Vol.2021-SE-207 No.30
2021/3/2

訓練履歴を用いた欠陥局所化による ディープニューラルネットワーク修正技術の開発

徳井 翔梧[1,a)]　徳本 晋[1,b)]　菊池 慎司[1,c)]　石川 冬樹[2,d)]

概要：ディープニューラルネットワーク（DNN）を用いたシステムは従来のシステムとは異なりデータによってモデルが構築されるため，誤判定を検出した場合，通常ではデータを追加して再訓練することでDNN モデルを修正する．データによる修正は誤判定となるデータ（失敗データ）を成功判定に修正すると同時に，成功判定となるデータを誤判定に変えてしまう（退行）可能性がある．そこで，探索的に DNN モデルのパラメータ（重み）を変更することで，再訓練なしで誤判定を修正する方法が提案されている．従来手法では，失敗データに影響する重みを特定する欠陥局所化を行い，誤りが減少するような重みの値を粒子群最適化により探索した．しかし，従来手法の欠陥局所化は失敗データの影響のみで重みを特定するため，退行につながることが多い．本研究では，訓練履歴を用いて，DNN モデルの訓練過程で成功判定から誤判定に変化するデータ (退行データ）を検出し，退行データにのみ影響する重みを特定する欠陥局所化を行うことで，退行を防ぐ新たな DNN 修正技術を開発した．評価実験では，画像分類のデータセットを用いた 4 種類の訓練済みモデルに対しモデルの修正を適用し，モデルの修正による退行を従来手法の 1/10 以下に抑えつつ，モデルの精度を下げることなく 1%〜10%の失敗データを修正できたことを示す．

キーワード：ディープニューラルネットワーク，DNN 修正，欠陥限局，粒子群最適化

1. はじめに

近年，ディープラーニングは音声認識や自動翻訳，物体認識や画像分類などのシステムに用いられ，医療技術や自動運転技術などの領域で使われている．ディープニューラルネットワーク（DNN）を用いたシステムは従来のシステムとは異なりデータによってモデルが構築される．DNN を用いたシステムの運用中に異常なふるまいを検出した場合，通常ではデータを追加して再訓練することで DNN モデルを修正する．しかし，再訓練のために異常なふるまいを修正するための必要なデータを収集する必要があり，追加したデータによって必ずしも修正できるとは限らない．そこで，再訓練なしに DNN モデルを修正する手法として，探索的に DNN モデルのパラメータを変更し異常なふるまいを修正する手法が研究されている．

Sohn らは，画像分類問題に対して，探索的に DNN モデルのパラメータ（重み）を変更することで，誤判定となるデータ（失敗データ）を成功判定となるデータ（成功データ）に修正する手法 Arachne を提案した [6]．従来手法は，失敗データに影響する重みを特定する欠陥局所化を行い，誤りが減少するような重みの値を粒子群最適化 [2], [9] により探索した．

しかし，従来手法の欠陥局所化は失敗データの影響のみで重みを特定するため，従来手法は成功データを誤判定に変えてしまう（退行）可能性が高かった．退行が発生すると，修正前に正常に動作していた機能が修正後に誤判定に変わり，利用者は予期せぬ損失を被る恐れがある．特に医療技術や自動運転技術など高信頼性が求められる領域では，退行がビジネスや社会に与える損失は大きく，可能な限り退行を少なくすることが求められる．

本研究では，退行を抑止しつつ画像分類の DNN モデルを修正する技術として，訓練履歴を用いた欠陥局所化による DNN 修正技術を開発した．従来手法の欠陥局所化では，失敗データのみにより特定された重みが一部の成功データにも影響を与えるものであったため，失敗データを修正するための重み修正により一部の成功データが退行したと考えられる．提案手法の欠陥局所化では，DNN モデルの訓練過程で誤判定から成功判定に変化したデータ（改善デー

[1] 富士通研究所
Nakahara, Kawasaki, Kanagawa 211–8588, Japan
[2] 国立情報学研究所
Hitotsubashi, Chiyoda, Tokyo 101–8430, Japan
[a)] tokui.shogo@fujitsu.com
[b)] tokumoto.susumu@fujitsu.com
[c)] skikuchi@fujitsu.com
[d)] f-ishikawa@nii.ac.jp

タ）に影響せず，成功判定から誤判定に変化したデータ（退行データ）に影響し，訓練過程で値が大きく変化した重みを特定する．特定した重みに対して粒子群最適化を行い，成功データに影響する重みを変更せずに DNN モデルを修正する．

評価実験では，画像分類の 2 つのデータセット GTSRB[8], CIFAR-10[3] を訓練して作成した，訓練回数の異なる 4 種類の訓練済みモデルに対して DNN 修正を適用し，修正後モデルの精度と判定結果が変化したデータ数を調べた．モデルの修正による退行を従来手法の 1/10 以下に抑えつつ，モデルの精度を下げることなく 1%〜10%の失敗データを修正できた．

次節以降の本論文の構成を述べる．2 章では本研究の背景として DNN と従来手法 Arachne について述べる．3 章では提案手法について述べる．4 章では提案手法の評価実験と考察について述べる．5 章では本研究の関連研究について述べる．6 章では本研究のまとめと今後の課題を述べる．

2. 背景

本章では，本研究の対象である DNN と，探索的に DNN モデルの重みを修正する従来の DNN 修正手法 Arachne について述べる．

2.1 ディープニューラルネットワーク（DNN）

DNN とは，入力層，出力層，2 つ以上の隠れ層で構成されたニューラルネットワークである．特に，分類問題などを解く基本的な DNN として順伝播型ニューラルネットワークが知られている．順伝播型とは，入力層，隠れ層，出力層を順に情報が伝播し，入力に対する予測ラベルを出力するニューラルネットワークである．

DNN モデルの訓練方法について説明する．DNN モデルとは層の構成と隠れ層の各パラメータ値を指す．訓練では，訓練データを用いて隠れ層の各パラメータ値を調整する．入力層から与えられた入力 x に対して，隠れ層で 2 種類のパラメータ重み w とバイアス b を用いて $o = wx + b$ に変換し，活性化関数と呼ばれる微分可能な非線形関数 A を用いて $x' = A(o)$ を出力する．出力層では，隠れ層の出力の最大の要素のインデックスを求め，入力に対する予測結果を示す．入力に対する予測結果と正解ラベルの誤差を表す関数は損失関数 L と呼ばれ，二乗誤差などの関数が適用される．入力データに対する損失が小さいほど良いモデルとされ，DNN モデルの訓練では損失を小さくするために誤差逆伝播法 [5] を用いてパラメータを調整する．誤差逆伝播法は学習率 α を用いて，重みを $w = w - \alpha \frac{\partial L}{\partial w}$ に調整する最急降下法を行う．n epoch 訓練する場合は，誤差逆伝播法を n 回繰り返す．

画像認識分野では畳み込みニューラルネットワーク（CNN）が多く使われる．CNN とは，主に画像データを入力とし，画像処理を行う畳み込み層などが隠れ層に追加された DNN である．畳み込み層は前段のニューロンの一部の領域を畳み込んだ 1 つの特徴量として後段に伝播するのに対し，前段と後段のすべてのニューロンが結合されている層は全結合層と呼ばれる．基本的な CNN は，入力層，畳み込み層，全結合層，出力層の 4 つで構成される．

DNN を用いたシステムはデータによって訓練することでモデルが構築される．DNN を用いたシステムの運用中あるいはテスト中に誤判定を検出した場合，データを追加して再訓練することで DNN モデルを修正する．しかし，再訓練のために失敗データを修正するための必要なデータを収集する必要があり，追加したデータによって必ずしも修正できるとは限らない．そこで，再訓練なしに DNN モデルを修正する手法として，直接 DNN モデルのパラメータを変更し失敗データを修正する手法が研究されている．

2.2 従来手法 Arachne

探索的に DNN モデルのパラメータ（重み）を変更することで，再訓練なしに局所的な修正ができる DNN 修正技術 Arachne が提案されている [6]．Arachne は，訓練済みモデルに対して失敗データの判定結果に影響する重みを特定し，粒子群最適化 [2], [9] を用いて重みの値を調整し，失敗データを成功判定に修正する手法である．

Arachne は欠陥局所化と粒子群最適化の 2 つのステップで構成される．欠陥局所化では，修正対象の DNN モデルが誤判定したテストデータ（失敗データ）に対して，DNN モデルの各重みの損失関数の勾配と各層の出力値を重みの影響度として算出し，失敗データの判定結果に影響する重みを特定する．粒子群最適化では，欠陥局所化で特定した重みの値を fitness 関数に基づいて，誤判定が修正されるデータ数を増やす方向に最適化する．2.2.1 節と 2.2.2 節で欠陥局所化と粒子群最適化の詳細を述べる．

2.2.1 欠陥局所化

DNN モデルには数万を超える重みが存在するため，すべての重みを同時に粒子群最適化で調整するには非常にコストがかかる．最適化する重みを絞り込むために，Arachne は，最終層に連結する重みから失敗データの誤判定に影響する重みを特定する欠陥局所化を行う．Arachne は誤判定への影響度として，DNN モデルの訓練時に重みやバイアスの調整に用いられる損失関数の勾配と，ニューロン活性化を示す順伝播の出力値を利用する．

Arachne の欠陥局所化は，失敗データを入力として DNN モデルの最終層の損失関数の勾配と順伝播の出力値で重みを順位付け，上位の重みを粒子群最適化の対象とする．損失関数の勾配，すなわち損失関数 L を重み w で微分した値 $\frac{\partial L}{\partial w}$ は，最終層の順伝播の出力 o を用いて $\frac{\partial L}{\partial w} = \frac{\partial L}{\partial o}\frac{\partial o}{\partial w}$ として計算される．最終層の j 番目のニューロンの出力を

o_j とすると，その一段前の層の i 番目のニューロンとの重み $w_{i,j}$ に対する損失関数の勾配は $\frac{\partial L}{\partial w_{i,j}} = \frac{\partial L}{\partial o_j}\frac{\partial o_j}{\partial w_{i,j}}$ として計算される．重み $w_{i,j}$ に対する順伝播の出力値は，活性化関数で非線形に変換される前の値，すなわち一段前の層の出力 o_i に重み $w_{i,j}$ を乗算して $o_i \cdot w_{i,j}$ として計算する．

欠陥局所化では，損失関数の勾配に基づいて選択される重みの候補の個数を事前に N_g 個と決定する．損失関数の勾配に従って重みをソートし，上位 N_g 個の重みを候補とする．選択した N_g 個の重みに対して，損失関数の勾配と順伝播の出力値を 2 つの目的関数として多目的最適化を行い，パレート解の重みの集合を粒子群最適化の対象として抽出する．

2.2.2 粒子群最適化

Arachne は，欠陥局所化により特定した重みに対し，粒子群最適化（Particle Swarm Optimisation）を用いて DNN モデルの誤判定を修正する [2], [9]．粒子群最適化は連続な空間における最適化に効果的であることが知られており，実数の範囲で制限がない重み修正するのに適している．

Arachne は，欠陥局所化で特定した重みの集合をベクトル $\vec{x}$ として，粒子群最適化の粒子の位置を表す．現在の粒子ベクトル $\vec{x}_t$ は速度ベクトル $\vec{v}_{t+1}$ を用いて更新する（式 1）．現在の速度ベクトル $\vec{v}_t$ は，現在の粒子ベクトル $\vec{x}_t$ と，その粒子において過去に観測した中で最良な fitness 値を取る粒子ベクトル $\vec{p}_l$ と，群全体で最良な fitness 値を取る粒子ベクトル $\vec{p}_g$，さらに，一様乱数 $U(\phi)\,(0 \leq U(\phi) \leq \phi)$ を用いて更新する（式 2）．ϕ_1 と ϕ_2 は収縮係数と呼ばれ，それぞれ局所成分と大域成分に明示的な速度の境界を設定することなく群内の粒子の収束を制御する．式 3 の χ も収縮係数であり，ϕ_1 と ϕ_2 から計算される（式 3）．Arachne は ϕ_1 と ϕ_2 で同じ値を使用している．Arachne は，粒子ベクトルの初期値 $\vec{x}_0$ を重みの分布から定めた正規分布から抽出し，初期速度 $\vec{v}_0$ を $\vec{0}$ とした．

$$\vec{x}_{t+1} \leftarrow \vec{x}_t + \vec{v}_{t+1} \tag{1}$$

$$\vec{v}_{t+1} \leftarrow \chi(\vec{v}_t + U(\phi_1)(\vec{p}_l - \vec{x}_t) + U(\phi_2)(\vec{p}_g - \vec{x}_t)) \tag{2}$$

$$\chi \leftarrow \frac{2}{\phi - 2 + \sqrt{\phi^2 - 4\phi}},\ \text{where}\ \phi = \phi_1 = \phi_2 \tag{3}$$

式 2 の $\vec{p}_l$ と $\vec{p}_g$ は，式 4 に示した fitness 関数に基づいて過去に観測した粒子の中で最良の fitness 値となる粒子を用いる．I_{neg} は修正対象の失敗データの集合，I_{pos} はランダムに一定数選択した成功データの集合とする．N_{pached} は I_{neg} のうち成功判定に変化したデータの数であり，N_{intact} は I_{pos} のうち誤判定に変化したデータの数である．

$$fitness = \frac{N_{patched} + 1}{L(I_{neg}) + 1} + \frac{N_{intact} + 1}{L(I_{pos}) + 1} \tag{4}$$

3. 提案手法

従来手法 Arachne は欠陥局所化で失敗データに影響する重みを特定し，粒子群最適化を用いて重みを修正し，局所的に DNN モデルの重みを修正した．しかし，すべての失敗データに対して Arachne で DNN モデルの修正を行うと，一部の失敗データを修正できる一方で，成功データが誤判定に変化するデータも多くなる問題がある．原因の 1 つとして Arachne が失敗データのみを用いて欠陥局所化したことが考えられる．欠陥局所化によって特定された重みが，一部の成功データにも影響を与えるものであったため，失敗データを修正するための重み修正により成功データの一部が誤判定に変化したと考えられる．我々は，訓練過程で成功判定になったことがある失敗データは修正が容易であると考えた．また，成功データの判定結果に影響する重みを変化させなければ，DNN 修正時に退行の発生を減らすことができると考えた．

そこで，本研究では，訓練履歴を用いた欠陥局所化による DNN 修正を提案する本手法の欠陥局所化では，訓練過程で誤判定から成功判定に変化したデータ（改善データ）と成功判定から誤判定に変化したデータ（退行データ）を検出し，改善データに影響せず退行データに影響し，訓練過程で値が大きく変化した重みを特定する．特定した重みに対して，粒子群最適化を用いて DNN モデルの重みを修正する．本手法の訓練履歴を用いた欠陥局所化は，以下の 4 つのステップで実行する．

STEP I　訓練履歴を用いたデータ分類
STEP II　重み差分と影響度の計算
STEP III　集合演算による欠陥局所化
STEP IV　粒子群最適化

STEP I では，訓練履歴を用いてテストデータを退行データと改善データを取得する．STEP II では，退行データ／改善データが重みに与える影響度と重みの差分を計算する．STEP III では，退行データ，改善データ，重み差分に影響する重みの集合をそれぞれ特定し，集合演算で重みを絞り込む．STEP IV では，粒子群最適化を用いて DNN モデルの重みを修正する．以降，各ステップの詳細を述べる．

3.1 STEP I 訓練履歴を用いたデータ分類

STEP I では訓練履歴から退行データと改善データを検出する．n epoch 訓練後モデルを提案手法の修正対象とするとき，訓練履歴から，$n-k$ epoch 訓練後モデル M_{n-k} と n epoch 訓練後モデル M_n の重みを取得する．それぞれのモデルに対してデータセットの判定結果を調べ，M_{n-k} で誤判定だが M_n で成功判定に変化したデータを退行データ，M_{n-k} で成功判定だが M_n で誤判定に変わるデータを成功データとして分類する．本研究の評価実験では $k = 1$

としてデータセットを分類した．

3.2 STEP Ⅱ 重み差分と影響度の計算

STEP Ⅱでは，修正すべき重みを特定するための重みの5つの影響度，重み差分 w_{diff}，退行データに対する後方影響度 $back_{deg}$ および前方影響度 fwd_{deg}，改善データに対する後方影響度 $back_{imp}$ および前方影響度 fwd_{imp} を計算する．本研究では，後方影響度は損失関数の勾配を指し，前方影響度は順伝播の出力値を指す．

重み差分 w_{diff} が大きい重みほど，データの判定結果を変化させる原因となっている可能性が高いと考えられる M_n の重み配列 w_n と M_{n-k} の重み配列 w_{n-k} を取得し，その差分 $w_{diff} = w_n - w_{n-k}$ を計算する．

後方影響度 $back$ は，特定の重み $w_{i,j}$ と修正対象の層の j 番目のニューロンの出力 o_j に対して，$back = \frac{\partial L}{\partial w_{i,j}} = \frac{\partial L}{\partial o_j}\frac{\partial o_j}{\partial w_{i,j}}$ として計算する．損失 L とニューロンの出力 o_j は重み w と入力データに依存する．退行データを入力として得られる後方影響度を $back_{deg}$，改善データを入力として得られる後方影響度を $back_{imp}$ とする．

前方影響度 fwd は，活性化関数で非線形に変換される前の値，すなわち一段前の層の出力 o_i に重み $w_{i,j}$ を乗算して $fwd = o_i \cdot w_{i,j}$ として計算する．fwd は重み w と入力データに依存する．退行データを入力として得られる前方影響度を fwd_{deg}，改善データを入力として得られる前方影響度を fwd_{imp} とする．

提案手法は最終層のみでなく，DNN モデルのすべての全結合層の重みを修正対象とする．DNN モデルの各層に対して後方影響度 $back$ と前方影響度 fwd を計算する．本研究では初期段階の仮説検証のため，全結合層のみを対象としたが，今後は畳み込み層の修正も検討する予定である．

3.3 STEP Ⅲ 集合演算による欠陥局所化

STEP Ⅲでは，STEP Ⅱで取得した5つの影響度 w_{diff}, $back_{deg}$, fwd_{deg}, $back_{imp}$, $back_{imp}$ それぞれに対して重みをソートし，それぞれ上位 N_g 個の重みの集合 W_{diff}, B_{deg}, F_{deg}, B_{imp}, F_{imp} を取得し，式5の集合演算に従って特定される重みを DNN モデル修正の対象とする．

$$(B_{deg} \cap F_{deg} \cap W_{diff}) \backslash (B_{imp} \cap F_{imp}) \tag{5}$$

式5によって，退行データに影響を及ぼし，重み差分が大きい重みから，改善データに影響を及ぼす重みを除いた集合を得ることができる．退行データ／改善データに影響を及ぼす重みは後方影響度と前方影響度の両方で大きい値を持つと考えた．したがって，退行データに影響を及ぼす重みを $B_{deg} \cap F_{deg}$ とし，改善データに影響を及ぼす重みを $B_{imp} \cap F_{imp}$ とする．式5による重みの集合演算によって，退行を抑えつつ DNN モデルを修正する欠陥局所化となる．

3.4 STEP Ⅳ 粒子群最適化

STEP Ⅳでは，STEP Ⅲで特定した重みを，粒子群最適化を用いて修正する．2.2.2 節に示した Arachne の粒子群最適化を利用するが，fitness 関数と fitness 関数の計算に用いる成功データのサンプルの選択方法を変更した．

fitness 関数には以下の式を用いた．Arachne が用いた fitness 関数は，テストデータの絶対数に依存するため，本手法では安定した結果を得るために比率に変更した．失敗データの修正率から失敗データの損失の増加率を除算した値と成功データ退行率から成功データの損失の増加率を除算した値の和を fitness 関数とした．L_{before} を修正対象モデルの損失関数，L_{after} を修正後モデルの損失関数とする．

$$fitness = \frac{N_{patched}}{|I_{neg}|} / \frac{L_{after}(I_{neg})}{L_{before}(I_{neg})} + \frac{N_{intact}}{|I_{pos}|} / \frac{L_{after}(I_{pos})}{L_{before}(I_{pos})}$$

Arachne は成功データのサンプルをランダムに選択した．本手法では修正対象モデルの成功データに対する予測値と正解値の二乗誤差に従って成功データをソートし，誤差が大きい順に一定のサンプル数を取得した．誤差が大きいほど，分類境界に近いデータである，すなわち重みの変化に敏感なデータであるため，そのデータの退行を防ぐことで他の成功データの退行も防ぎやすいと考えた．

4. 評価実験

本章では，提案手法の評価実験について述べる．本実験では，画像分類のデータセットを用いた4種類の訓練モデルに対し，提案手法と従来手法によるモデルの修正を行う．実験結果として，モデルの修正による退行を従来手法の 1/10 以下に抑えつつ，モデルの精度を下げることなく 1%〜10%の失敗データを修正できた．

本実験では，従来手法との比較，提案手法の設計の妥当性，および汎化性能を評価する．画像分類のデータセットである GTSRB[8] と CIFAR10[3] を用いて訓練し，データセットと訓練回数が異なる4つの訓練済み CNN モデルを用意する．4つの訓練済みモデルに対して，テストデータにおける失敗データを対象とする DNN 修正を行い，テストデータを用いて修正後のモデルを評価する．試行を 10 回実施し，計測結果の平均をその手法の評価とする．

本実験では DNN 修正の性能を測る指標として，Accuracy, Repair Rate, Break Rate を以下の式で計測する．

$$Accuracy(ACC) = \frac{\#I_{pos}}{\#I_{all}} \times 100$$

$$RepairRate(RR) = \frac{\#I_{imp}}{\#I_{neg}} \times 100$$

$$BreakRate(BR) = \frac{\#I_{deg}}{\#I_{pos}} \times 100$$

表 1 訓練済み CNN モデル

Model Name	Group	epoch	#weights	ACC	#pos	#neg	#deg	#imp
Model_gtsrb(5)	GTSRB	5	570,720	96.120	12,140	490	151	241
Model_gtsrb(10)	GTSRB	10	570,720	96.572	12,197	433	173	221
Model_cifar(5)	CIFAR10	5	553,824	67.430	6,743	3,257	882	1,057
Model_cifar(10)	CIFAR10	10	553,824	74.290	7,429	2,571	806	653

表 2 汎化性能評価のために 2 分割したデータセット

			Group1					Group2				
Model Name	Group	epoch	ACC	#pos	#neg	#deg	#imp	ACC	#pos	#neg	#deg	#imp
Model_gtsrb(5)	GTSRB	5	96.136	6,071	244	65	127	96.105	6,069	246	86	114
Model_gtsrb(10)	GTSRB	10	96.659	6,104	211	82	110	96.485	6,093	222	91	111
Model_cifar(5)	CIFAR10	5	66.940	3,347	1,653	458	493	67.920	3,396	1,604	424	564
Model_cifar(10)	CIFAR10	10	74.140	3,707	1,293	400	301	74.440	3,722	1,278	406	352

ACC は全テストデータ中の成功データの割合であり，モデルの性能を示す．RR は修正前のモデルの失敗データのうち成功判定に変化したデータの割合を指す．BR は修正前のモデルの成功データのうち誤判定に変化したデータの割合を指す．本実験では，以上の 3 つの指標で評価を行う．

Arachne の実装は公開されていないため論文に基づいて実装した．本手法および Arachne は Python 3.6.9 で実装し，DNN の訓練モデルは Tensorflow 1.16.0 で実装した．本実験は，Intel Core Xeon Gold 5122 CPU，64GB RAM，NVIDIA Quadro P5000 GPU の環境で実験を行った．

4.1 実験対象

訓練済みの各モデルについて表 1 に示す．本実験の修正対象とする DNN モデルは，6 層の畳み込み層と 2 層の全結合層で構成された CNN モデルを用いる．CNN モデルに対して，画像分類データセットである GTSRB[8]，CIFAR10[3] をぞれぞれ 5 回と 10 回訓練し，データセットと訓練回数が異なる 4 つの訓練済み CNN モデルを作成する．

4.2 研究課題

本実験は，以下の 5 つの観点で実験を行う．

RQ1 従来手法（Arachne）より ACC，RR，BR が優れているか？

RQ2 前方影響度と後方影響度と重み差分は効果的か？

RQ3 退行データと改善データの利用は効果的か？

RQ4 最後段以外の層の重み修正は効果的か？

RQ5 汎化性能が高いか？

RQ1 では，従来手法 Arachne と提案手法をそれぞれ適用し性能を比較する．計測結果 10 回分の ACC，RR，BR の分布を箱ひげ図と散布図で可視化し，それぞれの手法の性能の分布を調査する．

RQ2，RQ3，RQ4 では，提案手法の設計の妥当性を調査する．欠陥局所化の一部を変更し，それぞれ 10 回ずつ実験し，提案手法の結果と比較する．RQ2 では前方影響度と後方影響度がどちらが欠陥局所化に良い効果をもたらしているかを調べる．RQ3 では欠陥局所化の集合演算に利用するデータである，退行データ，改善データ，重み差分，の 3 つの組合せを変更し，どのデータが欠陥局所化に良い効果をもたらしているかを調べる．RQ4 ではどの層を修正することがモデル修正に良い効果があるかを調べる．

RQ5 では，提案手法の汎化性能を調べるために，2 分割交差検証を行う．GTSRB と CIFAR10 のそれぞれのテストデータを 2 分割したものを Group1，Group2 とする．それぞれの Group1，Group2 に対して，各訓練済みモデルで判定した結果を表 2 に示す．

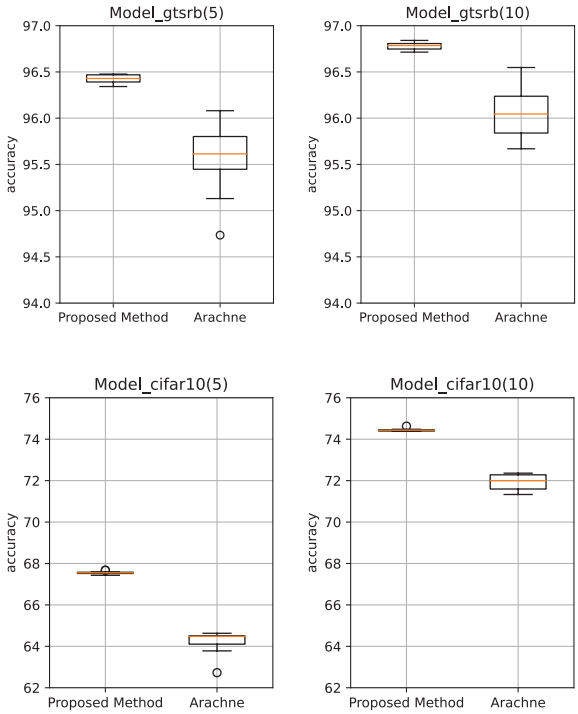

図 1 従来手法と提案手法の比較（ACC）

表 3 RQ1. 従来手法（Arachne）との比較

	Model_gtsrb(5)			Model_gtsrb(10)			Model_cifar(5)			Model_cifar(10)		
比較対象	ACC	RR	BR	ACC	RR	BR	ACC	RR	BR	ACC	RR	BR
提案手法	96.425	9.776	0.077	96.779	7.691	0.058	67.556	0.869	0.233	74.447	1.225	0.213
Arachne	95.559	10.225	0.997	96.070	11.778	0.938	64.210	8.299	8.784	71.924	12.653	7.564

表 4 RQ2. 欠陥局所化に利用する影響度の比較

	Model_gtsrb(5)			Model_gtsrb(10)			Model_cifar(5)			Model_cifar(10)		
比較対象	ACC	RR	BR	ACC	RR	BR	ACC	RR	BR	ACC	RR	BR
forward + back	96.425	9.776	0.077	96.779	7.691	0.058	67.556	0.869	0.233	74.447	1.225	0.213
forward	96.371	10.551	0.165	96.774	8.845	0.104	67.558	0.470	0.037	74.354	0.568	0.110
back	96.388	13.000	0.246	96.816	10.092	0.106	67.642	1.756	0.534	74.618	2.707	0.495

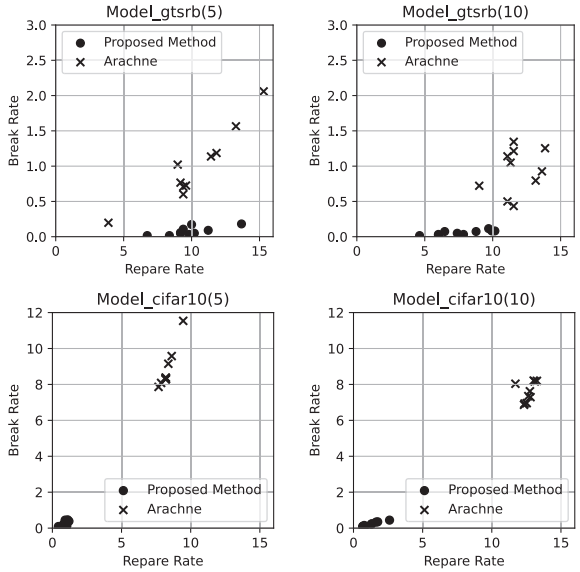

図 2 従来手法と提案手法の比較（RR,BR）

4.3 実験結果

本節では，RQ に対する実験結果を示す．

RQ1. 従来手法（Arachne）との比較

従来手法 Arachne と提案手法の実験結果を表 3，図 1，図 2 に示す．表 3 は，各実験結果に対する ACC，BR，RR の平均を示している．図 1 は，実験対象のモデルごとに実験結果の ACC の分布を表している．図 2 は，実験対象のモデルごとの RR と BR の分布を散布図として表している．図 2 の RR と BR の分布から，RR と BR の間には手法ごとに正の相関が見られた，BR が低く RR が高いほど修正後のモデルの ACC が高くなるため，グラフの右下に分布するほど良い DNN 修正手法であると考えることができる．

表 3 と図 1 から，すべての訓練済みモデルに対する実験結果について Arachne より提案手法の方が ACC は高く，修正前モデルと比べても ACC は減少しなかった．また表 3 と図 2 から，提案手法は Arachne と比べて BR を 1/10 以下に抑えることが確認できた．しかし，提案手法の RR についてはデータセットによって結果が偏った．GTSRB で訓練したモデルである Model_gtsrb(5), Model_gtsrb(10) では Arachne と同程度の RR を得られたが，CIFAR10 で訓練したモデルである Model_cifar(5), Model_cifar(10) では Arachne より RR が約 1/10 に下がった．Arachne よりも提案手法の方が RR が約 1/10 に下がっているのに ACC が高い理由は，BR を Arachne の 1/10 以下に抑えているからである．表 1 から成功データは失敗データより多いため，BR を抑えたことが ACC を高くすることに影響している．

RQ1 回答

すべてのモデルに対して, 提案手法は従来手法 Arachne に対して BR を 1/10 以下に抑えつつ，精度を下げることなく 1%〜10%の失敗データを修正できた．

RQ2. 欠陥局所化に利用する影響度の比較

提案手法では，重みに対して前方影響度と後方影響度の両方を用いて欠陥局所化をすることで，修正対象とする重みを重要な箇所に絞っている．RQ2 では，欠陥局所化に利用する影響度を，前方影響度のみに変更した場合と後方影響度のみに変更した場合でそれぞれ実験し，提案手法と比較する．

実験結果を表 4 に示した．Model_gtsrb(5) では提案手法が最も ACC が高かったが，他の 3 つのモデルでは後方影響度のみの場合が RR と ACC が高かった．しかし，後方影響度のみの場合は他の 2 つの場合と比べて，どのモデルに対しても BR が最も高かった．

RQ2 回答

提案手法が最も BR を抑えつつ ACC が高い場合があったが，提案手法が最も良いと言い切ることはできない．逆に，前方影響度や後方影響度どちらかのみの場合が良いと言い切ることはできない．

RQ3. 欠陥局所化に利用するデータの比較

提案手法では，退行データの影響度と重みの差分と改善データの影響度を利用して欠陥局所化する．RQ3 では，重み差分を考慮しない場合，改善データの影響度を考慮しない場合でそれぞれ実験し，提案手法と比較する．

実験結果を表 5 に示した．提案手法は，Model_gtsrb(10)

表 5　RQ3. 欠陥局所化に利用するデータの比較

	Model_gtsrb(5)			Model_gtsrb(10)			Model_cifar(5)			Model_cifar(10)		
比較対象	ACC	RR	BR	ACC	RR	BR	ACC	RR	BR	ACC	RR	BR
deg & diff – imp	96.425	9.776	0.077	96.779	7.691	0.058	67.556	0.869	0.233	74.447	1.225	0.213
deg – imp	96.379	16.571	0.400	96.923	13.718	0.123	67.442	3.433	1.640	74.489	3.201	0.840
deg & diff	96.438	11.816	0.147	96.805	8.291	0.052	67.616	1.142	0.276	74.492	1.715	0.322

表 6　RQ4. 欠陥の修正対象の層の比較

	Model_gtsrb(5)			Model_gtsrb(10)			Model_cifar(5)			Model_cifar(10)		
比較対象	ACC	RR	BR	ACC	RR	BR	ACC	RR	BR	ACC	RR	BR
最後段＋前段	96.425	9.776	0.077	96.779	7.691	0.058	67.556	0.869	0.233	74.447	1.225	0.213
最後段	96.413	10.245	0.110	96.755	6.236	0.032	67.533	0.669	0.171	74.433	2.205	0.571
前段	96.367	8.306	0.079	96.712	4.804	0.025	67.514	0.384	0.061	74.412	0.720	0.085

以外のモデルで BR を最も低く抑えていた．しかし，Model_gtsrb(10) では重み差分を考慮しない場合が最も ACC が高く，他の 3 つのモデルでは改善データを考慮しない場合が ACC が高かった．RR に着目すると，どのモデルに対しても重み差分を考慮しない場合が最も高くなっていた．

RQ3 回答

提案手法が最も BR を抑えていた．重み差分，改善データの影響度はいずれも BR を抑えることに貢献しているが，同時に RR が下がってしまうため，結果として提案手法の ACC が他の場合の ACC より低くなる．提案手法が良いと言い切ることはできない．

RQ4. 欠陥の修正対象の層の比較

提案手法では，すべての全結合層の重みを候補として欠陥局所化する．本実験の対象とした CNN モデルには 2 層の全結合層がある．RQ4 では，特定の層のみ欠陥局所化する場合を実験し，提案手法と比較する．

実験結果を表 6 に示した．提案手法は，すべてのモデルに対して ACC が最も高かった．しかし，BR に着目すると，Model_gtsrb(5) 以外のモデルに対しては最後段の前段の方が最も低く抑えていた．最後段で欠陥局所化した場合と，最後段の前段で欠陥局所化した場合を比較すると，RR,BR いずれも最後段の場合の方が値が高くなっている．したがって，最後段の重みを修正する方が判定結果に影響が強いと考えられる．

RQ4 回答

提案手法がすべてのモデルに対して ACC が最も高かった．最後段の前段のみ欠陥局所化する場合に BR が最も抑えられてることがあるが，RR が比較的高くないため ACC も提案手法より低くなっていた．

RQ5. 汎化性能への影響

提案手法の汎化性能を調査するために 2 分割交差検証を行う．DNN 修正に利用するデータセット GTSRB, CIFAR10 をそれぞれ Group1,Group2 に分割する．一方のデータセットを用いて DNN 修正して他方のデータセットを用いて評価を行う．また，評価の基準として，修正と評価を分割した同じデータセットに対して実験を行う．

実験結果を表 7 に示した．交差検証した結果から，全体的に BR が 2%以下に抑えられており，修正に利用していないデータに対して提案手法が効果があると考えられる．加えて，RR はすべて 1%以上得られており，一部の失敗データを修正できている．ACC が修正前のモデルより高くなったモデルも見られるため，提案手法は汎化性能が高いと言える．

RQ5 回答

交差検証では，すべての場合で BR は 2%以下に抑えつつ，RR は 1%以上得られていた．さらに，いくつかのモデルでは修正前のモデルより ACC が向上しているため，提案手法は汎化性能が高いと言える．

4.4　考察

Arachne は局所的な修正ができること，すなわち特定の失敗データや特定のラベルの誤判定を修正できることが Sohn らの主張である．しかし評価実験では，我々の提案手法に合わせてすべての失敗データを入力として実験を実施し，結果として Arachne の修正後のモデルの精度が下がっている．特定ラベルに対する修正できるか，Arachne と比較する必要がある．

RQ2, RQ3, RQ4 の結果から，実験対象のデータセットによっては，提案手法とした影響度の組合せより，一部の影響度を除外した組合せの方が ACC を高くする場合が見られた．本実験では 2 つのデータセットのみを実験対象としており，現状ではどの影響度の組み合わせが良いか判断できないため，さらに実験対象を増やした評価が必要だと考えられる．

RQ5 に関する実験結果から，修正に利用していないデータセットに対して，BR を抑えつつ一部の失敗データを修

表 7　RQ5. 汎化性能への影響

		Model_gtsrb(5)			Model_gtsrb(10)			Model_cifar(5)			Model_cifar(10)		
修正対象	評価対象	ACC	RR	BR	ACC	RR	BR	ACC	RR	BR	ACC	RR	BR
Group1	Group2	96.356	5.438	0.126	96.549	1.374	0.092	67.894	1.297	0.569	73.864	2.592	1.792
Group2	Group1	96.166	6.034	0.214	96.708	2.713	0.137	67.460	2.711	0.865	74.290	0.873	0.314
Group1	Group1	96.358	8.320	0.104	96.751	4.313	0.111	67.184	1.887	0.568	73.670	4.060	2.050
Group2	Group2	96.314	9.472	0.166	96.793	11.802	0.110	68.366	5.673	2.023	74.516	1.346	0.360

正できたことを確認した．したがって，提案手法の汎化性能は高いと考えられる．しかし，本実験では，DNN の層の構成として 6 層の畳み込み層と 2 層の全結合層の一種類の構成のみを対象としている．提案手法が本実験で対象とした層の構成に依存していないことを示すため，異なる層の構成でのさらなる評価が必要である．

5. 関連研究

本章では，本研究の関連研究として，再訓練に基づく DNN 修正手法と再訓練なしで DNN 修正する手法の例を示す．

再訓練に基づく DNN 修正手法として直近で提案されている 3 本の論文を示す．Ren らは失敗テストデータを混ぜて訓練して修正する手法 FSGMix を提案した [4]．Gao らは遺伝的アルゴリズムで損失の大きいデータを選択しながら訓練することでモデルの堅牢性を向上させる手法 SENSEI を提案した [1]．Srivastava らは損失関数に互換性に関する罰則項を追加したモデル学習を提案した [7]．

一方で，提案手法や従来手法の Arachne と同様に再訓練なしで DNN 修正を行う手法として，Zhang らは Apricot を提案した [10]．Apricot は，分割した訓練データで訓練したモデルの重みを参考に修正する手法である．

6. まとめと今後の課題

本研究では，訓練履歴を用いて退行データと改善データを検出し，粒子群最適化を用いた DNN 修正において退行を抑えるための欠陥局所化手法を提案した．評価実験では，従来手法の Arachne と比べて BR を 1/10 以下に抑えつつ，モデルの精度を下げることなく 1%〜10%の失敗データを修正できたことを示した．さらに，二分割交差検証を実施し，本手法が退行を抑えることに関して汎化性能が高いことを確認した．

今後の課題として，以下が挙げられる．

- 再訓練による DNN 修正との比較
- 実験対象の CNN モデルに VGG16 などよく利用されるモデルの追加
- 欠陥局所化の対象とする範囲を拡充するために本手法を畳み込み層に対する実装
- 特定ラベルなど局所的な修正に対する評価
- 提案手法による粒子群最適化の設計妥当性の評価
- 画像分類問題以外を扱う DNN モデルに対する修正方法の検討

謝辞　本研究の一部は JST 未来社会創造事業 JPMJMI20B8 の支援を受けたものです．

参考文献

[1] Gao, X., Saha, R. K., Prasad, M. R. and Roychoudhury, A.: Fuzz testing based data augmentation to improve robustness of deep neural networks, *Proc. of ICSE'20*, IEEE, pp. 1147–1158 (2020).
[2] Kennedy, J. and Eberhart, R.: Particle swarm optimization, *Proc. of ICNN'95*, Vol. 4, IEEE, pp. 1942–1948 (1995).
[3] Krizhevsky, A.: Learning multiple layers of features from tiny images, Technical report (2009).
[4] Ren, X., Yu, B., Qi, H., Juefei-Xu, F., Li, Z., Xue, W., Ma, L. and Zhao, J.: Few-Shot Guided Mix for DNN Repairing, *Proc. of ICSME'20*, IEEE, pp. 717–721 (2020).
[5] Rumelhart, D. E., Hinton, G. E. and Williams, R. J.: Learning representations by back-propagating errors, *nature*, Vol. 323, No. 6088, pp. 533–536 (1986).
[6] Sohn, J., Kang, S. and Yoo, S.: Search Based Repair of Deep Neural Networks, *arXiv preprint arXiv:1912.12463*, (online), available from ⟨http://arxiv.org/abs/1912.12463⟩ (2019).
[7] Srivastava, M., Nushi, B., Kamar, E., Shah, S. and Horvitz, E.: An Empirical Analysis of Backward Compatibility in Machine Learning Systems, *Proc. of KDD'20*, pp. 3272–3280 (2020).
[8] Stallkamp, J., Schlipsing, M., Salmen, J. and Igel, C.: Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition, *Neural networks*, Vol. 32, pp. 323–332 (2012).
[9] Windisch, A., Wappler, S. and Wegener, J.: Applying Particle Swarm Optimization to Software Testing, *Proc. of GECCO'07*, Association for Computing Machinery, pp. 1121–1128 (online), available from ⟨https://doi.org/10.1145/1276958.1277178⟩ (2007).
[10] Zhang, H. and Chan, W.: Apricot: A weight-adaptation approach to fixing deep learning models, *Proc. of ASE'19*, IEEE, pp. 376–387 (2019).