer.trainer(
#     trainer_name='densenet_p2',
#     dataframe=df,
#     data_proportion=10**-2,
#     y_col=y_col,
#     lr=10**-7,
#     batch_size=32,
#     initial_epoch=0,
#     epochs=50)
# trainingx.lr_finder('densenet_p2')

# trainingx = myTrainer.trainer(
#     trainer_name='densenet_p1',
#     dataframe=df,
#     data_proportion=10**-1,
#     y_col=y_col,
#     lr=10**-7,
#     batch_size=32,
#     initial_epoch=0,
#     epochs=50)
# trainingx.lr_finder('p1')

# trainingx = myTrainer.trainer(
#     trainer_name='densenet_p0',
#     dataframe=df,
#     data_proportion=10**-0,
#     y_col=y_col,
#     lr=10**-7,
#     batch_size=32,
#     initial_epoch=0,
#     epochs=50)
# trainingx.lr_finder('densenet_p0')


# In[ ]:


# import myTrainer
# training1 = myTrainer.trainer(
#     trainer_name='densenet_lr7_p3',
#     dataframe=df,
#     data_proportion=10**-3,
#     y_col=y_col,
#     lr=10**-7,
#     batch_size=32,
#     initial_epoch=0,
#     epochs=10)
# training1.training()
# training1.resume_training('densenet_lr7_p3_1')


# In[ ]:
```

```
training3 = myTrainer.trainer(
    trainer_name='densenet_lr3_p3',
    dataframe=df,
    data_proportion=10**-3,
    y_col=y_col,
    lr=10**-3,
    batch_size=32,
    initial_epoch=0,
    epochs=50)
training3.training()
training3.evaluating()


# In[ ]:


training2 = myTrainer.trainer(
    trainer_name='densenet_lr3_p2',
    dataframe=df,
    data_proportion=10**-2,
    y_col=y_col,
    lr=10**-3,
    batch_size=32,
    initial_epoch=0,
    epochs=50)
training2.training()
training2.evaluating()


# In[ ]:


training1 = myTrainer.trainer(
    trainer_name='densenet_lr3_p1',
    dataframe=df,
    data_proportion=10**-1,
    y_col=y_col,
    lr=10**-3,
    batch_size=32,
    initial_epoch=0,
    epochs=50)
training1.training()
training1.evaluating()


# In[ ]:


training0 = myTrainer.trainer(
    trainer_name='densenet_lr3_p0',
    dataframe=df,
    data_proportion=10**-0,
    y_col=y_col,
    lr=10**-3,
    batch_size=32,
    initial_epoch=0,
    epochs=50)
training0.training()
training0.evaluating()
```

```
# # ニューラルネットワークの評価
# ## 学習の進捗

# In[ ]:


df_training_log = pd.DataFrame()
for _ in range(4):
    tmp = pd.read_csv(
        './logs/training_log_densenet_lr3_p' + str(_) + '.csv', sep='\t')
    tmp['power'] = _
    df_training_log = pd.concat([df_training_log, tmp])

vars = ['acc', 'loss', 'auc', 'fscore', 'precision', 'recall']
epoch = df_training_log['epoch'].unique() + 1
sns.set()
cmap = plt.get_cmap("Blues")
for var in vars:
    for power in range(4):
        training = df_training_log.query('power==' + str(power))[var]
        val = df_training_log.query('power==' + str(power))['val_' + var]
        plt.plot(
            epoch,
            training,
            'bo',
            label=str(10**-power) + ': Training ' + var,
            color=cmap((4 - power) / 4))
        plt.plot(
            epoch,
            val,
            'b',
            label=str(10**-power) + ': Validation ' + var,
            color=cmap((4 - power) / 4))
    plt.title('Training and validation ' + var)
    plt.legend()
    plt.savefig("logs/Effusion_Train_Val_p0-p3_" + var + ".svg")
    plt.figure()

plt.show()


# In[ ]:


for var in vars:
    for power in [0]:
        training = df_training_log.query('power==' + str(power))[var]
        val = df_training_log.query('power==' + str(power))['val_' + var]
        plt.plot(
            epoch,
            training,
            'bo',
            label='Training ' + var,
            color=cmap((4 - power) / 4))
        plt.plot(
            epoch,
            val,
            'b',
            label='Validation ' + var,
            color=cmap((4 - power) / 4))
```

```
    plt.title('Training and validation ' + var)
    plt.legend()
    plt.savefig("logs/Effusion_Train_Val_p0_" + var + ".svg")
    plt.figure()

plt.show()


# ## データ量とテストデータでの評価項目の関連

# In[ ]:


df_test_log = pd.DataFrame()
for _ in range(4):
    tmp = pd.read_csv(
        './logs/test_result_densenet_lr3_p' + str(_) + '.csv', sep='\t')
    tmp['power'] = _
    tmp['frac'] = 10**(-_)
    df_test_log = pd.concat([df_test_log, tmp])

vars = ['acc', 'loss', 'auc', 'fscore', 'precision', 'recall']
sns.set()
cmap = plt.get_cmap("Blues")
#cmap = sns.cubehelix_palette(4, as_cmap=True)
for var in vars:
    y = df_test_log[var]
    x = df_test_log['frac']
    plt.plot(x, y, 'b', color=cmap(1.0))
    plt.title('Test ' + var)
    plt.legend()
    plt.xscale("log")
    plt.savefig("logs/Effusion_Test_" + var + ".svg")
    plt.figure()

plt.show()


# ## ROC 曲線

# In[ ]:


#http://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html
roc_auc_list, fpr_list, tpr_list = training0.auc_test()
plt.figure()
for i, finding in enumerate(y_col):
    fpr = fpr_list[i]
    tpr = tpr_list[i]
    roc_auc = roc_auc_list[i]
    plt.plot(
        fpr,
        tpr,
        #color='darkorange',
        lw=2,
        label=finding + ' (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
```

```
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.savefig("logs/Effusion_Test_ROC.svg")
plt.show()


# ## Grad-CAM

# In[ ]:


training0.Grad_CAM('00021381_013.png')


# In[ ]:


training0.Grad_CAM('00001437_012.png')
training0.Grad_CAM('00001558_016.png')
training0.Grad_CAM('00029894_000.png')
training0.Grad_CAM('00013337_000.png')
training0.Grad_CAM('00021181_002.png')
training0.Grad_CAM('00012045_009.png')
'''
00002395_007.png
00027028_017.png
00010007_168.png
00028974_016.png
00016291_002.png
00023058_004.png
00020277_001.png
00030634_000.png
00012834_034.png
00018427_011.png
00007034_016.png
00012834_122.png
00016972_025.png
00023283_019.png
00013285_026.png
00017714_006.png
00027631_000.png
00020751_003.png
'''
```

A.2.2 myTrainer.py

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, auc
from keras import backend as K
from keras import layers
from keras import models
from keras import optimizers
from keras import callbacks
from keras_preprocessing.image import (array_to_img, img_to_array, load_img)

import myGenerator
from myMetrics import tf_auc, recall, precision, fscore
```

```
import cv2


def create_generator(horizontal_flip, dataframe, directory, y_col, batch_size,
                     shuffle):
    datagen = myGenerator.myImageDataGenerator(
        rescale=1. / 255, horizontal_flip=horizontal_flip)
    generator = datagen.flow_from_dataframe(
        dataframe=dataframe,
        directory=directory,
        x_col='Image Index',
        y_col=y_col,
        has_ext=True,
        target_size=(224, 224),
        color_mode='rgb',
        classes=[0, 1],
        class_mode='binary',
        batch_size=batch_size,
        shuffle=shuffle,
        seed=None,
        save_to_dir=None,
        save_prefix='',
        save_format='png',
        subset=None,
        interpolation='nearest')
    return generator


class trainer:
    def __init__(self, trainer_name, dataframe, data_proportion, y_col, lr,
                 batch_size, initial_epoch, epochs):
        self.trainer_name = trainer_name
        self.data_proportion = data_proportion
        self.y_col = y_col
        self.lr = lr
        self.batch_size = batch_size
        self.initial_epoch = initial_epoch
        self.epochs = epochs

        self.train_df = dataframe.query('split == "train"').sample(
            frac=data_proportion)
        self.validation_df = dataframe.query('split == "validation"').sample(
            frac=data_proportion)
        self.test_df = dataframe.query('split == "test"')
        self.train_df.reset_index(drop=True, inplace=True)
        self.validation_df.reset_index(drop=True, inplace=True)
        self.test_df.reset_index(drop=True, inplace=True)

        self.train_generator = create_generator(
            horizontal_flip=True,
            dataframe=self.train_df,
            directory='images',
            y_col=self.y_col,
            batch_size=self.batch_size,
            shuffle=True)

        self.validation_generator = create_generator(
            horizontal_flip=False,
            dataframe=self.validation_df,
            directory='images',
            y_col=self.y_col,
            batch_size=self.batch_size,
            shuffle=True)

        self.test_generator = create_generator(
            horizontal_flip=False,
            dataframe=self.test_df,
            directory='images',
```

```
            y_col=self.y_col,
            batch_size=self.batch_size,
            shuffle=False)

        self.train_steps_per_epoch = np.math.ceil(
            self.train_generator.n / self.batch_size)
        self.validation_steps_per_epoch = np.math.ceil(
            self.validation_generator.n / self.batch_size)
        self.test_steps_per_epoch = np.math.ceil(
            self.test_generator.n / self.batch_size)

        train_df_total = self.train_df.shape[0]
        self.class_weight_list = []
        for num in self.train_df[self.y_col].sum():
            train_df_neg = train_df_total - num
            weight = {
                0: num / train_df_total,
                1: train_df_neg / train_df_total
            }
            self.class_weight_list.append(weight)

        #Define Callbacks
        tb_cb = callbacks.TensorBoard(log_dir="logs/", histogram_freq=0)
        cp_cb = callbacks.ModelCheckpoint(
            filepath='logs/' + self.trainer_name + '_best.h5',
            monitor='val_auc',
            verbose=1,
            save_best_only=True,
            mode='max')
        cl_cb = callbacks.CSVLogger(
            'logs/training_log_' + self.trainer_name + '.csv',
            separator='\t',
            append=False)
        reduce_lr = callbacks.ReduceLROnPlateau(
            monitor='val_auc',
            factor=0.5,
            verbose=1,
            patience=5,
            mode='max',
            min_delta=0.001)
        self.cb = [tb_cb, cp_cb, cl_cb, reduce_lr]

    def training(self, lr_search=False, **lr_search_dict):
        #Define Callbacks
        if lr_search is True:
            cb = []
            lr = lr_search_dict['lr']
            initial_epoch = 0
            epochs = 1
        else:
            cb = self.cb
            lr = self.lr
            initial_epoch = self.initial_epoch
            epochs = self.epochs

        #http://marubon-ds.blogspot.com/2017/10/inceptionv3-fine-tuning-model.html
        from keras.applications.densenet import DenseNet121
        input_tensor = layers.Input(shape=(224, 224, 3))
        inc_model = DenseNet121(
            input_tensor=input_tensor, weights='imagenet', include_top=False)

        # get layers and add average pooling layer
        x = inc_model.output
        x = layers.GlobalAveragePooling2D()(x)

        # add output layer
        predictions = layers.Dense(len(self.y_col), activation='sigmoid')(x)

        model = models.Model(inputs=inc_model.input, outputs=predictions)
```

```
    model.compile(
        optimizer=optimizers.Adam(lr=lr, beta_1=0.9, beta_2=0.999),
        loss='binary_crossentropy',
        metrics=['accuracy', precision, recall, fscore, tf_auc])

    history = model.fit_generator(
        self.train_generator,
        steps_per_epoch=self.train_steps_per_epoch,
        epochs=epochs,
        initial_epoch=initial_epoch,
        validation_data=self.validation_generator,
        validation_steps=self.validation_steps_per_epoch,
        use_multiprocessing=True,
        callbacks=cb,
        class_weight=self.class_weight_list,
        verbose=False)

    model.save('logs/' + self.trainer_name + '.h5')

    return history

def evaluating(self):
    model = models.load_model(
        'logs/' + self.trainer_name + '_best.h5',
        custom_objects={
            'precision': precision,
            'recall': recall,
            'fscore': fscore,
            'auc': tf_auc
        })

    test_result = model.evaluate_generator(
        self.test_generator,
        use_multiprocessing=True,
        steps=self.test_steps_per_epoch)
    test_result = pd.DataFrame(test_result, index=model.metrics_names).T
    test_result.to_csv(
        'logs/test_result_' + self.trainer_name + '.csv', sep='\t')

def lr_finder(self, filename):
    losses = []
    for power in range(7):
        lr = 10**(-power - 1)
        hist = self.training(lr_search=True, lr=lr)
        losses.append([hist.history['loss'][0], power, lr])
    losses = pd.DataFrame(losses, columns=['loss', 'power', 'lr'])
    losses.to_csv('logs/lr_finder_' + filename + '.csv', sep='\t')

def resume_training(self, pretraining_name):
    model = models.load_model(
        'logs/' + pretraining_name + '.h5',
        custom_objects={
            'precision': precision,
            'recall': recall,
            'fscore': fscore,
            'auc': tf_auc
        })

    history = model.fit_generator(
        self.train_generator,
        steps_per_epoch=self.train_steps_per_epoch,
        epochs=self.epochs,
        initial_epoch=self.initial_epoch,
        validation_data=self.validation_generator,
        validation_steps=self.validation_steps_per_epoch,
        use_multiprocessing=True,
        callbacks=self.cb,
        class_weight=self.class_weight_list,
```

```
        verbose=False)

    model.save(self.trainer_name + '.h5')

    return history

def auc_test(self):
    model = models.load_model(
        'logs/' + self.trainer_name + '_best.h5',
        custom_objects={
            'precision': precision,
            'recall': recall,
            'fscore': fscore,
            'auc': tf_auc
        })

    y_pred_class = model.predict_generator(
        self.test_generator, steps=self.test_steps_per_epoch)

    fpr_list = []
    tpr_list = []
    roc_auc_list = []
    for i, finding in enumerate(self.y_col):
        fpr, tpr, _ = roc_curve(
            self.test_generator.classes[:, i],
            y_pred_class[:, i],
            pos_label=1)
        roc_auc = auc(fpr, tpr)
        fpr_list.append(fpr)
        tpr_list.append(tpr)
        roc_auc_list.append(roc_auc)
    return roc_auc_list, fpr_list, tpr_list

def Grad_CAM(self, filename):
    model = models.load_model(
        'logs/' + self.trainer_name + '_best.h5',
        custom_objects={
            'precision': precision,
            'recall': recall,
            'fscore': fscore,
            'auc': tf_auc
        })

    img_path = 'images/' + filename
    img = load_img(img_path, target_size=(224, 224))
    display(img)
    x = img_to_array(img)
    x = np.expand_dims(x, axis=0)
    x /= 256

    preds = model.predict(x)
    x_output = model.output
    last_conv_layer = model.get_layer('bn')
    grads = K.gradients(x_output, last_conv_layer.output)[0]
    pooled_grads = K.mean(grads, axis=(0, 1, 2))
    iterate = K.function([model.input],
                         [pooled_grads, last_conv_layer.output[0]])
    pooled_grads_value, conv_layer_output_value = iterate([x])

    for i in range(len(pooled_grads_value)):
        conv_layer_output_value[:, :, i] *= pooled_grads_value[i]

    heatmap = np.mean(conv_layer_output_value, axis=-1)
    heatmap = np.maximum(heatmap, 0)
    heatmap /= np.max(heatmap)
    plt.matshow(heatmap, cmap=plt.cm.magma)
    plt.savefig("logs/" + filename + "_heatmap.png")
    plt.show()
```

```
        img = cv2.imread(img_path)

        heatmap = cv2.resize(heatmap, (img.shape[1], img.shape[0]))
        heatmap = np.uint8(255 * heatmap)
        heatmap = cv2.applyColorMap(heatmap, cv2.COLORMAP_JET)
        superimposed_img = heatmap * 0.4 + img

        #Overlay Bounding Box
        bbox = pd.read_csv('BBox_List_2017.csv', usecols=range(6))
        bbox.columns = ['Image_Index', 'Finding_Label', 'x', 'y', 'w', 'h']
        box_list = bbox.query(
            'Image_Index == @filename and Finding_Label in @self.y_col')
        box_list = box_list[['x', 'y', 'w', 'h']].astype('int16')
        box_list = box_list.values.tolist()
        box_list = [[(x[0], x[1]), (x[0] + x[2], x[1] + x[3])]
                    for x in box_list]
        if len(box_list) > 0:
            for box in box_list:
                cv2.rectangle(
                    superimposed_img, box[0], box[1], (0, 255, 0), thickness=3)

        cv2.imwrite("logs/" + filename + "_superimposed_img.png",
                    superimposed_img)
        img = load_img(
            "logs/" + filename + "_superimposed_img.png",
            target_size=(224, 224))
        display(img)
```

A.2.3 myMetrics.py

```
import functools
from keras import backend as K
import tensorflow as tf

#https://stackoverflow.com/questions/41032551/how-to-compute-receiving-operating-
characteristic-roc-and-auc-in-keras
def as_keras_metric(method):
    @functools.wraps(method)
    def wrapper(self, args, **kwargs):
        """ Wrapper for turning tensorflow metrics into keras metrics """
        value, update_op = method(self, args, **kwargs)
        K.get_session().run(tf.local_variables_initializer())
        with tf.control_dependencies([update_op]):
            value = tf.identity(value)
        return value

    return wrapper

tf_auc = as_keras_metric(tf.metrics.auc)
#tf_precision = as_keras_metric(tf.metrics.precision)
#tf_recall = as_keras_metric(tf.metrics.recall)


def binary_cut(y):
    cutoff = 0.5
    return (K.cast(K.greater(y, cutoff), dtype='float32'))


def true_positive(y_true, y_pred):
    y_pred_bin = binary_cut(y_pred)
    return K.sum(y_true * y_pred_bin)
```

```
def false_positive(y_true, y_pred):
    y_pred_bin = binary_cut(y_pred)
    y_true_inv = 1 - y_true
    return K.sum(y_true_inv * y_pred_bin)


def false_negative(y_true, y_pred):
    y_pred_bin = binary_cut(y_pred)
    y_pred_inv = 1 - y_pred_bin
    return K.sum(y_true * y_pred_inv)


def precision(y_true, y_pred):
    tp = true_positive(y_true, y_pred)
    fp = false_positive(y_true, y_pred)
    return tp / (tp + fp + K.epsilon())


def recall(y_true, y_pred):
    tp = true_positive(y_true, y_pred)
    fn = false_negative(y_true, y_pred)
    return tp / (tp + fn + K.epsilon())


def fscore(y_true, y_pred):
    pre = precision(y_true, y_pred)
    re = recall(y_true, y_pred)
    return 2 * pre * re / (pre + re + K.epsilon())
```

A.2.4 myGenerator.py

```
import numpy as np
import os
from keras_preprocessing.image import (ImageDataGenerator, Iterator,
                                       _list_valid_filenames_in_directory,
                                       load_img, img_to_array)

class myImageDataGenerator(ImageDataGenerator):
    def flow_from_dataframe(self, dataframe, directory,
                            x_col="filename", y_col="class", has_ext=True,
                            target_size=(256, 256), color_mode='rgb',
                            classes=None, class_mode='categorical',
                            batch_size=32, shuffle=True, seed=None,
                            save_to_dir=None,
                            save_prefix='',
                            save_format='png',
                            subset=None,
                            interpolation='nearest'):
        return myDataFrameIterator(dataframe, directory, self,
                                   x_col=x_col, y_col=y_col, has_ext=has_ext,
                                   target_size=target_size, color_mode=color_mode,
                                   classes=classes, class_mode=class_mode,
                                   data_format=self.data_format,
                                   batch_size=batch_size, shuffle=shuffle, seed=seed,
                                   save_to_dir=save_to_dir,
                                   save_prefix=save_prefix,
                                   save_format=save_format,
                                   subset=subset,
```

```
                                interpolation=interpolation)

class myDataFrameIterator(Iterator):
    def __init__(self, dataframe, directory, image_data_generator,
                 x_col="filenames", y_col="class", has_ext=True,
                 target_size=(256, 256), color_mode='rgb',
                 classes=None, class_mode='categorical',
                 batch_size=32, shuffle=True, seed=None,
                 data_format=None,
                 save_to_dir=None, save_prefix='', save_format='png',
                 follow_links=False,
                 subset=None,
                 interpolation='nearest',
                 dtype='float32'):
        super(myDataFrameIterator, self).common_init(image_data_generator,
                                                     target_size,
                                                     color_mode,
                                                     data_format,
                                                     save_to_dir,
                                                     save_prefix,
                                                     save_format,
                                                     subset,
                                                     interpolation)
        try:
            import pandas as pd
        except ImportError:
            raise ImportError('Install pandas to use flow_from_dataframe.')
        if type(x_col) != str:
            raise ValueError("x_col must be a string.")
        if type(has_ext) != bool:
            raise ValueError("has_ext must be either True if filenames in"
                             " x_col has extensions,else False.")
        self.df = dataframe.drop_duplicates(x_col)
        self.df[x_col] = self.df[x_col].astype(str)
        self.directory = directory
        self.classes = classes
        if class_mode not in {'categorical', 'binary', 'sparse',
                              'input', 'other', None}:
            raise ValueError('Invalid class_mode:', class_mode,
                             '; expected one of "categorical", '
                             '"binary", "sparse", "input"'
                             '"other" or None.')
        self.class_mode = class_mode
        self.dtype = dtype
        white_list_formats = {'png', 'jpg', 'jpeg', 'bmp',
                              'ppm', 'tif', 'tiff'}
        # First, count the number of samples and classes.
        self.samples = 0

        if not classes:
            classes = []
            if class_mode not in ["other", "input", None]:
                classes = list(self.df[y_col].unique())
        else:
            if class_mode in ["other", "input", None]:
                raise ValueError('classes cannot be set if class_mode'
                                 ' is either "other" or "input" or None.')
        self.num_classes = len(classes)
        self.class_indices = dict(zip(classes, range(len(classes))))

        # Second, build an index of the images.
```

```
    self.filenames = []
    filenames = _list_valid_filenames_in_directory(
        directory,
        white_list_formats,
        self.split,
        class_indices=self.class_indices,
        follow_links=follow_links,
        df=True)
    if has_ext:
        ext_exist = False
        for ext in white_list_formats:
            if self.df.loc[0, x_col].endswith("." + ext):
                ext_exist = True
                break
        if not ext_exist:
            raise ValueError('has_ext is set to True but'
                             ' extension not found in x_col')
        self.df = self.df[self.df[x_col].isin(filenames)].sort_values(by=x_col)
        self.filenames = list(self.df[x_col])
    else:
        without_ext_with = {f[:-1 * (len(f.split(".")[-1]) + 1)]: f
                            for f in filenames}
        filenames_without_ext = [f[:-1 * (len(f.split(".")[-1]) + 1)]
                                 for f in filenames]
        self.df = (self.df[self.df[x_col].isin(filenames_without_ext)]
                   .sort_values(by=x_col))
        self.filenames = [without_ext_with[f] for f in list(self.df[x_col])]
    self.samples = len(self.filenames)
    self.classes = np.zeros((self.samples,), dtype='int32')
    if class_mode not in ["other", "input", None]:
        self.classes = self.df[y_col].values
        #self.classes = np.array([self.class_indices[cls] for cls in classes])
    elif class_mode == "other":
        self.data = self.df[y_col].values
        if type(y_col) == str:
            y_col = [y_col]
        if "object" in list(self.df[y_col].dtypes):
            raise TypeError("y_col column/s must be numeric datatypes.")
    if self.num_classes > 0:
        print('Found %d images belonging to %d classes.' %
              (self.samples, self.num_classes))
    else:
        print('Found %d images.' % self.samples)

    super(myDataFrameIterator, self).__init__(self.samples,
                                              batch_size,
                                              shuffle,
                                              seed)

def _get_batches_of_transformed_samples(self, index_array):
    batch_x = np.zeros(
        (len(index_array),) + self.image_shape,
        dtype=self.dtype)
    # build batch of image data
    for i, j in enumerate(index_array):
        fname = self.filenames[j]
        img = load_img(os.path.join(self.directory, fname),
                       color_mode=self.color_mode,
                       target_size=self.target_size,
                       interpolation=self.interpolation)
        x = img_to_array(img, data_format=self.data_format)
```

```
            # Pillow images should be closed after `load_img`,
            # but not PIL images.
            if hasattr(img, 'close'):
                img.close()
            params = self.image_data_generator.get_random_transform(x.shape)
            x = self.image_data_generator.apply_transform(x, params)
            x = self.image_data_generator.standardize(x)
            batch_x[i] = x
        # optionally save augmented images to disk for debugging purposes
        if self.save_to_dir:
            for i, j in enumerate(index_array):
                img = array_to_img(batch_x[i], self.data_format, scale=True)
                fname = '{prefix}_{index}_{hash}.{format}'.format(
                    prefix=self.save_prefix,
                    index=j,
                    hash=np.random.randint(1e7),
                    format=self.save_format)
                img.save(os.path.join(self.save_to_dir, fname))
        # build batch of labels
        if self.class_mode == 'input':
            batch_y = batch_x.copy()
        elif self.class_mode == 'sparse':
            batch_y = self.classes[index_array]
        elif self.class_mode == 'binary':
            batch_y = self.classes[index_array].astype(self.dtype)
        elif self.class_mode == 'categorical':
            batch_y = np.zeros(
                (len(batch_x), self.num_classes),
                dtype=self.dtype)
            for i, label in enumerate(self.classes[index_array]):
                batch_y[i, label] = 1.
        elif self.class_mode == 'other':
            batch_y = self.data[index_array]
        else:
            return batch_x
        return batch_x, batch_y

    def next(self):
        """For python 2.x.

        # Returns
            The next batch.
        """
        with self.lock:
            index_array = next(self.index_generator)
        # The transformation of images is not under thread lock
        # so it can be done in parallel
        return self._get_batches_of_transformed_samples(index_array)
```