機械学習品質評価・向上技術に関する報告書

機械学習品質マネジメントガイドライン付属文書

第 4 版

2024 年 8 月 1 日

国立研究開発法人産業技術総合研究所

デジタルアーキテクチャ研究センター
テクニカルレポート DigiARC-TR-2024-01

サイバーフィジカルセキュリティ研究センター
テクニカルレポート CPSEC-TR-2024001

人工知能研究センター
テクニカルレポート

まえがき

国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）受託業務（JPNP 20006）「機械学習システムの品質評価指標・測定テストベッドの研究開発」では、機械学習品質を説明可能にするために機械学習品質マネジメントガイドライン[1]を開発しており、このガイドライン開発と並行して、機械学習品質の評価・向上技術の調査・研究開発も行っている。本報告書は、機械学習品質マネジメントガイドラインに記載されている品質評価を技術的な側面から補足するために、本調査・研究開発（2019～2023 年度）の成果をまとめたものである。

目次

1　はじめに

統計的機械学習を利用した各種産業製品の品質を明確に説明可能にするために、機械学習品質マネジメントガイドラインが開発されている[1]。このガイドライン第 4 版では、機械学習システムに対する 14 個の内部品質特性（学習モデルの安定性やプログラムの信頼性等）に着目しているが、これら内部品質特性の評価や向上の技術は必ずしもまだ確立していない。本報告書は、ガイドラインの開発と並行して行われている内部品質特性の評価・向上技術の調査・研究開発に関する成果をまとめたものである。

1.1　本調査・研究開発の概要

図 1.1 に、2019~2023 年度に調査・研究開発した機械学習品質評価・向上の技術（図 1.1 の中央の黄色い四角が本稿で説明する技術を表す）と機械学習モデルのライフサイクルの各フェーズ、14 個の内部品質特性との関係を示す。以下、各技術については 2 章以降で詳しく説明するが、ここで簡単に説明しておく。

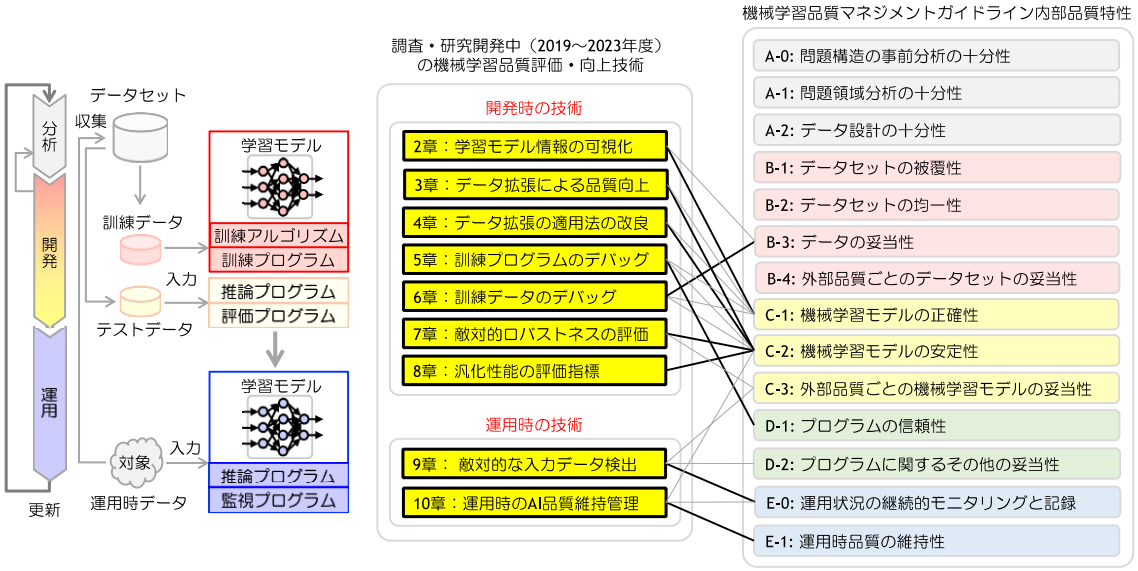


図 1.1　機械学習品質評価・向上技術（2019～2023 年度調査・研究開発）

- **2 章　学習モデル情報の可視化：**
  モデル同士の差分比較や各モデルに反映されている作業者（アノテータ、モデル設計者）の感性の可視化を主な目的として、モデル自体の構造や精度に加えてモデル作成に関わる作業者の作業手順やモデルへの影響を複合的に可視化するツールの実装を進めた[2][3]。主要な機能では使用した学習データ、モデル構造、最適化アルゴリズムの各差分や、作業者による調整の意図、モデルに対する印象・評価などを可視化する。このツールを用いて複数の機械学習モデル調整作業履歴の可視化結果を作成した。

- **3 章　データ拡張による品質向上：**

学習データを加工してデータを増強するデータ拡張手法が画像識別の品質評価に与える影響とその品質改善方法について検討を行った。様々なデータ拡張手法（とその組み合わせ）に対して実験を行い、識別精度の平均値だけでなく分散も計測することが、品質を評価するために重要であるとの結論を得た。

- **4 章　深層学習におけるデータ拡張の適用法の改良による品質改善：**
  多様なデータを生成するデータ拡張手法として、Latent DA 法や FC-mixup 法[24]を提案し、画像識別問題においてモデルの品質が向上することを確認した[4][5]。Latent DA 法については、データ拡張に最適な層を動的に発見する手法 AdaLASE を開発し、その効果を検証した。FC-mixup 法については、CNN のチャネル単位で混合する FC-channel とピクセル単位で混合する FC-pixel、および既存手法の Manifold mixup とのハイブリッドを比較し、提案手法は既存手法よりも高い精度を示すことを確認した。また、適切なデータ拡張ポリシーを効率よく探索するために、探索フェーズを短くとり、Affinity および Diversity を組み込んだ新しい指標を用いる方法を提案し、実験によりその有効性を確認した。

- **5 章　深層 NN ソフトウェアのデバッグ・テスティング：**
  深層学習型の機械学習モデルの不具合の原因を推論時の直接原因（予測・推論プログラムによる）と訓練時の根本原因（訓練・学習プログラム、学習モデル、学習データセットによる）に分けて整理し、訓練・学習プログラムのバグの有無を、推論時の機械学習モデルの内部情報（ニューロンカバレッジ）によって推定するための検査指標と分析手法を提案し、実験によってその検査指標の有効性を確認した[6]-[10]。

- **6 章　訓練データのデバッグ・テスティング：**
  深層学習型の機械学習モデルの不具合の根本原因が訓練データの偏りにある場合を対象とし、訓練データの偏りをモデル正確性とモデル・ロバスト性の 2 つの品質観点から判断する方法を検討し、実験を行った。実験では、訓練済み学習モデル内部の活性状態（ニューロン・カバレッジ NC）の低いデータと高いデータを訓練データから削除することによって、モデル正確性はほぼ保ったまま、モデル・ロバストネスが向上する結果が得られ、学習モデルの内部情報が訓練データのデバッグに役立つことを確認できた。このようなデバッグ問題に関して、本章の方法は統計的な見方とソフトウェア工学の方法を融合した方法とみなせる。

- **7 章　ロバストネスの評価・向上技術：**
  敵対的データを含む入力ノイズに対するロバストネスの指標の一つに最大安全半径（誤判断を生じないことを保証できるノイズの最大値）を計測する技術と、その半径を増加させる技術について調査した。最大安全半径の評価の計算コストは非常に高いが、近年、その近似値を効率よく見積もる手法が提案されてきている。

- **8 章　汎化誤差の評価技術：**
  機械学習品質マネジメントガイドラインの「機械学習モデルの安定性」を評価するた

めに、無作為摂動と最悪摂動を重み（訓練パラメータ）に付加した場合の汎化誤差上界に着目し、その計算法を調査して、その計算実験結果を基に、安定性の評価指標としての有効性について考察した。ここで、無作為摂動とは指定範囲内で無作為に付加する摂動であり、最悪摂動とは指定範囲内で不正解する方向に付加する摂動である。摂動付加汎化誤差上界に着目する理由は、摂動に対する耐性と汎化性能との間に相関関係があることが知られていること、汎化誤差上界はデータセットに含まれない未見の入力に対する不正解率の期待値を確率的に保証できることにある。

- **9 章　敵対的データ検出技術：**
  運用時に入力画像が敵対的データであるかを判別する方法を実用的に確立するために、最先端の敵対的データ検出手法を調査し、4 つの主要なカテゴリに分類するとともに追実験を行い、NIC 法が最も高い検出率を示すことを確認した。そこで、NIC 法に基づく敵対的データ検出を行うために NIC フレームワークを構築し、その高い検出率の理由を説明するために Kullback-Leibler 情報量を用いて NIC 法を評価した。

- **10 章　運用時における AI 品質管理技術：**
  運用時に発生するデータの変化や未知のデータの到来に対応するために、データ分布変化に対する検知・適応技術および同技術の関連技術であるドメイン適応技術の最新技術を幅広く調査した。適用可能性や運用コストの面から有望な教師なしコンセプトドリフト適応技術、訓練データに依存しない適応技術やラベルシフトなどの入力データの分布以外の変化にも対応可能な教師なしドメイン適応技術の研究開発が行われている。さらに、データのプライバシやその可搬性の観点から適応前に使用した訓練データに依存しない適応技術などの必要性も高まってきている。これらの調査結果をサーベイ[11][12][13]としてまとめた。

1.2　著者リスト

各章の著者は以下のとおり：

- 1 章：磯部 祥尚　（産業技術総合研究所）
- 2 章：宮城 優里　（産業技術総合研究所）
- 3 章：大西 正輝　（産業技術総合研究所）
- 4 章：高瀬 朝海　（産業技術総合研究所）
- 5 章：中島 震　（国立情報学研究所）
- 6 章：中島 震　（国立情報学研究所）
- 7 章：磯部 祥尚　（産業技術総合研究所）
- 8 章：磯部 祥尚　（産業技術総合研究所）
- 9 章：中島 裕生、西田 啓一　（テクマトリックス株式会社）
- 10 章：大川 佳寛、小林 健一　（富士通株式会社）

1.3 謝辞

本調査・研究開発は、国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）受託業務（JPNP20006）「機械学習システムの品質評価指標・測定テストベッドの研究開発」として行っています。

2 機械学習モデル情報の可視化

ブラックボックスである機械学習モデルの構造や挙動についての分析を支援する手法として、情報可視化技術の導入が進んでいる。このような機械学習モデル可視化に関し、以下の 2 点を目的として新たな手法の研究を進めた。

- 複数のモデル間の差分・比較結果の可視化
  （人間が解釈・理解しやすい表現を用いた可視化の実装）
- モデルに反映されている作業者（アノテータ、モデル設計者）の感性の可視化
  （品質評価に用いることができる新たな要素の提案）

本章では、まず近年の機械学習モデル可視化技術の調査結果について解説する。続いて 2020～2023 年度に作成した、モデルと作業者情報を観察するための可視化ツールの実行例と今後の実装方針について報告する。

2.1 機械学習支援のための可視化手法についての調査

機械学習を対象とした可視化手法の基本的な目的はモデルの解釈可能性の向上であり、近年注目されている XAI（Explainable AI）と大きな関連がある。XAI の定義や評価方法について確定的なものは未だ存在しない。しかし、XAI の分類を試みる論文等は多数発表されているため、これらに沿って可視化の目的や手法を考案することができる。[14]では解釈可能性を上げるためのアプローチについて以下の 4 通りに分類している。可視化が特に有効である項目は(2)(4)であり、研究事例も多いとみられる。

(1) 大局的な説明（簡易なモデルによる複雑なモデル構造の近似）
(2) 局所的な説明（モデルの出力結果に関する判断根拠の説明）
(3) 説明可能なモデルの設計（設計段階での可読性の高いモデルの作成）
(4) 深層学習モデルの説明（画像データ内でモデルが認識している部分のハイライトなど）

このような機械学習の可視化手法は継続的に研究されており、用途や対象事例が多様であるためサーベイ論文も増加している。例えば[15]では深層学習可視化手法について、5W1H の要素に沿う形で解説・分類している。深層学習可視化分野の全体的な方向性や課題についても多数提示しており、特に品質評価のための可視化手法の開発を目指す本研究と関わりの深いものとして「モデル評価のためのインタラクションの改良」「モデルに対する人の積極的な関与による解釈可能性の向上」などが挙げられる。

機械学習可視化の研究が進み、実社会でも利用される機会が増加するにつれて、1 つの可視化画面で複合的な分析が行われる傾向が強まっている。従来は単体のモデルの詳細な分析や、データ（[16]）かモデル構造（[17]）のいずれかに特化した可視化などが多かった。しかし近年はデータとモデル構造の複合的な可視化手法や、複数のモデルを比較することを目的とした可視化手法についても研究が進んでいる。機械学習モデルを構成する要素は

膨大なものとなるため、例えばモデル単体の可視化結果をモデル個数分作成し、並べて比較するには大きな手間がかかる。また、比較しようとするモデル間の構造や精度の差が小さく、モデルの特徴などを見落とす可能性も考えられる。そのため、差分を強調する表現を用い、限られた画面内で効率的に差分を発見できるようにする可視化手法の必要性が高い。（例えば[18]では、10 種類以上のモデルについてデータの入力から出力までのパイプライン、ハイパパラメータの値などを一画面で比較できる。）

ここまでで、モデル自体の性質や精度に関する可視化手法についての傾向と事例を紹介した。これと並行して、モデルの作成や評価に関わる作業者（アノテータ、設計者、エンドユーザ）とモデルとの関わり方についても調査を進めた。画像認識などの分野では人の認識能力を超える精度を持つモデルが開発される一方で、「モデルの精度を向上させるためには積極的な人の介入が望ましい」という提言は分野を問わず根強い。AI と人との関係性や、モデル作成過程での効果的な介入方法に関し、以下のような項目に注目して議論した論文が多く存在する。

- 学習過程での、モデルの精度を向上させるための操作（調整や評価）の導入方法
- 操作性が良く、作業者のモチベーションを維持しやすいインタフェースの設計方法
- 関連分野（認知科学、心理学など）との連携

一例として[19]では、モデルの評価・改善のためのフィードバックを課された作業者の心理状態について検証しており、共通する傾向として「モデルに正しい処理手順を直接指示できることを好む」「自身の行動によってモデルの精度が改善されていることが可視化されると、モチベーションの向上やより積極的なフィードバックを見込める」などを紹介している。このような作業者自身の情報や、各作業者がモデルに与えた影響についての可視化事例は少ないという印象だが、以下のように品質保証の根拠として採用できると考えられる。

- 分析対象の事例、あるいは機械学習の専門家がモデル作成に参加していた場合、彼らの知識が十分にモデルの挙動に反映されていることを示す
- どの作業者の行動がモデルに強く反映されているのかを示し、調整するべき要素（データ、パラメータなど）を特定するための手がかりとする

2.2 機械学習モデルと作業者情報の可視化手法の提案

以上のような調査結果から、本研究では機械学習モデル可視化手法のうち特に「複数モデルの比較可視化」「作業者の感性に関する可視化」を重要視し、これらの性質を併せ持つような可視化ツールの設計を進めた。図 2.1 は提案手法の概要図である。本研究では作業者をアノテータ、モデル設計者、エンドユーザの 3 種類に分類しており、特にモデル設計者を中心とした分析を進めている。

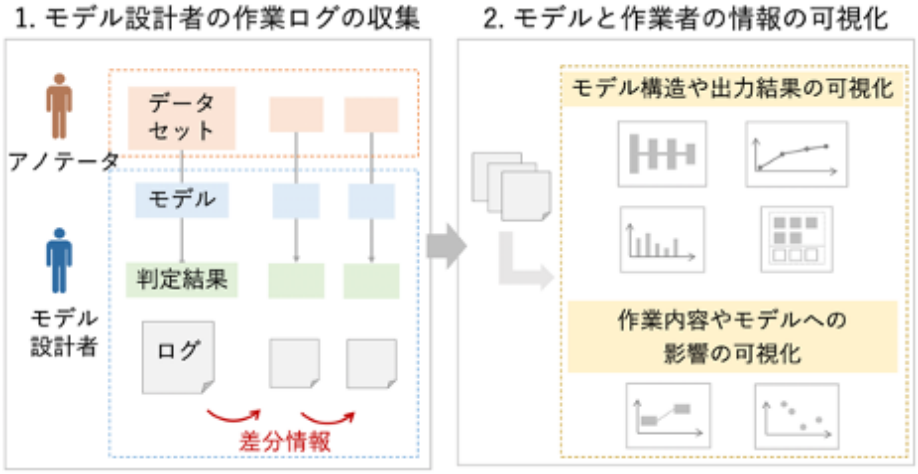


図 2.1　機械学習モデルと作業者情報の可視化手法の概要

2.2.1　モデル同士の差分に関するログの収集

まず、可視化の対象とするモデルの構造や調整過程、テスト結果などのログを収集する。現在の実装では画像分類または回帰分析の事例を想定している。機械学習の実験管理ツールである Comet.ml による記録や機械学習のコンペティションに投稿された記事（使用したコードや結果、およびそれらの解説）から、モデル設計者によるパラメータ調整過程やテストの結果などをテキストファイルとして取得する。なお、アノテータについては直接的に作業ログを収集せず、モデル設計者がどのようにデータを選択し、前処理を適用したのかによって間接的に作業内容を評価する。

これらのログから 2 つのモデル同士の差分（過去に使用したモデルからの変化量）を算出する。モデルに関する差分は「学習データ」「モデル構造」「最適化アルゴリズム」の 3 カテゴリに分類する。学習データの差分は、使用したデータやクラスの数、前処理に用いたパラメータの差分などを足し合わせて求める。モデル構造の差分は、2 つのモデルを構成するレイヤー同士のペアを作成し、各ペアの非類似度（レイヤーの種類やパラメータの差分）を合計することで求める（図 2.2）。最適化アルゴリズムの差分については、アルゴリズムの種類が異なる場合には定数を付与し、同一である場合にはパラメータの差分から計算する。3 種類の差分を求めたのち、これらの値を合計したモデルの総合的な変化量も求める。なおそれぞれのカテゴリの差分計算については、学習データの数やモデルのレイヤーの数といった基本的な情報による差分値に、事例ごとに独自の要素（例えば画像分類の場合はデータ拡張に関する処理など）から算出される差分値を加算して求める。この方法では事例ごとに差分の計算式が変わるため、異なる事例における差分値同士を直接比較することは難しいが、同一の事例内における特定のモデルの発展の過程や、複数の作業者が作成したそれぞれのモデルの比較が可能である。

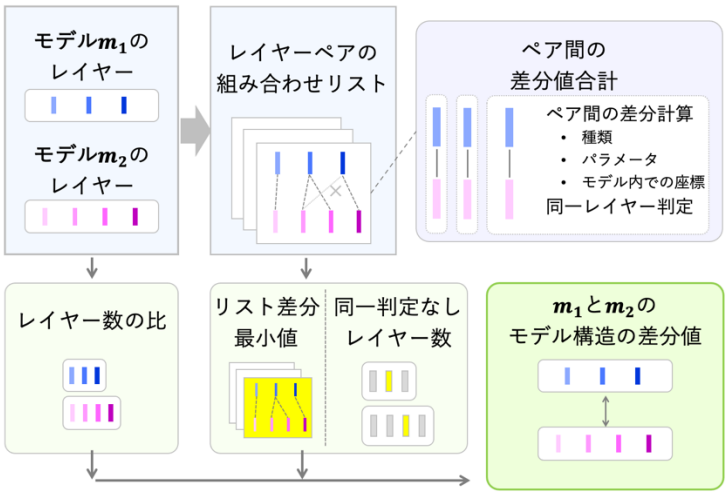


図 2.2　モデル構造差分計算の流れ

2.2.2　　モデルの構造や作業者の情報の可視化

収集したログを用いて、モデルの構造や作業者の情報の可視化を行なう。本ツールの主なユーザはモデルの設計者と想定する。可視化に不慣れなユーザが含まれる可能性を考慮して基本的な可視化手法（折れ線グラフ、棒グラフなど）を組み合わせ、これらの連携（関連部分のハイライトなど）を積極的に行うという方針で実装を進めた。2020 年度にはモデルの構造や出力結果などの基本的な情報に関するビューを実装し、2021～23 年度には作業者によるモデルの調整やテストの経過を可視化するビューを作成した。

図 2.3 は 2020 年度に試作した可視化ビューの一覧であり、2 つの簡易なモデルについて MNIST の学習・出力結果を可視化した例である。

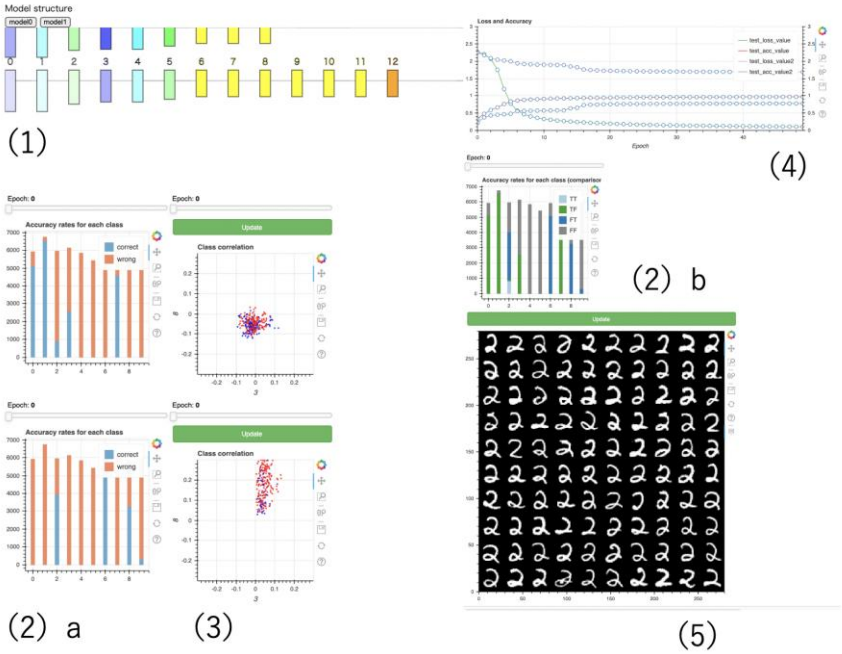


図 2.3　モデルの構造と出力結果に関する可視化ビューの一覧

JupyterLab 上で機械学習ライブラリ PyTorch、可視化ライブラリ Bokeh などを用いて以下のようにビューを作成し、2 つのモデルの特徴を比較できるようにした。

(1) 各モデル構造のネットワーク
(2) クラスごとの出力結果の棒グラフ
    a) モデルごとの可視化
    b) 2 つのモデル間の差分の可視化
(3) モデルごとの、選択した 2 クラス間の出力結果相関の散布図
(4) 精度の折れ線グラフ
(5) 特に高い（低い）確信度で分類されたデータのサムネイル一覧

図 2.4 は、2 つのモデルについて MNIST に対する出力を分類した結果の例である。横軸は 0〜9 のクラス、縦軸はデータ数を表す。それぞれの棒の色分けは、2 つのモデルについての正解（T）・不正解（F）の組み合わせを表したものである。TF（FT）は「モデル 1（2）のみ正しく分類できたデータ」を意味する。学習開始直後（図 2.4 左）は、モデル 1 がクラス 0、1、7、モデル 2 がクラス 2、6、8 での正解率が高く、それぞれのモデルが異なる強みを持っていたことがわかる。学習が進んだ段階（図 2.4 右）ではどちらのモデルも多くのクラスで正解率が高くなっている。特にモデル 1 はモデル 2 が不得意とするクラス 3、4、5、7 を含めて正答率が高く、この段階ではモデル 1 の学習の方がモデル 2 よりも進んでいることがわかる。

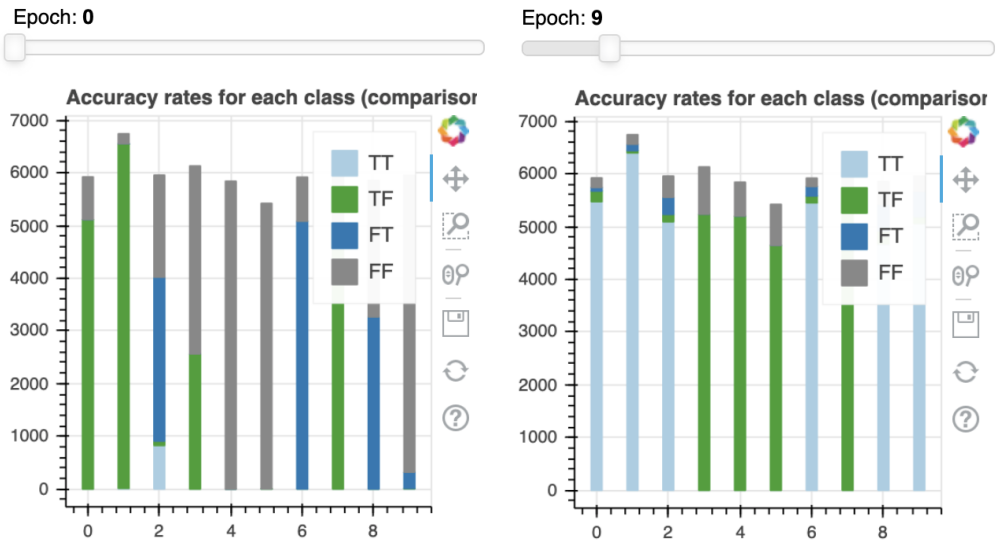


図 2.4　2 つのモデルの出力結果比較の例

図 2.5 は 2021 年度〜2023 年度に実装したモデル調整作業履歴の時系列可視化ビューの構成である。ここまでの処理で得たモデルの構造やテスト結果についてのログを入力として、ネットワークグラフによる可視化を行う。

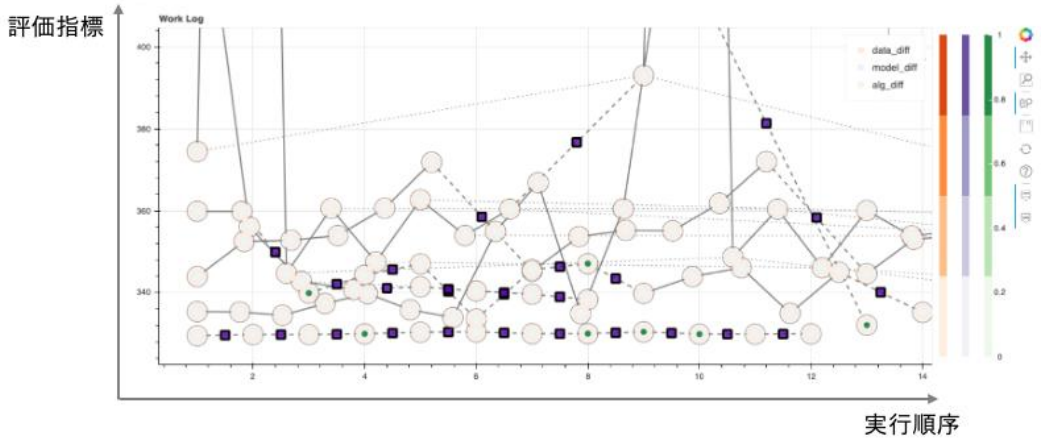


図 2.5　モデル調整作業履歴可視化画面の全体像

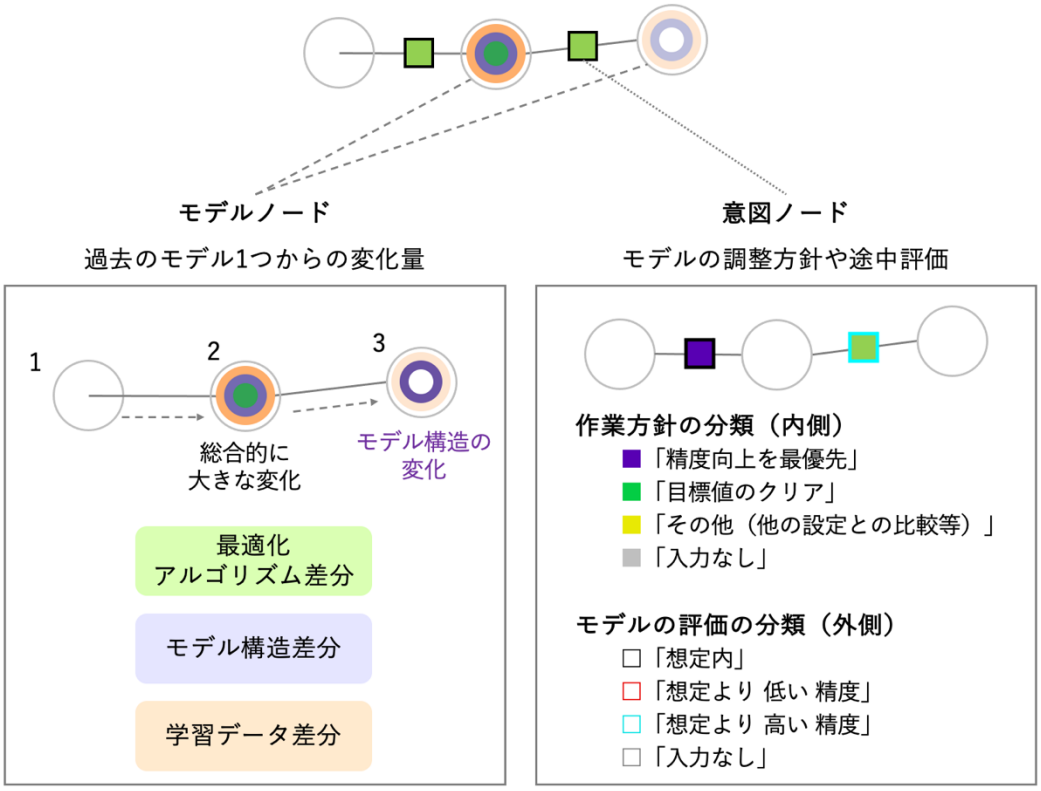


図 2.6　モデル調整作業履歴のグラフの構成

縦軸をモデルの評価指標、横軸を時間軸としてグラフを配置する。グラフは図 2.6 のように種類の異なる形状のノードを交互に配置する構成としており、それらを「モデルノード」、「意図ノード」と定義する。モデルノードは使用したモデル 1 つについての調整内容とテスト結果を表したものである。モデルノードの内部は 3 色のリングに分割されている。リングの色は過去に使用したモデルのうちの 1 つからの差分を表し、外側から順にデータ差

分（橙）、モデル構造差分（紫）、最適化アルゴリズム差分（緑）を意味する。彩度が高いほど比較対象のモデルからの差分値が大きいことを表しており、ビューの右端に配置したカラーバーで色と差分値の対応関係を確認できる。通常モデルノードは横軸方向には等間隔に配置することでモデルが使用された順序を示すが、特定のパラメータについて複数の設定を用意し同時に比較実験を行なった場合などには、並列で使用した複数のモデルをグループ化して近い位置に配置することもできる。このとき差分計算の対象とする過去のモデルの選出は図 2.7 のように行なう。この例では段階 1 で 1 つのモデルを使用し、段階 2 で 3 つのモデル、段階 3 で 2 つのモデルを並行して作成・使用したものとする。段階 1 から段階 2 のように、前の段階で複数のモデルが並列的に使用されていない場合には直前の段階のモデルと比較する（青の矢印）。段階 2 から段階 3 のように複数のモデルが比較の候補となる場合には、構造が似ているモデルとの比較（赤の矢印）や精度が高かったモデルとの比較（緑の矢印）を優先する。これにより、特定の構造のモデルの細かなパラメータ調整による発展の過程の観察や、精度が良く継続的に採用される可能性が高いパラメータ設定のモデルに注目して比較・評価をおこないやすくする。

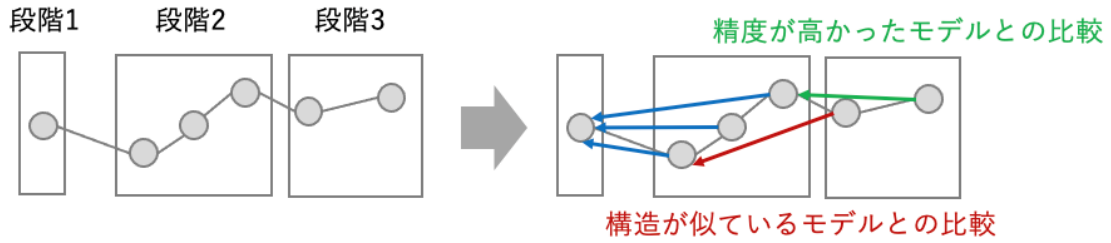


図 2.7　比較対象とするモデルノードの選出のイメージ

意図ノードは 2 つのモデルノードの間に配置し、内側と外側の 2 種類の色を割り当てる。内側の色は、その意図ノードの直後にあるモデルを作成した際の作業者の意図や方針を表したものである。意図や方針の情報は、ログの作成段階において事前に用意された選択肢の中から該当するものを選択でき、その内容に応じてノードの色が変化する（図 2.6）。一方意図ノードの外側の部分は、直後のモデルについての作業者の印象・評価を可視化するものである。現時点では「作成したモデルの精度が想定より下降（上昇）した」という場合に、ログに追記しておくことで赤色（青色）に着色するように設定している。想定外の結果となった部分をハイライトすることで、作業方針の分析や修正を行うべきポイントを明示する。

これらのノードは 3 種類のエッジ（実線、太い点線、細い点線）で接続し、モデルが使用された順序や発展の流れを示す（図 2.8）。複数のモデルノードを連結する実線のエッジは、それらのモデルが並列的にテストされたグループであることを意味する。太い点線は並列実験が終了したタイミングに該当しており、前のモデルの結果を確認し設定の変更を適用した上で次のモデルを使用したことを意味する。したがって太い点線のエッジの中間には意図ノードが配置される。細い点線のエッジは、後続のモデルがパラメータの設定を引き継いだ場合などのモデルの派生関係を表す。これは必ずしも連続で使用したモデル同士を接続するものではないが、特定の構造のモデルを微調整して使用した際の長期的な発展関係を確認できる。

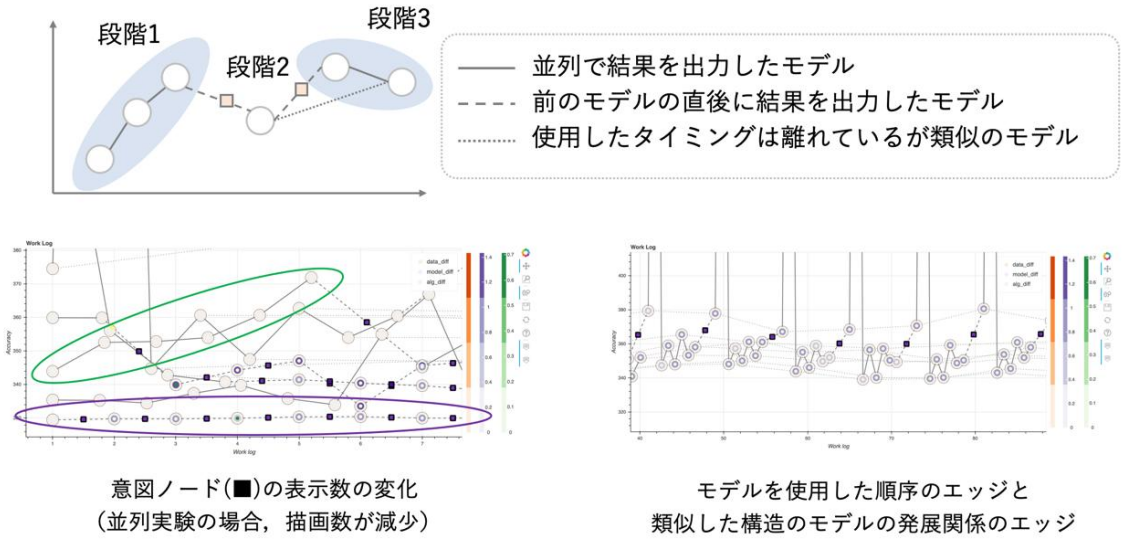


図 2.8　エッジの種類と表示例

これらの機能を用いて、3 つの事例の作業履歴データについての可視化結果を作成した。

(1) 機械学習モデルの可視化ツールの研究におけるユーザテスト時のモデル調整履歴
(2) Kaggle のコンペティション Digit Recognizer 参加者 1 名のモデル調整履歴
(3) Kaggle のコンペティション Prediction of Wild Blueberry Yield 参加者 6 名のモデル調整履歴

図 2.9 は(1)のデータを用いて、CIFAR-100 を学習した複数の ResNet モデルを順番にテストした際の精度とモデル構造の変化量を可視化した図である。

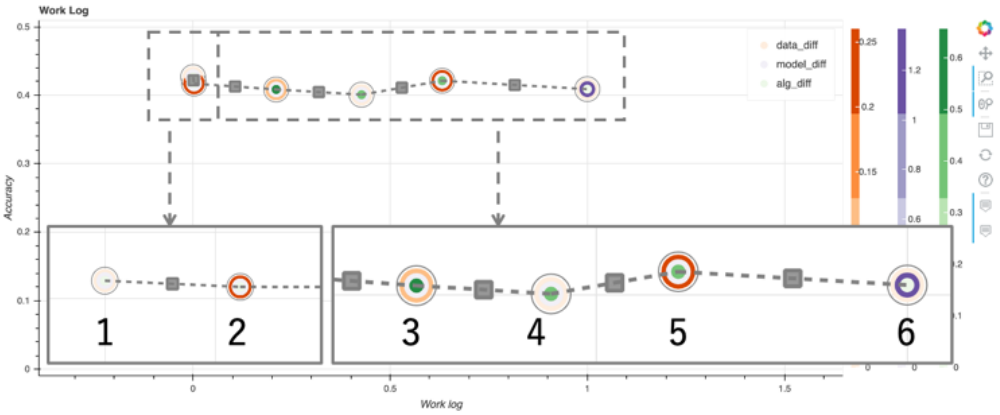


図 2.9　作業者のモデル調整履歴可視化の一例

[20]で実施されたハイパパラメータ調整のシナリオを参照し、表 2.1 のようにパラメータを変更しながらモデルの学習とテストを 6 回実行しログを収集した。lは学習率、mはモー

メンタムの値である。p、aは学習データに Random Erasing を適用した際の Erasing probability と Max erasing area である。dは使用した ResNet モデルの深さである。なお、この事例では意図ノードに関する情報の入力がなかったため意図ノードが全て灰色となっている。図 2.9 に記載した数字は表 2.1 の Index に対応する。2 と 5 ではモデルノード内のリングのうち、学習データの変化を表す一番外側のリングが赤でハイライトされている。モデル 1 から 2、および 4 から 5 に進む際に、学習データに関するパラメータであるp、aを大きく変更したことを反映している。同様に 3、4、5 ではノードが緑色にハイライトされており、最適化アルゴリズムに関する変化を示している。具体的にはlとmの変更を反映したものとなっている。また、6 のみノードが紫色で着色されており、モデル構造差分の変化が生じていることがわかる。

表 2.1 作業履歴の可視化に用いたモデルのパラメータ設定

Index	l	m	p	a	d
1	0.1005	0.9	0	0.4	18
2	0.1005	0.9	0.45	0.4	18
3	0.06	0.5495	0.409	0.55	18
4	0.06	0.919	0.409	0.55	18
5	0.0335	0.919	0.2955	0.287	18
6	0.0335	0.919	0.2955	0.287	34

続いて、(2)の機械学習コンペティションの参加記録を対象として可視化を行った例を紹介する。Kaggle で入門者向けに開催されているコンペティションの一つである Digit Recognizer に投稿されたパラメータ比較実験のコードとその解説の記事を 1 点選出し、パラメータ設定や得られた精度の情報を可視化ツールに入力した。実験は図 2.10 のように 5 段階に分かれており、5 段階目を除いて特定の要素（パラメータやモデル構造）に着目して最適な設定を探し、設定を引き継いで次の要素を検証するという作業を繰り返したものとなっている。図 2.11 はその可視化結果である。調整内容は主にモデル構造と Dropout 率を変化させるものであるため、いくつかのモデルノードの色には濃い紫色と緑色が見られる。意図ノードの色の変化を見ると、前半は緑色であることから計算資源の制約を考慮しながらパラメータの調整を進めており、紫色となっている後半では精度を高める設定を優先して取得しようとしていることがわかる。モデルの派生関係を表す細い点線のエッジに注目すると、派生元となっているモデル（細い点線のエッジの左端のモデルノード）は、必ずしも並列で実験したモデル（実線で接続されたモデルノード）の中で最適な精度を出したものではないことがわかる。意図ノードと同様に、計算資源のコストを考慮して「ある程度高い精度を出しながらもコストが大きくなりすぎないような設定」を優先して選出したことが反映されたものとなっている。

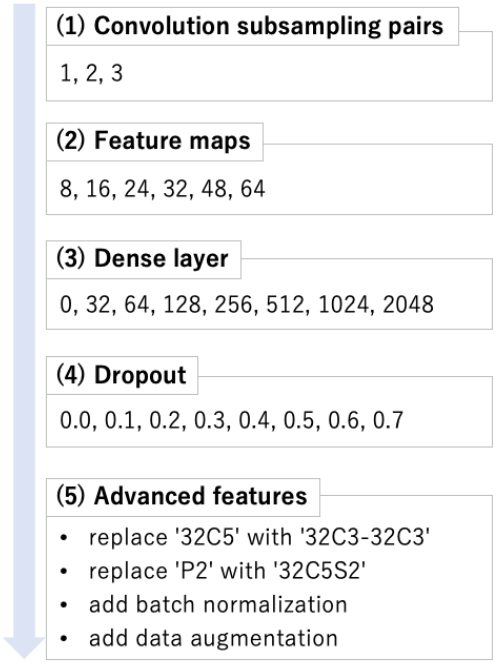


図 2.10　Digit Recognizer に投稿されたモデルの調整過程のパラメータ設定

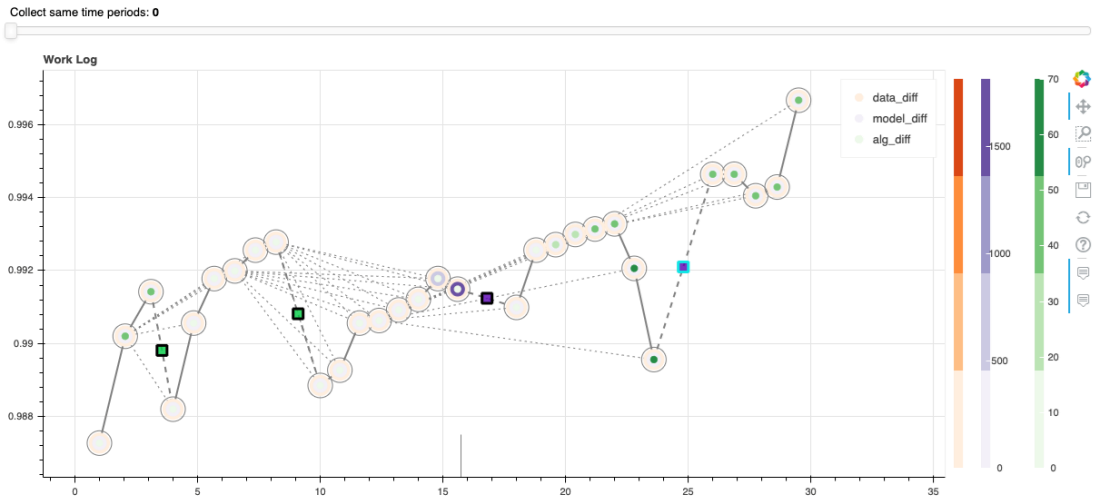


図 2.11　Digit Recognizer に投稿されたモデルの調整過程の可視化結果

(3)のデータは Kaggle のコンペティション Prediction of Wild Blueberry Yield の参加者 6 人分の作業記録であり、図 2.12 と図 2.13 はその可視化結果である。この事例では(1)(2)と異なりモデルの評価指標として MAE を使用しているため、縦軸の値が小さい方が良い結果を示している。図 2.12 は 6 人分の記録をまとめて描写したもので、図 2.13 は一人ずつ描画した場合である。作成されたグラフの全体的な形状やノードの数に差が見られ、MAE の変化の流れやテストを行った回数が作業者ごとに大きく異なっていることがわかる。 一方で作業履歴の長さに大きな差があり、図 2.12 では作業履歴の先頭を揃えて左から順にノードを配置した結果、画面の左端に極端にノードが集中しすぎており作業履歴の後半部分の比較が難しいという問題も起きている。また、一部のモデルの変化量が極端に大きいためにそ

のモデルノードのみが濃く着色され、他の相対的に変化の少ないモデルのノードの色の変化が少なくなり細かい差分値を観察しにくくなっているため、差分値とノードの色との対応関係の見直しが求められる。

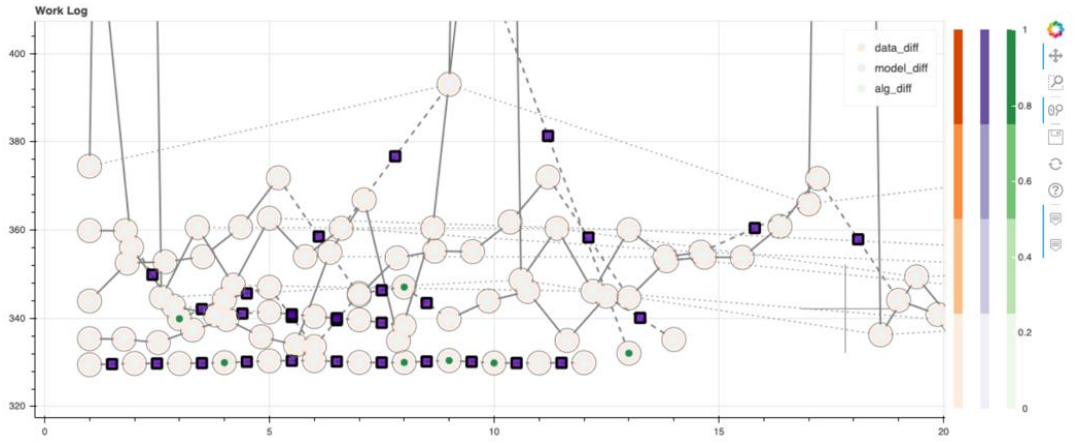


図 2.12　Prediction of Wild Blueberry Yield の参加者の作業記録の可視化例（6 人分）

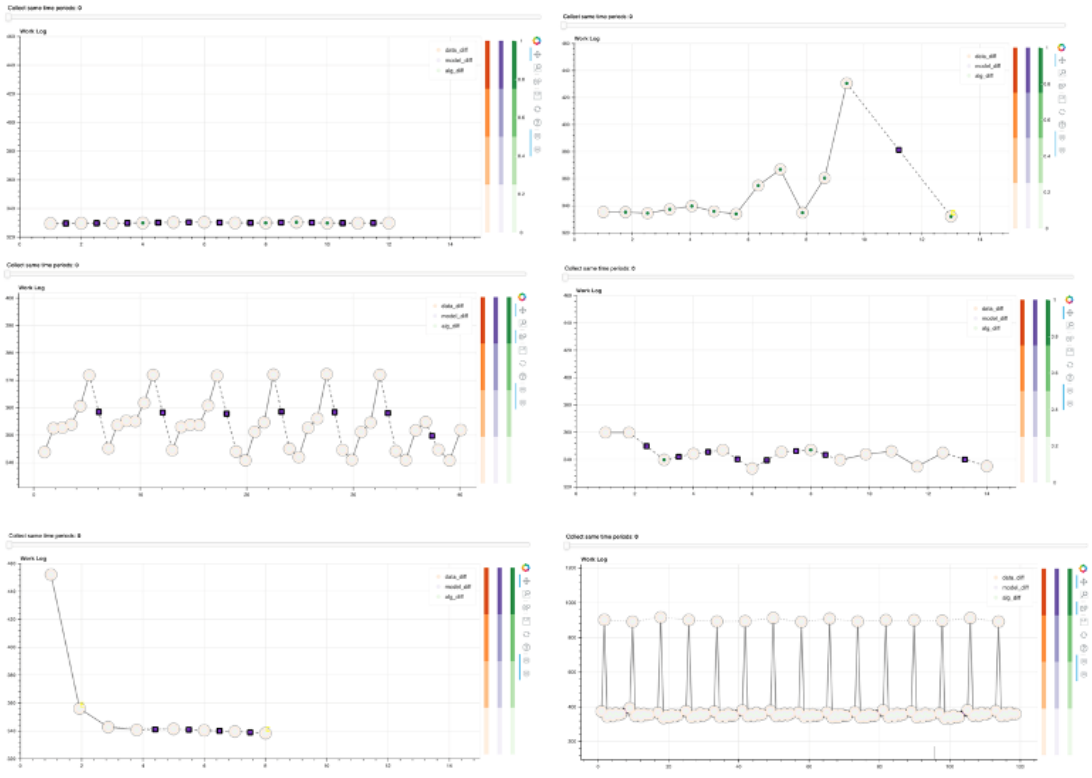


図 2.13　Prediction of Wild Blueberry Yield の参加者の作業記録の可視化例（個別）

2.3 今後の方針

今後は以下のような方針で可視化機能の拡張に取り組みたい。まず今回使用したデータについて、より詳細な差分値計算を導入して多様な項目での比較を行なえるようにする。その後、多数の作業者の作業パターンの比較を行い、モデルまたは作業者同士の類似度や分類結果を俯瞰的に可視化する。これらのような可視化結果を観察することで、作業者のスキルレベルの推定や作業の特徴（作業パターン）の分類を行えるようにしたい。さらに、差分値の計算結果や可視化結果の内容とともにモデル改善策の推薦結果を提示することを目指す。

また、現時点では手動でのモデル設計を前提としているが、NAS（Neural architecture search）などと組み合わせることで、より幅広い事例に本手法を適用できるように拡張することも有用と考えられる。

本手法の評価のため、生成された可視化結果画像を観察する形での可視化の機能や可視化された内容に関するユーザテストも実施したい。

3 データ拡張による品質向上

深層学習を用いた画像認識の中でも物体の識別の品質評価に焦点をあてて研究を進める。学習データを加工することでデータを増強する Data Augmentation（データ拡張）が画像識別の品質評価にどのように影響を与えるのか、品質を改善するためにはどうすればいいのかについての検討を行った。

特に議論の中で「品質評価に関して論文では state-of-the-art を競うことが多いが、産業応用を考えた場合には精度が 98%という場合にはデータセットを変えた場合にでも 98%の精度が出ることが望ましい。それが保証されるなら 98%ではなく 80%でも構わない」という意見があった。これまでに研究会や国際会議で発表されている手法においてもデータセットに過学習することで精度が上がっているように見えるという問題は散見している。

3.1 学習データ数と識別率の関係（予備実験）

最初に予備実験として以下の 2 点を検証した。

予備実験① 学習データを増加していくことで識別率はどのように変化していくのか
予備実験② 学習データを変えることで識別率はどのように変化するのか

両予備実験において深層学習のモデルとして WideResNet28-10 を用い、データセットは CIFAR10 を用いる。CIFAR10 は 10 クラスの画像で構成され、各クラス毎に 6000 枚のデータ（合計 6 万枚）が用意されている。これらの画像を左右反転に加え、上下左右にずらすことでデータを 10 倍に増やし 60 万枚とし、この 60 万枚の世界がデータによって構成される全ての世界であると仮定する。予備実験①では 60 万枚の全ての世界の中から N 枚が観測されたとして学習を行い、全ての世界の 60 万枚で評価を行い識別率を求める。つまり N = 600000 で学習する際には世界の全てが観測できたと仮定して識別機を作成し、識別率を求めることとなる。これは十分に識別性能が発揮されるネットワークモデルにおいては識別率が 100% になることが報告されているが、実際にそうなるかを確かめる実験でもある。識別結果を図 3.1 に示す。横軸はデータ数の対数グラフ、縦軸は識別率である。

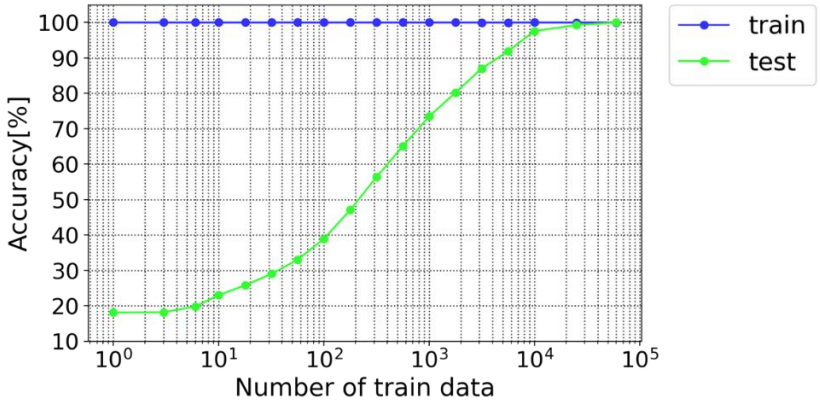


図 3.1 学習データ数と識別率の関係

これまでの研究で報告されているように 60 万画像という大量のデータであっても世界の全ての画像を学習データに利用することが可能であれば識別率は 100%を実現できることが分かる。また、各クラス 100 枚の学習データで 60 万枚の画像を識別すると 40%程度の識別精度であることが分かる。そこで予備実験②を行うための学習データ数は 10 クラス×100 枚の N = 1000 枚とした。オリジナルの各クラス 6000 枚の画像を 100 枚×60 セットに分割し、60 セットの学習データを用いてニューラルネットワークを学習し、60 万枚の画像で評価を行った。それぞれの学習データに対する識別精度のヒストグラムを図 3.2 に示す。

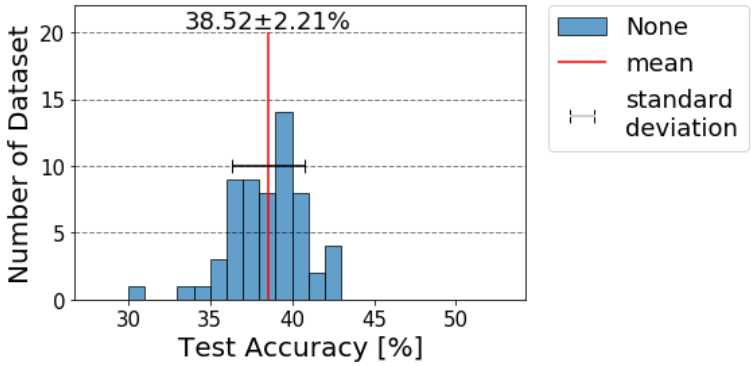


図 3.2 学習データの違いによる識別精度のヒストグラム

学習データがある組み合わせの場合には 42%の識別率が得られる一方で、学習データがある組み合わせの場合には識別率は 30%しか得られないことが分かり、これまでの識別精度だけの評価指標では品質の評価という観点では適していないことが明らかになった。なお、平均は 38.52%であり標準偏差は 2.21 である。この予備実験の結果から品質を評価する際には従来のような識別率だけで評価するのではなく、品質の安定性を評価するためには分散や標準偏差を考慮した評価が必要であることが分かる。

3.2 識別率の平均値と標準偏差の評価

これまでに提案されている様々なデータ拡張について以下の実験③を行うことで各データ拡張手法を品質の観点で評価した。

実験③ 様々なデータ拡張に対して予備実験②の方法で平均識別率と標準偏差を求める

基本的な実験設定は予備実験②と同じである。データ拡張としては変形や回転などの幾何学変換として Skew, Scale Augmentation, Shear, Rotate, Rotate Zoom の 5 種類、ノイズ付加として PCA Color Augmentation, Gaussian Noise, Patch Gaussian, Salt Pepper Noise の 4 種類、色情報の非線形変換として Gamma Transform, Contrast Transform の 2 種類、ぼかし処理として Gaussian Filter と Smoothing Filter の 2 種類、マスク処理として Cutout, Random Erasing, Cut Mix の 3 種類、データ合成として Manifold-Mixup を実装し、各評価を行った。評価の結果を図 3.3 に示す。

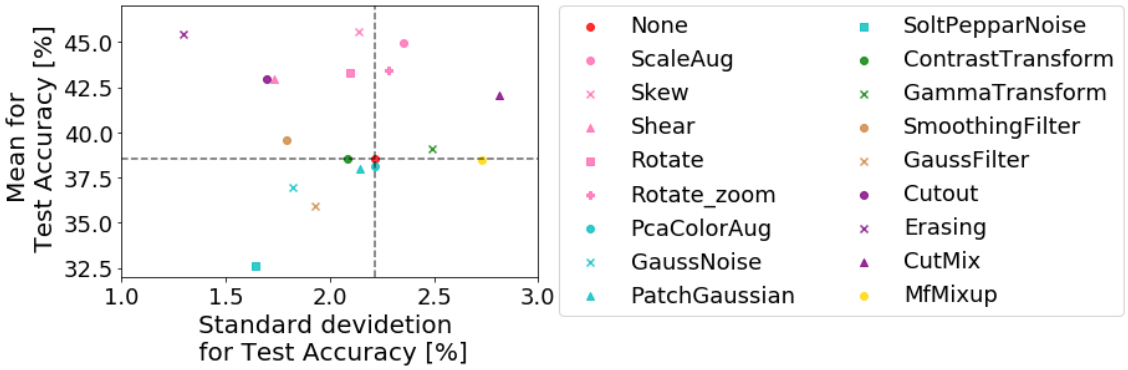


図 3.3　データ拡張による品質評価

それぞれのデータ拡張手法について縦軸に識別精度、横軸に標準偏差をプロットした散布図となっている。データ拡張を何もしていない None（赤点）を基準点として、基準点の左上の第二象限は精度を向上させながら標準偏差が小さくなっているため識別精度を上げながらもばらつきを抑えるデータ拡張であると言うことができ、品質を高める手法であると結論付けることができる。一方で基準点の右上の第一象限は識別精度の向上は認められるものの標準偏差が大きくなっているため、データによっては効果があるが、そのばらつきは大きいと考えられる。基準点の左下の第三象限は認識精度は下がるもののばらつきは小さくなっているため、識別精度は必ずしも上がらないが、品質を保証するという意味では貢献できるデータ拡張手法であるといえる。基準点の右下の第四象限は識別精度を下げつつばらつきを大きくするというデータ拡張に適していない処理がプロットされるエリアであるが、このようなデータ拡張手法は今回の適用例には見られなかった。

従来の state-of-the-art な評価という観点では上に行くほど良いデータ拡張手法であると言えるが、標準偏差を考慮した品質という点で考えると左上に行くほど良いデータ拡張手法であると言える。傾向としては幾何学変換のデータ拡張手法（図中のピンクのプロット）は上へと評価が上がる傾向にあり、ノイズ付加（水色のプロット）は左下へと下がる傾向にある。一方でマスク処理（紫のプロット）は左上へと上がる傾向にあり、データ拡張の中でも特に優れている手法であると評価することができる。

3.3　データ拡張手法の組み合わせの効果

次にデータ拡張としてばらつきを下げる効果のあった幾何学変換とノイズ付加、マスク処理の 3 つについてそれぞれのデータ拡張を全て行う場合とランダムに行う場合に対して一定の確率でデータ拡張する場合と徐々にデータ拡張をする割合を増やした場合を組み合わせて、識別率がどのように変化するかを実験④で検証した。実験④の検証項目は以下の通りである。

実験④ 品質を向上させるデータ拡張手法の組み合わせを明らかにする

幾何学変換、ノイズ付加、マスク処理の 3 つのデータ拡張について、それぞれのデータ拡

張を全て行う場合（All）、ランダムに行う場合（Random）、一定の割合で行う場合（const）、行う割合を線形で増やす場合（Linear）で実験を行い散布図を作成した。図 3.4 にその結果を示す。

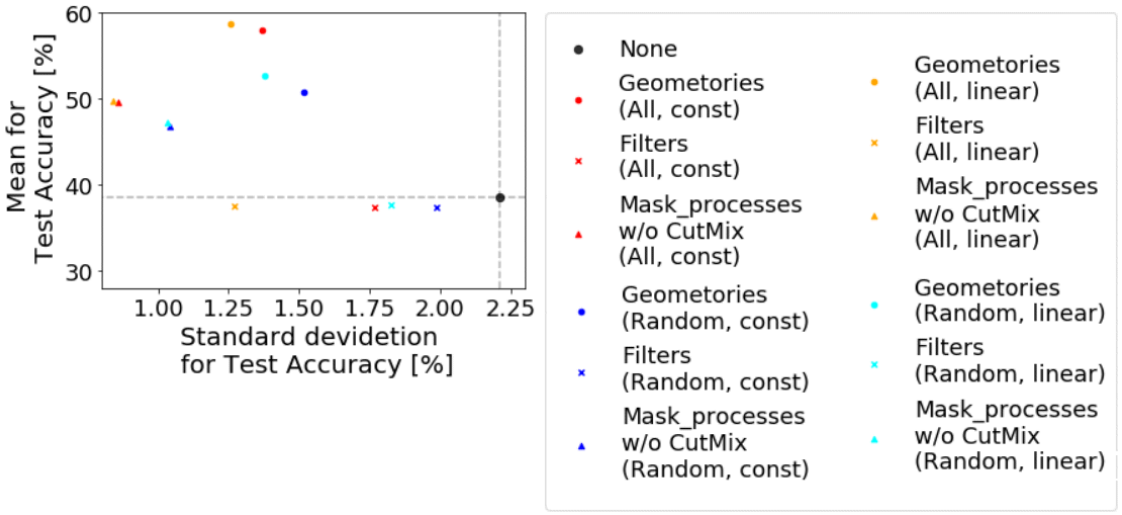


図 3.4　データ拡張手法の組み合わせ実験による品質評価

実験の結果から幾何学変換とマスク処理のデータ拡張においてはランダムに行うよりも全てを行う方が品質がよくなり、一定の確率で行うよりも徐々に行う確率を増やしていった方が識別精度を高く、かつ標準偏差を小さくできることが分かった。一方でノイズ付加に関しては識別精度を上げることはできなかったが、大きく下げることなく標準偏差を小さくする効果があることは分かった。

3.4　まとめ

これまでの多くの論文で行われている識別精度だけで機械学習システムの品質を評価するのではなく、分散や標準偏差も考慮することで品質を評価する必要があることが明らかになり、品質を向上させるためのデータ拡張としてはマスク処理と幾何学変換の組み合わせが有効であることが分かった。

4 データ拡張の適用法の改良による品質改善

本章では、ニューラルネットワーク学習におけるデータ拡張の新しい適用法の開発を行い、実験を通して学習品質への影響を評価した結果について述べる。

4.1 研究目的

データ拡張は、データに変形を加えることでデータ数を増やす技術であり、訓練データ数が少ないときに性能が落ちてしまうという性質をもつ深層学習において、高い効果を発揮する。一方で、データ拡張の有効性は用いるデータに強く依存するため、データ拡張手法の選択や各手法がもつパラメータを適切に設定しなければならない。しかし、データ拡張の理論解析は難しく、汎用的な使用法が確立していないという現状があり、経験的、慣例的な使用がなされるケースが多い。これは意図せず不適切な使用をすることにつながり、学習の品質を損ねてしまう。実際に、各データ拡張手法の変形量、例えばマスクサイズや回転角度などを不適切な値に設定してしまうことにより、学習の性能が落ちてしまうケースや、実際に用いる実データに対して、どのようなデータ拡張手法を選択すればよいのかと頭を悩ませるケースがよく見受けられる。

そこで、データ拡張の経験的な使用からの脱却を目指して、本研究はデータの多様性に焦点を当てた。多様性を高めることはデータ拡張の本質的な目的であり、多様性の増加が汎化性能の向上に大きく影響を及ぼすことは、[21]の研究において実証されている。近年、複数のデータ拡張操作からランダムに選ばれた操作を学習中に動的に適用する RandAugment [22]という手法が注目されているが、これは多様性を大きく向上させる半面、調整が必要なパラメータも多く、効果的に利用するのは容易ではない。本研究では、データの多様性に関連したデータ拡張の適用法として、以下の 2 点を新たに提案し、それぞれのアルゴリズムの改良および性能の評価を行った。

- ニューラルネットワークの中間層を含めた様々な層でデータ拡張を適用し、適用層の自動的な最適化を行う(4.2 節)。
- 有力なデータ拡張手法である Mixup 法[23]を改良し、サンプルの新しい混ぜ合わせ方を提案する(4.3 節)。

さらに、データ拡張を利用するうえで、実用上問題となってくるのが計算コストである。データ拡張は多数の種類が存在するうえ、各手法は前述したような回転角度などの独自のハイパーパラメータをもっている。適切なデータ拡張を選択するために、何度も学習を行わなければならず、高い計算コストを要する。そこで、効率よく適切なデータ拡張を見つけるために、以下の方法を提案した。

- Affinity, Diversity を考慮した新しいデータ拡張指標を用い、短い学習ステップ数でデータ拡張ポリシーをうまく探索する(4.4 節)。

また、4.3 節に示す研究成果は、国際誌 Neural Computing and Applications[24]において発表された。

4.2 データ拡張の適用層の改良

4.2.1 中間層におけるデータ拡張

一般に、データ拡張は入力データに適用するものであると考えられているが、ニューラルネットワークでは中間層で出力された特徴量を取り出し、データ拡張を適用することが可能である。これに関していくつか先行研究が存在するが、手法を mixup[23]に限定する Manifold mixup[25]や、ほかにも特殊なネットワークやデータを必要とするなど、汎用性の低い手法が多い。本研究では、画像データに対して用いられるアフィン変換やマスク処理といった様々なデータ拡張手法を、中間層で適用することを考えた。CNN では、階層的に特徴が抽出されるため、ミニバッチごとにランダムに選ばれた様々な層でデータ拡張を行うことで、多様なサンプルが生成される。入力画像への適用と同じように、中間層で得られる特徴マップに対してデータ拡張を適用することができるので、実装も容易である。

実際に入力画像および特徴マップに対して、マスク処理と平行移動を適用した例を図 4.1 に示す。ここでは、学習中のモデルにサンプルを入力し、同じパラメータ（マスク位置、移動量）に設定したデータ拡張を異なる層で適用した直後の画像を、サイズを揃えて上段に表示している。そのサンプルの最終層における特徴マップを下段に示しているが、それらはデータ拡張を適用した層によって異なる画像となっている。この結果から、様々な層でデータ拡張を行うことは、生成データの多様性の向上につながり、入力データのみにデータ拡張を適用する場合とは異なった学習が行われることがわかる。

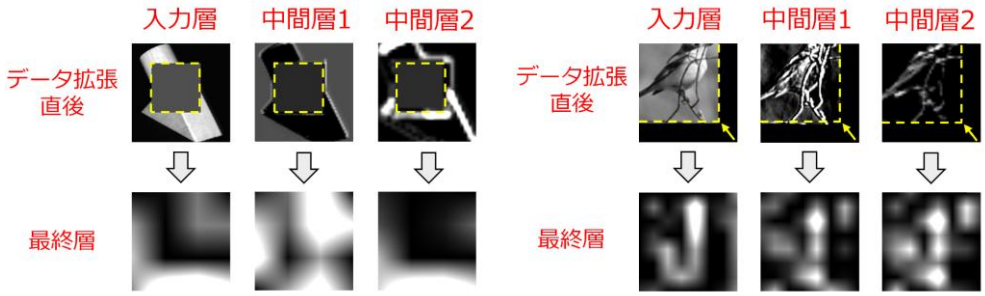


図 4.1 入力画像および中間層で得られる特徴マップにデータ拡張を適用した例

入力層におけるデータ拡張と特徴マップへのデータ拡張の性能を比較するために、様々なデータ拡張を用い、教師ありで学習したモデルのテスト精度を求めた。ここでは、CIFAR-10, Fashion-MNIST, SVHN（補助データを含まない）データセットを用いて、WideResNet28-10 を 200 エポックの間学習した。結果を図 4.2 に示している。各図において、横軸は従来手法（Input DA）、縦軸は提案手法（Latent DA）の精度[%]を表している。これらの結果からわかるように、従来手法よりも提案手法の方が高い精度を示す傾向があり、Crop を用いた結果のように、従来手法では精度が低くなる場合においても、提案手法は高い精度を与えた。この結果から、ランダムな層へのデータ拡張の適用により生成された多様なサンプルは、性能の向上に効果的であることがわかった。

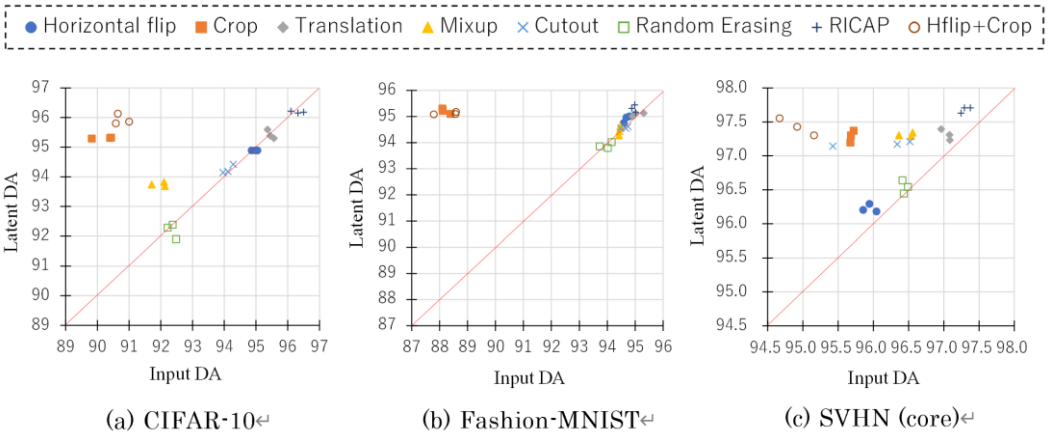


図 4.2　Input DA と Latent DA によるテスト精度の比較

4.2.2　データ拡張に最適な層の選択

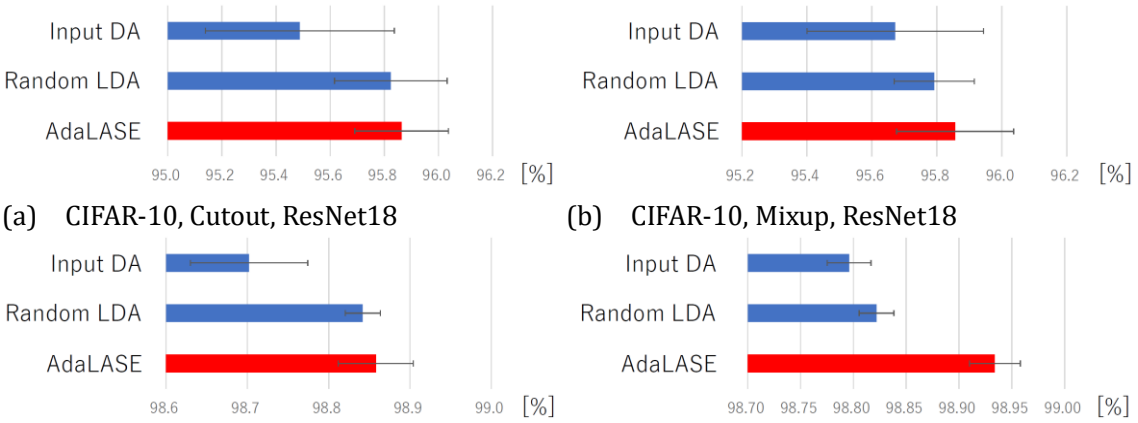
中間層でデータ拡張を行うことは効果的であることがこれまでの研究によりわかったが、どの層でデータ拡張を行うのが最適であるのかという問題が出てくる。データ拡張を行う層を変えながら訓練を繰り返し行い、バリデーション精度の値を比較することで最適な層を発見することは可能であるが、全体の訓練時間が増えてしまうため、非効率的な方法であり、実用的ではない。そこで本研究では、1 回の訓練でデータ拡張に最適な層を動的に発見する方法を開発することに取り組んだ。

アプローチとしては、採択率というパラメータを各層で用意し、学習中、採択率の更新を行い、採択率に従って確率的に選ばれた層でデータ拡張を適用する。採択率の更新は、以下に示すような勾配降下法を用いて行う。

$$q_l \leftarrow q_l - \eta \frac{\partial L_{val}}{\partial q_l}$$

ここで、q_lは l 層の採択率、L_{val}はバリデーションデータを入力したときの誤差の値、ηは更新のステップ幅を表す。実際には、バリデーションデータの値をアルゴリズムに含めるべきではないので、データ拡張を加えた訓練データで擬似的にバリデーションデータを作り出し、更新を行っている。学習の初期状態では、すべての採択率を和が 1 になるように均等な値にセットし、ミニバッチごとに採択率の更新を行う。これにより、データ拡張の適した層の採択率は増加し、適さない層の採択率は減少されるといった最適化が行われ、その結果、汎化性能が向上することが期待される。

この手法を Adaptive Layer Selection (AdaLASE)と名付け、従来手法とテスト精度を比較した。データは CIFAR-10 および MNIST、モデルは ResNet18 および中間層を 1 層もつ多層パーセプトロン（MLP）を用い、入力にデータ拡張、ランダムな層でデータ拡張、および AdaLASE を用いた。拡張法として Cutout および Mixup を用いた。図 4.3 の結果において、初期値を変えた 5 回の精度の平均と標準偏差が、手法ごとに示されている。これらの結果から、AdaLASE は従来手法と同等以上の性能が出せることがわかった。今後の計画として、学習中の採択率の変化を見て、層がどのように選択されていくのか、AdaLASE が正しく機能しているかについて、詳しい解析を行う。



(a)　CIFAR-10, Cutout, ResNet18　　(b)　CIFAR-10, Mixup, ResNet18

(c)　MNIST, Cutout, MLP　　(d)　MNIST, Mixup, MLP

図 4.3　AdaLASE と従来法とのテスト精度の比較

4.3　Mixup の改良による新しい混ぜ合わせ方法の提案

4.3.1　Feature Combination Mxiup

実際の訓練において、データ拡張は、例えば切り抜き、回転、および反転のように、複数の手法を同時に用いることが多い。そこで、このように複数の手法を用いるときの、手法間の相性に着目し、特にデータの多様性という観点から相性を議論することを考える。そのためのアプローチの第一歩として、既存の手法を変形した新しい手法を提案し、元の手法と同時に利用することによって、生成されるデータの多様性を高め、性能の向上を図ることを考えた。多様性を定式化する方法については、[21]の研究において述べられているが、本研究ではまず、精度だけを比較し、提案手法によって性能が向上するかどうかを検証する。

ここでは、データ拡張手法の一つである Mixup[23]の改良を行った。これは、2 つのサンプルの線形補間によって新たなサンプルを生成する手法であり、次式に示されるように、入力値およびラベルのそれぞれについて同じ比率で線形補間をとる。

$$\begin{cases} \tilde{x} = \lambda x_i + (1-\lambda)x_j \\ \tilde{y} = \lambda y_i + (1-\lambda)y_j \end{cases}$$

ここで、(x_i, y_i)および(x_j, y_j)はi番目およびj番目のサンプルの入力値を表し、λはベータ分布からサンプルした混合率を表す。本研究で、Mixup を対象に選んだ理由は、その汎用性の高さにあり、画像だけでなく時系列データなど多くの数値データに利用することができるため、Mixup 法を改良することによる影響が大きいと考えたからである。

Mixup をニューラルネットワークの中間層でも行えるように改良したものは Manifold mixup[25]と呼ばれるが、いずれの Mixup も 2 点間の線分上というデータ分布上の局所的な範囲にしかサンプルが生成されず、またその線分上の点の性質が非線形に変化する分布をもつデータセットに対しては不適切である。

本研究[24]で提案する Feature Combination Mixup（FC-mixup）は、従来の Mixup とは異なる方法でサンプルを混ぜ合わせる手法であり、その概要を図 4.4 に示す。同じミニバッチ内に含まれる 2 つのサンプル A と B が、ランダムに選ばれた層において、Z_AとZ_Bという特

徴量を出力するとする。dをその層の特徴量の総数とすると、FC-mixup は、Z_Aから$d\lambda$個、Z_Bから$d(1-\lambda)$個の特徴量をランダムに抽出し組み合わせて新たなサンプルZ_Xを生成する。その組み合わせの数は、一つのλの値について多数考えられるため、乱数に応じて異なったデータが生成され、したがってデータ分布上の広い範囲にサンプルを生成することができる。FC-mixup は次式のように表現されるので、この式が満たされるようにZ_AとZ_Bを混合する。

$$|Z_A \cap Z_X| = d\lambda$$

このように 2 つのデータがもつ各パーツの組み合わせにより新たなデータを生成する技術は、Puzzle Mix[26]においてもみられるが、これは対象が入力画像に限定される。また、Adversarial mixup resysthesis[27]において類似した手法が利用されているが、オートエンコーダでの使用に限られており、FC-mixup はより汎用的な使用を想定して設計されている。

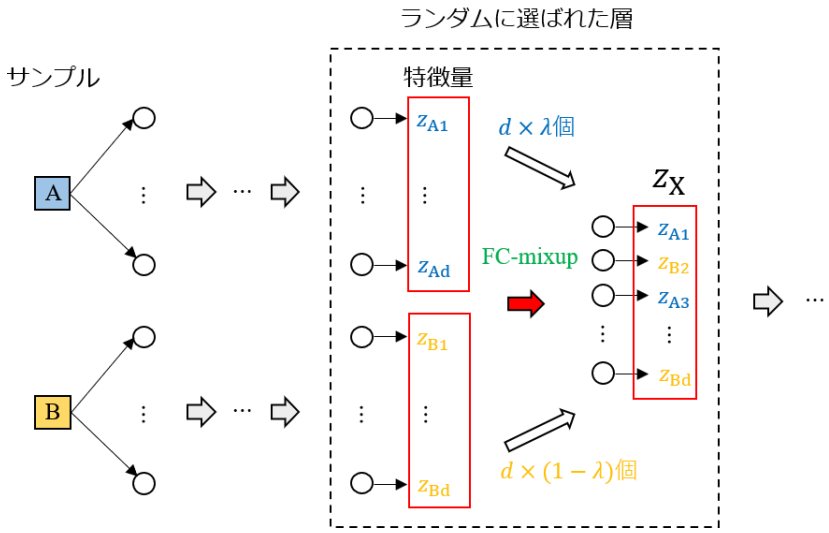


図 4.4　FC-mixup の概要

実験では、複数の多クラス分類データセットを用い、従来手法（データ拡張なし、入力層での mixup[23], Manifold mixup[25]）と提案手法（FC-mixup, Hybrid 法）とで、テストデータの識別精度を比較した。データには、MNIST, CIFAR-10, CIFAR-100, および SVHN を用いた。モデルには、小さい畳み込みニューラルネットワーク(CNN)および ResNet18 を用いた。フルサイズのデータに加えて、1,000 サンプルをランダムに抽出した少数データでも実験を行った。また、Cutout と Mixup を組み合わせた有力なデータ拡張法である CutMix とも比較を行った。初期値を変えた 5 回の試行における平均と標準偏差を求めた。

表 4.1 に結果を示しており、太字は最も高い精度を表している。この結果から、多くの場合で提案手法 FC-mixup(channel-wise)は最も高い精度を出していることがわかる。Manifold mixup や CutMix[28]などの既存手法よりも優れており、FC-mixup の性能の高さが示される結果となった。汎用的なデータ拡張であるので、Manifold mixup に加えて FC-mixup を利用することは、実用的にも有用であると考えられる。

表 4.1　多クラス分類データにおけるテスト精度の比較

	CIFAR-10 ResNet18	SVHN ResNet18	CIFAR-100 ResNet18
No mixup	93.75 (±0.42)	96.28 (±0.05)	74.61 (±0.29)
Input mixup	95.20 (±0.32)	96.72 (±0.13)	76.84 (±0.27)
Manifold mixup	94.92 (±0.45)	97.10 (±0.08)	**78.58 (±0.39)**
FC-mixup (channel-wise)	**95.23 (±0.16)**	**97.12 (±0.06)**	78.54 (±0.20)
CutMix	95.04 (±0.15)	96.96 (±0.14)	77.38 (±0.29)
	CIFAR-10 (1000) small CNN	SVHN (1000) small CNN	
No mixup	58.98 (±0.93)	72.45 (±1.13)	
Input mixup	61.09 (±0.15)	74.44 (±0.57)	
Manifold mixup	60.04 (±0.39)	74.18 (±0.52)	
FC-mixup (channel-wise)	**62.60 (±0.91)**	**78.05 (±0.40)**	
CutMix	58.29 (±0.59)	72.25 (±0.58)	

4.3.2　ピクセル単位で行う FC-mixup

4.3 節で提案した FC-mixup は、MLP であればユニット単位で、CNN であればチャネル単位でサンプルが混ぜ合わせられるというものであった。しかし、CNN に関しては、ピクセル単位で混ぜ合わせることも可能である。チャネル単位で行う FC-mixup を FC-channel、ピクセル単位で行う FC-mixup を FC-pixel と呼ぶこととする。図 4.5 にそれらの違いを示している。FC-channel では、各サンプルのチャネルを組み合わせて新しいサンプルのチャネルを作るのに対し、FC-pixel では、各サンプルのピクセルを組み合わせて新しいサンプルのピクセルを作る。FC-pixel におけるピクセルの混ぜ合わせ方はすべてのチャネルで同じである。

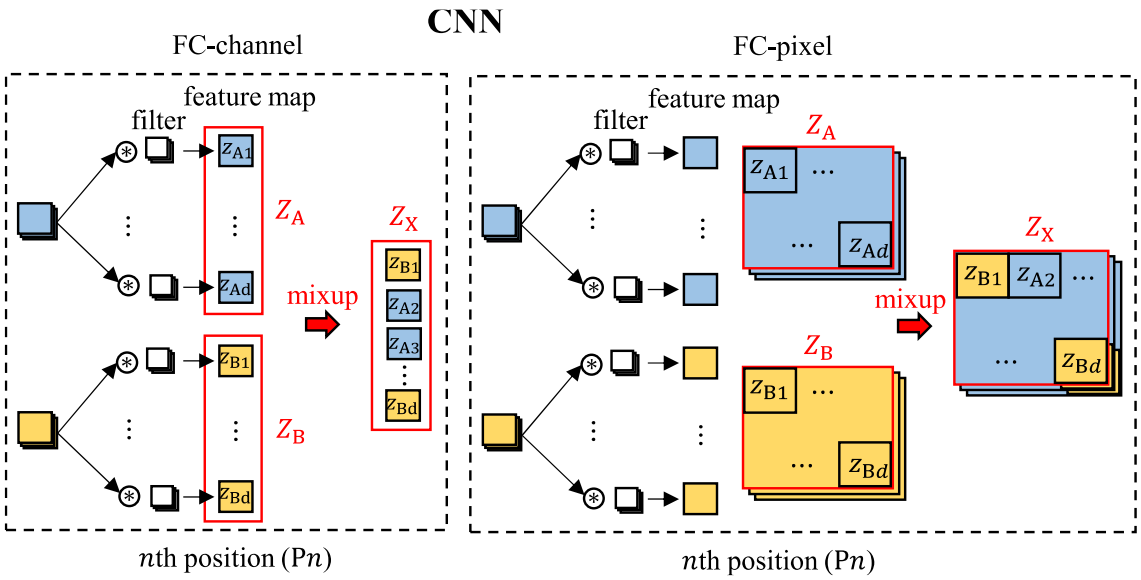


図 4.5　CNN における 2 種類の FC-mixup

Manifold mixup および 2 種類の FC-mixup によって生成されたサンプルを可視化した結果を図 4.6 に示す。これは、2 つのサンプルをある中間層で混ぜ合わせ、生成された特徴マップを示しており、各サンプルは複数のチャネルを有しているが、そのうち 3 つのチャネルを取り上げて示している。この結果から、Manifold mixup および 2 種類の FC-mixup によって生成された特徴マップは大きく異なっていることが見て取れる。Manifold mixup を用いた場合、2 つのサンプルを重ね合わせた画像が生成されている。FC-channel を用いた場合、チャネルごとに、元の 2 つのサンプルのいずれかの特徴マップが採用されていることがわかる。この例では、ch 1 は Sample 2, ch 2 は Sample 2, ch 3 は Sample 1 の特徴マップが採用されて新しい画像が生成されている。FC-pixel を用いた場合、どちらかのサンプルのピクセルを採用して新たな特徴マップを生成するので、各チャネルの画像が大きく変形しているようにみえる。

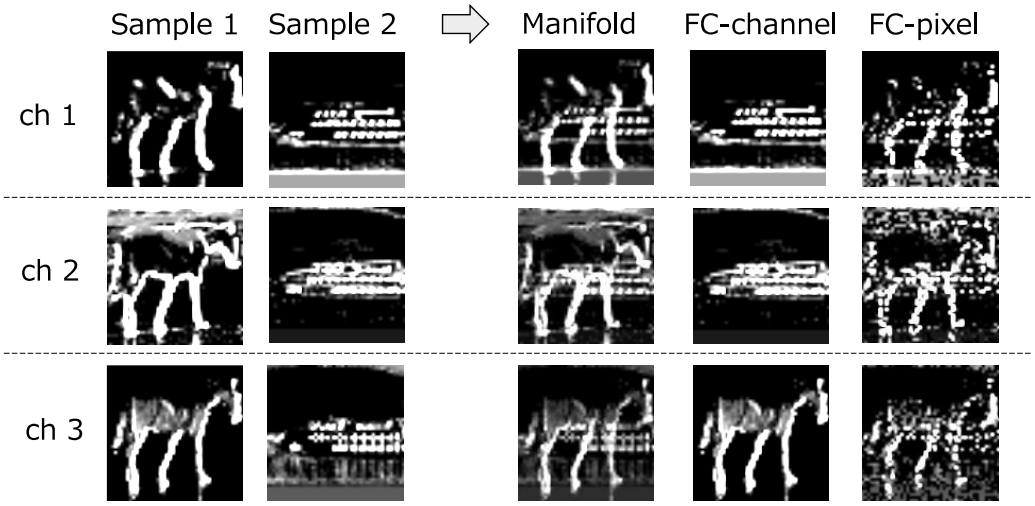


図 4.6 各 Mixup により生成された特徴マップの可視化

4.3.3 Mixup のハイブリッド利用

生成データの多様性を高めるために、FC-mixup と Manifold mixup[25]を併用する Hybrid 法を考える。ここで、図 4.7 に示すように 2 種類の Hybrid 法が考えられる。Hybrid 1 は、同じ 2 つのサンプルに対して Manifold mixup と FC-mixup を別個に適用して得られた画像を 0.5 倍して足し合わせる手法である。Hybrid 2 は、まず FC-mixup を適用し、画像を生成した後、その生成画像のうち 2 つを Manifold mixup によって混ぜ合わせる手法である。

FC-channel および FC-mixup を単体で用いた場合と、2 種類の Hybrid 法において用いた場合を、既存手法の中で最も精度が高いものと比較した。表 4.2 の結果から、提案法は既存手法よりも高い精度を示した。Hybrid 法が常に最高の精度を示したわけではなく、どの FC-mixup が最適であるかはデータセットやモデルなどの条件に依存することがわかった。

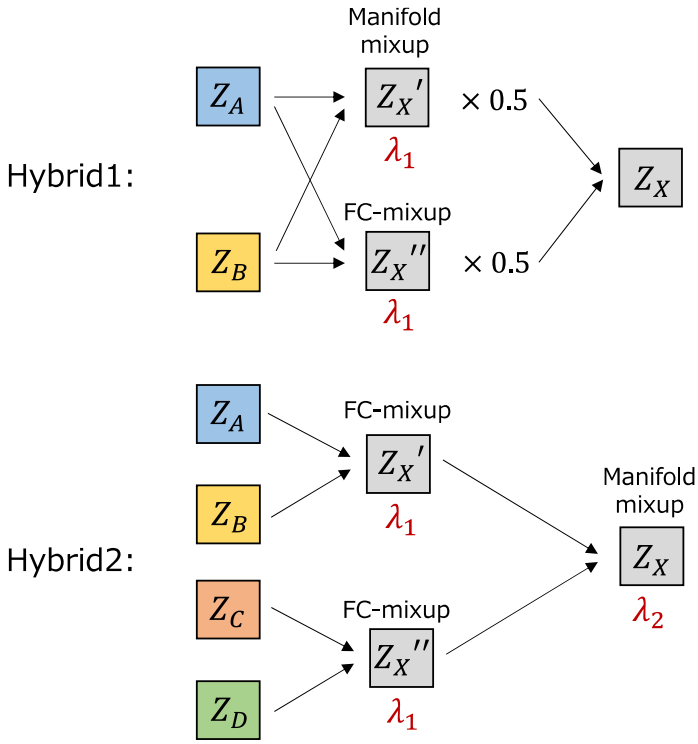


図 4.7　2 種類の Hybrid 法の提案

表 4.2　複数の FC-mixup 法と既存手法とのテスト精度の比較

	CIFAR-10 ResNet18	SVHN ResNet18	CIFAR-100 ResNet18
Conventional method	95.20 (±0.32)	97.10 (±0.08)	78.58 (±0.39)
FC-mixup (channel-wise)	**95.23 (±0. 16)**	97.12 (±0.06)	78.54 (±0.20)
FC-mixup (pixel-wise)	95.08 (±0.16)	96.95 (±0.10)	**78.74 (±0. 11)**
Hybrid 1 (channel-wise)	95.06 (±0.18)	96.93 (±0.07)	78.68 (±0.37)
Hybrid 1 (pixel-wise)	94.88 (±0.20)	96.97 (±0.09)	78.72 (±0.25)
Hybrid 2 (channel-wise)	95.00 (±0.33)	**97.15 (±0. 11)**	78.72 (±0.23)
Hybrid 2 (pixel-wise)	95.00 (±0.24)	97.01 (±0.10)	**78.74 (±0. 10)**
	CIFAR-10 (1000) small CNN	SVHN (1000) small CNN	MNIST (1000) small CNN
Conventional method	61.09 (±0.15)	74.44 (±0.57)	96.54 (±0.10)
FC-mixup (channel-wise)	**62.60 (±0. 91)**	78.05 (±0.40)	96.74 (±0.07)
FC-mixup (pixel-wise)	59.61 (±0.53)	75.41 (±0.57)	96.83 (±0.23)
Hybrid 1 (channel-wise)	61.08 (±0.77)	76.09 (±0.64)	96.52 (±0.15)
Hybrid 1 (pixel-wise)	59.89 (±0.70)	74.63 (±0.62)	96.61 (±0.20)
Hybrid 2 (channel-wise)	61.54 (±0.92)	**78.37 (±0. 74)**	96.79 (±0.12)
Hybrid 2 (pixel-wise)	59.60 (±0.64)	75.81 (±0.53)	**96.94 (±0. 21)**

4.4　Affinity, Diversity を用いたデータ拡張ポリシー探索の効率化

4.4.1　Affinity, DIversity

データ拡張には多数の手法が存在し、各手法はハイパーパラメータを有している。そのため、適切な拡張ポリシーを決めるためには、多くの学習を行う必要があり、計算コストがかかってしまう。本研究では、拡張ポリシー探索に要する計算コストの軽減を目的とした新たな手法の開発に取り組んだ。

一般に、適切な拡張ポリシーの探索は、バリデーション精度を用いて行われるが、本研究では、Affinity、Diversity というデータ拡張用に提案された指標[21]に着目した。図 4.8 に示されるように、Affinity は元データと拡張後のデータの分布の重なり具合を表し、以下の式で計算される。

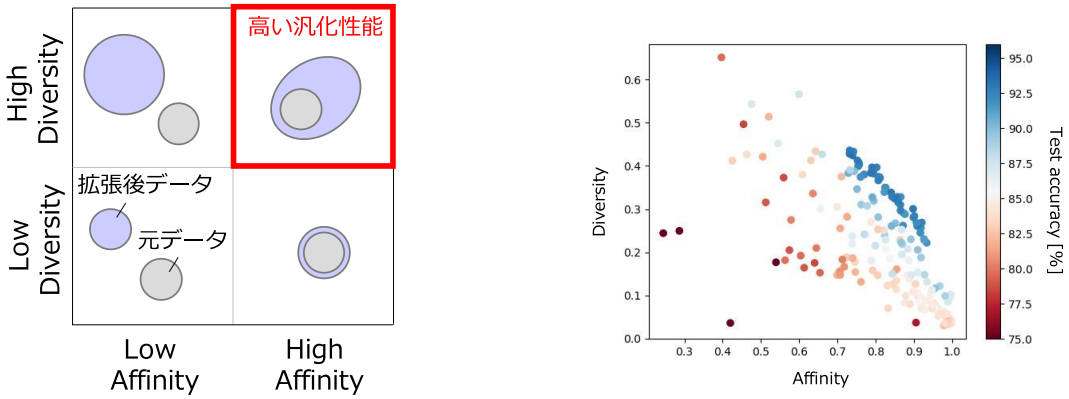
$$Aff \coloneqq A(m, D'_{val}) \,/\, A(m, D_{val})$$

ここで、$A(m, D'_{val})$はクリーンなデータを用いて学習したモデルにおけるデータ拡張を加えたバリデーションデータの精度を表し、$A(m, D_{val})$は同モデルにおけるクリーンなバリデーションデータの精度を表す。Diversity は、拡張後のデータ分布の広がりを表し、以下の式で計算される。

$$Div \coloneqq E_{D'_{train}}[L_{train}] \,/\, E_{D_{train}}[L_{train}]$$

ここで、$E_{D'_{train}}[L_{train}]$はデータ拡張を加えた訓練データを用いて学習されたモデルの誤差関数の値を表し、$E_{D_{train}}[L_{train}]$はクリーンな訓練データを用いて学習されたモデルの誤差関数の値を表す。

図 4.8(a)の赤枠で囲まれた分布のように、Affinity と Diversity の値がともに大きくなるようなデータ拡張ポリシーを用いて学習すると、テスト精度が高くなることが知られている。これは、図 4.8(b)の結果においても確認することができる。この図で、各点は手法やハイパーパラメータが異なるデータ拡張を用いた学習の結果を示している。図の右上部分の精度が高くなっていることが確認でき、Affinity と Diversity の両方が重要であることを示している。



(a) 指標の大小による分布の違い　　(b) テスト精度との関係

図 4.8　Affinity, Diversity の性質

4.4.2 Affinity, Diversity を考慮した指標による探索フェーズの短縮

データ拡張ポリシーの探索に要する計算コストを減らすために、本研究では探索フェーズを短縮する、つまり探索のための学習ステップ数を少なくすることを提案する。これによって容易に計算コストを軽減することが可能となるが、バリデーション精度を用いる場合、短いステップ数の学習ではテスト精度をうまく見積もることができないという問題が発生する。そこで、Affinity と Diversity を考慮した指標$aff \times div^{\alpha} \times val$を提案する。これは、Affinity、Diversity、およびバリデーション精度がすべて大きな値をとるときに大きくなる指標となっている。後述するように Diversity は学習序盤において不安定な指標であるため、0 から 1 までの値をとるαを用いて、Diversity の影響を小さくなるようにしている。この提案指標を用いた学習法を図 4.9 に示す。従来の方法だと、探索フェーズを最終ステップまで行い、バリデーション精度を用いてテスト精度を見積もる。一方、本手法では、短い探索フェーズの後、Affinity、Diversity、およびバリデーション精度を計算して提案指標を評価し、最も大きい値を出したデータ拡張ポリシーを選択する。その拡張ポリシーを用いて最終ステップまで学習を行う。この方法をとると、複数のデータ拡張を用いて個別に学習を行う探索フェーズが短縮されることで、全体の計算コストが減少する。例えば、全体の学習が 200 エポックで、本手法における探索フェーズが 5 エポックであるならば、大まかに考えると計算コストを 0.025 倍にできたことになる。

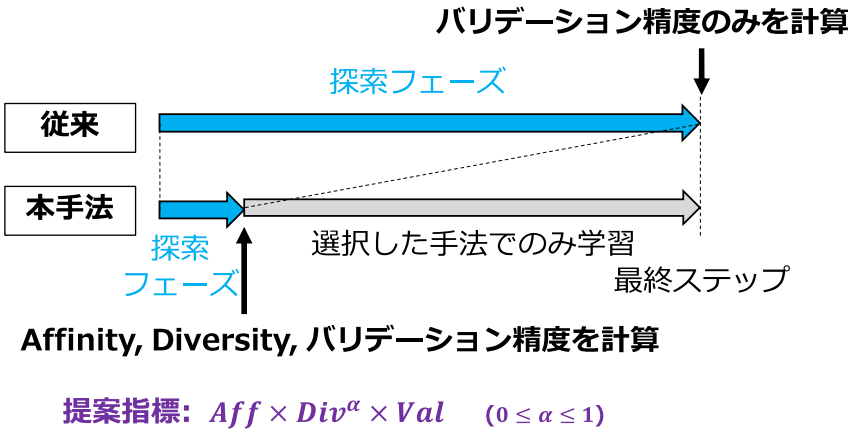


図 4.9 提案法の概要

複数のデータセットを用いて提案法の効果を検証する実験を行った。モデルは ImageNet の場合は ResNet50 を用い、それ以外のデータセットの場合は ResNet18 を用いた。データ拡張手法は、PatchGaussian[29], FlipLR, FlipUD, Crop, Cutout, Crop+FlipLR+Cutout, Rotate, ShearX および ShearY の 9 種類で、それぞれ複数のハイパーパラメータをもち、全体で、ImageNet の場合は 100 個、それ以外のデータセットでは 216 個のデータ拡張を用意した。5 エポックの探索フェーズの後、異なる指標を用いてデータ拡張ポリシーを選択する。その

データ拡張を適用して学習を最終エポック(ImageNet の場合は 100 エポック、他のデータセットでは 200 エポック)まで行い、テスト精度を比較した。実験結果を表 4.3 に示す。この結果から、バリデーション精度のみを用いた場合(Val acc)と比べ、提案指標を用いた場合(Proposed)は、高いテスト精度が得られる拡張ポリシーを選択できていることがわかる。Ground truth は、最終エポックまですべてのデータ拡張を用いて学習した中で最も精度の高かった結果であるが、提案法はこれに近い結果が得られていることがわかる。Affinity と Diversity を考慮に入れたことで、短い探索フェーズでも精度よくテスト精度を見積もることができるようになった。

表 4.3 指標ごとに選択された拡張ポリシーを用いた学習のテスト精度の比較

Dataset	Metric	Selected data augmentation	Selected hyperparameter	Test acc [%]
CIFAR-10	Val acc	ShearX	6, 1.0, -, variable	85.7
	Affinity	FlipLR	0.25, -, -, fixed	86.5
	Diversity	PatchGaussian	32, 2.0, -, fixed	80.4
	Proposed	Crop + FlipLR + Cutout	0.5, 0.25, 0.25, fixed	92.4
	Ground truth	Crop + FlipLR + Cutout	1.0, 0.5, 1.0, fixed	93.7
CIFAR-100	Val acc	PatchGaussian	4, 1.5, -, fixed	59.6
	Affinity	FlipLR	0.25, -, -, fixed	65.3
	Diversity	Rotate	45, 0.75, -, fixed	59.8
	Proposed	Crop + FlipLR + Cutout	0.25, 0.5, 0.25, fixed	70.9
	Ground truth	Crop + FlipLR + Cutout	1.0, 0.75, 0.25, fixed	72.5
SVHN	Val acc	PatchGaussian	16, 2.0, -, fixed	95.7
	Affinity	ShearY	6, 0.25, -, variable	95.8
	Diversity	PatchGaussian	28, 0.3, -, fixed	94.6
	Proposed	Rotate	20, 0.75, -, variable	96.2
	Ground truth	Crop	1.0, -, -, fixed	96.8
ImageNet	Val acc	ShearY	4, 1.0, -, fixed	70.8
	Affinity	ShearY	5, 1.0, -, fixed	70.9
	Diversity	Crop + FlipLR + Cutout	0.25, 0.5, 0.5, fixed	71.2
	Proposed	Crop + FlipLR + Cutout	0.5, 0.5, 0.25, fixed	71.8
	Ground truth	Rotate	60, 0.5, -, fixed	72.2

また、CIFAR-10 を用いた場合の、各指標におけるエポック 5 とエポック 200 の間の相関を図 4.10 に示す。各点は異なるデータ拡張を用いた学習の結果を示している。これらの結果から、Affinity は非常に正の相関が強いことがわかる。これは、Affinity の計算には、クリーンなデータで学習したモデルを必要とするため、学習序盤において、データ拡張による精度の変化の大きい不安定な学習が行われないためであると考えられる。逆に、Diversity の計算にはデータ拡張を用いた学習を必要とし、また学習序盤における誤差関数の値は最終的なテスト精度の値を必ずしもうまく反映しないため、相関が小さくなっている。提案指標において Diversity の影響を小さくしているのもそのためである。

まとめると、本研究はデータ拡張ポリシーの探索における高い計算コストの問題に取り組み、探索フェーズにおける学習ステップを大きく短縮することおよびバリデーション精度に加え、Affinity と Diversity を考慮した指標を用いることによって、高い精度で拡張ポリシー探索を行いつつ、全体の計算コストを減らすことができた。

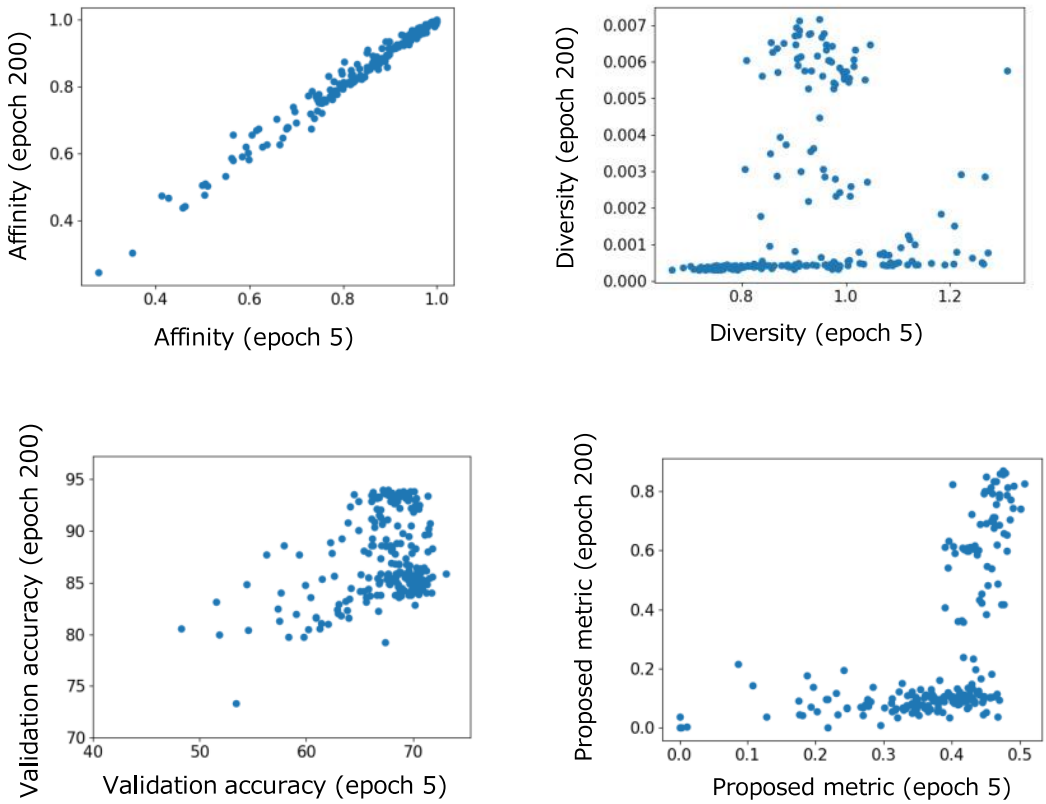


図 4.10 CIFAR-10 を用いた学習における各指標のエポック 5 とエポック 200 の間の相関

5 深層 NN ソフトウェアのデバッグ・テスティング

深層 NN ソフトウェア開発の初期段階では 3 つの観点（実現機能の具体化、学習に用いるデータセットの整備、深層 NN 学習モデルの選択）から試行錯誤的な繰り返しを通して、要求機能ならびに予測性能が達成可能かを確認する。この試行錯誤過程は従来のプログラム開発のデバッグ作業に対応するが、深層 NN ソフトウェア（DNN ソフトウェア）の場合、デバッグ・テスティングの入力データセット生成、訓練学習進行状況の監視と評価、要求実現を阻害する原因の特定と除去といった作業になる。以下、令和 2 年度に実施したデバッグ・テスティングの方法を報告し、得られた実験結果の考察と今後の計画を整理する。

5.1 不具合の直接原因

教師あり DNN 学習の標準的な方法では、訓練・学習と予測・推論の 2 種類のプログラムが関わる。学習データを与えられて学習タスクが決まった時、目的とする DNN ソフトウェアの実現に必要な学習モデルを選び、また、訓練学習過程で用いる方式を決める。利用可能な OSS の学習フレームワーク提供機能を用いる場合、フレームワークのパラメータを決めれば良い。次に、学習データから訓練データセットを構築する。そして、学習モデル・訓練データセットを入力として訓練・学習プログラム（学習フレームワーク提供）を作動させ、その結果、訓練済み学習モデルを導出する。より詳細には、訓練・学習プログラムが求めるのは、訓練済み学習モデルを定義する重みパラメータ値の集まりである。この訓練済み学習モデルが予測・推論プログラムの振舞いを規定する。

利用者からみると、DNN ソフトウェアの実体は予測・推論プログラムである。たとえば分類学習タスクの場合、入力データに対する分類の確からしさを求めるプログラムである。そして、この出力結果を調べることで、構築した DNN ソフトウェアが意図通りに作動しているかを判断する。期待する結果が得られず不具合があるとみなす時、訓練・学習プログラムの実行以前にもどって、欠陥の在り処を調べて除去する。つまり、デバッグ作業を行う。

不具合が生じる時、訓練・学習プログラム実行過程で用いる情報の何処かに欠陥があり、訓練データセット・学習モデル・学習機構のいずれか、あるいは、これらの複数が原因となる。一方、予測・推論結果に不具合をもたらす直接原因は、訓練済み学習モデルあるいは重みパラメータ値の集まりである。訓練データセット・学習モデル・学習機構の欠陥が不具合の根本原因である一方で、直接原因は重みパラメータ値にある。つまり、根本原因となった欠陥は重みパラメータ値の不具合として顕在化し、その不具合が示す訓練済み学習モデルの歪みが利用者からみた不具合の直接原因となる欠陥である[30]。この歪みを計測する方法が必要である。

本章では、重みパラメータ値を測定する内部指標を導入することで、DNN ソフトウェアの不具合を検知できるか否かを調べる。重みパラメータ値は訓練・学習プログラムの出力であるが、その出力値が妥当かを調べる直接の方法はない。その理由は、出力として期待する重みパラメータ値を予め知ることができないことによる。このような期待する重みパラメータ値が既知であれば、訓練・学習は不要になる。その既知の値を使えばよい。

5.2 内部指標

ニューロン・カバレッジ（NC）の考え方を紹介する。学習モデルをニューロンのネットワークとみなす。閾値を決めた時、出力値が閾値を超えるニューロンは活性状態にあるという。学習モデルを構成するニューロン数を N とし、活性状態のニューロン数を A とする時、活性ニューロンの比（NC = A/N）をニューロン・カバレッジと定義する。文献[31]は、NC を検査網羅性の基準と仮定し、評価用入力データの選び方が NC 値、すなわち、訓練済み学習モデルの検査網羅性に影響することを調べた。

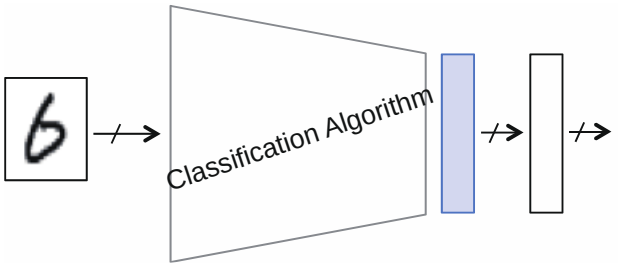


図 5.1 訓練済み学習モデル

本章では、NC を計算する対象ニューロンの選び方を工夫することで、不具合の有無を調べる内部指標[32]として用いる。図 5.1 に訓練済み学習モデルの模式的な図を示した。中間層の最終段階（網掛けした層）を対象とするニューロンに対して NC を定義し内部指標とする。

一般に、機械学習の技術では、此の Penultimate Layer を特別の観測対象とすることが多い。たとえば、画像分類タスクの場合、その前段までが画像認識などの具体的なアルゴリズムの役割を果たす相関分析（ピクセル値のパターンの分析）の処理であり、その計算結果が、此の層に集約されることが理由である。そして、本章では、想定される欠陥原因が、この内部状態として顕在化すると仮定する。さらに、この内部状態をもとに、さまざまな統計指標を導出することができる。調べたい不具合によって、何が適切な導出指標かを実験によって調べる。

5.3 実験の方法と結果

いくつかの実験結果を示し、先に定義した内部指標あるいは導出指標の有用性を考察する。最初に、訓練・学習プログラム（学習フレームワーク）に欠陥がある時の比較実験結果を示す。以下、BI は PC に欠陥挿入した訓練・学習プログラムである。

図 5.2 は学習モデルとして古典的な全結合ネットワークを用い、中間層のニューロン数を変化させて、試験データセットの正解率をプロットした。十分な数のニューロンを持つ時（横軸で 50）、PC と BI で正解率に大きな差がないことがわかる。つまり、正解率を調べても、PC と BI を区別することが困難であり、その結果、欠陥の有無がわからない。これは、これまでに得られた知見を再確認するものである。以下、この知見（図 5.2）に加えて、さらに状況を系統的に調べる実験の結果（図 5.3）を示す。

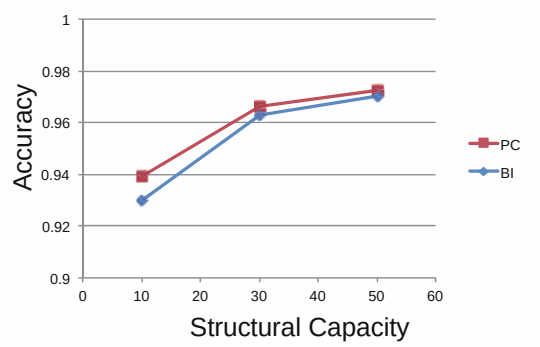


図 5.2　中間層の異なる学習モデル

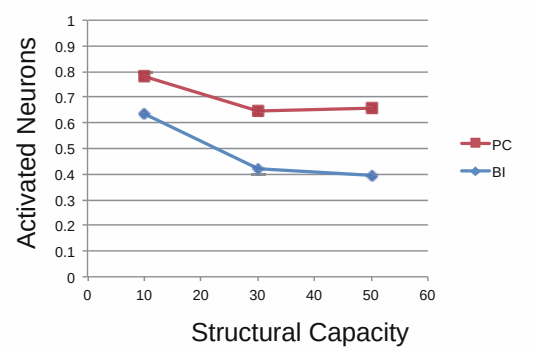


図 5.3　内部指標との関係

図 5.3 は、縦軸に内部指標（本章でのニューロン・カバレッジ）をプロットした。この指標の絶対値を参照すると、たとえば、BI の 10 と PC の 30 とは、共に 0.7 程度であって、BI と PC の区別がつかない。そこで、内部指標からの導出指標に適切なものがあるかを調べる。今、試験データセットに対するニューロン・カバレッジの集まりを得て、この平均値 μ と分散 σ^2 を求めて、さらに、σ / μ を計算する。横軸にこの導出指標 σ / μ を用いる場合を図 5.4 に示した。縦軸の値から図 5.3 を参照することで、どの学習モデルがどの値かを知ることができる。

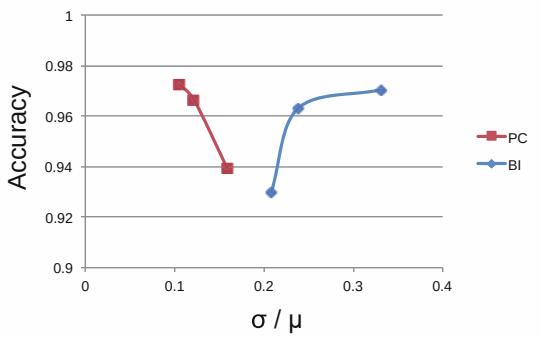


図 5.4　導出指標

図 5.4 によると、PC と BI を区別できることがわかる。つまり、内部指標では、キャパシ

ティ（中間層のニューロン数）が異なる PC と BI を区別できない（図 5.3）が、導出指標を工夫して σ/μ によって比較すると、ニューロン・カバレッジが有用な情報を与えることがわかる。

次に、欠損データを評価用に用いて、PC と BI が個々のデータに対して出力する分類の確からしさを散布図（図 5.5）で表す。

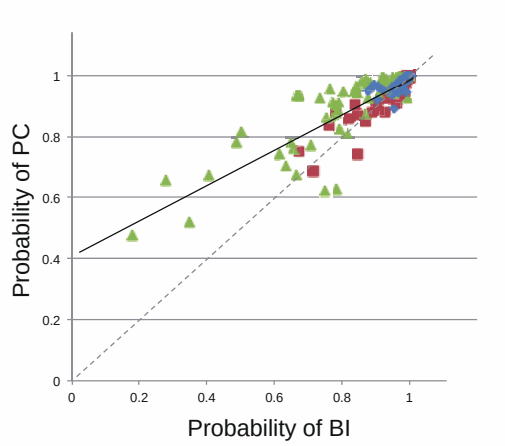


図 5.5　分類の確からしさ

図 5.5 で△が欠損データに対する出力値を表す。PC と BI で同等の値を出力すると仮定すると、原点を通る点線上に分布する筈である。実際、試験データセットから選んだ□は概ね此の線上にのることがわかる。一方、欠損データ（△）は実線上に分布し、PC がより良い分類の確からしさであること示す。つまり、欠陥混入した BI は、正解率は変わらない（図 5.2 を参照）が、頑健性に劣るといえる。

次の実験は、内部指標を用いることで頑健性の違いを検出できることを確認するものである。

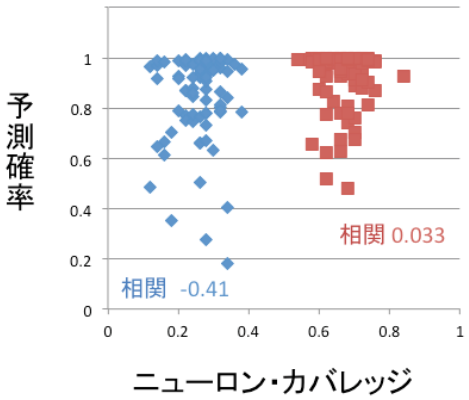


図 5.6　内部指標の違い

図 5.6 は、前記の欠損データを入力し、先に定義した内部指標を横軸にプロットした。右側に分布する□は PC、左側に分布する◇は BI による結果を示す。これによると、(1) PC の内部指標の値が大きいこと、(2) 内部指標と予測確率（分類の確からしさ）の相関が弱いこと、がわかる。次に、σ/μ を計算すると、PC は 0.0876 であり、BI は 0.2183 となった。図

5.6 は頑健性に影響する欠損データを用いる実験であり、σ/μ の値が頑健性と強く関係すると考えられる。

次に、訓練データセットに系統的な歪みを与えて、訓練済み学習モデルを導出する実験を行った。系統的な歪みが生成できることが、これまでの実験からわかっている。この実験は、訓練データセットの違いが、内部指標に影響を与えるかを調べることに相当する。同一の試験データセットに対する正解率をプロットした。

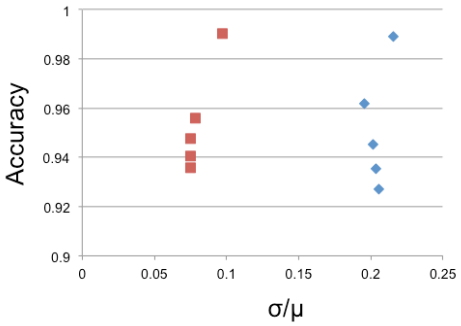


図 5.7　訓練データセットの違い

図 5.7 の上から下へ（正解率の良い方から悪い方へ）、大きい歪みの訓練データセットを用いた場合に対応する。つまり、試験データセットのデータ・シフトが訓練時に用いたデータセットから相対的に大きくなるので正解率が低下することを確認できる。一方で、横軸の値（σ/μ）は、PC（□）と BI（◇）で明らかに異なる。この実験の状況は、訓練データセット・シフトの一例と考えて良い。図 5.7 は PC と BI で独立した 2 系列を示す。正解率が示す正確性と σ/μ 値が示唆する頑健性とが独立な観点であることを確認できる。

以上から、訓練データセットの歪みは、内部指標では区別がつきにくいが、正解率に基づく方法で調べることができる。実務で行われているように、訓練データセットの良し悪しの検討に際しては、正解率を調べる方法が有用であると云えよう。一方、訓練・学習プログラムの欠陥など他の要因が関わる可能性がある（多重欠陥が想定される）場合、内部指標や導出指標（σ/μ）の値を同時に調べることが望ましい。

5.4　関連研究

ニューロン・カバレッジは DeepXplore [31]で導入されたメトリックスである。従来のソフトウェア・テスティングで用いられてきたテスト網羅性基準を参考に提案された。従来法は、実行文をノードとする制御フロー・グラフ（Control Flow Graph, CFG）でプログラムを表現する時、あるテスト入力データによって実行される文を基本単位として、テスト網羅性を定義する。最も簡単には、CFG のノードが入力テストによって実行される経路に含まれるか否か、つまり文が実行されるか否かを基準とする。これは、文網羅基準あるいは C0 基準と呼ばれる。DNN 学習モデルはネットワークとして表現されることから、CFG 上で定義された C0 基準との対比により、ノードに位置するニューロンが活性化（出力値が閾値を超えること）されるか否かによって、ニューロン・カバレッジ（NC）を定義した。そして、NC 値を増加させるように新しいテスト入力データを生成する方法を論じた。

ニューロン・カバレッジは従来のテスト網羅性基準から類推できる素直な考え方だろう。その後、NC を向上させる入力データの作成が難しくないという経験から、複数ニューロンの相関や異なる層の間での相関を考慮したメトリックスが提案された[33]。また、NC が増加するように、NC 値をガイドとして、データ補完の方法で有用なテスト入力データを系統的に生成する方法[34]がある。さらに、NC をガイドに用いる方法と GAN をベースとするテスト入力自動生成[35]とを組み合わせる方法が論じられている[36]。データ生成をガイドする指標として、NC が有用なことが確認されたと言ってよい。

テスティングの例として、文献[34]および[35]は回帰問題 DNN モデルの予測結果をもとに計算したステアリング角度というアプリケーションの機能を検査の観点とした。文献[37]は NC 値を大きくするテスト入力が欠陥発見に役立つかを調べた。何を欠陥とするかで NC が有用か否かの判断が異なることを論じている。逆に、この研究[37]は、正確性を中心とする外部指標と NC の間の相関が弱いことを述べている。本章では、両者に相関が弱いという観察から、NC に基づく内部指標を検査に用いた。文献[37]と矛盾しないどころか、同じ方向の議論を展開しているといえる。なお、網羅性は検査終了の判断基準であり、一方、欠陥発見はテスト入力データがコーナーケースを実行するかに依存する。両者は異なる側面を論じているともいえる。実際、従来のソフトウェア・テスティングにおいても、網羅性の向上が必ずしも欠陥発見の効率向上に結びつかないことが報告されている。

本章の方法は、文献[30][32]で論じられているように、NC 値を簡易的な検査指標に用いるというものである。従来の研究が NC を検査の網羅性基準に用いるのに対して、DNN モデルの欠陥が NC 値として現れる、という見方を採用した。実験では、この考え方に基づく具体的な検査として、訓練・学習プログラムの信頼性および訓練済み学習モデルの頑健性を調べることができた。

5.5 おわりに

本章では、Penultimate Layer でのニューロン・カバレッジに基づく内部指標を用いた。これはスカラーであることから、計測ならびに導出指標の定義が容易であり、検査指標として利用しやすい。一方、NC はニューロン個々の値に関する情報を捨象しており、有用な情報が欠落する。実際、文献[38]では、ニューロン値の分布を推定し、これをもとにテスト入力データが妥当かを論じる方法を提案している。これを応用すると、ニューロン値の分布は訓練データセットの歪みを、より詳細に表すと考えられるだろう。今後、この分布を利用する考え方を応用することで、訓練データセットをデバッグする方法を検討する。

6 訓練データのデバッグ・テスティング

6.1 ３つの問題設定

深層ニューラル・ネットワーク（Deep Neural Networks、DNN）の技術[39]を応用した DNN ソフトウェア開発の初期段階では、中核となる DNN コンポーネントの機能振舞いが期待通りであるかの確認を目的としたデバッグ・テスティングを行う。これは DNN コンポーネントに適切なデータを入力し、予測出力が意図通りかを調べる作業である。出力に何らかの不具合がある時、検査対象の DNN コンポーネントが欠陥を含む。デバッグの目的は、このような未知の欠陥を特定し除去することである。

DNN コンポーネントの欠陥は不具合の直接原因であるが、根本原因ではない。DNN コンポーネント構築の標準的な方法[40]では、(a) 学習基盤、(b) 学習モデル（DNN モデルの雛形)、(c) 訓練データ、といった 3 つの要素が複合的に関わる。その何処かに DNN コンポーネントが示す不具合を導いた根本原因がある。何処に根本原因を想定するかで検査の問題設定が異なる[41]。

DNN コンポーネント構築の基本は、膨大な数の学習データからなる訓練データセットを対象とし、訓練データに内在する情報を統計的な方法を用いて帰納的に導出して DNN モデル（非線形関数）を求めることである。素朴には、DNN モデルを調べて根本原因を特定すれば良い。しかし、DNN モデルは非線形関数で表現した処理手順なので、DNN モデルに評価データを入力し機能振舞いあるいは出力結果が妥当かを調べる間接的なテストオラクルによるソフトウェア・テスティング法を用いることになる[42]。

先の 3 つの要素の(a)では、学習基盤の実体は最適化問題を解く数値計算プログラムで、メタモルフィック・テスティング検査法が有用なことが知られている[43]。(b)の場合、学習モデルに明らかな欠陥があるわけではない。対象学習タスクに対する最良の学習モデルを見つけることであり、DNN 技術を応用する研究の主要な課題である[39]。本章では、(c)の場合について、つまり、訓練データのデバッグ・テスティングの問題を検討する。

6.2 訓練データのデバッグ問題

訓練データのデバッグ・テスティングは、学習データの偏りが DNN モデルに影響するという観察に基づき、意図通りの機能振舞いを示す DNN モデルが得られるように訓練データを改訂（追加・削除）することである。以下、教師あり分類学習タスクを対象として、訓練データのデバッグ問題を具体的に考える。

6.2.1 モデル正確性とモデル・ロバスト性

入力データをC種類に分類する教師あり学習タスクでは、多次元ベクトル x と正解タグ y からなるデータ点 z（$z = \langle x, y \rangle$）を考える。そして、与えられた訓練データセット S（$S = \{z^{(k)} \mid k = 1, \dots, N\}$）から導出された DNN モデルを検査対象とする。DNN モデルは入力の評価データ x に対応するC次元の分類確率ベクトル P_x を返す。P_xの j成分を $P_x[j]$として、値が最大の成分 j が y と一致（$y = \text{argmax}_{(j \in [1,C])} P_x[j]$）すれば正解とする。この時、多次元

ベクトル P_x、特に、正解の成分 $P_x[j]$ が示す予測確率は、データ x に対するモデル正確性を表す 1 つの指標となる。また、評価データの集まり E（$E = \{\langle x^{(\ell)}, y^{(\ell)}\rangle \mid \ell = 1, \ldots, M\}$）に対する正解の度数（正解率）を正確さ（Accuracy）とし、正確さもモデル正確性の指標のひとつである。さらに、C種類の分類カテゴリの正確さが不均等か否か（たとえば Gini 係数）もモデル正確性を表す指標となる。

DNN モデル構築過程では、訓練データセット S から求めた正確性と、S と異なるデータセット E に対するモデル正確性を求める。たとえば、S に対しては良い正解率を示す一方で、E に対しては正解率が悪化することがある。これは訓練データセットへの過適合として知られている。通常、S と E はひとつの大きなデータ・プール D から選んで構成することから、同じデータ分布に従う異なる標本と考える。S と E に対する正解率を調べることで S への過学習の程度を調べる。過学習が見られない時、DNN モデルは汎化性能に優れているとする。

訓練データのデバッグ問題では、評価データ E を D 以外から選ぶこともある。期待する振舞いを示すことの確認が目的の正常系テスティングでは、汎化性能の評価と同じように、D から S と異なる E を選べば良い。しかし、例外的な状況での振舞いを調べるには、D に含まれないデータを評価用データセット F としたい。F に対しては、正解率に代表されるモデル正確性は良い指標にならない。F の特定データに対する予測確率（モデル正確性の指標）からの判断ではなく、D のデータからの外れ具合に応じた予測確率の低下が許容範囲であるかを表すモデル・ロバスト性が評価の基準となる。

なお、実際の開発時においては、所与の D で期待する予測性能が得られない場合、新たなデータを収集し、訓練データそのものを改訂する。そして、新たな訓練データセットを用いて、DNN コンポーネントを導出する。検査時には、正常系ならびに例外系テスティングの双方から、モデル正確性とモデル・ロバスト性を評価する。学習データを当初の D に限定して考えることは実務に合わないことがわかる。

6.2.2 訓練データ記憶

過適合（あるいは過学習）は DNN モデルの予測性能（モデル正確性とモデル・ロバスト性）に大きく影響する。そこで、過学習の問題を緩和する方法を組み込んだ学習基盤の方式が従来から研究されてきた。正則化やドロップアウトといった手法である[44]。このような方法を学習基盤が持つ場合であっても、訓練データセットに偏りがあると、期待する予測性能を得ることができない。訓練データのデバッグ問題は、訓練データセットを改訂することで、DNN モデルの予測性能を改善することである。簡単には、不適切な偏りをなくすことと言える。しかし、偏りの度合い、ならびに、偏りの適切さ（不適切さ）を評価することは難しい。

訓練データ（標本）の偏りを評価する方法として、標本の統計的な特徴を調べるアプローチがある。たとえば、$S = \{\langle x^{(k)}, y^{(k)}\rangle \mid k = 1, \ldots, N\}$ とする時、正解タグによってC個の集まり $S^C = \{\langle x, c\rangle \mid \langle x, c\rangle \in S \text{ かつ } c = 1, \ldots, C\}$ に分けたとしよう。S^C の大きさが均等であれば、正解タグからみた標本数に偏りはないとしてよい。しかし、各々の S^C は何らかのデータ分布 ρ^C に従うし、この ρ^C から見て偏りがあるかは何もわからない。これを調べるには、ρ^C を推定する必要がある。しかし、データ x は多次元ベクトルであり、このような多次元データ分布 ρ^C を推定することは実務上容易でない。

DNN モデルの予測性能は、評価データを入力してのテスティング結果によって調べる。学習基盤の方式によって、同じ訓練データから導出した DNN モデルが異なる予測性能を示すこともある。つまり、訓練データのデバッグを目的とする検査では、訓練データの統計的な特徴を調べるだけでは不十分であり、訓練データの偏りが、どのように DNN モデルに反映されているか、どのような方式で訓練学習されたか、が影響する。

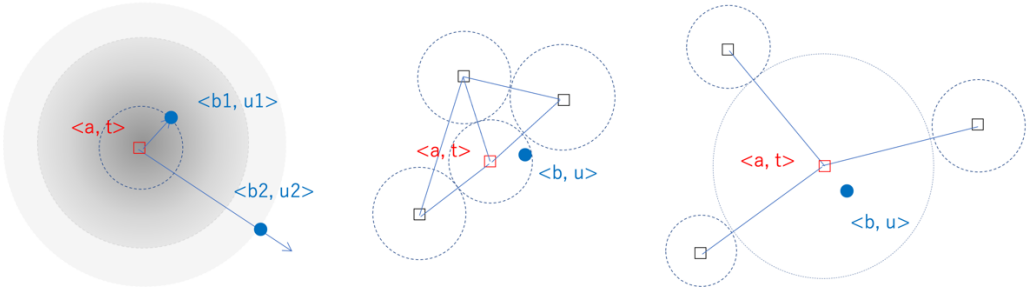
訓練データの偏りと DNN モデルの関係は、DNN モデルが訓練データの正解ラベルを記憶するという観点から論じることができる。今、訓練データ S がデータ点 $\langle a,t\rangle$ を含む時（$\langle a,t\rangle \in S$）、S から $\langle a,t\rangle$ を除去した訓練データを S' とする（$S' = S \setminus \{\langle a,t\rangle\}$）。$S$ あるいは S' を用いて訓練することで得られる DNN モデルを各々 M, M' とする。そして、M による入力 a に対する予測確率ベクトル P_a と M' による P'_a を求める。分類結果 t に対する $P_a[t]$ の確からしさが大きい一方、$P'_a[t]$ に対する確からしさが小さい時、M は訓練に用いたデータ $\langle a,t\rangle$ を記憶する（Memorize）という。この定義から、過適合の状況では、DNN モデルが訓練データを記憶することがわかる。データ記憶は、データの偏りの、ひとつの側面を表すと考えられる。

DNN モデルでは、帰属関係推定（Membership Inference）が可能なことが知られている。データ点 $\langle x,y\rangle$（$\langle x,y\rangle \in D$）が訓練データセット S に含まれていたか（$\langle x,y\rangle \in S$）を訓練済み学習モデルMに入力して得られる情報から調べる問題である。$M(x)$の実行結果の分類確率ベクトル P_x から調べるブラックボックス法[45]や $M(x)$ の実行過程で計算する損失関数 $\ell(Y(W;x),y)$ の情報を利用するホワイトボックス法[46]がある。ここで、W を学習パラメータあるいは重みとし、$Y(W;x)$ を入力 x に対する予測の内部表現とした。なお、これらを用いて、M は訓練データセット S の分布 ρ の下で、損失関数の平均値を最小化する問題の解 W^*（$W^* = \mathrm{argmin}_W\, \mathbb{E}_{\langle x,y\rangle\sim\rho}[\ell(Y(W;x),y)]$）をもとに定義する$Y(W^*;x)$ から得られる。

直感的には、帰属関係推定の方法は、データ点 $\langle x,y\rangle$ が訓練データセット S に含まれているか否かによって、P_x あるいは $\ell(Y(W;x),y)$ の分布が異なるという観察に基づく。このような指標の分布に違いが生じる原因は過適合を含む訓練データ記憶である[46]。そして、帰属関係推定の脅威を緩和するアプローチは、過適合が生じないような学習基盤の方式を利用することに加えて、記憶されやすいデータを訓練データセットから除去することである[47]。帰属関係推定は、訓練データの偏りの状況を反映する。

訓練データ記憶と関わる状況を調べる。分類問題を考えて図 6.1 で、$a \neq b$ であるが $t = u$ とする。図 6.1 (a) は訓練データ $\langle a,t\rangle$ から離れると共にデータ $\langle b,u\rangle$（$\langle b,u\rangle \notin S$）の予測確率が低下することを模式的に示す。図 6.1 (b)は S の中で訓練データ $\langle a,t\rangle$ が密集している時、そのデータを除去してもデータ $\langle b,u\rangle$ の予測確率が大きな影響を受けないことを表す。つまり、除去した訓練データは予測結果に大きな影響を与えないという点で記憶されない。図 6.1 (c)は訓練データが疎（外れ値）の時、図 6.1 (b)とは逆に、影響が大きくなり、強く記憶されていることを表す。このような外れ値データは DNN モデルの予測分類性能に大きく影響する。

最後に、帰属関係推定を訓練データ・デバッグの問題から考える。訓練データが記憶される状況では、データ点が訓練データセット S に含まれるか否かで、P_x や $\ell(Y(W;x),y)$ の分布が大きく異なる。帰属関係推定の方法は、訓練に用いた z から遠いデータ点 z' に対する予測性能が悪いことを利用している。つまり、帰属関係推定をモデル・ロバスト性の検査と考えることもできる。訓練データ記憶の現象はモデル・ロバスト性と関わる。



(a) 近傍での予測確率　　(b) 密集する領域　　(c) 疎な領域

図 6.1　訓練データの配置と予測の確からしさ

ここで、図 6.1 の模式的な状況を参照する。図 6.1 (b)に示す密集データを除去しても $\langle b, u\rangle$ のモデル正確性への影響は少ないだろう。一方、図 6.1 (c)のようなデータを除去するとモデル・ロバスト性は改善するが、予測分類をサポートするデータがなくなることから、その近傍のモデル正確性が低下する。あるいは、このデータを除去しないで、その近傍に新たなデータを追加すれば密になり、局所的なモデル正確性が向上する。したがって、訓練データ・デバッグでは、訓練データセット S の外れ値を検知することが大切である。

図 6.1 の模式的な説明は、入力データの予測分類性能が訓練データとの位置関係に影響されることを示す。ところが、どのようにして位置関係を定義するのか、つまりデータのどのような特徴から位置関係を定義するのか、を述べていない。逆に、予測分類性能の違いを議論する際に、どのように位置関係を定義すべきなのかという問題ともいえる。位置関係の基準を決めることで、外れ値検知の問題設定があきらかになる。

6.3　外れ値とニューロン・カバレッジ

訓練データ・デバッグを目的とする外れ値検知の方法を考察する。

6.3.1　訓練データの外れ値

訓練データのデバッグ問題は、訓練データセットから外れ値（Outliers）を見つけ出し、開発対象 DNN モデルの目的に応じて、外れ値の取扱いを決めることである。外れ値を除去する、外れ値周辺に新たなデータを追加する、などの方策が考えられるが、その取り扱いは、DNN モデルに対する要求仕様と関わる。訓練データ・デバッグの一般的な議論としては、外れ値検知の技術を確立すればよい。

一般に、外れ値は多数を占めるデータと異なる特徴を持つデータであり、外れ値であるか否かは対象データの集まりが示すデータ分布（統計的なデータ・モデル）を前提として定義する[48]。たとえば、データ分布の確率密度関数が既知であれば、データ x の尤度をもとに外れ値であるかを調べればよい。

素朴には、訓練データの経験分布をもとに、外れ値であるかを考える。ところが、訓練データは多次元ベクトルであり、経験分布を簡便な形式で知ることが困難である。たとえば、

カーネル密度推定（Kernel Density Estimation, KDE）などの手法の適用が難しく、その結果、尤度に基づく外れ値の検出は実用的ではない。あるいは、ソフトウェア・テスティングの分野で知られている組み合わせテスト法（Combination Testing）に準じる分析方法を応用することが考えられる。経験分布に大きな影響を与えると思われる成分（因子）を選び、このような代表的な次元に着目して調べることで経験分布に対する検査を代用する。実務的に有用な一方、外れ値は元来の定義から稀なデータであり、この近似的な方法の有効性に疑問が残る。

少し視点を変えて、モデル・ロバスト性を分析する標準的な方法のロバスト半径[49]を考える。2 つのデータ点 $\langle x, y\rangle$ と $\langle x', y'\rangle$ を選び、各々の出力の予測分類結果を $P_x[y]$ と $P_{x'}[y']$ とする。ロバスト半径 δ は、出力の違いの許容水準 ε が与えられた時、ε を満たす入力データの違いの最大値 δ（$|P_x[y] - P_{x'}[y']| \leq \varepsilon$ の時、$|x - x'|_p \leq \delta$）である。ここで、入力データの違いをL_pノルムで定義する。素朴には、与えられた ε に対して、ロバスト半径 δ が大きいとモデル・ロバスト性が良いと考える。しかし、ノルムL_pの選び方によってロバスト半径 δ_p が異なる。ロバスト半径によるモデル・ロバスト性の定義は厳密である一方、入力データの空間での分析は、用いたノルムが妥当かといった議論あるいは解釈を必要とし、問題の状況が複雑化する。なお、この議論は$|x - x'|_p$が大きくなる（離れる）と$|P_x[y] - P_{x'}[y']|$が大きくなること、つまり、孤立点の議論である。

ここで、異なる分類カテゴリーのデータ $\langle b2, u2\rangle$ への訓練データ $\langle a, t\rangle$（$t \neq u2$）の影響について考える（図 6.1 (a)）。2 つは分類カテゴリーが異なるので、入力データ空間では離れているとして良い。訓練データ記憶の議論と同様に、訓練データセット S が $\langle a, t\rangle$ を含み、S から $\langle a, t\rangle$ を除去した訓練データ S' とする。各々から得られた DNN モデルを M, M'とし、データ $\langle b2, u2\rangle$ に対する予測分類結果をP_{b2}, P'_{b2} とする。S と S' の誤差関数への影響を分析する影響関数（Influence Functions）の方法によって、$P_{b2}[u2]$ と $P'_{b2}[u2]$ の確からしさが異なる $\langle b2, u2\rangle$ が存在することがわかる[50]。つまり、訓練データ $\langle a, t\rangle$ の有無が $\langle b2, u2\rangle$ 予測分類確率に影響することを表す。したがって、入力データ空間での違いによる分析（$a \neq b2$）だけでは目的とする情報を得ることが難しい。

以上から、訓練データの集まりを対象として、つまり、入力のデータ空間での分析によって、目的とする外れ値を系統的に検知することが難しい。その理由は、モデル正確性やモデル・ロバスト性が訓練データだけではなく、学習方式など訓練学習に関わる多様な要素に影響されるからである。ただし、入力データ空間での分析が全く効果ないことを主張するものではない。このような分析によって、訓練データの経験分布を大まかに把握することができるだろう。

本章では、入力データ空間の特徴がモデル正確性やモデル・ロバスト性と関わるとしても、系統的な訓練データ・デバッグの方法としては不十分と考える。訓練データの外れ値を検知する系統的な方法を検討する。

6.3.2 活性ニューロン

ニューロン・カバレッジは、対象ニューロン数に対する活性ニューロンの割合で定義される[51]。ここで、$M^S(x)$ とした入力データ x の信号が DNN モデル中を伝播し、各々のニューロンを活性化すると考える。与えられた閾値を超える出力を持つ時、活性ニューロンと呼

ぶ。

ニューロン・カバレッジは、当初、カバレッジ駆動テストデータ生成（Coverage-driven Test Data Generation）の網羅性基準として提案された[51]。入力データ x による活性ニューロンは出力結果に影響を与えるという点で有用な働きを示す。一方で、入力データ x に対して予測推論に関わらない時、不活性ニューロンが生じると解釈する。そして、不活性ニューロンを生じる入力データでは、全てのニューロンを調べていないと考え、十分なテストができていないとする。不活性ニューロンを活性化させるような新しい入力テストデータを求めれば良い。

文献[13]による提案の後、ニューロン・カバレッジのテスト網羅性基準としての有用性などに関する研究が進められた[52][53][54]。特に、ソフトウェア・テスティングの C0 基準がそうであるように、ニューロン・カバレッジを達成することは難しくなく、テスト網羅性基準として弱いことがわかっている。

一方、そもそも訓練データに不備があり、予測推論に関わらない不活性ニューロンが生じたと解釈することもできる。この場合、新たな入力として得られたデータを訓練データに追加して再学習する[51]。つまり、ニューロン・カバレッジを M^S のモデル品質を評価する基準とする考え方を示唆する。以下、このモデル品質評価基準としてのニューロン・カバレッジという観点[55]から考察する。

分類学習の DNN モデル M^S は入力に近い上流層が符号化 E'（Encoding）を行い、その後に分類 C'（Classifying）を担うという流れを示す（ $M^S = C' \circ E'$ ）。ここで、_∘_は関数結合を表し、$M^S(x) = (C' \circ E')(x) = C'\big(E'(x)\big)$ である。出力を分類確率ベクトルとする場合、C' の logits と呼ぶ出力の最終層に softmax 関数を配置する。$M^S = \mathrm{SOFTMAX} \circ C \circ E'$ のように表現できる。次に、E' と C をつなぐ層に全結合ネットワーク（Fully Connected Network）FCN を配置し、$M^S = \mathrm{SOFTMAX} \circ C \circ \mathrm{FCN} \circ E$ に変形する。FCN 層に符合化処理 E の結果が移され、さらに C に伝播されると考える。

ニューロン・カバレッジを計算する場合、閾値の選び方ならびに対象ニューロンの決め方が問題となる。FCN は全結合なので同一層内でのニューロンの入れ替えに対して出力が保存されるスワップ不変性（Swap Invariant）が成り立つ。そこで、 FCN 層のニューロン活性状況を要約する情報としてニューロン・カバレッジが有用だろう。一方、SOFTMAX や CNN のように構成ニューロンが特定の機能的な役割を担う場合、すべてのニューロンを同等に考えるニューロン・カバレッジが有用な情報を提供するかは疑問がある。実際、CNN に対しては、2 つの異なる定義が知られており、どちらを採用するかで、ニューロン・カバレッジの値が異なる[56]。本章では FCN 層に対してニューロン・カバレッジを考える。

文献[56]は、データ補完の方法で学習データを系統的に変化させて、E' や C でのニューロン・カバレッジへの影響を調べた。その結果、訓練データの違いが E'層のニューロン・カバレッジに大きく影響する一方で C層への影響は小さいこと、評価用の試験データの違いは C の最終層（Penultimate Layer）にほとんど影響しないことなどがわかった。先に導入したように $E' = \mathrm{FCN} \circ E$ とする時、この FCN 層に訓練データの違いが反映されると考えてよいだろう。

また、これまでの実験[43][55]において、C の最終層に設けた FCN のニューロン・カバレッジを測定したところ、分類予測確率とニューロン・カバレッジの相関がほとんどないことがわかった。したがって、ニューロン・カバレッジはモデル正確性に寄与する情報と独立し

た側面を表すと考えられる。仮にモデル・ロバスト性との間に何らかの相関関係があるとしたら、ニューロン・カバレッジが外れ値を検知する方法として有用と期待できる。

6.4 実験と考察

ニューロン・カバレッジとモデル・ロバスト性の関係を実験的に調べ、ニューロン・カバレッジに関わる情報が訓練データのデバッグに有用か否かを考察する。

6.4.1 実験の概要

実験は、準備作業を含めて、5 つのステップからなる。MNIST データセットを用い、学習モデルとして、$M^S = \text{SOFTMAX} \circ C \circ \text{FCN} \circ E$ の形を想定した。以下、MNIST の訓練データセットを LS（大きさ 60,000）、試験データセットを TS（大きさ 10,000）と表記する。

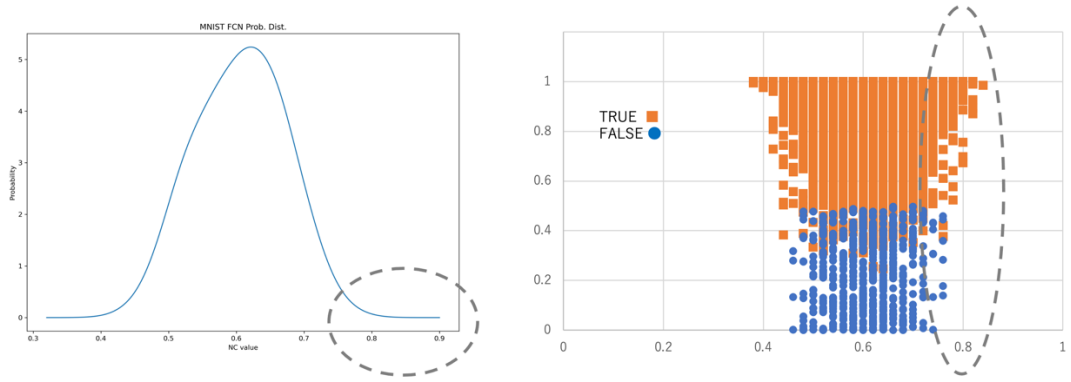
- ステップ 1：LS が生起する活性ニューロンを測定する。M^Sを LS で訓練して得た訓練済み学習モデルをMとする。Mの FCN 層を対象として、多次元ベクトルデータx（$\langle x, \_\rangle \in LS$）が生起（$M(x)$）する活性ニューロンの集まりを$Act^x$とする。FCN 層を構成するニューロン数を$|FCN|$として、ニューロン・カバレッジは$NC^x = |Act^x|/|FCN|$である。LS の要素全体に対して、$N^{LS} = \{NC^x \mid \langle x, _\rangle \in LS\}$とする。$N^{LS}$のデータ分布（$NC^x$の度数分布）を求める。
- ステップ 2：N^{LS}のデータ分布を基準として、LSから系統的に訓練データを選び出し、LS_j^Pを得る。具体的な選び方ならびに添字Pとjの意味は 6.4.2.2 節で説明する。
- ステップ 3：M^SをLS_j^Pで訓練して得た訓練済み学習モデルをM_j^Pとする。試験データ TS を入力して、正解率を計算する。また、予測の確からしさ（正解タグの SOFTMAX 値）を分類クラスごとに求め、分類クラス間のばらつきを表す Gini 係数を計算する。すなわち、LS_j^Pのモデル正確性への影響を調べる。
- ステップ 4：モデル・ロバスト性を評価するデータを TS から合成する。合成関数をT_kとし、いくつかの方法で、評価データES_kを得る（$ES_k = T_k(TS)$）。具体的な合成方法は 6.4.3.1 節で説明する。なお、データ合成方法T_kによっては、必要に応じて訓練済み学習モデルMを利用する。
- ステップ 5：訓練済み学習モデルM_j^PをデータES_kで評価し、モデル・ロバスト性の指標を求め、M_j^Pの違い、すなわち、LS_j^Pのモデル・ロバスト性への影響を調べる。

6.4.2 ニューロン・カバレッジ分布

6.4.2.1 ニューロン・カバレッジの測定

検査対象の訓練データセット LS のニューロン・カバレッジの集まりN^{LS}を求め、そのデータ分布（ニューロン・カバレッジ分布）を測定した結果を図 6.2 に示した。図 6.2 (a)は

横軸に入力データに対するニューロン・カバレッジ、縦軸に対応するニューロン・カバレッジ値を得た入力データ度数を表す（KDE の結果）。図 6.2 (b)は横軸に入力データに対するニューロン・カバレッジ、縦軸に同じ入力データの予測確率をプロットした散布図である。赤い点は予測が正しいデータ、青い点は不正解のデータを表す。なお、訓練データ正解率と試験データ正解率は共に 99%であった。



(a) ニューロン・カバレッジ vs データ度数　(b) ニューロン・カバレッジ vs 予測確率

図 6.2　訓練データの分析結果

図 6.2 (a)は測定したニューロン・カバレッジが、0.38 から 0.84 の間に分布することを示す。また、中央値ならびに平均値は 0.60 である。図 6.2 (b)はニューロン・カバレッジと予測の確からしさ（予測確率値）の相関がほとんどないことを示す。図 6.2 中、たとえば、楕円で囲んだ領域は、平均よりも大きなニューロン・カバレッジとなる訓練データが対応する。ニューロン・カバレッジの値が同じであっても、正解タグの予測確率値が広くバラつくだけではなく、正解・不正解の両方を含む。ニューロン・カバレッジの値が小さい領域に着目しても、同様な傾向を示す。ニューロン・カバレッジは予測推論過程の内部状態を表すが、end-to-end の出力値とは相関がない。

ニューロン・カバレッジが小さいことは、入力データが分類に寄与する有意な情報を持たないことを示唆する。極端な場合、入力画像のピクセル値が 0 であれば活性化されるニューロンはなく、その結果、ニューロン・カバレッジがゼロになる。測定結果では 0.38 程度が小さい領域であり、「弱い」入力信号を活用することで予測分類結果を導いたと考えられる。一方、ニューロン・カバレッジが大きいとは、活性化されるニューロンが多いことを表すが、ニューロン・カバレッジ値が大きいことが正解の予測確率を向上させるわけではない。入力データが有意な分類結果を導く情報を持たないこと、異なる分類かを区別できないことを示唆する。実際、測定対象層のニューロンが少なく構造的なキャパシティが小さい場合、正解率が劣る一方でニューロン・カバレッジが大きくなるという測定結果（図 5.2 と図 5.3）と整合している。構造的なキャパシティが十分であるという前提のもとで、ニューロン・カバレッジのデータ分布の中央付近のデータを訓練に用いると、適切に分類に寄与するデータを積極的に選ぶことができ、モデル正確性に影響することなく、モデル・ロバスト性に寄与すると期待できる。

本章では、以上の観察を 2 つの仮説に整理する。デバッグ対象の訓練データセットを LS とする時、LS の要素 x を入力した時のニューロン・カバレッジの集まり N^{LS} のデータ分布を

考える。

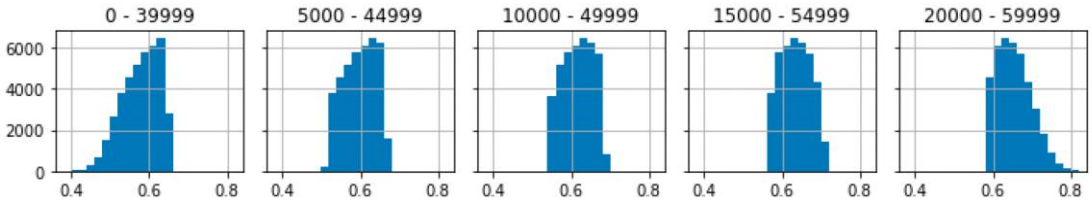
［仮説 1］ニューロン・カバレッジはモデル・ロバスト性と相関がある
［仮説 2］N^{LS}のデータ分布が外れ値の基準となる

以降、仮説が妥当かを実験で確認し、訓練データ・デバッグの指針を整理する。

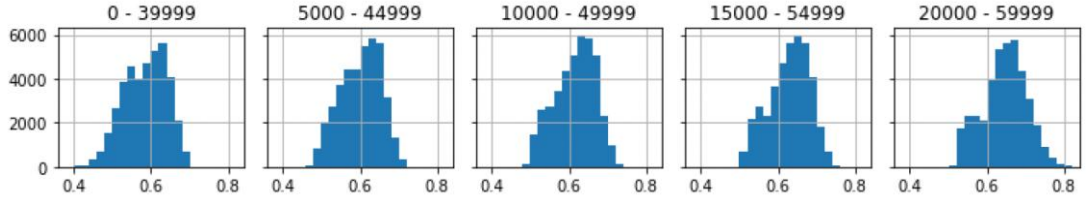
6.4.2.2　訓練データの区分け

訓練データのニューロン・カバレッジN^{LS}のデータ分布（図 6.2 (a)）を基準にして、LSから系統的に訓練データを選び出し、LS_j^Pを得る。60,000 個のLSから、同数（40,000 個）の訓練データLS_j^Pを系統的に抽出する。MNIST の訓練データLSを、分類クラス C に分けて$LS^{\langle C\rangle}$（$LS = \cup LS^{\langle C\rangle}$）として、2 つの抽出方法を考える。選択モード P が N の時、LS全体を対象として、ニューロン・カバレッジの大きさ区間が$[5000 \times j,\ 5000 \times j + 39999]\ (j = 0,1,2,3,4)$となる 5 つのデータセット$LS_j^N$を得る。また、選択モード P が C の時、$LS^{\langle C\rangle}$ごとに、同様な方法で 400 個ずつ選択したのち組み合わせ、LS_j^Cを得る。各々の度数分布を図 6.3 に示す。

図 6.3(a)のLS_j^Nと(b)のLS_j^Cを比較すると分布の外形が異なることがわかる。もともと分類クラス C によって$LS^{\langle C\rangle}$の分布が異なる。実際の測定によると、たとえば、0 と 6 はニューロン・カバレッジの小さい値の領域（< 0.5）に、3 と 8 は大きい領域（> 0.5）に偏る。$LS^{\langle C\rangle}$毎に同数のデータを抽出し組み合わせたLS_j^Cは、「切り立った崖」のような形状を示すLS_j^Nよりもなだらかになる。



(a) 訓練データ全体を対象とした区分け（LS_j^N）



(b) 分類クラスごとの区分けの組みわせ（LS_j^C）

図 6.3　訓練データの区分け

6.4.2.3　モデル正確性の検査指標

大きさ 40,000 の訓練データLS_j^P（計 10 種類）を用いて得られる訓練済み学習モデルM_j^Pのモデル正確性を測定する。評価用データとして、MNIST 試験データセットTSを用いる。TSは、LSと同等のデータ分布を持つが、LS_j^Pとは異なると考えられる。実際、ニューロン・カバレッジで比較すると、N^{LS}とN^{TS}は同じ外形の分布を示す。したがって、LS_j^PとTSは、ニューロン・カバレッジ分布も異なる。この違いは、モデル正確性の定量的な結果に影響すると予想される。ここでは、訓練データLS_j^Pの違いがモデル正確性に、どのような影響を定性的に与えるかを調べる。なお、N^{LS}とN^{TS}が同じ外形の分布を示すということは、LS と TS に標本選択の偏りがないことと整合している。

モデル正確性の指標として、正解率と Gini 係数を図 6.4 に示した。ここで、青は全体から抽出する場合（M_j^N）、赤はクラス毎に抽出する場合（M_j^C）を表す。また、横軸は訓練済み学習モデルの違いを表す（$M_K = M_j^P$、$K = 1,2,3,4,5$）。正解率は 96%程度で、M_Kに対して、大きな違いがない。M_K間の相対的な違いは、全体抽出の場合で 0.6%程度、クラス毎抽出の場合で 0.3%程度である。クラス間のバラつき具合を表す Gini 係数は、小さい値（0.002 以下）であり、クラス間で均等な正確性を示す。以上から、LS_j^Pの違いはモデル正確性と相関が小さいことが確認できた。

Gini 係数については、全体抽出（青）とクラス毎抽出（赤）の場合で傾向が異なる。前者は緩やかな単調減少、後者は緩やかな単調増加である。この違いについては分析が未だ進んでいない。しかし、図 6.4 からわかるように、全体抽出（青）とクラス毎抽出（赤）の場合で、分類クラスごとのデータ数に違いが生じる。正解率に比べて、Gini 係数は他分類クラスの訓練データからの影響が大きいことが単調性の違いの原因と関わると考えられる。なお、正解率と予測確率の平均値は強い相関がある。

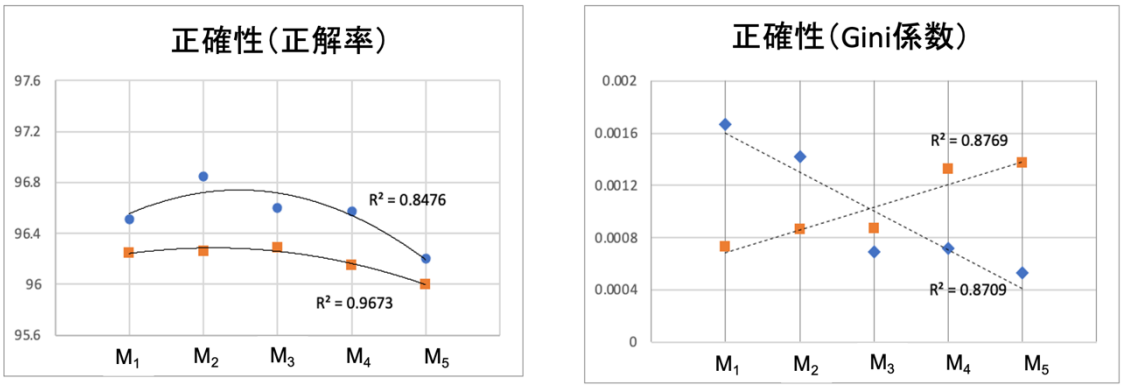


図 6.4　モデル正確性

6.4.3　モデル・ロバスト性の測定

モデル・ロバスト性の経験的な評価では、訓練データあるいは試験データと異なる統計的な性質を有する評価用データを準備する必要がある。

6.4.3.1　評価用データの生成

モデル・ロバスト性を論じる基本的な考え方を整理する。ロバスト半径δは、与えられたεに対して、$|P_x[y]-P_{x'}[y']| \le \varepsilon$ を満たす、$|x-x'|_p \le \delta$ として定義される[49]。δが大きいほど、ロバスト性が良い。

測定実験では、基準となるデータ点 $\langle x,y\rangle$ がある時、x'を系統的に求める変換関数 T を導入する（$x'=T(x)$、分類タスクでは $y'=y$）。ここで、L_2 ノルムを前提として、データの距離 $d_T(x)=|x-T(x)|$ を定義する。与えられたεに対して、$d_T(x)$を変化させて予測確率の違い調べれば、ロバスト半径δを経験的に求めることができる。しかし、後に説明するように、$d_T(x)$を滑らかに変化させるような変換関数 T を適切に決めることが難しい。そこで、定性的な性質が異なるデータを生成する変換関数 T を考える。具体的には、ガウスノイズ挿入、小領域の欠損、意味ノイズ挿入（フレーム、下線）、アフィン変換（回転、縮小・拡大）を用いた。MNIST 試験データ TS の要素を基準データとし、変換関数 T_k を適用して評価用データ（$ES_k=\{\langle T_K(x),y\rangle \mid \langle x,y\rangle \in TS\}$）を生成するものである。また、$diff_K(x)=\left|p_x(y)-p_{T(x)}(y)\right|$ に対して、$Diff_K=\{diff_K(x) \mid x \in TS\}$ とする。さらに、$d_K(x)=|x-T_K(x)|$ とする。そして、LS で訓練して得た訓練済み学習モデルMを用いて、ES_kの特徴を調べる。

図 6.5 に、生成した画像イメージ、横軸に TS データの予測確率$p_x(y)$・縦軸に ES データの予測確率 $p_{T(x)}(y)$ の散布図、$d_K(x)$ の度数分布図を、8 種類の変換について示した。画像イメージから、いずれも、基準として選んだデータ x の目視上の特徴を保存していることがわかる。散布図は、評価用データ生成方式に外形の分布が違い、予測確率への影響が全く異なることを示している。

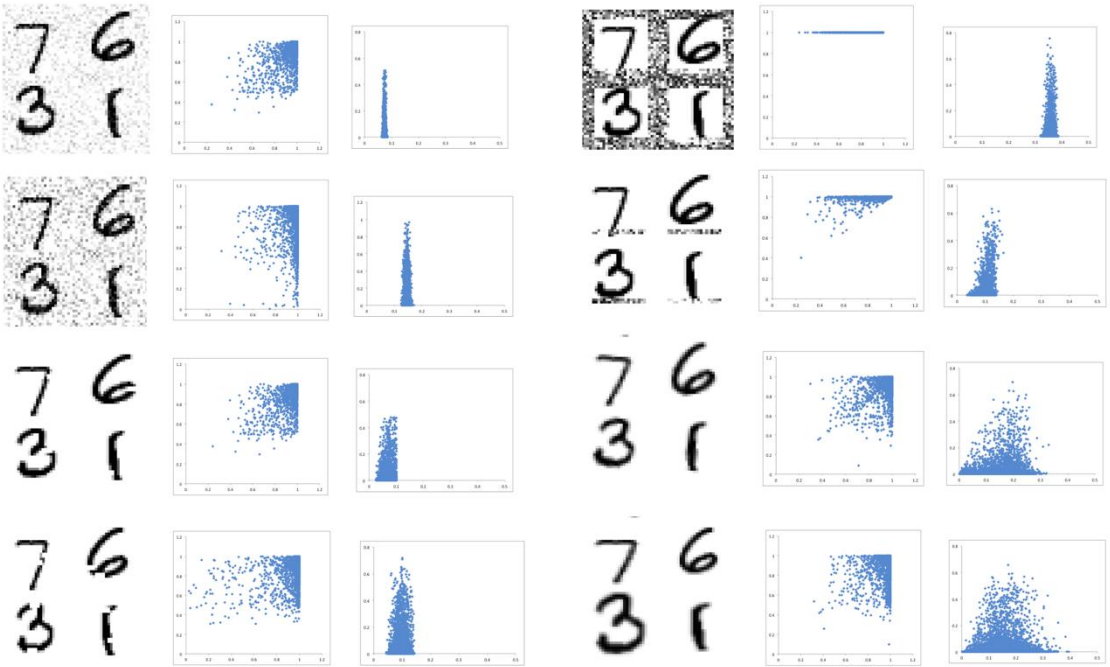


図 6.5　モデル・ロバスト性の評価用データ

図 6.5 左上の 2 つはガウスノイズを加えたもので、ノイズの大きさが異なる。ノイズが大きくなると、予測確率のバラつきが大きくなる。与えたガウスノイズの大きさの指定方法から $d_K(x)$ の度数分布は鋭いピークを示す。ノイズを大きくすれば $d_K(x)$ を変化させることは容易であるが、一方で、目視上の画像への影響が大きく、評価用データとしての適切さを欠く。左下の 2 つは欠損データである。欠損領域の大きさを変えること $d_K(x)$ を変化させることができるが、各 $d_K(x)$ の度数分布は鋭いピークを示す。欠損領域を極めて大きくとると、もとの画像を壊し、評価用データとしての有用性が低下する。

右上の 2 つは意味ノイズ挿入（フレーム、下線）である。ノイズ挿入の方法の性質[43]から、$p_{T(x)}(y)$の値が、ほぼ 1 になる。目視から明らかなように、フレームの方が $d_K(x)$ の値が大きい。しかし、度数分布は鋭いピークを示す。右下の 2 つのアフィン変換は度数分布がなだらかとなる。

全体をまとめると、変換方法によって、$d_K(x)$ の度数分布が偏るが、各々は特徴的なピークを示すことがわかる。また、図 6.6 は、横軸に $d_K(x)$ の平均値・縦軸に $diff_K(x)$ の平均値をプロットしたものである。評価用データES_kの全体を合わせると、さまざまな $d_K(x)$ の値での評価ができる。定性的な性質の異なる多様なデータを用いてモデル・ロバスト性の評価を行なっていることになる。どのような性質のデータを評価に用いるかが大切であり、これは、従来のソフトウェア・テスティングでも重要な「テストケース」の問題と同じである。

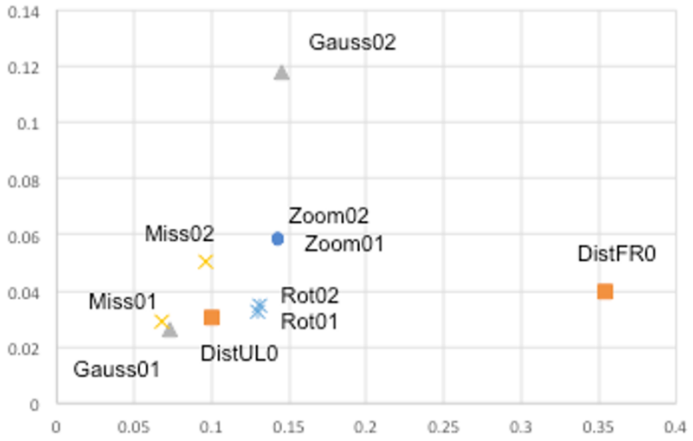


図 6.6　評価用データ距離

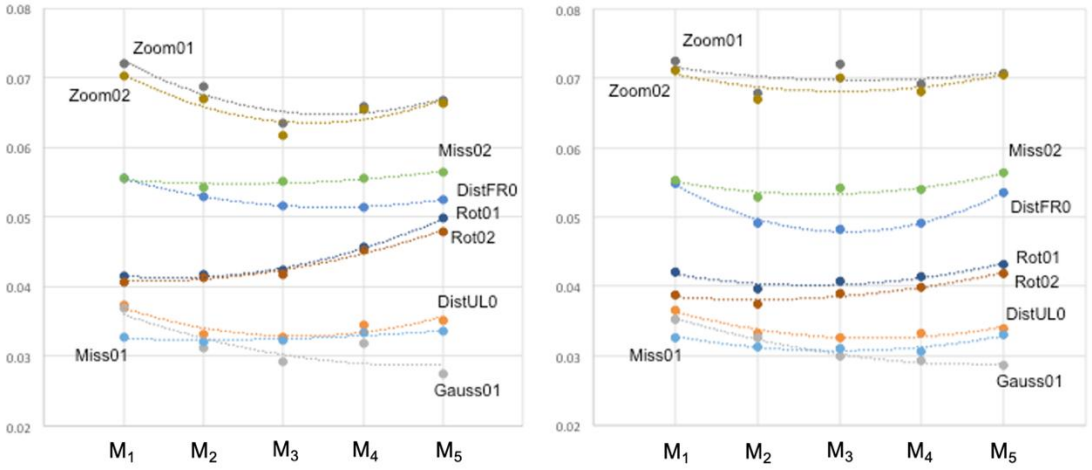
6.4.3.2　モデル・ロバストの検査指標

本実験では、評価用データを用いる経験的なモデル・ロバスト性評価の指標として、先に定義した $Diff_K$ の平均値を用いる。モデル正確性検査の場合と同様に、訓練データ LS_j^P を用いて得られる訓練済み学習モデル M_j^P を対象に、$Diff_K$ 各々の平均値を測定する。その結果を図 6.7 に示した。図 6.7(a)は M_j^C 、図 6.7(b)は M_j^N である。

評価用データによって、多少の違いがあるものの、大まかな傾向として、M_K間で単調性を示さない。モデル正確性の場合（図 6.4(a)）、M_K間の相対的な差が 1%未満であるのに対して、図 6.7 では 10%程度の違いがある。M_Kの違い、つまり、LS_j^Pの違いが、モデル・ロ

バスト性に有意な影響を与える。

訓練データ LS_j^P の構成方法から推測すると、ニューロン・カバレッジ分布の両側裾野を除去した中央付近（もともと度数の大きな領域）の訓練データが、モデル・ロバスト性に寄与していると考えられる。



(a) クラス抽出　　**(b) 全体抽出**

図 6.7　モデル・ロバスト性の指標

6.4.4　まとめ

訓練データのニューロン・カバレッジ N^{LS} のデータ分布をもとに系統的に抽出した訓練データ LS_j^P を用いて得られた訓練済み学習モデル M_j^P を対象に、モデル正確性とモデル・ロバスト性の評価を行った。M_j^P のモデル正確性は同等な一方、モデル・ロバスト性に影響を与えることがわかった。［仮説 1］（ニューロン・カバレッジはモデル・ロバスト性と相関がある）と整合性がある。また、N^{LS} のデータ分布の両側裾野を外れ値として除外することで、モデル・ロバスト性への影響から、M_2^C および M_2^C が概ね妥当な値を示すことがわかった。［仮説 2］（N^{LS} のデータ分布が外れ値の基準となる）と整合する。

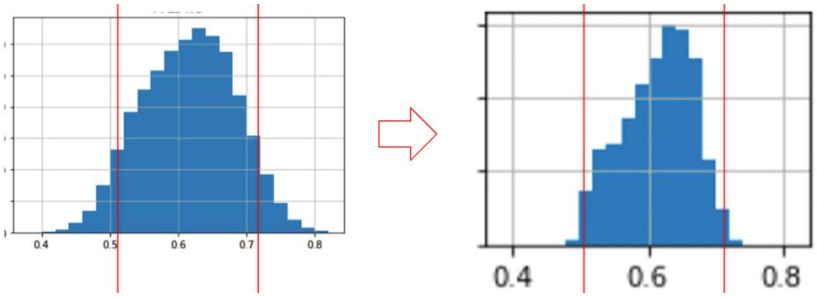


図 6.8　訓練データのデバッグ戦略

ここで、図 6.8 を参照して、訓練データのデバッグ戦略を整理する。与えられた訓練データセットのニューロン・カバレッジが左側のデータ分布を示すとき、妥当なデータ数を含む

範囲の中央領域を選び、つまり、両側裾野に相当する訓練データを捨象し、右側のデータ分布を示す訓練データを得る。次に、新たな学習データの候補の中から、そのニューロン・カバレッジ値が中央領域に相当する訓練データを選び、右側のデータ分布の外形を保持しながら、十分なデータ数を得られるまで、訓練データセットに追加する。

6.5 おわりに

最後に、深層 NN ソフトウェアのデバッグ問題に関して、ソフトウェア工学分野での最近の研究状況を概観し、本章で論じた技術の位置付けを明らかにする。一般に、深層 NN ソフトウェアに不具合がある時、その直接原因は訓練済み学習モデルの欠陥である。一方、このモデル構築には、(a) 学習基盤、(b) 学習モデル（DNN モデルの雛形)、(c) 訓練データ、といった要素が複合的に関わり、何が根本原因かが明らかでないことが多い。

さて、実務上、深層 NN ソフトウェア開発は、既存の機械学習フレームワーク上で行われる。つまり、プログラム（フレームワーク API 利用スクリプト）作成作業を含む。この時、フレームワークに問題があり、訓練機構（数値最適化問題の解法プログラム）自身が不適切だったり、スクリプトから呼び出される API のライブラリ関数（たとえば学習モデル定義関数）が欠陥を持ったりすること[57]がある。これは、上記の(a)の場合であって、フレームワーク提供者の責任になる。

一方、機械学習フレームワークを利用して、深層 NN ソフトウェア開発を行う立場では、スクリプトの欠陥が不具合の原因となる[58]。欠陥の種類によって、要求仕様と異なる学習モデルが生成されたり、データ型（テンソルの次元）の誤りが生じたり、設定されたハイパーパラメーターが不適切だったりする。上記の分類では(b)に相当する。機械学習の研究は、与えたれた学習タスクに対して、適切な学習モデルを選定する論理的なデザインの問題である。これに対して、このような上位仕様にしたがった適切な「プログラム」を実現しているかが問題となる。訓練過程（誤差関数や勾配）を監視することで、取り扱っている学習モデルの欠陥箇所を推定する方法[59]などが研究されている。

深層 NN ソフトウェアの機能振る舞いは、訓練に用いた学習データに支配される。そこで、学習データが適切な特徴量（あるいは属性）から構成されているかが問題となる。一般論として取り扱うことは難しいが、入力データの機微属性が結果に影響することが多い公平性の問題にアプローチする研究[60]がある。これは、訓練データに着目するという点で、上記の(c)に分類できるが、特徴量選定の問題であり、データ定義に関わるデバッグ問題である。

本章は、(c)の問題について、訓練に用いた学習データ分布の偏りが不具合の根本原因となる場合を対象とした。直感的には、次のような問題設定である。与えられた学習データを用いて訓練した DNN モデルが期待するモデル性能(特にモデルロバスト性)を示さない時、目的を達成する DNN モデルを得られるように、学習データの選別ならびに追加を行なって、データ分布を変形することである。

一般に、ソフトウェア工学からデバッグ問題にアプローチする研究は、スクリプトのように明示的な記号表現を持つことが前提である。連続量のデータ分布をデバッグ対象とする方法の検討が進んでいない。一方、機械学習技術からの研究では、データ分布の取り扱いは重要と認識されているものの、統計学に基づく方法によって、分布関数が既知の場合を取り

扱う。しかし、機械学習の対象は多次元ベクトルであり、データ分布を簡便に表現することが難しい。理論的な整理が可能であっても、実際のデバッグ作業に、直接、役立つわけではない。

本章の方法は、データの経験分布（連続量）を「デバッグ」のガイド情報として利用し、データ個々の取捨選択（離散量）を行うものである。特に、デバッグ対象データの性質をニューロン・カバレッジ分布で表すことで、1 次元のデータ分布を取り扱う方法に帰着することで、取り扱う問題を簡単化した。統計的な見方とソフトウェア工学の方法を融合させたといえる。

7　ロバストネスの評価・向上技術

本章におけるロバストネスは、入力ノイズ（敵対的データも含む）に対する機械学習モデルの耐性である。例えば、どの程度ノイズを付加しても推論結果を維持できるかを評価する。そのロバストネスの指標の一つに最大安全半径がある。本章では、順伝播型ニューラルネットワークを用いた分類器を対象として、敵対的データと最大安全半径について説明した後、最大安全半径を計測する技術と増加させる技術についての調査結果について報告する。

7.1　ロバストネスの指標（最大安全半径）

機械学習ソフトウェアにおいては、訓練済み学習モデル（正確には、学習モデルにもとづく推論プログラム）は、わずかにノイズを付加された入力データによっても誤推論する問題が知られている。そのような誤推論させる入力データは敵対的データ（adversarial example）[61]と呼ばれており、近年、敵対的データに対する研究が活発に行われている。入力データ$x \in \mathbb{R}^n$（$\mathbb{R}$は実数の集合）のδ近傍（半径$\delta \in \mathbb{R}$の球の内側）に含まれる全ての敵対的データの集合$Adv_\delta(x)$は次のように定義できる。

$$Adv_\delta(x) = \{x' \mid \| x - x' \| \leq \delta \ \wedge \ f(x) \neq f(x')\}$$

ここで、$f(x)$は入力xに対する機械学習モデルの出力（分類結果）を表す関数、$\|x - x'\|$は 2 つのデータx, x'の距離（差）を表す。距離の定義にはpノルムが使われることが多い。

図 7.1 を用いて敵対的データについて説明する。図 7.1 の左側はニューラルネットワークへの入力空間、右側がニューラルネットワークからの出力空間を表す。入力空間の赤い球の中心がパンダ画像（元データ）であり、その球（半径δ）の内側が大きさδ未満の微小ノイズを付加した画像の集合（δ近傍）である。このδ近傍の入力画像の集合に対するニューラルネットワークの出力の集合が、右側の出力空間の赤い領域である。ここで、出力側の赤い領域の右下の決定境界を超えて猿の領域に入っている部分が誤分類を表しており、この部分にマップされる入力が敵対的データである。

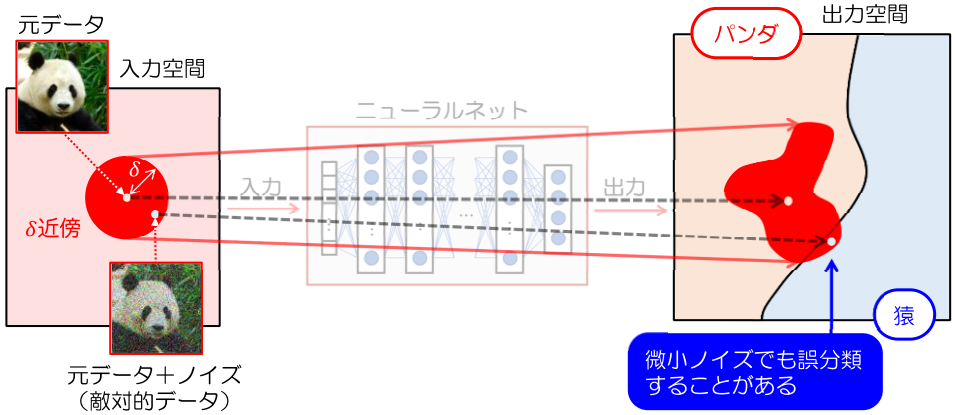


図 7.1　敵対的データ（ノイズ付加パンダ画像を猿に誤分類）

入力データxのδ近傍（xを中心とする半径δの球の内側）に敵対的データが存在しないと

き、δをxの安全半径という。特にxの安全半径の中で最大の半径（最大安全半径, Maximum Safe Radius）$MSR(x)$は次のように定義される。

$$MSR(x) = \max\{\delta \mid Adv_\delta(x) = \emptyset\}$$

この最大安全半径が大きいほど、敵対的データによる攻撃は難しくなるため、機械学習モデルのロバストネス（敵対的データを含む入力ノイズに対する耐性）の指標のひとつとして最大安全半径を使うことができる。

図 7.1 のδ近傍の入力画像の一部は猿に誤分類されるため、このδは安全半径ではない。一方、図 7.2 のδ近傍の入力画像は誤分類されることはないため、このδは安全半径であり、これ以上δを大きくすると誤分類される可能性がある（敵対的データを含む）ため、図 7.2 のδは最大安全半径である。

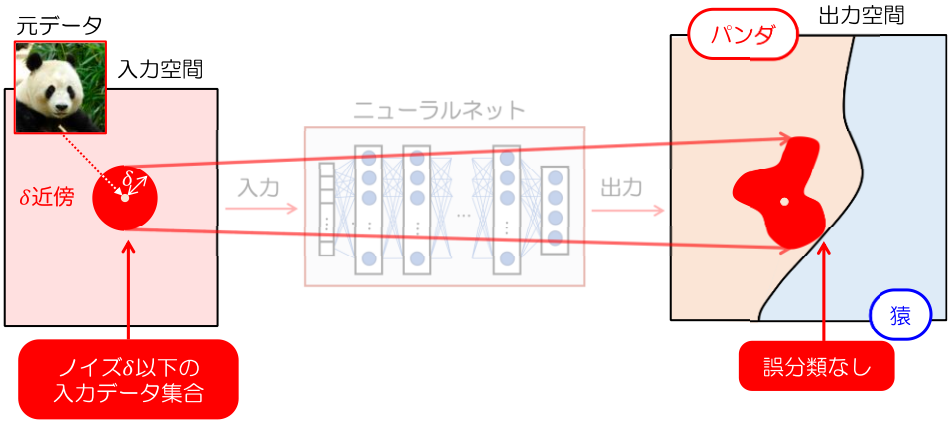


図 7.2　最大安全半径 δ

7.2　ロバストネスの評価・向上技術調査結果

ロバストネスの評価と向上に関する研究論文について調査した結果を表 7.1 に示す。ロバストネスについては多くの研究論文が発表されているが、比較的良い結果が得られている最近の論文を代表として示している。表 7.1 の各論文の下には、その論文で提案されている手法を適用できるスケールの参考情報として、その手法の評価実験に使用されたニューラルネットワークの情報を記入している。表 7.1 は次の観点から整理している。

- 横方向（技術の用途）：
  - ロバストネスの評価：最大安全半径の見積り
  - ロバストネスの向上：指定された最大安全半径をもつデータ数の増加
- 縦方向（評価の信頼度・精度）：
  - 保証あり：δ近傍に敵対的データが存在しないことを保証できる
    - 厳密：最大安全半径の厳密な見積り
    - 近似：最大安全半径よりも小さめ（安全半径のひとつ）の見積り
      - 決定的：δ近傍に敵対的データは存在しない（100%安全）
      - 確率的：δ近傍に敵対的データが存在しない確率はp%
  - 保証なし：δ近傍に敵対的データが存在しないことを保証できない

表 7.1 ロバストネス（最大安全半径）の評価と向上に関する技術

<table>
<tr><th colspan="3"></th><th>ロバストネスの評価</th><th>ロバストネスの向上</th></tr>
<tr><td rowspan="4">保証あり</td><td rowspan="2">厳密</td><td rowspan="2"></td><td>最大安全半径の厳密な見積り
Katz et al. 2017 (Reluplex) [62]
ACAS-XU-DNN, 300 ReLU nodes
6 hidden layers,
数百ノード程度が上限

Tjeng et al. 2019 [63]
CIFAR-10, ResNet, 9-CNN, 2-layer,
107,496 ReLU units,
Reluplex より 100～1,000 倍程度高速</td><td></td></tr>
<tr><td rowspan="3">近似</td><td>決定的</td><td>最大安全半径の小さめの見積り
Weng et al. 2018 (Fast-Lin) [64]
CIFAR, 6-layer, 12,288 ReLU units
Reluplex より 10,000 倍程度高速

Boopathy et al. 2019 (CNN-Cert)[65]
CIFAR-10 (32x32x3), 5-layer,
10 filters, 29,360 hidden nodes,
Fast-Lin より高速</td><td>近傍に敵対的データがないように訓練
Wong and Kolter 2018 [69]
SVHN (32x32x3), 2-conv, 32-ch,
100, 10 hidden units, ReLU,
ImageNet への適用は困難</td></tr>
<tr><td>確率的</td><td>確率的最大安全半径の小さめの見積り
Weng et al. 2019 (PROVEN) [66]
CIFAR, 5-layer, CNN, ReLU
CNN-Cert と同程度</td><td>訓練後に保証付ランダムスムージング
Lecuyer at el. 2019 [70]
ImageNet (299x299x3),
Inception-v3 + auto-encoder

Cohen at el. 2019 [71]
ImageNet (299x299x3),
ResNet-50 (50-layer)
Lecuyer [70]より精度向上</td></tr>
<tr><td>保証なし</td><td colspan="2"></td><td>最大安全半径の大きめの見積り
Carlini and Wagner 2017 [67]
ImageNet (299x299x3),
Inception-v3

最大安全半径のおおよその見積り
Weng et al. 2018 (CLEVER) [68]
ImageNet (299x299x3),
ResNet-50 (50-layer)</td><td>近傍の敵対的データを探索しながら訓練
Madry et al. 2018 [72]
CIFAR (32x32x3),
28-10 wide ResNet</td></tr>
</table>

以降、小節 7.2.1～7.2.7 で表 7.1 の各手法について簡単に説明する。

7.2.1 ロバストネス評価、保証あり（厳密）

Katz 等は、与えられた性質を機械学習モデルが満たすことを判定する方法 Reluplex を提案した[62]。その方法を実装した実証用ツールも公開されている。性質は入出力関係につい

ての制約であり、入力データの δ 近傍に敵対的データが存在しないことを網羅的に厳密に（健全かつ完全に）判定できる。すなわち、二分探索などで半径 δ を変えながら敵対的データの有無を判定することによって、最大安全半径を見積もることができる。Reluplex は Simplex 法（線形計画問題の解法のひとつ）に ReLU 関数用の規則を追加した解法であり、実数を扱える充足可能性判定ツール（SMT-Solver）によって実装されている。ロバストネス以外の性質も判定できる強力なツールであるが、計算コストが高く、扱えるニューロン数が ReLU 数百個程度という制約がある。

Tjeng 等は、最大安全半径を効率よく計算する方法を提案し、その方法を混合整数線形計画法ソルバ（MILP）上に実装して、10 万個の ReLU 型ニューロンをもつネットワークの最大安全半径を厳密に求められることを示した[63]。まだ実用的な機械学習モデルに適用するには十分とは言えないが、このようにスケーラビリティ改善の研究は進んでいる。

7.2.2　ロバストネス評価、保証あり（近似、決定的）

Weng 等は、ReLU 型ニューラルネットの最大安全半径を近似的に見積る方法 Fast-Lin を提案した[64]。Fast-Lin では、図 7.3 に示すように、出力領域を多角形で線形近似し、最大安全半径よりも少し小さめの近似値 δ を見積もる。この近似値 δ は最大安全半径を超えないため（健全）、δ 近傍に敵対的データが存在しないことを保証できる。すなわち、δ は安全半径のひとつであり、最大安全半径の下界である（$\delta \leq MSR(x)$）。多角形で線型近似することによって、厳密な方法（Reluplex）よりも 1 万倍以上速い結果が得られたことが報告されている。

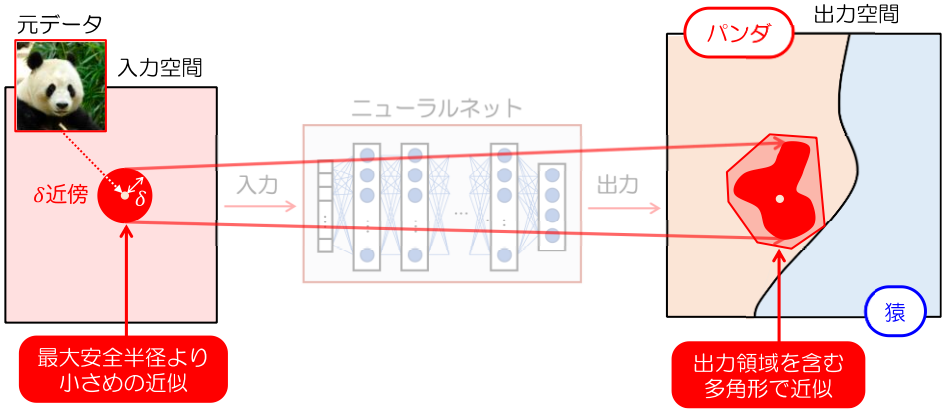


図 7.3　最大安全半径の近似値（少し小さめ）の見積り

Boopathy 等は Fast-Lin を改良した手法 CNN-Cert を提案した[65]。CNN-Cert では、ReLU 以外の活性化関数（sigmoid, tanh, arctan）を含む Convolutional ネットワークにも対応し、Fast-Lin よりも近似精度と計算速度を向上した。

7.2.3　ロバストネス評価、保証あり（近似、確率的）

Weng 等は、確率的な最大安全半径を近似的に見積もる方法 PROVEN を提案した[66]。安全確率 ρ の最大安全半径 δ は、図 7.4 に示すように、δ 近傍に敵対的データが存在しない確率

がρである、すなわち、$(1-\rho)$の確率で敵対的データが存在することを許容する最大の半径である（決定的な最大安全半径は安全確率1の最大安全半径に相当する）。PROVEN は、CNN-Cert をベースに開発されおり、計算量は CNN-Cert から大きくは増えていない。

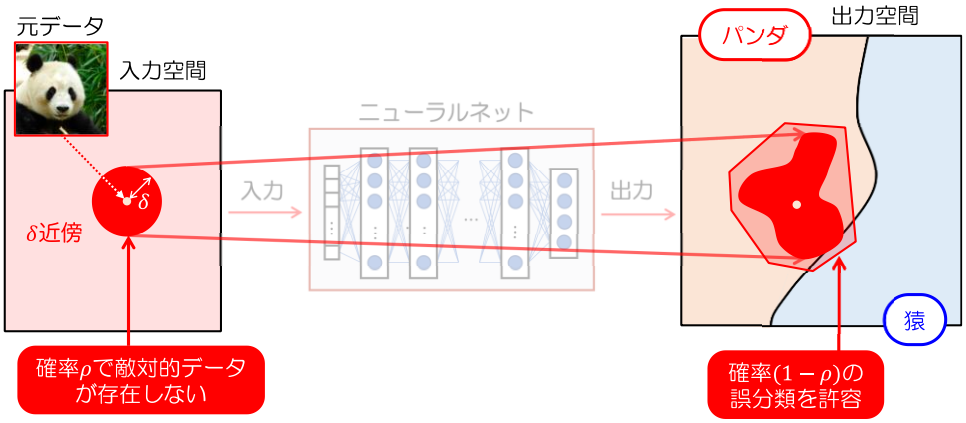


図 7.4　安全確率ρの最大安全半径の近似値の見積り

7.2.4　　ロバストネス評価、保証なし

Carlini と Wagner は、既存の最適化ツール（Adam）を用いて、入力データxに最も近い敵対的データを探索し、その距離δを見積もる方法を提案した[67]。ただし、この方法で得られる距離δが実際に敵対的データまでの最短距離である保証はなく、その距離よりも近いところに敵対的データが存在する可能性はある。すなわち、最大安全半径の上界である（$msr(x) \leq \delta$）。この方法で得られる距離δが安全半径である保証はないが、最大安全半径の目安として、最近のロバストネスの論文ではしばしば評価に使われている。

Weng 等は、攻撃方法に依存しないロバストネスの評価値として、おおよその最大安全半径を求める方法 CLEVER を提案した[68]。比較的大きなネットワークにも適用可能な方法であり、画像認識モデル Inception-v3 を 10 秒程度で評価できたと報告している。入力のわずかな変化が出力に与える影響の最大値を極値理論によって推定し、最大安全半径に近い値δを見積もっている。図 7.5 に示すように、見積もった値δは実際の最大安全半径より大きいこともあるため、δ近傍内に敵対的データが存在する可能性はある（安全半径である保証はない）。

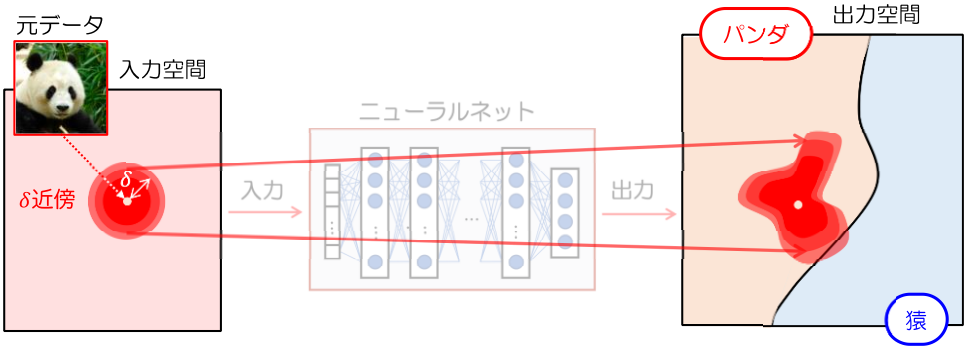


図 7.5　最大安全半径の近似値の見積り（保証なし）

7.2.5　ロバストネス向上、保証あり（近似、決定的）

Wong 等は、訓練データセットの各データの最大安全半径がδ（指定値）になるように訓練する方法（ロバスト訓練）を提案した[69]。この方法は訓練後に全ての訓練データに対して最大安全半径δを保証するものではないが、各入力データの最大安全半径の近似値（安全半径）を見積もる方法を与えている。このロバスト訓練では、各訓練データのδ近傍で最も誤推論する可能性のあるデータに対して正しい推論をするように訓練する。

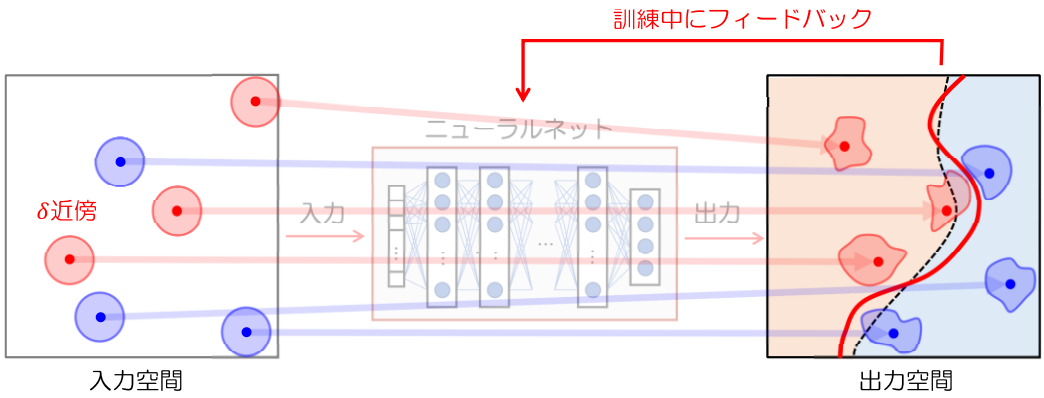


図 7.6　ロバスト訓練（入力のδ近傍も含めた訓練）

ロバスト訓練の様子を図 7.6 に示す。図 7.6 の出力空間の点線は通常の訓練によって学習した決定境界、赤色線はロバスト訓練によって学習した決定境界を表す。入力空間の 6 個の訓練データについては両方の境界線によって正しく分類されているが、各訓練データのδ近傍のデータについては点線の境界線（通常訓練）では誤分類が生じている。一方、ロバスト訓練では赤色の境界線のように、δ近傍のデータについても正しい分類になるように訓練を行う。Wong 等のロバスト訓練は、ロバストな学習モデルを保証付で訓練する方法であるが、スケーラビリティが低く、訓練可能なネットーワークのサイズを大きくできない問題がある。この論文は、MNIST（28×28）と SVHN（32×32）のデータセットに対して有効性を示しているが、ImageNet（256×256）には適用できなかったと報告している。

7.2.6　ロバストネス向上、保証あり（近似、確率的）

Lecuyer 等はランダムスムージングによって確率的に保証可能な最大安全半径を見積もる方法を提案した[70]。ランダムスムージングとは、入力データに防御用のノイズを付加した推論を繰り返し、その複数の出力の平均値を最終出力とする方法である。

ランダムスムージングの様子を図 7.7 に示す。図 7.7 の出力空間の点線は防御用ノイズを付加しない場合の決定境界、赤色線は防御用ノイズを付加した場合の決定境界を表す。ランダムスムージングは決定境界を滑らかにすることによってロバストネスを向上させる技術であり、ImageNet（299×299×3）のような大きな入力データに対する機械学習モデルのロバストネスの保証にも成功している。付加するノイズの分散を増加させると保証可能な最大安全半径も増加するが、一方で正解率等の推論精度は低下する。この論文では、差分プライバシの技術（類似した 2 つの入力に対する出力をノイズ等によって統計的に区別でき

なくする技術）を適用し、保証可能な最大安全半径、防御ノイズ付推論回数、許容される敵対的データの存在確率等の関係を明確にした。

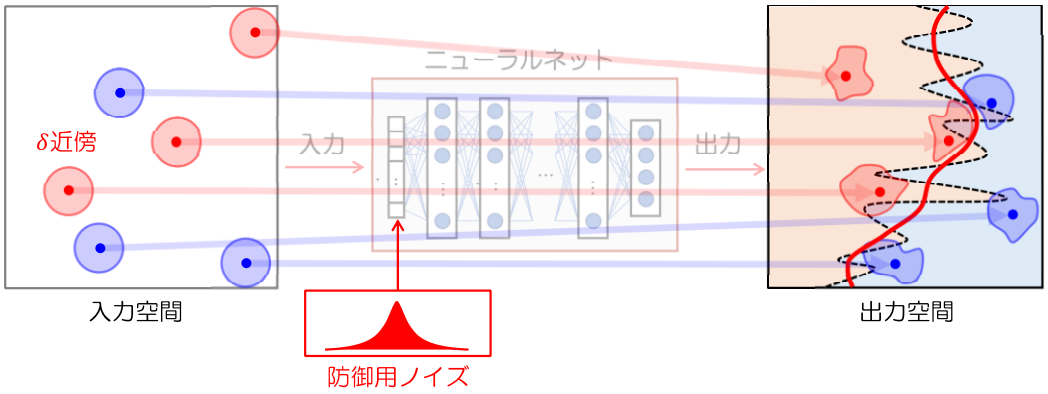


図 7.7　ランダムスムージングによるロバストネスの向上

Cohen 等は、ランダムスムージングによって確率的に保証可能な最大安全半径を、Lecuyer 等の方法[70]よりも精度良く（より大きく）見積もる方法を提案した[71]。

ランダムスムージングは 1 回の推論を行うために内部で複数回（実験的には数十～数百回程度）推論する必要はあるが、Lecuyer や Cohen 等の研究によって、大規模なネットワークに対しても、ロバストネスを確率的に保証することが可能になる。

7.2.7　ロバストネス向上、保証なし

Madry 等は訓練データセットの各データの最大安全半径がδ（指定値）になるように訓練する方法（敵対的訓練）を提案した[72]。この方法では、訓練中に各訓練データのδ近傍で敵対的データになる可能性のあるデータを探索し、そのデータに対しても正しい推論を行えるように訓練する。Wong 等の保証付のロバスト訓練[69]と比較して、ロバストネスを保証することはできないが、ロバスト訓練よりも大きなネットワークに適用可能である。ランダムスムージングのように推論時に複数回の推論を行う必要がなく、ロバストネス向上のための候補技術になりうる。

7.3　まとめ

一般に、ロバストネスを向上させると正解率が低下する傾向にあり、現在は正解率などの評価指標の方が重視されることが多い。しかし、ロバストネスを考慮しない場合、わずかなノイズでも正解率が低下する可能性があるため、誤判断によるリスクが高い場合はロバストネスによる評価も重要である。今回調査したロバストネスに関する技術は本報告書執筆時（2019 年頃）の論文で提案されたものであり、まだこれらの技術を容易に利用できる評価環境は整備されていないが、技術的には実用的なニューラルネットワークにも適用可能になりつつある。今後、そのような評価環境が整備されれば、ロバストネスもニューラルネットワークの一般的な評価指標のひとつになりうると考える。

8　汎化誤差の評価技術

第 1 章で紹介した機械学習品質マネジメントガイドライン[1]では、機械学習要素の品質特性として、機械学習モデルの**正確性**「データセットの入力に対して十分に正確な推論が行われること」と機械学習モデルの**安定性**「データセット以外の入力に対しても安定した推論が行われること」が定義されている。データセットによる従来の評価指標（正解率や F 値など）は、正確性を評価するためには有効であるが、データセット以外の入力を対象とする安定性を保証するためには十分ではない。

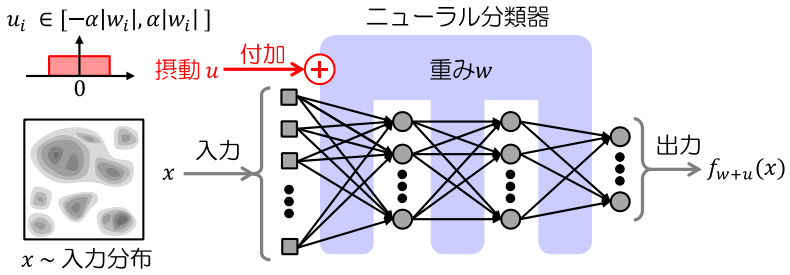


図 8.1　重み摂動付加ニューラル分類器 f_{w+u} による推論

本章では、安定性の評価指標の策定を目標として、主に文献[78]を参照し、図 8.1 に示すような順伝播型ニューラルネットワークを分類目的で訓練した**ニューラル分類器**（以下、単に**分類器**とよぶ）を対象に、その安定性（データセットに含まれない未見のデータに対しても確率的に性能）を保証するため、**重み摂動付加汎化誤差**の上界見積法について説明する。重み摂動付加汎化誤差とは、図 8.1 に示すように、任意の入力（未見のデータを含む）に対する推論過程で、分類器の重み（訓練パラメータ）に摂動を付加した場合の不正解率の期待値である。重み摂動付加汎化誤差は汎化性能との相関関係が高いことが報告されており[73][74]、安定性の評価に有効であると考えられる。

まず、8.1 節で重み摂動付加汎化誤差を定義し、8.2 節で重み摂動付加汎化誤差上界の見積法を紹介して、8.3 節で最悪重み摂動のための閾値の決め方について説明する。8.4 節で実際に重み付加汎化誤差上界の見積実験の結果を示す。8.5 節で関連技術について述べ、8.6 節で重み摂動付加汎化誤差上界の特長をまとめる。なお、本章では摂動は（入力ではなく）重みに付加することを前提としており、文脈から判断できる場合は、しばしば“重み摂動”を単に“摂動”と略記する。

8.1　重み摂動付加汎化誤差

本章では、重みに付加する摂動として、**無作為摂動**と**最悪摂動**の二種類の摂動を扱う。ここで、無作為摂動とは指定された範囲内で無作為に付加される摂動であり、最悪摂動とは指定された範囲内で不正解するように付加する摂動である。最悪摂動でも不正解にできるとは限らないが、実際に不正解にできる摂動を**敵対的摂動**とよぶ。

図 8.2 は二つの分類器 A, B の出力空間での決定境界と、指定範囲内の微小な摂動を重みに付加したときの出力の変動範囲の例である。図 8.2 左側の拡大図中の点が摂動なしの場合の出力であり、その周りの領域は微小な摂動（指定範囲内）を重みに付加した場合の出力

の変動範囲を表している。この拡大図の網目の領域内（不正解）に出力を変える摂動が敵対的摂動である。無作為摂動は、図 8.2 の分類器 A のように敵対的摂動の割合が大きい場合の評価には有効であるが、分類器 B のようにわずかな敵対的摂動の可能性を評価するためには適さない。分類器 B のような場合、無作為摂動では不正解になる確率は非常に低いが、意図的に敵対的摂動を生成することは可能であり、わずかな敵対的摂動の存在、すなわち、危険性を評価することも重要である。最悪摂動はそのような危険性を評価するために有効である。

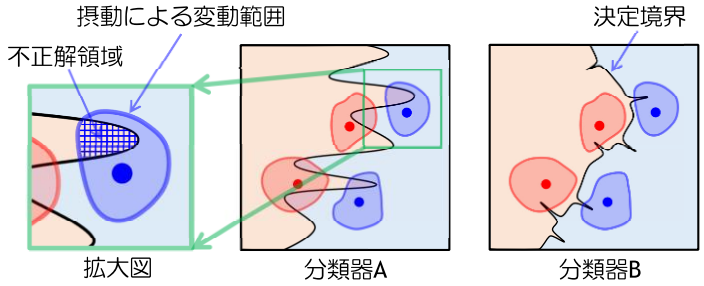


図 8.2　重み摂動付加分類器の出力と決定境界

本節では、分類器を入力xと出力yとの関係 ($y = f_w(x)$) を表す関数 f_w によってモデル化する。ここで、$w \in \mathbb{R}^\omega$ はニューラルネットワークのニューロン間の結合の重み（訓練パラメータ、総数 ω ）である。この重み w に付加する摂動 $u \in \mathbb{R}^\omega$ の集合 $U_{w,\alpha}$ を、各重みの大きさに比例（係数 α ）するように与える。

$$U_{w,\alpha} \coloneqq \{(u_1, \ldots, u_\omega) \in \mathbb{R}^\omega \mid \forall i. |u_i| \leq \alpha |w_i|\} \tag{8.1}$$

この摂動集合 $U_{w,\alpha}$ から無作為に一つの摂動を選択するための多次元一様分布を $\mathcal{U}_{w,\alpha}$ と表記する。すなわち、$u \sim \mathcal{U}_{w,\alpha}$ ならば、$u_i \sim \mathbf{U}(-\alpha|w_i|, \alpha|w_i|)$ である。以降、文脈からw, α が明確な場合は $U_{w,\alpha}$, $\mathcal{U}_{w,\alpha}$ を U , $\mathcal{U}$ のように略記する。

入力データ x とそのクラス y の組 (x, y) の分布を $\mathcal{D}$ とする。このとき、各データ (x, y) について、任意の摂動 $u \sim \mathcal{U}$ を付加した場合の不正解率の期待値（**摂動付加個別誤差**とよぶ）を次のように定義する。

$$\mathbf{r}^{\alpha}_{(x,y)}(f_w) \coloneqq \mathbb{E}_{u \sim \mathcal{U}}[\ell(f_{w+u}(x), y)] \tag{8.2}$$

ここで、$\ell(y, y')$ は損失関数であり、本章では次の 0-1 損失関数を用いる。

$$\ell(y, y') \coloneqq \mathbb{I}[y \neq y'] \tag{8.3}$$

ここで、$\mathbb{I}(b)$ は次のように定義される指示関数であり、上の式(8.3)では、不正解ならば 1（この値は損失値と呼ばれる）、正解ならば 0 を返す関数として使われている。

$$\mathbb{I}(b) \coloneqq \text{ if } b \text{ then } 1 \text{ else } 0 \tag{8.4}$$

摂動付加個別誤差は図 8.2 の拡大図の“摂動による変動範囲”に対する“網目の領域（不正解）”の割合である。このとき、**無作為摂動付加汎化誤差**は、任意のデータに対する摂動付加個別誤差の期待値であり、次のように定義される。

$$\mathbf{R}^{\alpha}(f_w) \coloneqq \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbf{r}^{\alpha}_{(x,y)}(f_w)\right] \tag{8.5}$$

無作為摂動付加汎化誤差は、分類器Bのようにわずかな摂動付加個別誤差（わずかな敵対的摂動領域）の影響はほとんど受けない。そこで、各データ (x,y) に対する敵対的摂動がわずかでも存在する場合 $\left(\mathbf{r}^{\alpha}_{(x,y)}(f_w) > \theta_{(x,y)}\right)$ は**頑健ではない**として、**最悪摂動付加汎化誤差**を次のように定義する。

$$\mathbf{W}^{\alpha}_{\theta}(f_w) \coloneqq \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbb{I}\left[\mathbf{r}^{\alpha}_{(x,y)}(f_w) > \theta_{(x,y)}\right]\right] \tag{8.6}$$

$$\Theta \coloneqq \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\theta_{(x,y)}\right] \tag{8.7}$$

ここで、$\theta_{(x,y)}$ はデータ (x,y) ごとに設定可能な閾値であり、全ての閾値がゼロ $(\Theta = 0)$ ならば、最悪摂動付加汎化誤差は、敵対的摂動が一つでも存在するデータ数の（全データ数に対する）割合になる。現実的には、閾値ゼロの条件は強すぎることもあり、計算コストの観点からも適切な微小な閾値に設定することが妥当である。この閾値設定法については 8.3 節で説明する。

8.2 重み摂動付加汎化誤差上界の見積法

一般に入力対象と摂動は無数に存在するため、摂動付加汎化誤差を正確に計算することは困難であるが、その上界に関するさまざまな既存研究がある。本節では、無作為摂動付加汎化誤差と最悪摂動付加汎化誤差の上界の見積式を紹介する。

無作為摂動付加汎化誤差上界については多くの既存研究がある。例えば、よく知られている汎化誤差上界に関する Maurer の定理[75]とサンプル収束上界の定理[76]との組合せにより、不確実度 $\delta \in (0,1)$に対して、次の不等式が確率（信頼度）$(1-\delta)$ 以上で成り立つ[77]。

$$\mathbf{R}^{\alpha}(f_w) \leq \overline{\mathbf{R}}^{\alpha}_{T,V,\delta_0,\delta}(f_w) \tag{8.8}$$

ここで、$T\sim\mathcal{D}^n$ は（訓練に使われていない）テストデータセット（サイズn）、$V\sim\mathcal{U}^m$ は無作為摂動サンプル（サイズm）、$\delta_0 \in (0,\delta)$ は摂動サンプルによる不確実度であり、式(8.8)の右辺の上界$\overline{\mathbf{R}}^{\alpha}_{T,V,\delta_0,\delta}(f_w)$は次式により与えられる。

$$\overline{\mathbf{R}}^{\alpha}_{T,V,\delta_0,\delta}(f_w) \coloneqq kl^{-1}\left(\overline{\overline{\mathbf{R}}}^{\alpha}_{T,V,\delta_0}(f_w), \frac{1}{n}\ln\left(\frac{2\sqrt{n}}{\delta-\delta_0}\right)\right) \tag{8.9}$$

ここで、$\overline{\overline{\mathbf{R}}}^{\alpha}_{T,V,\delta_0}(f_w)$ は摂動サンプルVによる摂動付加テスト誤差 $\widehat{\overline{\mathbf{R}}}^{\alpha}_{T,V}(f_w)$ の上界であり、次式により与えられる。

$$\overline{\overline{\mathbf{R}}}^{\alpha}_{T,V,\delta_0}(f_w) \coloneqq kl^{-1}\left(\widehat{\overline{\mathbf{R}}}^{\alpha}_{T,V}(f_w), \frac{1}{m}\ln\left(\frac{2}{\delta_0}\right)\right), \tag{8.10}$$

$$\widehat{\overline{\mathbf{R}}}^{\alpha}_{T,V}(f_w) \coloneqq \frac{1}{nm}\sum_{u\in V}\sum_{(x,y)\in T}\ell(f_{w+u}(x),y) \tag{8.11}$$

ここで、$kl^{-1}(q,b)$は次のように定義される値であり、

$$kl^{-1}(q,b) \coloneqq \sup\{p \in [q,1] \mid kl(q \parallel p) \leq b\} \tag{8.12}$$

$kl(q \parallel p)$は次の二項 KL（Kullback-Leibler）情報量である。

$$kl(q \parallel p) \coloneqq q \ln\left(\frac{q}{p}\right) + (1-q)\ln\left(\frac{1-q}{1-p}\right) \tag{8.13}$$

最悪摂動付加汎化誤差上界についての研究は少ないが、例えば、文献[78]より、不確実度 $\delta \in (0,1)$に対して、次の不等式が確率（信頼度）$(1-\delta)$ 以上で成り立つ。

$$\mathbf{W}_{\theta}^{\alpha}(f_w) \leq \bar{\mathbf{W}}_{\theta,T,V,\delta_0,\delta}^{\alpha}(f_w) \tag{8.14}$$

ここで、各パラメータは式(8.8)の場合と同様に、$T \sim \mathcal{D}^n, V \sim \mathcal{U}^m, \delta_0 \in (0,\delta)$ であり、式(8.14)の右辺の上界$\bar{\mathbf{W}}_{\theta,T,V,\delta_0,\delta}^{\alpha}(f_w)$は次式により与えられる。

$$\bar{\mathbf{W}}_{\theta,T,V,\delta_0,\delta}^{\alpha}(f_w) \coloneqq kl^{-1}\left(\bar{\bar{\mathbf{W}}}_{\theta,T,V,\delta_0}^{\alpha}(f_w), \frac{1}{n}\ln\left(\frac{2}{\delta-\delta_0}\right)\right) \tag{8.15}$$

ここで、$\bar{\bar{\mathbf{W}}}_{\theta,T,V,\delta_0}^{\alpha}(f_w)$ はテストデータセットに対する敵対的摂動が存在する割合の上界であり、摂動サンプルVに対する摂動付加個別誤差の上界$\bar{\mathbf{r}}_{(x,y),V,\delta_1}^{\alpha}(f_w)$から次式により与えられる。

$$\bar{\bar{\mathbf{W}}}_{\theta,T,V,\delta_0}^{\alpha}(f_w) \coloneqq \frac{1}{n}\sum_{(x,y)\in T}\left[\mathbb{I}\left[\bar{\mathbf{r}}_{(x,y),V,\frac{\delta_0}{n}}^{\alpha}(f_w) > \theta_{(x,y)}\right]\right] \tag{8.16}$$

$$\bar{\mathbf{r}}_{(x,y),V,\delta_1}^{\alpha}(f_w) \coloneqq kl^{-1}\left(\hat{\mathbf{r}}_{(x,y),V}^{\alpha}(f_w), \frac{1}{m}\ln\left(\frac{2}{\delta_1}\right)\right) \tag{8.17}$$

$$\hat{\mathbf{r}}_{(x,y),V}^{\alpha}(f_w) \coloneqq \frac{1}{m}\sum_{u\in V}\ell(f_{w+u}(x), y) \tag{8.18}$$

最悪摂動付加汎化誤差の閾値の期待値θについては、次の 8.3 節で説明する。

8.3 最悪重み摂動のための閾値

無作為摂動付加汎化誤差上界 $\bar{\mathbf{R}}_{T,V,\delta_0,\delta}^{\alpha}(f_w)$ は、式(8.11)にしたがって、テストデータセットTに摂動サンプルVを付加したときの不正解率の平均値 $\hat{\bar{\mathbf{R}}}_{T,V}^{\alpha}(f_w)$ を計測すれば計算することができる。一方、最悪摂動付加汎化誤差上界は閾値θに依存しており、固定閾値方式と適応閾値方式の二つの計測法[78]が実用的である。以下、この二つの方式について 8.3.1 小節と 8.3.2 小節で説明する。

8.3.1 固定閾値

摂動サンプルV（サイズm）とテストデータセットT（サイズn）が与えられたとき、Vの中に敵対的摂動が存在するデータの集合 $T_1 \subseteq T$

$$T_1 \coloneqq \{(x,y) \in T \mid \exists u \in V.\ f_{w+u}(x) \neq y\} \tag{8.19}$$

とそれ以外の集合 $T_0 = T - T_1$ に、テストデータセットTを分割し、集合 T_0 と T_1 の各サイズ

n_0 と n_1 を計測する。このとき、データ非依存の閾値 $\theta^{fix}_{m,n,\delta_0}$ を

$$\theta^{fix}_{m,n,\delta_0} \coloneqq 1 - \left(\frac{\delta_0}{2n}\right)^{\frac{1}{m}} \tag{8.20}$$

にとると、式(8.16)のテスト誤差上界$\bar{\bar{\mathbf{W}}}^{\alpha}_{\theta,T,V,\delta_0}(f_w)$ は次式により求められる。

$$\bar{\bar{\mathbf{W}}}^{\alpha}_{\theta^{fix}_{m,n,\delta_0},T,V,\delta_0}(f_w) = \frac{n_1}{n} \tag{8.21}$$

すなわち、摂動サンプルVに対して、一つでも不正解になるデータの個数n_1を数えればよい。

固定閾値 θ_0 が指定される場合は、その閾値が式(8.20)の閾値に等しくなるように、次式により摂動サンプルサイズ m を決定する。

$$m \geq \frac{\ln(\delta_0 \,/\, (2n))}{\ln(1-\theta_0)} \tag{8.22}$$

例えば、$n = 5000,\ \delta_0 = 0.05, \theta_0 = 0.01$（閾値 1%）ならば、摂動サンプルサイズ$m$は1215である。高い頑健性を保証するためには閾値は小さい方が良いが、より多くの摂動サンプルが必要になる。実際に計算可能な妥当な閾値を設定する必要がある。なお、固定閾値の期待値θは固定閾値$\theta^{fix}_{m,n,\delta_0}$である。

8.3.2　適応閾値

図 8.2 の分類器 B のように敵対的摂動の割合が非常に小さい場合、それが摂動サンプルVに含まれる可能性は低い。そこで、損失関数の勾配等を利用した敵対的摂動探索（例. 文献[73]の Algorithm 3）の併用が考えられる。そこで、8.3.1 小節の固定閾値方式のデータの集合T_0（m個の摂動サンプル中に敵対的摂動なし）に対して、さらに勾配法等の敵対的摂動探索を適用して、この探索によっても敵対的摂動が発見されないデータの集合$T_{00} \subseteq T_0$と発見されたデータの集合 $T_{01} = T_0 - T_{00}$ に分割し、T_{00} と T_{01} の各サイズ n_{00} と n_{01}を計測する。このとき、データ依存の閾値 $\theta^{ada}_{m,n,\delta_0}$ を

$$\theta^{ada}_{m,n_{00},\delta_0,(x,y)} \coloneqq \begin{cases} 0 & \text{if } (x,y) \in T_{01} \cup T_1 \\ \theta^{fix}_{m,n_{00},\delta_0} & \text{if } (x,y) \in T_{00} \end{cases} \tag{8.23}$$

にとると、式(8.16)のテスト誤差上界$\bar{\bar{\mathbf{W}}}^{\alpha}_{\theta,T,V,\delta_0}(f_w)$ と閾値の平均値は次式により求められる。

$$\bar{\bar{\mathbf{W}}}^{\alpha}_{\theta^{ada}_{m,n_{00},\delta_0},T,V,\delta_0}(f_w) = \frac{n_{01}+n_1}{n} \tag{8.24}$$

$$\hat{\theta}_T = \frac{n_{00}}{n}\theta^{fix}_{m,n_{00},\delta_0} \tag{8.25}$$

効果的な探索によって多くの敵対的摂動を発見できれば、n_{00} は減少し、閾値平均 $\hat{\theta}_T$ を低くすることができる。

適応閾値の最悪摂動付加汎化誤差上界は、式(8.24)を式(8.15)に代入して計算できる。また、適応閾値の期待値の上界は、敵対的データを発見できないデータ数の割合(n_{00}/n) の上界 $\left(kl^{-1}\left((n_{00}/n), \ln(2/\delta)/n\right)\right)$ を求め、式(8.25)の n_{00} をその上界 $\bar{n}_{00}$ に置換して得ることができる。

8.4　摂動付加汎化誤差上界の見積実験

本節では、8.3 節で説明した式(8.9)と式(8.15)による摂動付加汎化誤差上界の見積実験について説明する。本実験では、手書き数字の画像データセット MNIST（28 × 28画素, 輝度[0,1]）で畳み込み型ニューラルネットワーク CNN（重みとバイアスの総数：121,930）を訓練した 8 個の分類器 CNN#1~8 の摂動付加汎化誤差上界を見積る。これら 8 個の分類器 CNN#1~8 の訓練設定の組合せを表 8.1 に示す。評価に使用したテストデータセット T のサイズ n は5000、摂動サンプル V のサイズ m は1215、汎化誤差上界の不確実度 δ は 0.1（10%）、摂動に関する上界の不確実度 δ_0 は 0.05（5%）である。摂動サンプルサイズは、8.3.1 小節の計算例で説明した固定閾値を1%にするためのサイズである。適応閾値方式で用いた敵対的摂動探索法は、摂動範囲（超直方体）に勾配法を適用して、損失関数が増加する方向の頂点を繰り返し探索する手法である。なお、摂動は重みとバイアスに付加し、バッチ正規化の訓練パラメータ（スケールγとシフトβ）には付加しない。

表 8.1　訓練設定と分類器番号（CNN#1~8）

バッチ正規化		無		有	
ドロップアウト率		0	0.1	0	0.1
正則化 L^2係数	0	#1	#3	#5	#7
	0.001	#2	#4	#6	#8

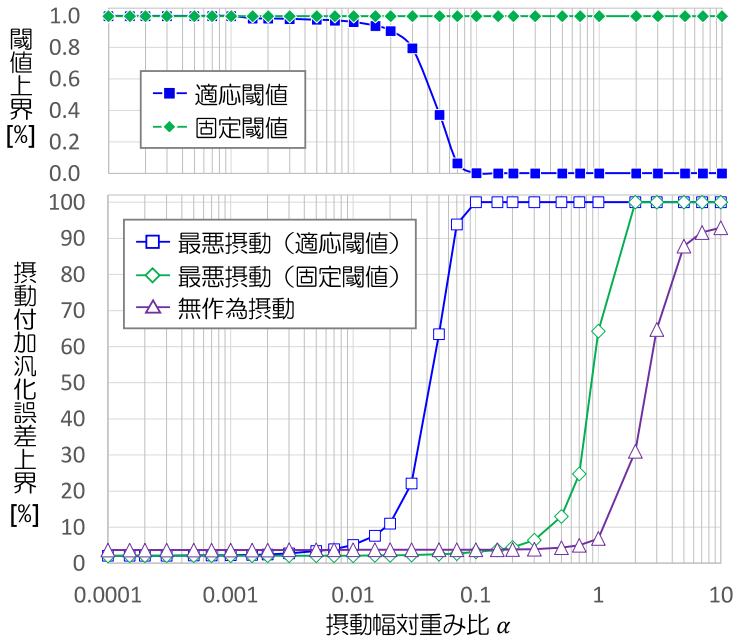


図 8.3　CNN#4 の無作為/最悪摂動付加汎化誤差上界と閾値上界の見積結果

図 8.3 に、式(8.9)と式(8.15)による分類器 CNN#4 の無作為摂動と最悪摂動（固定閾値/適応閾値）付加汎化誤差上界の見積結果を示す。横軸は重みの大きさに対する摂動最大幅の比 α である。無作為摂動付加汎化誤差上界は $\alpha = 1$ 程度から、最悪摂動付加汎化誤差上界は固定閾値（1%）の場合は 0.1 程度から、適応閾値の場合は 0.01 程度から増加している。図 8.3

には閾値の期待値上界も示しており、探索によって発見される敵対的摂動数（誤差）の増加とともに、適応閾値の減少を確認できる。

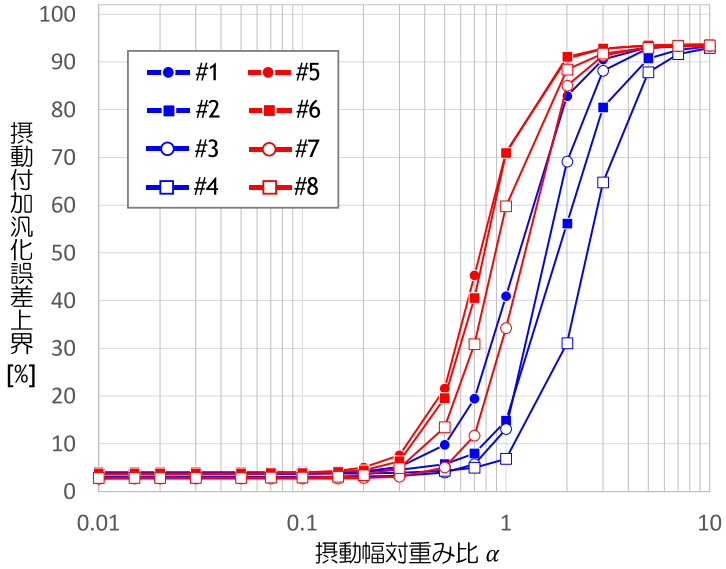


図 8.4　無作為摂動付加汎化誤差上界の見積結果

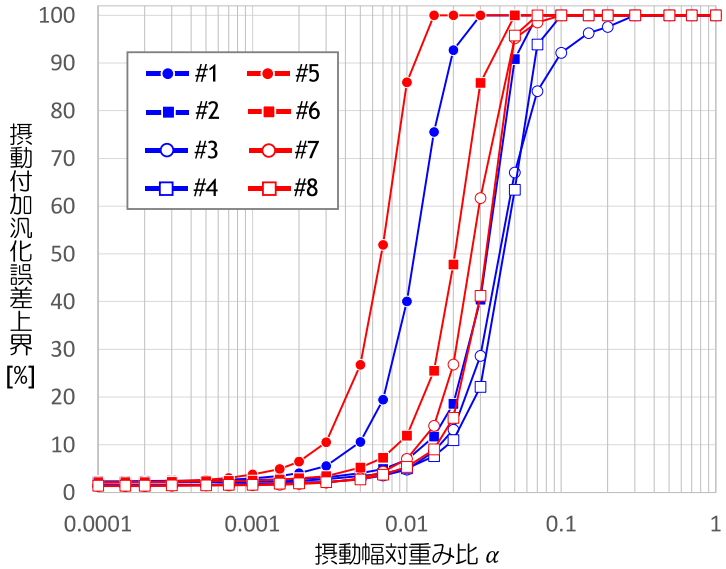


図 8.5　最悪摂動付加汎化誤差上界（適応閾値）の見積結果

図 8.4 と図 8.5 に、8 個の分類器 CNN#1~8 の無作為摂動付加汎化誤差上界と最悪摂動付加汎化誤差上界（適応閾値）の見積結果を示す。最悪摂動付加の方が無作為摂動付加よりも小さな摂動幅で誤差が増加しているだけでなく、無作為摂動付加と最悪摂動付加では、各分類器の誤差の増加傾向に違いがみられる。例えば、無作為摂動に対しては、分類器 CNN#3 の方が CNN#4 よりも明らかに耐性が弱い（誤差の増加が速い）が、最悪摂動では CNN#3 と CNN#4 の耐性に大差は見られない。

図 8.6 と図 8.7 に、無作為摂動付加汎化誤差上界（$\alpha = 0.3$）と最悪摂動付加汎化誤差上界（適応閾値, $\alpha = 0.003$）の内訳を示す。図中の TE は摂動付加なしのテスト誤差、Δ_N は摂動付加による誤差の増分、Δ_G は汎化による誤差の増分（汎化ギャップ）である。汎化ギャップは1~2%程度であり、タイトな上界が見積もられている。摂動なしテスト誤差 TE が同程度の場合でも、摂動を付加することによって違いが顕在化している。

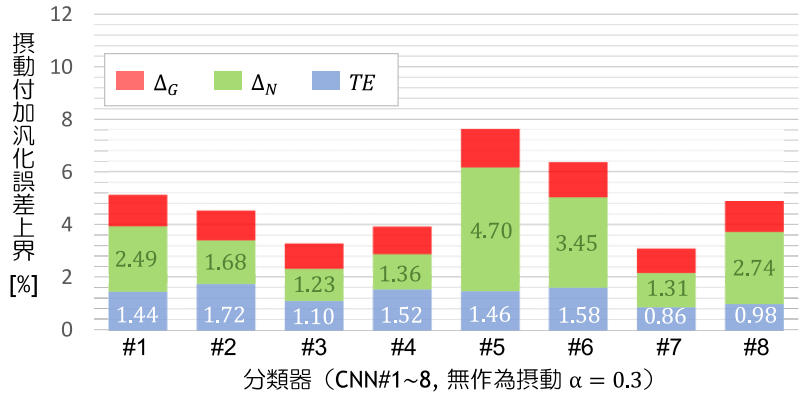


図 8.6　無作為摂動付加汎化誤差上界の内訳

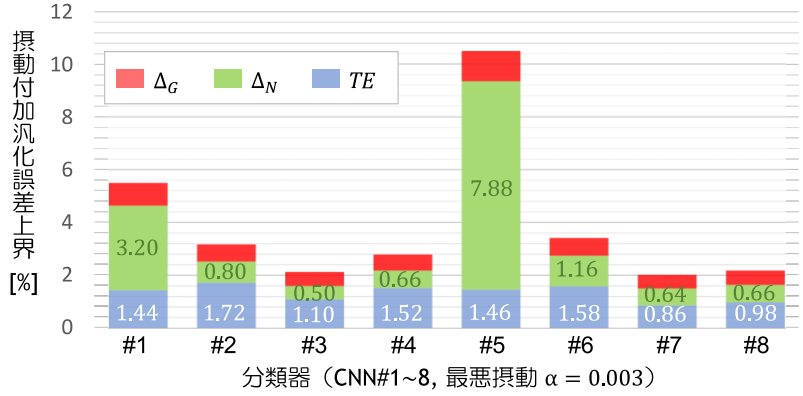


図 8.7　最悪摂動付加汎化誤差上界（適応閾値）の内訳

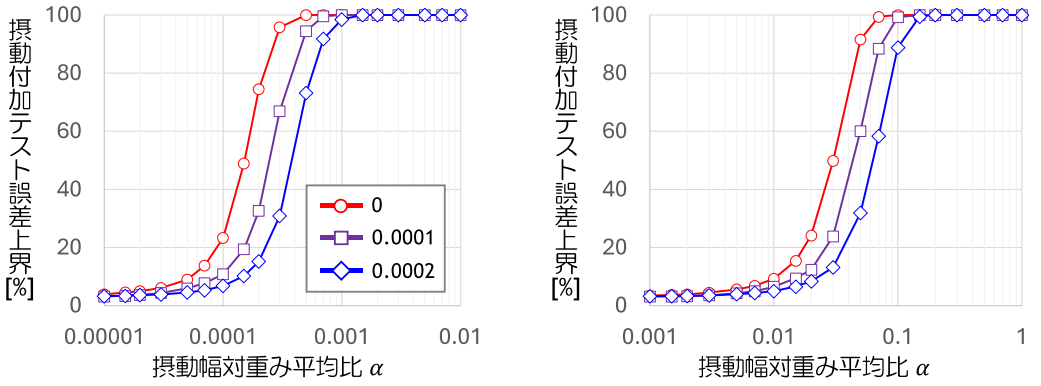
本節の見積実験で使用したプログラムは WP-GEB-Estimator（WP-GEB：Weight-Perturbed Generalization Error Bounds）という名前で website[79]から公開されている。

8.5　関連技術

汎化誤差上界を確率的に保証する定理は、テスト誤差（テストデータセット）ベースの定理（例. Chernoff bound[80]）と訓練誤差（訓練データセット）ベースの定理（例. PAC-Bayes bound[75][81]）に大別できる。訓練誤差ベースの汎化誤差上界の利点は、汎化誤差を小さくする訓練アルゴリズムの理論的な研究に適用できることである。また、訓練用とは別にテスト用のデータサンプルを用意できない場合でも学習モデルを評価できることもあげられる。しかし、訓練誤差ベースの汎化誤差上界では、その上界の計算結果（不正解率の上界）が 100% に近い意味のない値になることが指摘されている[82]。最近では、例えば、PAC-

Bayes bound において学習モデル（入出力関数）の確率分布を用いる手法[82]、訓練にランダムラベルのデータを含める手法[83]、モデル圧縮による手法[84]など、訓練誤差ベースの汎化誤差上界を低く抑える手法も提案されているが、訓練誤差と上界の差を小さくすることは容易ではない。一方、テスト誤差ベースの汎化誤差上界では、訓練データセットの他にテストデータセットを用意する必要はあるが、訓練誤差ベースと比較して、十分に汎化誤差に近い上界を得ることができる。例えば、図 8.6 と図 8.7 では汎化ギャップ Δ_G は2%以下に抑えられている。本章では、実用性の観点からテスト誤差ベースの汎化誤差上界の見積法を基本として紹介した。

無作為摂動付加汎化誤差上界に関する既存研究と比較して、最悪摂動付加汎化誤差上界に関する既存研究は少ないが、Tsai 等[85]は、最悪摂動をニューラル分類器の重みに付加した場合の影響をニューラルネットワークの構造から厳密に形式化し、最悪摂動を付加した場合に各層で生じる誤差を積算して分類器の各クラスの出力差上界の計算式を与えた（定理 2[85]）。図 8.8 に簡単な全結合型 3 層ニューラルネットワークを MNIST で訓練した 3 個の分類器（正則化L^2係数：0, 0.0001, 0.0002）の最悪摂動付加テスト誤差上界を、Tsai 等[85]の定理 2 と本章で紹介した式(8.24)で計算した結果を示す。Tsai 等の定理 2 を適用した図 8.8(a)の場合は、1 個でも敵対的摂動が存在する場合は頑健ではないと判定されるため（閾値0%, 信頼度100%に相当）、本章で紹介した式(8.24)を適用した図 8.8(b)の場合（適応閾値1%以下、信頼度95%以上）よりも大きなテスト誤差が得られているが、3 個の分類器の耐性について同様な傾向が得られている。Tsai 等の計算式は厳密であり、最悪摂動の影響を計算するために非常に有効であるが、ネットワークの内部構造に依存するため、適用可能な分類器に制約があることに注意が必要である（論文[85]には全結合層と畳み込み層に対する誤差の計算式が与えられている）。



(a) Tsai 等[85]の定理 2 の計算式による　　(b) 本章で紹介した計算式(8.24)による

図 8.8　最悪重み摂動付加テスト誤差上界計算例

8.6　「機械学習モデルの安定性」評価に向けて

本章では、機械学習モデルの**安定性**「データセット以外の入力に対しても安定した推論が行われること」[1]を評価するため、ニューラル分類器を対象として**無作為/最悪摂動付加汎**

化誤差上界に着目し、その計算法を説明して、その有効性を実験により示した。以下の理由により、テスト誤差ベースの摂動付加汎化誤差上界は安定性の評価指標になりうると考えられる。

(1) 無作為/最悪摂動付加の理由：
8.4 節の実験結果（例えば、図 8.4、図 8.5 参照）にみられるように、(摂動のない) 汎化誤差上界のみではみえにくい性能の差を、摂動を付加することによって明確に評価できる。実際、学習モデルの汎化性能と摂動に対する耐性には相関関係があることが報告されている[73][74]。無作為摂動付加と最悪摂動付加では耐性に異なる傾向を示すことがあり、最悪摂動の方が無作為摂動よりも汎化性能との相関が高いとの報告もある[73]が、さまざまな観点から評価することは、学習モデルの性能を理解するために有益である。

(2) 入力摂動ではなく重み摂動付加の理由：
摂動はしばしば入力（主に画像）に付加されるが、順序付されていない入力（例. 表形式データの地名）に適用することは難しい。重み摂動は任意の入力形式のニューラル分類器に適用することができる。

(3) 汎化誤差上界の理由：
摂動付加汎化誤差上界を求めなくても、実際に計測可能な摂動付加テスト誤差平均（$TE + \Delta_N$）でほぼ同様な評価（例えば、図 8.6 と図 8.7 参照）ができるようにみえるが、摂動付加テスト誤差平均では、データセットサイズや摂動サンプルサイズの根拠を示すことが難しい。摂動付加汎化誤差上界は、統計的学習理論に基づいて、ある信頼度で（母集団の分布$\mathcal{D}$に従う）任意の入力に対する不正解率の期待値の上界を保証することができる。摂動付加汎化誤差上界は摂動付加テスト誤差平均から容易に計算でき、その計算時間も訓練時間や推論時間と比較して無視できるほど短い。

(4) テスト誤差ベースの理由：
最近の汎化誤差上界に関する研究は訓練誤差ベースが主流であるが、訓練誤差ベースの汎化誤差上界の計算結果は 100%に近い意味のない値になることが多い（例えば、[82]参照）。訓練誤差ベースの汎化誤差上界を低く抑える研究は進められているが、現時点（2024 年 3 月）では、訓練済み学習モデルの評価としては、テスト誤差ベースの汎化誤差上界の方が実用的と考えられる。

訓練済み学習モデルを第三者に提供するときに、その評価項目の一つに無作為/最悪摂動付加汎化誤差上界の計算結果を加えることによって、例えば、図 8.4 と図 8.5 に示すように、正解率等からではみえにくい情報を与えることができる。また、汎化誤差は、ある信頼度で任意の入力に対する不正解率の上界を保証できるため、未見のデータに対する学習モデルの性能を示すことができる。このような情報は訓練済み学習モデルを利用する際に有益であると考える。なお、本章ではデータセット MNIST による実験結果を紹介したが、文献[78]では乗り物や動物のデータセット CIFAR10 による実験結果が紹介されている。

9 敵対的データ検出技術

9.1 研究概要

与えられた入力画像が敵対的データ（Adversarial Example）であるかを判別する方法を実用的に確立することを目標として、敵対的データを生成する攻撃と検出手法について、下記の点に着目して代表的な技術の調査および実装を実施している。

- 敵対的データ検出プログラムコードの裏付けと計算実験による確認
- 敵対的データ検出手法の論文の実験結果の再現
- 機械学習システム品質評価テストベッドへの敵対的データ検出フレームワークの構築

敵対的データ検出（Adversarial Examples Detection）とは、与えられた入力の中から敵対的データを検出することであり、既存の最先端の敵対的データ検出方法は次の 4 つの主要なカテゴリに分類できる。

① メトリックベースアプローチ（例. [86]）
② ディノイザーアプローチ（例. [87]）
③ 予測不整合ベースアプローチ（例. [88]）
④ ニューラルネットワーク不変式チェック（NIC）アプローチ（例. [89]）

本章では、これら①～④の各アプローチに基づく敵対的データ検出手法を比較・評価するために追試実験を行った結果について報告する。論文[89]に報告されているように、④のアプローチ（NIC：Neural Network Invariant Checking）が①～④の中で最も高い検出率を示すことを確認できた。この追実験において、①～③については公開されている実装コードを使用したが、④については実装コードが公開されていなかったため、論文[89]にしたがって NIC を実装して追実験を行うとともに、NIC に基づく敵対的データ検出を行うための NIC フレームワークを構築した。そのため、本章では主に④の NIC について説明する。

以降、4 つのアプローチの概要を説明したのち、NIC による敵対的データの検出方法を説明して、その実装方法について述べる。次に、各アプローチの追試実験と NIC による実験の結果を示し、最後に NIC フレームワークの実装とその有効性評価結果について報告する。

9.2 敵対的データ検出アプローチの概要

以下、最先端の敵対的データ検出のための 4 つのアプローチの概要について説明する。

9.2.1 メトリックベースアプローチ

入力（および各ニューロンの出力）の統計的測定を実行して、敵対的データを検出する方法であり、Ma 等は、最近、Local Intrinsic Dimensionality（LID）と呼ばれる測定の使用を提案した[86]。この方法では、サンプルの距離分布と個々のレイヤーの近隣の数を計算することによって、サンプルを囲む領域の空間充填能力を評価する LID 値を推定し、敵対的データが大きな LID 値を持つ傾向がある性質を用いて、敵対的データを検出している。LID は、敵

対的データの検出に対して、従来のカーネル密度推定（KD）やベイジアン不確実性（BU）よりも優れており、現在この種の検出器の最先端の技術となっている。

9.2.2 ディノイザーアプローチ（**Denoiser**、ノイズ除去）

各入力に対して前処理ステップでノイズを除去することによって敵対的データを検出する方法である。この方法では、学習モデル内の主要なコンポーネントを強調できるように、学習モデルまたはノイズ除去器（エンコーダーおよびデコーダー）をトレーニングして画像をフィルター処理する。このフィルターを用いて、攻撃者が敵対的データを生成するために追加したノイズを除去し、誤分類を修正することができる。MagNet[87]は、検出器とリフォーマー（トレーニング済みの自動エンコーダーと自動デコーダー）を使用して、敵対的データを検出する方法である。

9.2.3 予測不整合ベースのアプローチ（**Prediction inconsistency based approach**）

元のニューラルネットワークと人間の知覚可能な属性で強化されたニューラルネットワークとの間の不一致を測定して、敵対的データを検出する方法である。この方法の最先端の検出技術であるフィーチャスクイージング（Feature Squeezing）[88]は、さまざまな攻撃に対して非常に高い検出率を実現することができる。フィーチャースクイージングは、ディープニューラルネットワーク DNN の不必要に大きな入力特徴空間によって攻撃者が敵対的データを生成できることに着目しており、勾配ベースの攻撃の検出に焦点を当てている。フィーチャースクイージングによる敵対的データの検出手順を以下に示す。

1. 元の入力画像にスクイージング技術（画像の色深度を減らし、画像を平滑化する技術）を適用して複数のスクイーズ画像を生成する。
2. 元の入力画像と複数のスクイーズ画像をディープニューラルネットワークに入力し、入力画像の推論結果（予測ベクトル）と各スクイーズ画像の推論結果との距離を測定する。
3. 元の入力画像とスクイーズ画像の差（距離）の一つがしきい値を超える場合に、元の入力画像を敵対的データとして検出する。

9.2.4 ニューラルネットワーク不変性チェック（**NIC**）アプローチ

NIC[89]では、ニューラルネットワーク内部の値の不変量（VI: Value Invariants）と来歴不変量（PI: Provenance Invariants）に着目する。値の不変量 VI は各層の可能なニューロン値の分布であり、来歴不変量 PI は 2 つの連続した層の可能なニューロン値パターン（2 層にわたるフィーチャ間の相関の要約）である。ある入力がこれらの不変量に違反している場合に、その入力は敵対的データとして検出される。それらの不変量 VI と PI を良性の入力データで訓練し、敵対的データを検出する 1 クラス分類（OCC）問題としてモデル化する。上で説明した①～③に基づく手法よりも高い検出率が報告されている[89]。以降、NIC のシステム設計概要と実装について、各々9.3 節と 9.4 節で詳しく説明する。

9.3　NIC のシステム設計概要

NIC の検出器の構築と検出の手順（ステップ A～C：訓練時、D～E：実行時）を図 9.1 を用いて説明する[89]。この不変量 VI, PI の訓練では、敵対的ではない良性のデータのみを使用する。

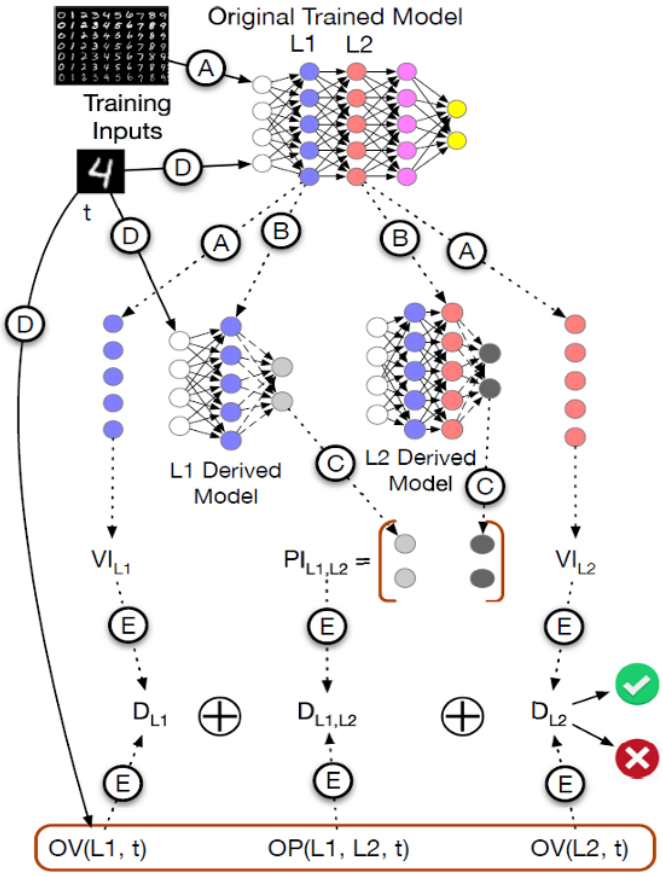


図 9.1　システム設計概要（論文[89]の図 8）

- **ステップ A**：各訓練データ入力の各層で各ニューロンの出力値を収集する。
- **ステップ B**：各層 k（例えば、L1、L2）に対して、入力層から k 層までのサブモデルを抽出し、元のモデルと同じ出力ラベルをもつ新しい softmax 層を追加して、派生モデル（図 9.1 中の DerivedModel）を作成する
- **ステップ C**：すべての派生モデルに対して各良性訓練データを入力し、これらのモデルの最終出力（すなわち、個々のクラスの出力確率値）を収集する。連続する層の組ごとに、この派生モデルの分類結果の分布を用いて訓練する。この訓練された分布が、これらの 2 つの層の PI となる。
- **ステップ D**：各テストデータ t（例えば、図 9.1 中の“4”の画像）を、元のモデルの他に、すべての派生モデルにも入力して、元のモデルの各層の活性化値を観測値 OV（例えば、図 9.1 中の OV(L1,t)）と、連続した層の派生モデルの分類結果（組）を収集す

る。この分類結果から観測された出処 OP（例えば、OP(L1,L2,t)など）を得る。

- **ステップ E**：OV と OP が対応する VI と PI の分布に適合する確率 D を計算する。入力 t が敵対的である可能性を、これらの D 値をすべて集約して同時予測する。

9.4 NIC のシステム実装

NIC に基づいて敵対的データを検出するために、PI と VI から直和空間（ベクトル）を構成し、このベクトルを分類するための OSVM（One Class Support Vector Machine）を構築する。訓練済みの DNN（Deep Neural Network）のモデル（以降、これを M と記述する）の層 l に対する入力を x_l とするとき、層 l の出力 f_l は次式により与えられる。

$$f_l = \sigma(x_l \cdot w_l^T + b_l)$$

ここで、σ は層 l の活性化関数、w_l^T は重み行列、b_l はバイアスである。このとき、VI と PI、および OSVM で分類する直和空間は次のように求められる。

- **VI の計算**：モデル M の各層 l の VI は以下の最適化問題を解いて決定する。

$$VI_l = \min\left[\sum_{x \in X_b} J(f_l \circ f_{l-1} \circ \cdots \circ f_1(x) \cdots w^T - 1)\right]$$

ここで、J はエラー評価関数、X_b は M を作成する際に使用したバッチである。また、$\circ$はモノイドであり、この場合では、f_k をベクトル化したものである。

- **PI の計算**：$PI_{l,l+1}(x)$は層 l および層$l+1$ の派生モデルの分類出力に基づいて、x が良性である（敵対的でない）確率は、次の最適化問題を解いて推測する。

$$PI_{l,l+1}(x) = \min\left[\sum_{x \in X_b} J(concat(D_l(x), D_{l+1}(x)) \cdots w^T - 1)\right]$$

ここで、層 l の派生モデルD_l は、層 l の後に softmax 層を追加してのように定義される。

$$D_l = \text{softmax} \circ\ f_l \circ f_{l-1} \circ \cdots \circ f_1$$

- **PI と VI の直和空間**：上記の最適化により求めた VI と PI から、モデル M の学習データのバッチごとに以下の直和空間（ベクトル）を作成する。

$$VI_1 \oplus PI_{1,2} \oplus VI_2 \oplus PI_{2,3} \cdots VI_B \oplus PI_{B-1,B} \oplus VI_B$$

このベクトルは $L \times 3$ 次元（L は M の層数）であり、これは個数B のベクトル空間（直和空間）になる。NIC ではこの空間に対して OSVM を行う。

9.5 計算機実験

敵対的データ検出技術（NIC)の効果を確認するため、下記の実験環境で、論文[89]の実験の追試を行なった。

- ハードウェア環境：産総研 ABCI[90]
- データセット：画像分類の実験には、MNIST[91]、CIFAR-10[92]の 2 つの一般的な画像データセットを用いた。MNIST は手書き数字認識に使用されるグレースケール画像データセットであり、CIFAR-10 はオブジェクト認識に使用されるカラー画像データセットである。なお、NIC に対しては、LFW（顔画像）[93]についても実験を行った。
- 攻撃：敵対的データの生成には、非標的型攻撃（FGSM L^2, L^∞）、標的型攻撃 JSMA、勾配ベースの攻撃（CW L^2）の方法を使用した。FGSM と JSMA の実装には、Cleverhans ライブラリ[94]を使用した。

最初に、①～③の各アプローチに基づく敵対的データ検出手法を評価するため、LID[86]、MagNet[87]、フィーチャスクイージング[88]の公開されている実装コードを用いて、各論文の追試実験を行った。その結果、各論文に報告されている検出率を確認でき、この 3 つの中では、フィーチャスクイージングが最も高い検出率を示していた。

次に、④のアプローチに基づく敵対的データ検出手法を評価するため、9.4 節で実装した NIC のコードを用いて実験を行った。表 9.1～表 9.3 に、各々、MNIST、CIFAR-10、LFW のデータセットに対する敵対的データ検出計算実験の結果示す。ここで、正答率は、9.4 節で説明した分類器（OSVM）に敵対的データを入力し、敵対的データであると判定された割合である。なお、実験に使用した CNN モデルは LeNet5、OSVM の Kernel は RBF（MNIST：$\gamma = 0.1\sim0.27$, CIFAR-10：$\gamma = 0.11\sim0.2$, LFW：$\gamma = 0.005\sim0.90$）である。本実験結果では、論文[89]で報告されていたデータセットや攻撃方法だけでなく、報告されていないデータセット LFW、攻撃方法（FGSM L^∞）についても高い検出性能を確認することができた。

表 9.1　MNIST データセットに対する敵対的データ検出計算実験結果

Data Set	Attack	Invariant	正答率	データ件数	論文[89]正答率
MNIST	FGSM L^2	VI	97%	2800	100%
		PI	98%		84%
		NIC	97%		100%
	FGSM L^∞	VI	98%	2800	—
		PI	98%		—
		NIC	98%		—
	JSMA	VI	100%	280	83%
		PI	100%		100%
		NIC	100%		100%
	CW2	VI	100%	280	95%
		PI	100%		96%
		NIC	100%		100%
	Trojan	VI	100%	3200	100%
		PI	100%		100%
		NIC	100%		100%

表 9.2 CIFAR-10 データセットに対する敵対的データ検出計算実験結果

Data Set	Attack	Invariant	正答率	データ件数	論文[89]正答率
CIFAR-10	FGSM L^2	VI	99%	6400	100%
		PI	99%		52%
		NIC	99%		100%
	FGSM L^∞	VI	100%	6400	—
		PI	100%		—
		NIC	100%		—
	JSMA	VI	97%	320	62%
		PI	95%		100%
		NIC	96%		100%
	CW2	VI	98%	320	88%
		PI	95%		89%
		NIC	96%		100%
	Trojan	VI	100%	3200	100%
		PI	100%		100%
		NIC	100%		100%

表 9.3 LFW データセットに対する敵対的データ検出計算実験結果

Data Set	Attack	Invariant	正答率	データ件数	論文[89]正答率
LFW	FGSM L^2	VI	98%	28222	—
		PI	98%		—
		NIC	98%		—
	FGSM L^∞	VI	100%	2822	—
		PI	100%		—
		NIC	100%		—
	JSMA	VI	100%	280	—
		PI	100%		—
		NIC	100%		—
	CW2	VI	100%	840	—
		PI	100%		—
		NIC	100%		—
	Trojan	VI	100%	3200	—
		PI	100%		—
		NIC	100%		—

9.6 NIC フレームワークの実装

9.5 節で NIC の有効性の確認を目的とした計算機実験を行うため、簡易に、9.3 節と 9.4 節に基づき NIC 法の実装をおこなった、その際、原論文[45]に実装上の問題点などが明らかとなった。これら問題点を明らかにしつつ、敵対的データ（Adversarial Examples）に対する脆弱性をベンチマークする環境（攻撃・防御・検出）を構築し、敵対的データの高い検出率を目標とする NIC フレームワークをテストベッドに構築するため、アルゴリズムの再検討を行なった。

9.6.1 NIC フレームワークの概要

NIC フレームワークは、各層からの出力取出し、正常データの VI, PI 計算、敵対的データの VI, PI, NIC の計算、OSVM の評価と結果の表示の 5 つのパートで構成されている。NIC フレームワークのユースケースを図 9.2 に示す。また、敵対的データを検出する処理手順を図 9.3 に示す。

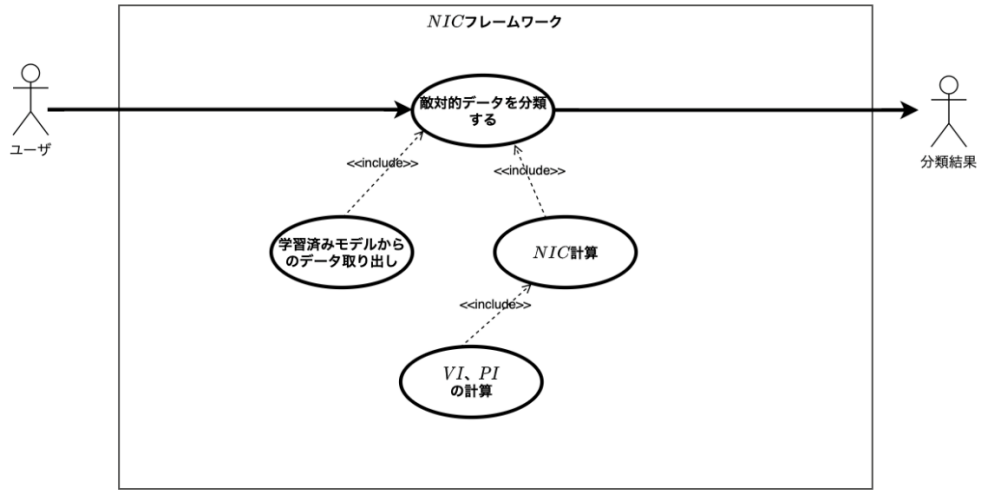


図 9.2 NIC フレームワークのユースケース

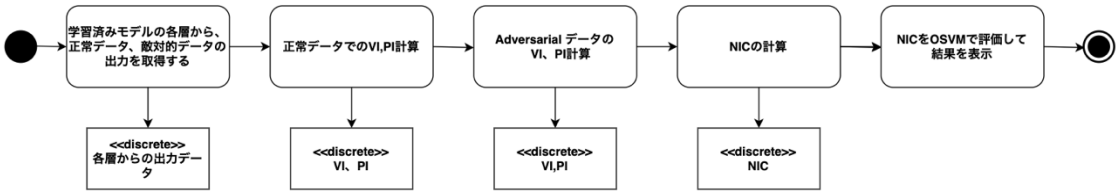


図 9.3 NIC フレームワークによる敵対的データ検出の処理手続き

図 9.3 に示すように、NIC フレームワークによる敵対的データ検出全体の処理手続きは 5 つのパートから構成されている。各パートの機能（入力、処理、出力）を表 9.4 に示す。

表 9.4 敵対的データ検出処理手続きを構成する各パートの機能

各層からの出力取り出し	
入力	正常データ（画像）
	敵対データ（画像）
	正常データで学習した、学習済みモデル（正常データで学習したモデル）
処理	正常データ、敵対的データの学習済みモデルの各層の出力を取得して、「numpy の npy」形式で保存する。
出力	各層からの出力データ

正常データの VI, PI 計算	
入力	正常データの各層からの出力データ
処理	正常データの各層の出力データから VI、PI を計算する。
出力	VI, PI

敵対的データの VI, PI 計算	
入力	敵対的データの各層からの出力データ
	正常データで PI に計算時に作成された PI の派生モデル
処理	敵対的データの各層の出力から VI, PI を計算する。
出力	VI, PI

NIC の計算	
入力	正常データの VI, PI
	敵対的データの VI, PI
処理	正常データの VI, PI から正常データの NIC を作成、敵対的データの VI, PI から敵対的データの NIC をそれぞれ計算する。
出力	正常データの NIC、敵対的データの NIC

OSVM での評価と結果の表示	
入力	正常データの NIC
	敵対的データの NIC
処理	正常データで OSVM を学習させてモデルを作成、この学習済みモデルを使用して敵対的データを判定する。OSVM は sk-learn の one class svm API を使用している。その後、判定結果を表示する。
出力	評価結果

9.6.2 **OSVM** 評価結果の出力

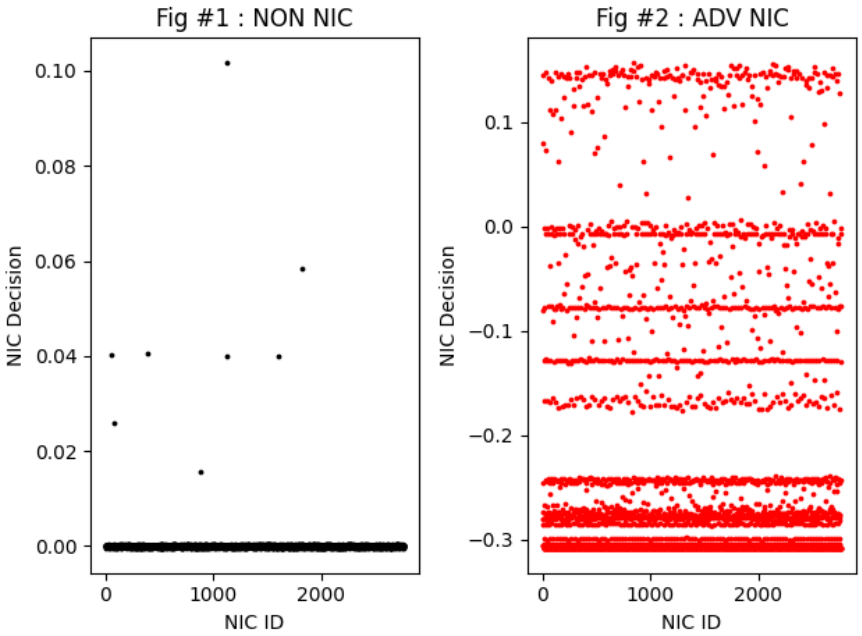
NIC フレームワークの実装には Python の機械学習のライブラリである scikit-learn を用いた。例えば、図 9.3 の最終パートである OSVM は scikit-learn の OneClassSVM の class を用いて次のように実装した。

```
class sklearn.svm.OneClassSVM(array, kernel='rbf', gamma='auto', nu=0.3)
```

ここで、各引数の意味は次の通りである。

- array：正常データの NIC でパラメータを学習し、敵対的データの NIC で推定する
- kernel：One Class SVM のアルゴリズムとして、RBF カーネルを使用する
- gamma：RBF カーネルの γ パラメータ、「auto」を指定
- nu：学習誤差の割合の上限とサポートベクトルの割合の下限を指定する（今回は 0.3）

NIC フレームワークに 1 個の正常データと、その敵対的データを入力したときの、各層からの出力値を図 9.4 に示す。ここで、横軸は各層で NIC を計算するために導出したモデルの ID（注. 例えば CNN の Convolution 層など、1 画像に対する各層からの出力は複数存在する）、縦軸は正常データの NIC の One Class SVM 分類超平面までの各 NIC の符号付き距離、すなわち、この場合、正常データへの近さを表している。図 9.4(a)の黒点が正常データに対する出力、図 9.4(b)の赤点が敵対的データに対する出力である。なお、図 9.4(b)の敵対的データの生成には FGSM L^{∞}の攻撃方法を用いた。



(a) 正常データ入力　　(b) 敵対的データ入力

図 9.4　NIC フレームワークの出力比較

正常データで One Class SVM を学習させた場合、One Class SVM の関数 $f(x)$ によって、$f(x) \geq 0$ ならば正常、$f(x) < 0$ ならば異常であると判断することができる。図 9.4 の結果は正常データを入力したときの出力がほぼ 0 の周辺に集中している（図 9.4(a)）のに対して、敵対的データを入力したときは約 94%が 0 未満となっている（図 9.4(b)）。この結果から NIC による敵対的データ検出の有効性を確認できる。

9.6.3　敵対的データの作成

図 9.3 に示すように、NIC フレームワークには敵対的データの作成プログラムは含まれな

い。敵対的データを作成する場合は CleverHans [95]の使用を推奨する。図 9.5 に、手書き数字画像（MNIST）の正常データとその画像から生成した敵対的データ（攻撃方法：FGSM L^2）および各画像データに対して推論されたラベルを示す。図 9.5(b)の推論結果（label 8）にみられるように、生成された敵対的データは全て 8 と誤判断されている。

(a) MNIST の元データ　　(b) 生成した敵対的データ

図 9.5　MNIST（手書き数字）画像からの敵対的データ生成例とその判定結果

9.6.4　VI、PI、NIC の計算コスト削減

原論文[75]の VI, PI, NIC の計算方法は 9.4 節で説明したが、そのまま計算するとそれぞれのデータ（ベクトル）の次元が非常に大きくなり、いわゆる「次元の悪魔」に囚われることになるため、なるべく次元が低下するように工夫している。以下、各計算の簡素化の方法について説明する。

- **VI の計算**：NIC フレームワークでは、入力データ（正常、敵対的双方）と、VI, PI, NIC との対応を明確にするため（検証の正確性を記すため）$X_B = 1$とする。また入力データは全て正規化して計算しているため、下記のように簡素化した式を使用している。

$$VI_l = f_l \circ f_{l-1} \circ \cdots \circ f_2 \circ f_1$$

- **PI の計算**：上記の VI と同様に、$X_B = 1$として計算している。従って下記のように簡素化した式を使用している。

$$PL_{l,l-1} = concat(D_l, D_{l-1}) \circ \cdots \circ concat(D_2, D_1)$$

・ **NIC の計算**：次元抑制のため $X_B = \left(\text{出力を取得した層の数}\right)$ としている。

9.7 Kullback-Leibler 情報量による NIC の有効性評価

Kullback-Leibler 情報量を用いて、正常データと敵対的データの画像、及び、NIC の乖離度を計算し、NIC の有効性を評価した結果について報告する。

9.7.1 Kullback-Leibler 情報量

Kullback-Leibler 情報量は 2 つの確率分布P（確率密度関数p）、Q（確率密度関数q）の間の乖離度を測る尺度である。ただし、厳密な距離の公理を満足しないので、厳密な意味での距離にはならない。Kullback-Leibler 情報量（以後、$KL(P \parallel Q)$と表す）は次式で定義される。

$$KL(P \parallel Q) = \int p(x) \log \frac{p(x)}{q(x)}$$

両者が同じ場合には 0 、乖離が進むと値が大きくなる（log があるため収束は保証されない）。図 9.6 に Kullback-Leibler 情報量の簡単な計算例を示す。図 9.6 左は、PとQ双方とも平均 0.5、分散 0.5 の正規分布の $KL(P \parallel Q)$ で、値は 0 である。右は、Pが平均 0.5、分散 0.5、Qが平均 0.55、分散 0 .55 の $KL(P \parallel Q)$ で、値は 0.053 である。

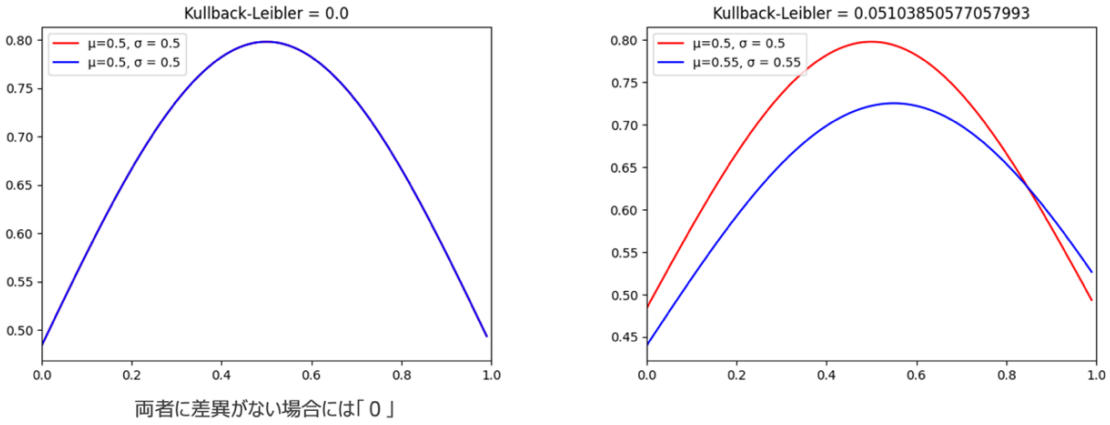


図 9.6　Kullback-Leibler 情報量の計算例

9.7.2 Kullback-Leibler 情報量の推定

Kullback-Leibler 情報量は比較する確率分布が確定していることが前提であるが、実際には、正常データ、敵対的データ双方ともに単なる画像の集合であり、分布は明確でない。しかし、確率分布が明確でない集合同士の Kullback-Leibler 情報量を近似する手法[96]が知られている。その近似計算の概略は、密度比 $r(x) = p(x)/q(x)$ を Kullback-Leibler 情報量を最小化することを制約として、θの線形多項式 $r_\theta(x)$ を最適化問題として解くことにある。

$$r_\theta(x) = \sum_{j=1}^{b} \theta_j \psi_j(x) = \boldsymbol{\theta}^T \boldsymbol{\psi}(x)$$

ここで、次式により定義される$\psi_j(x)$は RBF カーネルである。

$$\psi_j(x) = \exp\left(-\frac{\|x - x'\|^2}{2h^2}\right)$$

ここで、hは決定可能な定数でバンド幅である。

このとき、上記の線形多項式 $r_\theta(x)$ を用いて、Kullback-Leibler 情報量を求めるための目的関数は次式により与えられる。

$$\min_\theta J(\theta), \ \ where \ \ J(\theta) = \frac{1}{N'} \sum_{n'=1}^{N'} r_\theta(\boldsymbol{x}'_{n'}) - \frac{1}{N} \sum_{n=1}^{N} r_\theta(\boldsymbol{x}_n)$$

この最適化問題を解いて得られる線形多項式 $r_\theta(\boldsymbol{x})$ を用いて、Kullback-Leibler 情報量は次式により近似できる[96]。

$$KL(P \parallel Q) \sim \frac{1}{n} \sum_{i=1}^{n} \log r(\boldsymbol{x}_i)$$

9.7.3 NIC の有効性評価

9.5 節の実験結果によって NIC 法が敵対的データを異常値として検出する手段として有効であることを示したが、有効であることの理由を説明するため、正常データと敵対的データの Kullback-Leibler 情報量を比較して、双方のデータの乖離度を推定した。まず、正常データと敵対的データ(攻撃手法:FGSM L^2)の画像データ 50 個(図 9.5 参照)に対する Kullback-Leibler 情報量を求めた結果を図 9.7 に示す。図 9.7 の Kullback-Leibler 情報量の近似値は 0.46 である。なお、図 9.4 に示したように、1 画像に対して複数の NIC が存在するが、図 9.7 では、各画像の複数の Kullback-Leibler 情報量の平均値を示している。

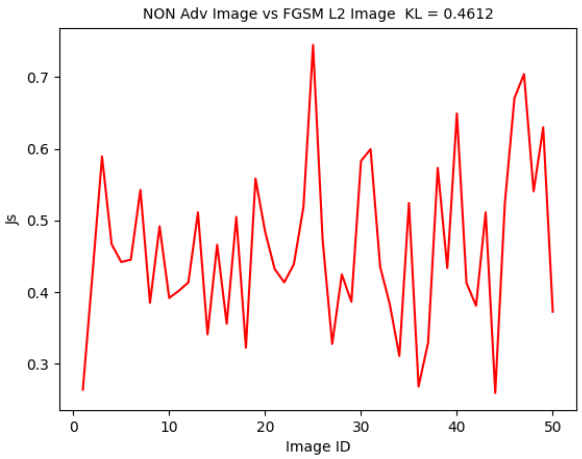


図 9.7 正常データと敵対的データの Kullback-Leibler 情報量

次に、正常データと敵対的データの NIC の Kullback-Leibler 情報量（図 9.5 の画像データから作成）を求めた結果を図 9.8 に示す。図 9.8 の Kullback-Leibler 情報量の近似値は 4.47 である。図 9.7 と図 9.8 の Kullback-Leibler 情報量には約 10 倍の違いがある。この違いが、正常データに加えられた摂動を NIC 法によってより集約された形で収集できることを表していると推定される。

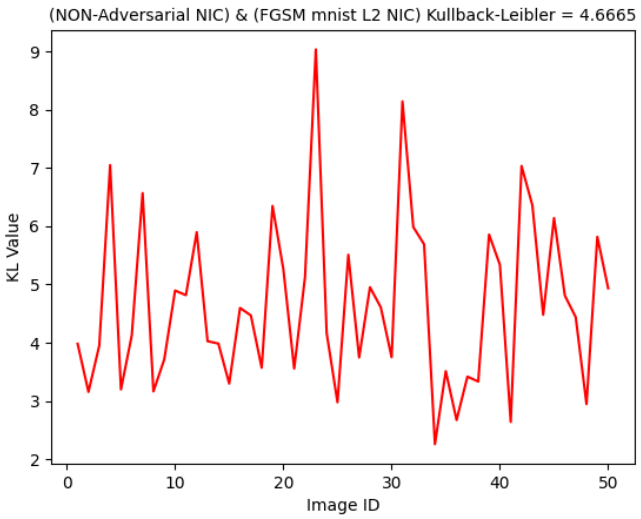


図 9.8　正常データと敵対的データの NIC の Kullback-Leibler 情報量

10 運用時における AI 品質管理技術

本章では、運用時における AI 品質管理技術として、コンセプトドリフトと呼ばれる時間経過に伴うデータ分布の変化に対し、その分布変化の検知と、変化後の分布に機械学習モデルを適応させる最新技術および同技術の関連技術である教師なしドメイン適応技術の最新技術に関する調査結果について報告する。

コンセプトドリフトは、運用中の AI システム内で稼働する機械学習モデルの性能低下を引き起こす主な原因の 1 つである。そのため、システムの運用開始時点で充足されていた品質を、運用期間を通じて維持するためには、ドリフトが生じているか否かを継続的に監視することに加え、必要に応じてシステム内の機械学習モデルを最新のデータを用いて再学習することで、ドリフト後の分布にシステムを適応させることが必要である。特に近年の機械学習技術の利用拡大に伴い、今後の AI システムの運用場面では、これまで扱われなかった種類のデータを含め、正解ラベル付けされていない大量のデータを短期間で処理することが求められる。加えて、データのプライバシやセキュリティの問題で運用前に使用した訓練データを再利用できない場合や、システムの適用先が多岐に渡ることも想定される。

そこで、2019～2020 年度において、運用中の機械学習モデルの性能維持を目的として、上記のコンセプトドリフトの検知および適応を行う最新技術に関する調査を行った。その結果、これまでに開発されている手法の多くが、検知および適応時に運用中に新たに取得した入力データの正解ラベルを用いる教師あり手法であった。しかしながら、正解ラベルは必ずしも入手できるとは限らず、入手できたとしてもコストがかかる場合が多い。そのため、適用可能性を広げるため、または運用コストを削減するためには、それらの正解ラベルを用いない「教師なし手法」や、少数の正解ラベルのみ限定的に用いる「半教師あり手法」が有望であることがわかった。そこで、その視点から整理し検討した調査結果をサーベイとしてまとめた。各サーベイに関する詳細については、検知手法に関しては機械学習品質マネジメントガイドライン[1] 9.13 節を、適応手法に関しては文献[97]をそれぞれ参照されたい。

一方、今後の AI システムの運用においては、新たに、データのプライバシやその可搬性の観点から適応前に使用した訓練データに依存しない適応技術や、入力データの分布以外の変化にも対応可能な適応技術の必要性も高まってきている。特に訓練データ（ソースデータ）に依存しない適応技術は、「ソースフリードメイン適応手法」や「test-time 適応手法」とも呼ばれ、ソースデータの管理や送受信に係る費用削減、同データの保管に関するセキュリティの観点からも注目を集めている。

そこで 2021 年度においては、上述の 2020 年度までの調査に引き続き、教師なしコンセプトドリフト適応技術および教師なしドメイン適応技術を中心に、2019 年以降の機械学習分野の主要国際会議で発表されたデータ変化に対する教師なし適応技術の最新の研究動向に関して調査を行った。その結果、近年においては、従来の教師なし適応技術に加え、上述のソースフリーな適応技術や、ラベルシフトなどの入力データの分布以外の変化にも対応可能な適応技術の開発が行われていることがわかった。さらには調査した論文に記載された幾つかの手法に関しては、画像分類問題だけでなく、セマンティックセグメンテーション問題や物体検知問題に対する有効性の検証も行われている。これら教師なし適応技術の研究動向は、データプライバシーの保持などの AI 運用における新たな課題に対して解決を図るとともに、今後の AI 運用における様々な場面においての活用へと広がっていると言える。

本サーベイに関する詳細については、文献[98]を参照されたい。

2022 年度においても、この領域についての注目度は大きく、test-time 適応手法においてより進歩的な研究がなされている。適応時に伴う忘却を抑制するための機構を明示的に取り入れる手法、適応手法の複雑化に伴う計算量増加に対する反面教師的なアプローチを採る手法、近年利用が広がっている Vision Transformer に適用可能な手法などがある。また、タスクについても 3 次元セマンティックセグメンテーションや、時間関係を考慮した物体認識などさらなる広がりを見せている。これら最新の手法について調査を行い、これまでの 2019～2021 年度に実施した調査を併せて包括的な整理を行った結果を、文献[99]にまとめた。詳細についてはそちらを参照されたい。

11 参考文献リスト

（1 章の参考文献）

[1] 大岩 寛他, 機械学習品質マネジメントガイドライン 第 4 版, Digiarc-TR-2023-03, CPSEC-TR-2023003, 2023 年 11 月. https://www.digiarc.aist.go.jp/publication/aiqm/
[2] 宮城 優里, 大西 正輝, 作業者情報に注目した機械学習モデル比較可視化手法, 第 24 回画像の認識・理解シンポジウム, I31-22, 2021 年 7 月.
[3] 宮城 優里, 大西 正輝, 機械学習モデルの品質保証・評価のための作業者情報比較可視化手法, 第 49 回可視化情報シンポジウム, OS12, 2021 年 9 月.
[4] 高瀬朝海, 星野貴行, 畳み込みニューラルネットワークの特徴マップへの Data Augmentation 適用, 第 23 回画像の認識・理解シンポジウム,2020 年 8 月.
[5] Tomoumi Takase, Dynamic batch size tuning based on stopping criterion for neural network training, Neurocomputing, Volume 429, pp.1-11, 2021 年 3 月.
[6] 中島 震, 敵対的なセマンティック・ノイズの実行時検知, 情報処理学会・ソフトウェア工学研究会, 2020 年 7 月.
[7] 中島 震, 統計的な部分オラクルによるテスティング方法, 日本ソフトウェア科学会大会, 2020 年 9 月.
[8] 中島 震, ニューラルネットワーク・ソフトウェアの頑健性検査, 情報処理学会・ソフトウェア工学研究会, 2020 年 11 月.
[9] Shin Nakajima (NII), Software Testing with Statistical Partial Oracles, 10th SOFL+MSVL, 2021 年 3 月.
[10] 中島 震, 訓練済み機械学習モデル歪みの定量指標, 電子情報通信学会・ソフトウェアサイエンス研究会, 2021 年 3 月.
[11] 大川 佳寛, 小林 健一, ラベルなし運用データに対するコンセプトドリフト適応技術に関するサーベイ, 第 35 回 人工知能学会全国大会, 2021 年 6 月.
[12] 大川 佳寛, 小林 健一, データ変化に対する教師なし適応技術に関する最新研究動向とその考察, 第 36 回 人工知能学会全国大会, 2022 年 6 月.
[13] 大川 佳寛, 金月 寛彰, 小林 健一, 運用時 AI 品質維持技術：コンセプトドリフト検知・適応から教師なしドメイン適応まで, 第 37 回 人工知能学会全国大会, 2023 年 6 月.

（2 章の参考文献）

[14] 原 聡, 私のブックマーク「機械学習における解釈性」, 人工知能, vol. 33, no. 3, pp. 366-369, 2018.
[15] Fred Hohman, Minsuk Kahng, Robert Pienta, Duen Horng Chau, Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers, IEEE Transactions on Visualization and Computer Graphics, vol. 25, no. 8, pp. 2674-2693, 2018.
[16] Bilal Alsallakh, Amin Jourabloo, Mao Ye, Xiaoming Liu, Liu Ren, Do Convolutional Neural

Networks Learn Class Hierarchy?, IEEE Transactions on Visualization and Computer Graphics, vol. 24, no. 1, pp. 152-162, 2018.

[17] Mengchen Liu, Jiaxin Shi, Kelei Cao, Jun Zhu, Shixia Liu, Analyzing the Training Processes of Deep Generative Models, IEEE Transactions on Visualization and Computer Graphics, vol.24, no.1, pp.77-87, 2018.

[18] Jorge Piazentin Ono, Sonia Castelo, Roque Lopez, Enrico Bertini, Juliana Freire, Claudio Silva, PipelineProfiler: A Visual Analytics Tool for the Exploration of AutoML Pipelines, IEEE Transactions on Visualization and Computer Graphics, vol.27, no.2, pp.390-400, 2021.

[19] Saleema Amershi, Maya Cakmak, W. Bradley Knox, Todd Kulesza, Power to the People: The Role of Humans in Interactive Machine Learning. AI Magazine, vol.35, no.4, pp.105-120, 2014.

[20] Heungseok Park, Jinwoong Kim, Minkyu Kim, Ji-Hoon Kim, Jaegul Choo, Jung-Woo Ha and Nako Sung, VISUALHYPERTUNER: VISUAL ANALYTICS FOR USER-DRIVEN HYPERPARAMTER TUNING OF DEEP NEURAL NETWORKS, 2019.

（4 章の参考文献）

[21] Cubuk, E. D., Dyer, E. S., Lopes, R. G., and Smullin, S., Tradeoffs in Data Augmentation: An Empirical Study. In Proceedings of International Conference on Learning Representations, 2021.

[22] Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q., RandAugment: Practical Automated Data Augmentation with a Reduced Search Space. In Neural Information Processing Systems, 33, 2020.

[23] Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D., Mixup: Beyond Empirical Risk Minimization. In International Conference on Learning Representations, 2018.

[24] Takase, T., Feature Combination Mixup: Novel Mixup Method Using Feature Combination for Neural Networks, Neural Computing and Applications, 2023.

[25] Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Lopez-Paz, D., and Bengio, Y., Manifold Mixup: Better Representations by Interpolating Hidden States. In International Conference on Machine Learning, pp. 6438–6447, PMLR, 2019.

[26] Kim, J-H., Choo, W., and Song, H. O., Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In International Conference on Machine Learning, 2020.

[27] Beckham, C., Honari, S., Verma, V., Lamb, A., Ghadiri, F., Hjelm, R. D., Bengio, Y., and Pal, C. On adversarial mixup resynthesis. In Neural Information Processing Systems, 2019.

[28] Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features. In Proceedings of the IEEE/CVF International Conference on Computer Vision, 6023–6032, 2019.

[29] Lopes, R. G., Yin, D., Poole, B., Gilmer, J., and Cubuk, E. D., Improving Robustness Without Sacrificing Accuracy with Patch Gaussian Augmentation. arXiv preprint arXiv: 1906.02611, 2019.

（5 章の参考文献）

[30] 中島 震，ソフトウェア工学から学ぶ機械学習の品質問題，丸善出版 2020.
[31] Pei, K., et al., DeepXplore: Automated Whitebox Testing of Deep Learning Systems, In Proc. 26th SOSP, 2017, pp.1-18.
[32] Nakajima, S., Distortion and Faults in Machine Learning Software, In Post-Proc. 9th SOFL+MSVL, 2020, pp.29-41.
[33] Ma, L., et al., DeepGauge: Multi-Granularity Testing Criteria for Deep Learning Systems, In Proc. ASE, 2018, pp.120-131.
[34] Tian, Y., et al., DeepTest: Automated Testing of Deep-Neural-Network-driven Autonomous Cars, In Proc. 40th ICSE, 2018, pp.303-314.
[35] Zhang, M., et al., DeepRoad: GAN-Based Metamorphic Testing and Input Validation Framework for Autonomous Driving Systems, In Proc. ASE, 2018, pp.132-142.
[36] Zhang, P, et al., CAGFuzz: Coverage-Guided Adversarial Generative Fussing Testing of Deep Learning Systems, arXiv:1911.07931, 2019.
[37] Harel-Canada, F., et al., Is Neuron Coverage a Meaningful Measure for Testing Deep Neural Networks? In ESEC/FSE, 2020, pp.851-862.
[38] Kim, J. et al., Guiding Deep Learning System Testing Using Surprise Adequacy, In Proc. 41st ICSE, 2019, pp.1039-1049.

（6 章の参考文献）

[39] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, The MIT Press 2016.
[40] Simon Haykin, *Neural Networks and Learning Machines (3ed.)*, Pearson India 2016.
[41] Shiqing Ma, Yingqi Liu, Wen-Chuan Lee, Xiangyu Zhang, and Anath Grama, MODE: Automated Neural Network Model Debugging via State Differential Analysis and Input Selection, In Proc. 26th ESE/FSE, pp.175-186, 2018.
[42] Shin Nakajima, Software Testing with Statistical Partial Oracles – Applications to Neural Network Software, In Proc. 10th SOFL+MSVL, pp.275-192, 2021.
[43] Shin Nakajima and Tsong Yueh Chen, Generating Biased Dataset for Metamorphic Testing of Machine Learning Programs, In Proc. 31st ICTSS, pp.56-64, 2019.
[44] Gregor Montavon, Genevieve B. Orr, and Klaus-Robert Muller (eds.), *Neural Networks: Tricks of the Trade (2ed.)*, Springer 2012.
[45] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov, Membership Inference Attacks Against Machine Learning Models, arXiv:1610.05820v2, 2017.
[46] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha, Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting, arXiv:1709.01604v5, 2018.
[47] Yunhui Long, Vincent Bindschaedler, Lei Wang, Diyue Bu, Xiaofeng Wang, Haixu Tang, Carl A. Gunter, and Kai Chen, Understanding Membership Inferences on Well-Generalized Learning Models, arXiv:1802.04489, 2018.
[48] Charu C. Aggarwal, *Outlier Analysis (2ed.)*, Springer 2017.

[49] Guy Katz, Clark Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer, Replux: An Efficeint SMT Solver for Verifying Deep Neural Networks, In Proc. 29th CAV, pp.97-117, 2017.
[50] Pang Wei Koh and Percy Liang, Understanding Black-box Predictions via Influence Functions, arXiv:1703.04730v3, 2020.
[51] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana, DeepXplore: Automated Whitebox Testing of Deep Learning Systems, In Proc. 26th SOSP, pp.1-18, 2017.
[52] Lei Ma, Felix Juefei-Xu, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Chunyang Chen, Ting Su, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang. DeepGauge: Multi-Granularity Testing Criteria for Deep Learning Systems, In Proc. 33rd ASE, pp.120-131, 2018.
[53] Yizhen Dong, Peixin Zhang, Jingyi Wang, Shuang Liu, Jun Sun, Jianye Hao, Xinyu Wang, Li Wang, Jin Song Dong, and Dai Ting. There is Limited Correlation between Coverage and Robustness for Deep Neural Networks. arXiv:1911.05904, 2019.
[54] Fabrice Harel-Canada, Lingxiao Wang, Muhammad Ali Gulzar, and Miryung Kim, In Proc. 28th ESEC/FSE, pp.851-862, 2020.
[55] Shin Nakajima, Distortion and Faults in Machine Learning Software, In Proc. 9th SOFL+MSVL, pp.29-41, 2019.
[56] Stephanie Abrecht, Maram Akila, Sujan Sai Gannamaneni, Konrad Groh, Christian Heinzemann, Sebastian Houben, and Matthjas Woehrle, Revisiting Neuron Coverage and Its Application to Test Generation, In Proc. SAFECOMP 2020 Workshop, pp.289-301, 2020.
[57] Anjiang Wei, Yinlin Deng, Chenyuan Yang, and Lingming Zhang, Free Lunch for Testing: Fuzzing Deep-Learning Libraries from Open Source, In Proc. 44th ICSE, pp.995-1007, 2022.
[58] Md Johirul Islam, Giang Nguyen, Rangeet Plan, and Hridesh Rajan, A Comprehensive Study on Deep Learning Bug Characteristics, In Proc. 27th ESEC/FSE, pp.510-520, 2019.
[59] Jiakun Cao, Meiziniu Li, Xiao Chen, Ming Wen, Yongqiang Tian, Bo Wu, and Shing-Chi Cheung, DeepFD: Automated Fault Diagnosis and Localization for Deep Learning Programs, In Proc. 44th ICSE, pp.573-585, 2022.
[60] Yanhui Li, Linghan Meng, Lin Chen, Li Yu, Di Wu, Yuming Zhou and Baowen Xu, Training Data Debugging for the Fairness of Machine Learning Software, In Proc. 44th ICSE, pp.2215-2227, 2022.

（7 章の参考文献）

[61] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus, Intriguing properties of neural networks, The International Conference on Learning Representations (ICLR 2014), pp.1-10, 2014. https://arxiv.org/abs/1312.6199
[62] Guy Katz, Clark Barrett, David Dill, Kyle Julian, and Mykel Kochenderfer, Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks, International Conference on Computer-Aided Verification (CAV), 2017. https://arxiv.org/abs/1702.01135
[63] Vincent Tjeng, Kai Xiao, and Russ Tedrake, Evaluating robustness of neural networks with mixed integer programming, International Conference on Learning Representations (ICLR),

2019. https://arxiv.org/abs/1711.07356

[64] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S. Dhillon, and Luca Daniel, Towards Fast Computation of Certified Robustness for ReLU Networks, International Conference on Machine Learning, PMLR 80, pp.5276-5285, 2018. https://arxiv.org/abs/1804.09699

[65] Akhilan Boopathy, Tsui-Wei Weng, Pin-Yu Chen, Sijia Liu, and Luca Daniel, CNN-Cert: An Efficient Framework for Certifying Robustness of Convolutional Neural Networks, The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI 2019), pp.3240-3247, 2019. https://arxiv.org/abs/1811.12395

[66] Tsui-Wei Weng, Pin-Yu Chen, Lam Nguyen, Mark Squillante, Akhilan Boopathy, Ivan Oseledets, and Luca Daniel, PROVEN: Verifying Robustness of Neural Networks with a Probabilistic Approach, International Conference on Machine Learning (ICML 2019), PMLR vol. 97, pp.6727-6736, 2019. http://proceedings.mlr.press/v97/weng19a.html

[67] Nicholas Carlini and David Wagner, Towards Evaluating the Robustness of Neural Networks, IEEE Symposium on Security and Privacy (SP), pp.39-57, 2017. https://arxiv.org/abs/1608.04644

[68] Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel, Evaluating the Robustness of Neural Networks: An Extreme Value Theory Approach, International Conference on Learning Representations (ICLR 2018), 2018. https://arxiv.org/abs/1801.10578

[69] Eric Wong and J. Zico Kolter, Provable defenses against adversarial examples via the convex outer adversarial polytope, International Conference on Machine Learning (ICML 2018), PMLR vol. 80, pp.5283-5292, 2018. https://arxiv.org/abs/1711.00851

[70] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana, Certified Robustness to Adversarial Examples with Differential Privacy, The IEEE Symposium on Security and Privacy (SP), 2019. https://arxiv.org/abs/1802.03471

[71] Jeremy M Cohen, Elan Rosenfeld, and J. Zico Kolter, Certified Adversarial Robustness via Randomized Smoothing, The 36th International Conference on Machine Learning (ICML 2019), 2019. https://arxiv.org/abs/1902.02918

[72] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, Towards Deep Learning Models Resistant to Adversarial Attacks, The Sixth International Conference on Learning Representations (ICLR 2018), 2018. https://arxiv.org/abs/1706.06083

（8 章の参考文献）

[73] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio, Fantastic Generalization Measures and Where to Find Them, International Conference on Learning Representations (ICLR 2020). https://arxiv.org/abs/1912.02178

[74] Gintare Karolina Dziugaite, Alexandre Drouin, Brady Neal, Nitarshan Rajkumar, Ethan Caballero, Linbo Wang, Ioannis Mitliagkas, and Daniel M. Roy, In search of robust measures

of generalization, NeurIPS 2020. arXiv:2010.11924. https://arxiv.org/abs/2010.11924
[75] Andreas Maurer, A Note on the PAC Bayesian Theorem, arXiv:cs/0411099, 2004. https://arxiv.org/abs/cs/0411099
[76] John Langford and Rich Caruana, (Not) Bounding the True Error, NIPS, 2001 https://papers.nips.cc/paper_files/paper/2001/hash/98c7242894844ecd6ec94af67ac8247d-Abstract.html
[77] María Pérez-Ortiz, Omar Rivasplata, John Shawe-Taylor, and Csaba Szepesvári, Tighter risk certificates for neural networks, Journal of Machine Learning Research, 2021. arXiv:2007.12911. https://arxiv.org/abs/2007.12911
[78] 磯部 祥尚, 最悪重み摂動付加ニューラル分類器の汎化誤差上界の見積法, 第 38 回 人工知能学会全国大会, 2024.
[79] Yoshinao Isobe, WP-GEB-Estimator -- WP-GEB: Weight-Perturbed Generalization Error Bounds. https://staff.aist.go.jp/y-isobe/wp-geb-estimator
[80] John Langford, Tutorial on Practical Prediction Theory for Classification, JMLR, vol.6, No.10, pp.273–306, 2005. https://jmlr.org/papers/v6/langford05a.html
[81] Oliver Catoni, PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning, Institute of Mathematical Statistics, Lecture Notes-Monograph Series, vol. 56, 2007. https://www.jstor.org/stable/i20461497
[82] Guillermo Valle-Pérez and Ard A. Louis, Generalization bounds for deep learning, arXiv:2012.04115v2, 2020. https://arxiv.org/abs/2012.04115
[83] Saurabh Garg, Sivaraman Balakrishnan, J. Zico Kolter, and Zachary C. Lipton, RATT: Leveraging Unlabeled Data to Guarantee Generalization, ICML 2021. arXiv:2105.00303. https://arxiv.org/abs/2105.00303
[84] Wenda Zhou, Victor Veitch, Morgane Austern, Ryan P. Adams, and Peter Orbanz, Non-vacuous Generalization Bounds at the ImageNet Scale: a PAC-Bayesian Compression Approach, ICLR 2019. https://arxiv.org/abs/1804.05862
[85] Yu-Lin Tsai, Chia-Yi Hsu, Chia-Mu Yu, and Pin-Yu Chen, Formalizing Generalization and Adversarial Robustness of Neural Networks to Weight Perturbations, NeurIPS 2021. https://proceedings.neurips.cc/paper/2021/hash/a3ab4ff8fa4deed2e3bae3a5077675f0-Abstract.html

（9 章の参考文献）

[86] X. Ma, Characterizing adversarial subspaces using Local Intrinsic Dimensionality, 2018.
[87] D. Meng , Magnet: a two-pronged defense against adversarial examples, in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2017.
[88] W. Xu, Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks, in Proceedings of the 2018 Network and Distributed Systems Security Symposium (NDSS), 2018.
[89] Shiqing Ma, NIC: Detecting Adversarial Samples with Neural Network Invariant Checking, Network and Distributed Systems Security Symposium (NDSS), NDSS 2019.

[90] 産業技術総合研究所, AI Bridging Cloud Infrastructure, https://abci.ai/ja/
[91] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE, vol. 86, no. 11, pp.2278–2324, 1998. [Online]. Available: http://yann.lecun.com/exdb/mnist/
[92] A. Krizhevsky and G. Hinton, Learning multiple layers of features from tiny images, 2009.
[93] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, Labeled faces in the wild: A database for studying face recognition in unconstrained environments, University of Massachusetts, Amherst, Tech. Rep. 07-49, October 2007.
[94] Nicolas Papernot, Ian Goodfellow, Ryan Sheatsley, Reuben Feinman, and Patrick McDaniel. cleverhans v1.0.0: an adversarial machine learning library. arXiv preprint arXiv:1610.00768, 2016.
[95] CleverHans, https://github.com/cleverhans-lab/cleverhans
[96] Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori, Density Ratio Estimation in Machine Learning, Cambridge University Press, 2012.

（10 章の参考文献）

[97] 大川 佳寛, 小林 健一, ラベルなし運用データに対するコンセプトドリフト適応技術に関するサーベイ, 第 35 回 人工知能学会全国大会, 2021 年 6 月.
[98] 大川 佳寛, 小林 健一, データ変化に対する教師なし適応技術に関する最新研究動向とその考察, 第 36 回 人工知能学会全国大会, 2022 年 6 月.
[99] 大川 佳寛, 金月 寛彰, 小林 健一, 運用時 AI 品質維持技術：コンセプトドリフト検知・適応から教師なしドメイン適応まで, 第 37 回 人工知能学会全国大会, 2023 年 6 月.