遺伝的アルゴリズムを利用した訓練済みモデルの再最適化

唐泊文

令和 7 年 2 月

九州大学大学院システム情報科学府

情報理工学専攻

Contents

1 概要

機械学習（Machine Learning）[1] は、コンピュータがデータをもとに自動で学習し、問題を解決する技術で、人工知能（Artificial Intelligence、AI）の中核となる分野のひとつである. 特に、深層学習（Deep Learning）の発展によって、画像認識、自然言語処理、強化学習などさまざまな分野で高い成果を上げている.

機械学習の中心的な要素は「モデル」であり、モデルはデータから規則やパターンを学習し、予測や分類といったタスクの実行に用いられる. 特に、ニューラルネットワーク (Neural Network)[2] は機械学習モデルの一種として、データから複雑なパターンを学習できる点で注目されている. ニューラルネットワークは脳神経の仕組みを模倣した構造を持ち、層を深くすることで（深層学習）、画像やテキストといった高次元データの特徴を捉えることを可能とする. このようなニューラルネットワークの性能が高いほど、タスクを効率的かつ正確に処理できるため、ニューラルネットワークの構造設計やパラメータの最適化は、機械学習研究における重要なテーマとなっている.

近年、機械学習や深層学習が急速に発展する中で、ニューラルネットワークの性能最適化は常に研究の焦点であり、難点でもある. 従来の最適化手法である勾配降下法やその派生アルゴリズムは、主に訓練プロセスに重点を置き、より適切な訓練方法を模索してきた. この手法により、多くのニューラルネットワークが構築され、さまざまな分野で実用化が進んでいる. しかし、異なるアルゴリズムで訓練されたニューラルネットワークは、その構造やパラメータの規模において大きな差異を持ち、最適化の難しさが残されている.

遺伝的アルゴリズム（Genetic Algorithm, GA）[3] は、自然選択と遺伝メカニズムに基づいた最適化アルゴリズムであり、生物進化過程における選択、交叉、突然変異などの操作を模倣することで、複雑な高次元空間での全体的な探索を可能にする. また、目標関数の勾配情報に依存しないため、非凸関数、離散的なパラメータ、明示的な目標関数の導関数が存在しない場合においても、優れた特性を発揮する. GA は、モデルの性能最適化における研究対象として注目されている. 特に、勾配降下法などの従来の最適化手法では克服しにくい非凸問題や高次元空間における探索において、GA は柔軟かつ強力な特性を発揮する.GA を用いた研究では、生物進化の過程を模倣し、選択、交叉、突然変異といった操作を通じて、最適解を探索する能力を最大限に活用することが目的とされる. また、勾配情報に依存しないという特性により、従来の手法では扱いにくい離散的なパラメータや複雑なニューラルネットワーク構造の最適化にも適用可能である. このため、GA はニューラルネットワークをはじめとする多様なモデルに対して、新たな可能性を提供する研究手法として広く議論されている.

本研究は、GA を使用して、訓練済みニューラルネットワークのパラメータを最適化する方法を探索することを目的としている. 深層学習ニューラルネットワークのパラメータ最適化に GA を適用し、交叉と突然変異操作を用いて、特定のアルゴリズムで訓練されたニューラルネットワークのパラメータにわずかな調整を加えることで、アルゴリズムの性能をさらに向上させる.

提案する方法の実現可能性を検証するため、本研究では強化学習とコンピュータビジョンという異なる分野における代表的なモデルを用いて実験を行った. 具体的には、強化学習分野では深層モンテカルロ法（DMC)[4] と深層 Q ネットワーク（DQN）[5]、コンピュータビジョン分野では深層畳み込みニューラルネットワーク（ResNet50）[6] を対象とし、GA がこれらのモデルの性能向上に与える影響を評価した.

本研究では、遺伝的アルゴリズム（GA）が異なるタスク領域における深層学習モデルの最適化に与える影響を実験的に評価した. 実験結果から、強化学習タスク（DMC および DQN）においては、GA によりモデルの性能が一定程度向上し、対戦環境でのパフォーマンスが改善されることが確認された. 一方で、コンピュータビジョンタスク（ResNet50）では、GA の最適化効果は顕著に見られず、これは適応度計算環境が固定されていることや、画像認識タスクにおいて勾配最適化手法がより優位であることが影響している可能性がある. さらに、初期性能が異なるモデルを比較分析した結果、GA は低性能モデルの最適化において大きな改善をもたらすが、すでに十分に訓練された高性能モデルに対しては最適化の余地が限られることが明らかになった. 本研究の結果は、GA の深層学習モデル最適化における適用可能性に関する新たな知見を提供するとともに、タスクの種類、適応値の計算方法、および初期モデルの性能が GA の最適化効果に影響を与えることを示している.

2 背景

2.1 機械学習

2.1.1 機械学習の紹介

機械学習は、データ駆動のアプローチを用いて問題を解決する技術であり、従来のエンジニアリング設計手法とは異なり、明確な数学モデルを手動で構築したり、専門家の知識に依存したりする必要がない. 代わりに、データから規則を学習することでタスクを遂行する. 従来のエンジニアリング手法では、例えば化学反応の設計、音声翻訳アルゴリズムの開発、画像圧縮アルゴリズムの設計など、大量の領域分析や専門家の関与が必要となる. これらの作業は、複雑な数学的モデリングや手動での最適化を伴うことが多い. 一方、機械学習では、大規模なデータセットを活用してモデルを訓練し、複雑な問題を効率的に処理できるため、開発コストと時間を削減できる

機械学習の核心的な目的は、データを用いてモデルを訓練し、未知のデータに対して予測や意思決定を行えるようにすることである. その基本的なプロセスには、問題の定義、モデルの選択、パラメータの最適化、モデルの評価が含まれる. 従来の方法と比べ、機械学習はデータ駆動型であり、人間が直接解決策を設計するのではなく、データから学習して解決策を生成する.

機械学習が適用される主なケースとして、以下のようなものがある：

- 入力と出力の対応関係を明確なルールで定義するのが困難な問題
- 大量の過去データが利用可能、または短期間でデータを収集できる場合
- 目標と評価基準が明確であり、ある程度の誤差が許容されるケース

さらに、機械学習の応用は幅広い分野で重要な役割を果たしている. 例えば、画像認識、自然言語処理、自動運転などの分野では、大量のデータを用いて学習した機械学習アルゴリズムが、従来の方法を上回る成果を示し、特定のタスクにおいては人間の能力をハイえる場合もある.

機械学習の学習プロセスの本質は、モデルを最適化し、与えられたデータに対してより良いパフォーマンスを発揮させること. 通常、損失関数を定義し、予測値と実際の値の誤差を最小化することで実現される. モデルの選択やパラメータの最適化は機械学習の重要な要素であり、モデルの複雑さを「アンダーフィッティング」と「オーバーフィッティング」のバランスの中で調整し、データへの適応と新しいタスクへの汎化を両立させる必要がある.

総じて、機械学習は従来の方法では対応が困難だった複雑な問題に対する新たな解決手段を提供する技術である. データの力を活用し、「経験から学習する」という概念をアルゴリズムやモデルに組み込むことで、人工知能や関連分野の急速な発展を支えている.

2.1.2 線形回帰と機械学習の基本概念

機械学習の核心思想を理解するために、まずはシンプルな例である線形回帰から始める. 線形回帰とは、データ間の関係を直線で表す手法であり、その核心は「学習」を通じて最も適した直線の傾き（スロープ）と切片（インターセプト）を見つけ出し、データのパターンをできるだけ正確に記述することにある.

例えば、「ある人の月の支出が月収と関係があるかどうか」を研究しているとする. 直感的には「収入が高い人ほど支出も多い」と考えられるが、その関係を具体的に数式で表すにはどうすればよいだろうか？ここで、線形回帰を使って「収入」と「支出」の関係をモデル化することができる. この関係を直線で表すと、次のようなシンプルな数式になる：

支出 $=$ 傾き $\times$ 収入 $+$ 切片

コンピュータは、過去の大量のデータをもとに直線の「傾き」と「切片」を何度も調整し、できるだけデータにフィットする最適な直線を見つける.

線形回帰では、コンピュータは「誤差」という指標を用いて、予測結果の良し悪しを評価する. 例えば、直線を使ってある人の支出を 3000 円と予測したが、実際の支出が 3500 円だった場合、その 500 円のズレが「誤差」となる. コンピュータは、この誤差をできるだけ小さくするように直線の位置や傾きを調整する. この誤差を最小化するプロセスこそが、機械学習における「学習」に相当する.

学習の過程では、次の 2 つの問題が発生することがある. 学習不足（アンダーフィッティング）の場合、直線があまりにも単純すぎて、データのパターンをうまく捉えられない状態になる. 例えば、常に一

定の支出を予測する水平線を用いると、収入が変化しても予測値が変わらないため、データの関係を正しく表現できない. 一方で、学習しすぎ（オーバーフィッティング）の場合、直線がデータのすべての点に過剰に適応し、ノイズ（誤ったデータ）までも考慮してしまう. これにより、過去のデータには非常によくフィットするが、新しいデータに対する予測精度が著しく低下する. この問題を防ぐために、機械学習では「正則化（Regularization）」という手法を用いる. これは、モデルに対して「データに適合しすぎず、しかし単純すぎないように」というバランスを取るルールを課すことで、汎化性能（新しいデータへの適応力）を向上させる技術である.

線形回帰はシンプルな手法に見えるが、機械学習の基本概念を理解する上で非常に重要である. これを通じて、コンピュータがデータからどのように規則を学習するのか、最適化をどのように進めるのか、そしてモデルの複雑さをどのように制御するかといった機械学習の核心的な概念を学ぶことができる.

2.1.3 分類タスクと機械学習の実際の応用

分類タスクは、機械学習において最も基本でありながら極めて重要な応用の一つである. その目的は、データのパターンを学習し、新しい入力データをあらかじめ定義されたカテゴリに分類することにある. これは、まるで各データポイントに適切なラベルを付ける作業のようなものである. 分類問題の核心は、ラベル付きデータを用いて特徴や規則を学習し、それを新しいデータの識別に応用することにある.

身近な分類タスクの代表例として、スパムメールフィルタリングがある. メールの内容を分析し、特定の単語やフレーズ（例：「無料」「当選」「期間限定」など）がスパムメールに多く含まれていることを学習し、逆に通常のメールにはあまり見られないことを理解する. そして、新しいメールが届いた際、学習したルールを基に、そのメールがスパムなのか通常のメールなのかを判定する. また、分類タスクは医療分野でも広く活用されている. 例えば、患者の健康診断データ（血圧、血糖値、体重など）をもとに、疾患リスクを予測することができる. さらに、画像認識においても分類タスクは重要な技術の一つである. 写真の特徴を分析することで、モデルは画像内に「猫」や「犬」が含まれているかを判断できるだけでなく、顔認識を活用してセキュリティ認証に応用することも可能である.

分類モデルの実装にはさまざまな方法があり、シンプルで計算効率の良いロジスティック回帰（Logistic Regression）や、分かりやすい構造を持つ決定木（Decision Tree）、そしてより複雑で強力なニューラルネットワーク（Neural Network）などがある. ロジスティック回帰は、分類タスクの中でも最も基本的なアルゴリズムの一つであり、データを分割する「境界線」を見つけることで分類を行う. 例えば、スパムメールの例では、ロジスティック回帰を使ってメール内の特定の単語の出現頻度を分析し、そのメールがスパムである確率を算出する. そして、この確率がある閾値（例えば 50%）を越えた場合に、そのメールをスパムとして分類する. 一方、決定木は、「当選」という単語がメールの件名に含まれているかどうかをまずチェックし、次に「期間限定」という単語が含まれているかを判定するといった、一連の条件分岐を通じて分類を行うモデルである. さらに、ニューラルネットワークはより高度な分類モデルであり、人間の脳の構造を模倣しており、特に画像や音声といった複雑なデータの処理に長けている. 画像認識では、ニューラルネットワークの「隠れ層（Hidden Layers）」を通じて画像のエッジや色、形状といった特徴を抽出し、それらをもとに画像のカテゴリを判別する.

分類タスクの重要性は、あらゆる分野での意思決定や予測に関わっている点にある. 例えば、金融分野では、顧客の行動データをもとに、ローンの延滞リスクを予測できる. 医療分野では、過去の診療記録を学習し、新しい患者の疾患の可能性を予測することが可能である. また、自動運転技術では、分類モデルが交通信号、歩行者、車両を認識し、安全な運転判断を下すのに活用されている. 分類モデルの強みは、大量のデータを処理できる点に加え、問題に応じてモデルの複雑さや予測精度を柔軟に調整できる点にある.

分類タスクの学習プロセスを理解することで、機械学習がどのようにデータを活用して問題を解決するのかを深く知ることができる. 分類モデルの背後には、データの特徴を分析し、最適なモデルパラメータを見つけ出すという継続的な最適化のプロセスがある. この学習の仕組みによって、機械学習は単なる単純なタスクをハイえ、より複雑な課題にも対応できるようになる. 分類タスクの研究と発展は、機械学習技術の進化を加速させ、現実世界のさまざまな問題に対する革新的なソリューションを提供している. スパムメールフィルタリングから顔認識、疾病予測、自動運転まで、分類モデルの応用範囲は広がり続けており、その柔軟性と実用性は、さらに高度な学習アルゴリズム（深層学習や生成モデルなど）の発展を支える基盤となっている.

2.1.4 ニューラルネットワーク：人間の脳の学習を模倣するモデル

ニューラルネットワーク（Neural Network）は、機械学習における重要なモデルの一つであり、人間の脳の神経細胞（ニューロン）の接続方式からインスピレーションを得ている. 脳は複雑なニューラルネットワークを通じて情報を処理する. コンピュータ科学者はこの仕組みを参考にし、人工ニューラルネットワーク（Artificial Neural Network, ANN）を設計し、画像認識、音声認識、自然言語処理などの複雑なタスクを解決するために活用している.

従来の線形回帰などのシンプルなモデルとは異なり、ニューラルネットワークは多層構造を持つことで強力な表現能力を持つ. ニューラルネットワークは複数の「ノード」または「ニューロン」から構成され、入力層、隠れ層、出力層に分けられる. 入力層はデータを受け取り、隠れ層は一連の複雑な非線形変換を通じて特徴を抽出し、出力層は最終的な予測や分類結果を生成する.

ニューラルネットワークの核心的な考え方は、層と層の間の接続にある. 各接続には「重み（weight）」と「バイアス（bias）」という重要なパラメータが含まれており、データが層を通過するときにこれらのパラメータを用いて加重和が計算される. その後、ReLU や Sigmoid などの活性化関数（Activation Function）を適用することで非線形性を導入する. この仕組みにより、ニューラルネットワークは単純な線形モデルでは処理できない複雑なパターンを学習し、例えば画像から特定の物体を認識することが可能になる.

ニューラルネットワークの訓練では、モデルの予測誤差を徐々に減らすことが目標となる. このプロセスは「誤差逆伝播（Backpropagation）」アルゴリズムを用いて実現される. 誤差逆伝播では、予測値と実際の値の誤差を計算し、その誤差をネットワークの各層へと逆伝播させることで、各層のパラメータを適切に調整する. さらに、勾配降下法（Gradient Descent）などの最適化アルゴリズムを利用して、重みとバイアスを更新し、誤差を最小化していく.

例えば、手書き数字認識のタスクにニューラルネットワークを用いる場合、入力層は画像のピクセル値を入力として受け取り、隠れ層は画像のエッジや形状といった特徴を抽出し、出力層が「0」から「9」までの数字のどれに該当するかを判定する. 十分な数のラベル付き画像データを用いて学習することで、ニューラルネットワークは手書き数字の特徴を理解し、見たことのない新しい画像でも高い精度で分類できるようになる.

ニューラルネットワークの登場により、機械学習はより複雑な問題を扱うことが可能になった. 多層構造によってモデルがデータから特徴を自動的に抽出できるため、専門家が手作業で特徴を設計する必要がなくなる. 前述した線形回帰や分類モデルと比較すると、ニューラルネットワークはより汎用性と適応性の高いツールであり、単純な関係性のモデリングだけでなく、多層の隠れ層を活用することで、高次元で複雑な課題にも対応できる.

総じて、ニューラルネットワークは機械学習の中心的な役割を担っており、その発展は人工知能の急速な進歩を支えている. 単純な二値分類問題から高度な画像認識や自然言語処理タスクに至るまで、ニューラルネットワークは強力なツールを提供し、深層学習の時代を切り開いた. この技術を深く理解することで、畳み込みニューラルネットワーク（Convolutional Neural Network, CNN）や再帰型ニューラルネットワーク（Recurrent Neural Network, RNN）といった、さらに高度な学習方法を探求し、より複雑な現実世界の問題を解決するための基盤を築くことができる.

2.1.5 人工ニューラルネットワーク

人工ニューラルネットワークは基本的なニューラルネットワークモデルの一種であり、その基本構成は以下の通りである：

- 入力：生物のニューロンのシナプスのように信号を受け取る.
- 重み（Weight）：入力信号の重要性を決定する. 正の値（強化）や負の値（抑制）を取ることができる.
- 活性化関数（Activation Function）：ニューロンが活性化されるかどうかを決める（生物のニューロンの閾値に相当する）.
- 出力：次の層のニューロンへ信号を伝達する.

ネットワーク構造：複数の人工ニューロンを組み合わせて、一つのネットワークを形成し、情報処理を行う.

図 1は、単純な三層 Neural Network の構造を示しており、具体的には入力層、隠れ層、出力層で構成されている.

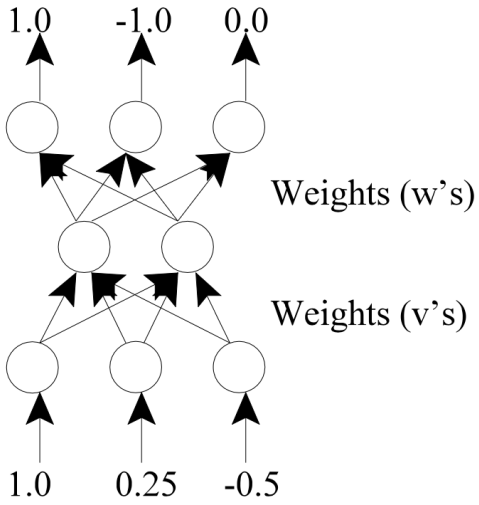

Figure 1: 論文 [2] 4.1 節の図より引用. 入力層 1 層、隠れ層 1 層、出力層 1 層を含む 3 層の人工ニューラルネットワークを示す

このネットワークの目的は、入力データを受け取り、層と層の間の重みの伝達と活性化関数の非線形変換を通じて、最終的な出力結果を生成することである.

入力層（Input Layer）は、元のデータを受け取る役割を持つ. この層には 3 つのノードが含まれ、それぞれ入力データの 3 つの特徴に対応する. 元のデータは通常数値ではないため、入力をデータ化（数値化）する必要がある. 隠れ層（Hidden Layer）は、ニューラルネットワークの中で特徴やパターンを抽出する重要な部分である. 各隠れ層のノードは、受け取った入力に対応する重みを掛け、バイアス（Bias）を加えた後、活性化関数を通じて出力を生成する. 出力層（Output Layer）は最終的な結果を生成する. このネットワークでは三つのノードを持つ.

2.1.6 ニューラルネットワークの最適化

ニューラルネットワークの性能は、主にネットワーク構造、ハイパーパラメータ設定、および訓練方法の 3 つの側面に影響を受ける. 現在、ニューラルネットワークの最適化に関する研究もこれらの側面に集中している. ニューラルネットワークにおいて、訓練アルゴリズムの選択はモデルの性能に重要な影響を与える. 訓練アルゴリズムの核心的な目標は、ニューラルネットワークのパラメータ（重みとバイアス）を最適化し、ネットワークがデータにより良く適合し、誤差を減少させることだ.[7] では、いくつかのニューラルネットワークの訓練アルゴリズムについて述べられている.

確率的勾配降下法（Stochastic Gradient Descent, SGD）SGD は基本的な最適化アルゴリズムであり、各訓練ステップで一部のデータ（バッチと呼ばれる）を使用して勾配を計算し、ネットワークパラメータを更新する. この方法は計算量を大幅に削減し、訓練効率を向上させることができる. しかし、SGD の主な欠点は、局所的最小値や鞍点に陥りやすく、最適化効果が不十分になることだ.

改善方法：

- モーメンタム法（Momentum）：モーメンタム項を導入し、過去の勾配の方向を累積することで、アルゴリズムが局所的最小値から「加速」して脱出するのを助ける.
- Nesterov モーメンタム法：モーメンタム法を基に、将来の位置を推定し、更新の精度を向上させる.

適応的勾配法（Adaptive Gradient Methods）適応法は、各パラメータの過去の勾配に基づいて学習率を動的に調整し、学習プロセスをより効率的にする. 一般的な適応法には以下がある：

- AdaGrad：各パラメータに異なる学習率を割り当て、スパースな特徴を持つデータセットに適しているが、訓練後期には学習率が小さくなりすぎる可能性がある.
- RMSProp：AdaGrad を改良し、指数移動平均を導入して勾配の累積効果を平滑化し、学習率が徐々に小さくなる問題を解決する.
- Adam：モーメンタム法と RMSProp の利点を組み合わせ、より優れた安定性と適応性を持ち、現在の深層学習で最も一般的に使用される最適化アルゴリズムの 1 つだ.

学習率スケジューリング（Learning Rate Scheduling) 学習率は、ニューラルネットワークの訓練効率に影響を与える重要なハイパーパラメータの 1 つだ. 訓練プロセス中に学習率を動的に調整することで、モデルが収束初期に適切な方向を迅速に見つけ、後期に大域的最適解を見逃すのを防ぐことができる:

- 一定の学習率：単純なタスクに適している.
- 段階的減衰：訓練が進むにつれて学習率を徐々に減少させる. 例えば、一定のステップごとに学習率を半分にする.
- コサインアニーリング（Cosine Annealing）：コサイン関数を使用して学習率を動的に調整し、更新ステップを滑らかに減少させる.

勾配問題への対応も重要である. 深層ニューラルネットワークでは、勾配消失や勾配爆発の問題が訓練プロセス中によく発生する. 以下の技術はこれらの問題を効果的に緩和することができる：

- 重み初期化：例えば Xavier 初期化や Kaiming 初期化は、初期勾配値が合理的な範囲にあることを保証する
- バッチ正規化（Batch Normalization）：各層の出力を標準化し、勾配の流れを改善し、訓練をより安定させる
- 勾配クリッピング（Gradient Clipping）：勾配値を制限し、勾配爆発問題を回避する

上記のアルゴリズムに加えて、以下の方法でもニューラルネットワークの訓練効果を向上させることができる：

- 正則化方法：例えば L2 正則化（重み減衰）や Dropout 技術は、過学習を効果的に防止する
- スキップ接続（Skip Connection）：ResNet50 の設計のように、入力をより深い層に直接伝達し、勾配消失問題を緩和する
- 早期停止（Early Stopping）：検証セットの誤差がそれ以上減少しなくなった時点で訓練を早期に終了し、過学習を回避する

2.2 遺伝的アルゴリズム (Genetic Algorithm; GA)

2.2.1 進化論から発想を得た手法

論文 [8] は、ダーウィンが提唱した、生物が時間とともに進化し、環境に適応し、多様性を形成する仕組みを説明する理論であり、その核心は自然選択にある. 進化論によると、生物個体の間には遺伝的な変異によって形態、行動、機能などに違いが生じる. これらの変異が自然選択の基盤となる. 資源が限られ、環境が絶えず変化する中で、個体間には必然的に競争が発生する. この生存競争によって、環境に適応した個体は生存しやすくなり、繁殖を通じてその有利な特徴を次世代に伝える. 一方で、環境に適応できない個体は淘汰されていく. この「適者生存」の原則が、世代を越えて集団の特性を最適化し、生物がその生態環境により適応するように変化していく. ダーウィンは、この過程が緩やかに進行し、小さな変異が長期間にわたって蓄積されることで、最終的に大きな進化の結果を生むと考えた.

また、進化論は生物多様性の起源についても説明している. ダーウィンは、地理的隔離、生態的適応、繁殖障壁などの要因によって、同じ種の異なる集団が長期間にわたって十分な違いを蓄積すると、新たな

種が生じることがあると考えた. この種分化の過程によって、新しい生物種が誕生し、地球上の豊かな生命の多様性が生まれるという理論的な根拠を提供した.

ダーウィンの進化論は、生物の進化のメカニズムを明らかにしただけでなく、科学界に新たな視点をもたらし、人類の自然界に対する理解を大きく変えた. 進化論は、生命の多様性が固定されたものではなく、変異、選択、分化などのプロセスを通じて常に変化し続けることを示している. この理論は生物学の基盤となっただけでなく、生態学、遺伝学、現代進化学の発展にも大きな影響を与えた. ダーウィンは『種の起源』の研究を通じて、自然界の内在的な法則を理解するための基礎を築き、現代科学の進歩に欠かせない理論的枠組みを提供した.

2.2.2 遺伝的アルゴリズム (GA)

遺伝的アルゴリズムは、進化論における自然選択と生物の遺伝的変異の核心的な考え方を取り入れている. 進化論によれば、生物個体は遺伝の過程で変異を生じ、資源が限られ環境が変化する中で、個体間で生存競争が発生する. 適応性の高い個体は生存しやすく、繁殖を通じて有利な特性を次世代に伝える. 一方で、環境に適応できない個体は淘汰される. 遺伝的アルゴリズムはこの過程を模倣し、最適化を行う. 個体は問題の候補解に対応し、適応値関数によって解の優劣が評価される. アルゴリズムは選択、交叉、変異といった操作を通じて生物の進化メカニズムを再現する. 選択操作では適応値の高い個体が優先的に残され、交叉操作では遺伝子の組み換えをシミュレーションし、変異操作ではランダム性を導入して種群の多様性を高める. これらの進化プロセスを繰り返すことで、遺伝的アルゴリズムは種群を徐々に最適化し、最終的に全局的最適解に近づける. このように進化論の概念を取り入れることで、遺伝的アルゴリズムは複雑な探索空間において高品質な解を見つけ出し、非凸最適化問題においても高い適応性を発揮する.

遺伝的アルゴリズムには多くの変種があり、その中でも最も基本的で広く適用されているのが Simple Genetic Algorithm (SGA) である.SGA の核心的な考え方は、自然選択と遺伝メカニズムを模倣し、選択、交叉、変異の三つの主要なステップを通じて種群内の個体を段階的に最適化し、その適応値を向上させることである. アルゴリズムはまずランダムに種群を初期化し、それぞれの個体が問題の候補解を表し、二進数または実数によって遺伝情報をエンコードする. その後、適応値関数を用いて各個体の適応値を評価し、選択された個体が繁殖に利用される確率を決定する.

選択段階では、ルーレット選択やトーナメント選択などの方法が一般的に用いられ、適応値の高い個体が優先的に残される. 交叉操作では、生物の遺伝過程における親の遺伝情報の組み換えを再現し、新たな個体を生成することで、より広範な解空間を探索する. 一方、変異操作では、個体の遺伝情報を小確率でランダムに変化させ、突然変異のプロセスをシミュレーションすることで、種群の多様性を維持し、早期収束を防ぐ. このように進化プロセスを繰り返すことで、SGA は全局的最適解に向かって収束していく.

しかし、SGA は多くの最適化問題において良好な結果を示すものの、局所収束が遅い、全局探索能力が限定的であるといった問題が存在し、これを克服するためにさまざまな改良アルゴリズムが開発されてきた. それでもなお、SGA は遺伝的アルゴリズムの基礎的な枠組みを提供し、関数最適化や経路計画といった分野で広く活用されている.

SGA を基盤として、研究者たちは遺伝的アルゴリズムの全局探索能力、収束速度、そして複雑な問題への適応能力を向上させるためにさまざまな改良手法を提案してきた. その中でも、Adaptive Genetic Algorithm (AGA) は交叉率と変異率を動的に調整するメカニズムを導入し、種群の適応値分布の状態に応じてパラメータを柔軟に変更し、全局探索と局所探索のバランスを最適化する手法である. また、Multi-Objective Genetic Algorithm (MOGA) は遺伝的アルゴリズムを多目的最適化問題へと拡張し、Pareto 最適解の概念を用いることで複数の目標関数を同時に最適化する. このアプローチの代表例として NSGA-II や SPEA などがある.

さらに、収束速度を向上させ解の精度を高めるために、Hybrid Genetic Algorithm (HGA) では局所探索や他の最適化手法（シミュレーテッドアニーリングや粒子群最適化など）を組み合わせることで、種群の進化の過程において局所最適解の探索を強化する手法が提案されている. また、大規模な計算負荷を軽減し、アルゴリズムの実行効率を向上させるために、Parallel Genetic Algorithm (PGA) では種群を複数のサブ種群に分割し、それぞれを分散計算によって独立に進化させながら、定期的に情報交換を行うことで計算効率を飛躍的に向上させる.

Chaotic Genetic Algorithm (CGA) ではカオス理論を取り入れ、カオスシーケンスを導入することで種群の多様性を向上させ、局所最適解への過剰適応を防ぐ. このように、遺伝的アルゴリズムは SGA を基盤としながら、さまざまな改良が加えられ、非線形問題、多目的最適化、動的環境における適応的最適化

といったより複雑な課題にも対応できるようになっている.SGA はこれらの改良の基礎となり、進化計算の研究や実用化の分野において重要な役割を果たしている.

2.3 遺伝的アルゴリズムを用いたニューラルネットワークの最適化研究

遺伝的アルゴリズムを用いた機械学習モデルの最適化に関する研究は、主にニューラルネットワークの構造最適化 [9]、パラメータ最適化 [9]、ハイパーパラメータ最適化 [10] の三つの分野に集中している.

論文 [9] では、GA-NINASWOT と呼ばれる手法が提案されている. これは遺伝的アルゴリズムに基づく Neural Architecture Search (NAS) の一種であり、優れたニューラルネットワークアーキテクチャを探索する際に、従来の方法のような時間のかかる完全な学習プロセスを回避することを目的としている.

具体的には、遺伝的アルゴリズムが生物進化のプロセス（選択、交叉、変異など）を模倣する. 各候補ニューラルネットワークのアーキテクチャは、遺伝的アルゴリズムの個体として扱われ、適応値関数を用いて評価される.GA-NINASWOT は、Neural Architecture Search Without Training (NASWOT) のスコアリングメカニズムを統合しており、各候補ネットワークを完全に学習せずにその性能を迅速に評価する. この NASWOT のスコアリング手法は、ニューラルネットワークの入力データに対する「感度」に基づいており、ネットワークの勾配変化を評価することで潜在的な性能を推定する.

さらに、スコアリングの精度を向上させるために、Noise Immunity（抗ノイズ性スコアリング）を導入している. この手法では、入力データにガウスノイズなどのランダムノイズを加え、その状態でのネットワークの挙動を観察することで、ネットワークのロバスト性や汎化能力を測定する. この改良により、GA-NINASWOT はより信頼性の高いアーキテクチャの選択が可能になる.

実験結果では、GA-NINASWOT が計算コストを大幅に削減することが確認された. 従来の NAS 手法では、候補アーキテクチャを完全に学習する必要があるが、GA-NINASWOT はこのプロセスを回避し、計算コストを約 99% 削減することができる. 計算量を大幅に削減しつつも、CIFAR-10、CIFAR-100、ImageNet-16-120[11] などのデータセットにおいて、完全な学習を要する NAS 手法（例：DARTS[12]、REA[13]）と同等のテスト精度を達成している. この結果は、高速な検索と性能保証のバランスが良好であることを示している.

しかし、この手法を適用する際には、既存のアルゴリズムやモデル構造に一定の変更が必要であり、異なるニューラルネットワークに適用する場合、遺伝的アルゴリズムの適用方法や位置を調整する必要がある. また、遺伝的アルゴリズムと組み合わせることで、必ずしも元のアルゴリズムを常に上回るとは限らないため、さらなる改良の余地がある. 例えば、より複雑なタスクに対してより精密なスコアリングメカニズムを設計することや、アーキテクチャの変更依存度を低減する方法を探ることが考えられる.

論文 [14] では、遺伝的アルゴリズムと人工ニューラルネットワーク（Artificial Neural Network, ANN）を組み合わせた高効率な最適化手法を提案し、Savonius 型風力タービンの設計パラメータ最適化に応用した. この Savonius タービンは、一般的な垂直軸型の風力発電装置であり、その性能は Coefficient of Power（Cp）によって評価される. しかし、重なり比（overlap ratio）、ブレード回転角、レイノルズ数、速度比など、多くの設計パラメータが影響を与えるため、それらの非線形な相互作用により、従来の最適化手法では高次元・多目的な問題を効率的に扱うのが困難だった. さらに、Cp の評価には通常、Computational Fluid Dynamics (CFD) シミュレーションが必要であり、高精度ではあるが計算コストが非常に高いという課題があった.

計算コストを削減するために、研究者は CFD シミュレーションの代替として人工ニューラルネットワーク（ANN）を使用し、ANN を目標関数の近似モデルとして学習させた. まず、限られた実験データや少量の CFD シミュレーション結果から、入力パラメータと出力性能の関係を抽出し、ANN に学習させることで、複雑な非線形関係をモデル化した.ANN の構造は、入力層・隠れ層・出力層からなり、重みとバイアスを調整することで予測誤差を最小化する. これにより、ANN は様々な設計パラメータの組み合わせに対する Cp の予測を高速に行うことが可能となり、最適化プロセスにおいて計算コストを大幅に削減できる. 訓練された ANN は高い予測精度を持ち、推論速度も非常に速いため、CFD モデルの代替として遺伝的アルゴリズムの適応値評価に使用された.

遺伝的アルゴリズムはグローバル最適化手法として、進化プロセスをシミュレーションしながら最適なパラメータを探索する. アルゴリズムは、ランダムに初期化されたパラメータセットを集団として扱い、選択・交叉・変異といった操作を通じて世代ごとに進化させる. 各世代では、ANN モデルを用いて個体（パラメータセット）の適応値（Cp）を迅速に評価し、適応値の高い個体を選択して次世代に継承させる. このプロセスを繰り返すことで、より良い設計パラメータを見つけ出すことができる. 従来の方法と比較

して、この手法は最適化の効率を大幅に向上させると同時に、性能向上の点でも優れた結果を示している.

実験結果では、遺伝的アルゴリズムと ANN を組み合わせた手法が、計算コストを大幅に削減しながら Savonius タービンの Cp を最適化できることが示された. さらに、この手法は複雑な工学的問題に対する適用可能性や汎用性を持つことが確認された. このハイブリッド手法は、ニューラルネットワークの内部構造を変更することなく、事前に訓練された ANN を補助ツールとして活用し、遺伝的アルゴリズムと組み合わせることで高効率な最適化を実現する. これにより、今後の複雑なシステムの多目的最適化において、新たなアプローチとしての可能性を示した.

論文 [10] では、遺伝的アルゴリズム（GA）に基づくハイパラメータ最適化手法を提案し、神経機械翻訳（Neural Machine Translation, NMT）モデルのハイパラメータを最適化することで、手動でのパラメータ調整の負担を軽減し、ニューラルネットワークの全体的な性能を向上させることを目的としている.

現在、ニューラルネットワークのハイパラメータ選択はモデルの性能にとって極めて重要である. しかし、従来のハイパラメータ最適化手法であるグリッドサーチ（Grid Search）やランダムサーチ（Random Search）では、ハイパラメータの数が増えるにつれて計算コストが指数関数的に増大し、大規模なニューラルネットワークでは効率的に適用することが困難である.

研究では、遺伝的アルゴリズム（GA）を用いたハイパラメータの自動探索手法を提案し、進化戦略を利用してハイパラメータを最適化することで、探索効率を向上させ、計算コストを削減する. 具体的には、各ニューラルネットワークのハイパラメータの組み合わせを GA の個体（Chromosome）として扱い、適応度関数に基づいて評価を行う. 適応度関数としては、機械翻訳の品質を評価する指標である BLEU スコアを使用し、ハイパラメータの組み合わせの優劣を判断する. 最適化対象とするハイパラメータは BPE サブワード単位数（Byte Pair Encoding Units）, エンコーダ/デコーダ層数（Encoder/Decoder Layers）, 単語埋め込み次元数（Word Embedding Dimensions）, 隠れユニット数（Hidden Units）, アテンションヘッド数（Attention Heads）, 初期学習率（Initial Learning Rate）. さらに、本研究では個体置換戦略を導入し、各世代の GA 進化の際に、最も適応度の低い個体のみを置き換えることで、探索の安定性を確保している.

研究では、日英翻訳データセット（Japanese-to-English dataset）を用いて実験を行い、GA とランダムサーチ（Random Search）のハイパラメータ最適化性能を比較した. 結果として、計算コストの削減に関しては、従来の NAS 手法ではすべての候補となるハイパラメータの組み合わせを完全に学習する必要があるが、研究の手法では完全な学習を行わずに最適なハイパラメータを探索できるため、計算コストを約 99% 削減し、わずかな計算資源で高性能なハイパラメータの組み合わせを見つけることが可能である. 翻訳精度に関しては、CIFAR-10、CIFAR-100、ImageNet-16-120 などのデータセットを用いた実験において、GA によるハイパラメータ最適化手法は、DARTS や REA などの完全学習を必要とする NAS 手法と同等の BLEU スコア [15] を達成している. 検索効率の向上に関しては、GA によるハイパラメータ探索はランダムサーチよりも収束が速く、より少ないイテレーション回数で高品質なハイパラメータの組み合わせを見つけることができ、平均すると GA はランダムサーチよりも 1.27 個体少ない探索回数で BLEU スコア 16 に到達した.

2.4 既存研究の課題

現在、遺伝的アルゴリズムをモデルの性能向上に利用する研究の多くは、遺伝的アルゴリズムを直接モデルの学習プロセスに組み込む方法を採用している. この方法は、遺伝的アルゴリズムのグローバル探索能力をある程度活かすことができるが、いくつかの大きな制約も伴う. まず、異なる組み合わせ方を試すたびに、モデルをゼロから再学習する必要がある. これは、学習に時間がかかり、計算リソースの要求が高いディープラーニングモデルにとって非常に高コストであり、実際の応用における実現可能性を大きく制限している. 次に、遺伝的アルゴリズムを導入することで、元のモデルの学習プロセスが変化し、従来の最適化戦略や学習規則が崩れる可能性がある. そのため、最終的に得られるモデルの性能が従来の学習方法よりも優れるとは限らない. さらに、この手法の調整は特定のタスクやモデルの特性に大きく依存し、それぞれのモデルに対して特別な組み合わせ方法を設計する必要がある. これにより、開発やデバッグの複雑さがさらに増してしまう. したがって、遺伝的アルゴリズムのグローバル最適化能力を活かしつつ、モデルを再学習せずに、元の学習プロセスの整合性を維持する方法を見つけることが、緊急に解決すべき課題となっている.

3 本研究における遺伝的アルゴリズムの使用

3.1 実験方法

本研究では、遺伝的アルゴリズムを用いた最適化手法を提案し、ゼロからモデルを再学習するのではなく、すでに学習済みのモデルのパラメータ調整に焦点を当てる. 具体的には、まず異なるハイパーパラメータを調整して一連の事前学習済みモデルを生成し、それらを遺伝的アルゴリズムにおける初期親集団と見なす. 次に、親集団内のモデルのパラメータに対して交叉および突然変異操作を実行し、新たなモデルを生成する. そして、適応値関数を設定して候補モデルを選別し、性能の高いモデルを残し、適応値の低いモデルを淘汰することで、世代を重ねながらモデルの性能を向上させる.

この方法は、既存研究における 2 つの主要な問題を効果的に解決する. まず、本手法は事前学習済みモデルを最適化の基盤とするため、手法を変更するたびにモデルをゼロから学習し直す必要がなく、高い計算コストを回避できる. これにより、特に学習に長時間を要するディープラーニングモデルにおいて、リソース消費を大幅に削減し、最適化プロセスの効率を向上させる. 次に、本手法は元のモデルの学習プロセスを変更せず、パラメータ最適化の段階でのみ遺伝的アルゴリズムを導入する. そのため、元の学習プロセスの整合性が保たれ、手法を組み合わせた後のモデル性能の安定性と信頼性が確保される.

さらに、適応値関数を手動で設定できるため、本手法は実際のニーズに応じて最適化の方向を柔軟に定義できる. 例えば、画像分類タスクでは、適応値関数を分類精度とすることで、初期集団がより高い分類精度を持つ方向に最適化される. また、推論速度やリソース消費（推論時間やメモリ使用量）を適応値関数に設定することで、軽量化されたモデルや、組み込み機器・リアルタイム処理に適したモデルを最適化できる. 生成タスクでは、適応値関数を生成サンプルの多様性や品質と定義することで、生成モデルのパフォーマンス向上を目指すことができる. また、多目的最適化問題に対しては、適応値関数を複数の指標（例えば、精度とリソース消費のトレードオフ）を組み合わせた形で設定することで、遺伝的アルゴリズムを用いて総合的な性能を満たす最適なパラメータの組み合わせを探索できる. この柔軟性により、本手法はさまざまなタスクの要求に応じたカスタマイズが可能となり、ディープラーニングモデルの性能向上に幅広い応用シナリオを提供する.

このように、本研究では遺伝的アルゴリズムのグローバル探索能力を十分に活用し、事前学習済みモデルと組み合わせることで、学習済みモデルの性能をさらに引き出しつつ、計算コストを抑え、高い汎用性を維持する. これにより、ディープラーニングモデルの最適化に対して、効率的かつ柔軟なソリューションを提供する.

3.2 アルゴリズムの紹介

3.2.1 DMC（Deep Monte Carlo）

DMC（Deep Monte Carlo、深層モンテカルロ）は、モンテカルロ法と深層学習を組み合わせた強化学習アルゴリズムであり、エージェントの戦略を最適化するために、開始状態から終了状態までの累積報酬（リターン）を評価する.DMC の核心となる考え方は、深層ニューラルネットワークを用いて方策関数や価値関数を近似することであり、高次元の状態空間や行動空間を持つ問題に対応できるようにすることである. 実際の応用において、DMC はまず大量の軌跡をサンプリングして生成し、それらの軌跡の長期累積報酬を計算する. これらの報酬を基に DMC はエージェントの戦略を更新し、累積報酬を最大化する. この方法は、特に複雑な環境やターン制ゲームに適しており、単一のステップでの即時報酬に依存するのではなく、軌跡全体を評価することに重点を置いている.

さらに、DMC は自己対戦（self-play）を訓練手法として活用している. エージェントが自身の現在の方策と対戦することで、現在の戦略の弱点を効率的に発見し、改良を加えることができる. この自己対戦のメカニズムにより、DMC は戦略的な対戦が求められる場面、例えばチェス、囲碁、そして闘地主（DouDiZhu）などのゲームにおいて優れた性能を発揮する. 実験では、DMC を闘地主というカードベースの戦略ゲームに適用した. 闘地主は 3 人でプレイするカードゲームであり、プレイヤーは地主（Landlord）と農民（Farmers）という 2 つの陣営に分かれる. 地主陣営は 1 人で構成され、農民陣営は 2 人が協力して地主と対戦する. ゲームの目的は、手札を使った戦略的なプレイによって、いち早く手札をすべて出し切ることである. 地主が最初に手札を出し切れば勝利し、逆に農民のどちらか 1 人でも先に手札を出し切れば農民陣営の勝利となる. 闘地主は対戦要素と協力要素を兼ね備えており、複雑な戦略や協調プレイを研究するための理想的な環境を提供する. ゲーム内で、地主は 2 人の農民の連携攻撃に対処する必要があり、

できるだけ早く手札を出し切るための攻撃的な戦略が求められる. 一方で、農民は協力しながら、交互に防御と攻撃を行い、地主の勝利を阻止する必要がある. 本研究では、AI がこの非対称なゲーム構造の中で、地主として独立して戦う戦略を学習すると同時に、農民として味方と協力する方法を習得することで、全体の戦略レベルを向上させることを目的としている. 闘地主のこの独特なルールは、強化学習アルゴリズムの最適化に対して、多役割・多目的の複雑なテスト環境を提供し、AI 研究にさらなる挑戦と機会をもたらす.

DouZero が DMC を用いて闘地主のエージェントを訓練する方法は、以下のいくつかの重要なステップに分けられる. まず、モデルの初期化として、DouZero は LSTM（Long Short-Term Memory）をニューラルネットワークモデルとして採用する. このネットワークは、過去のデータを記憶しつつ、ゲート機構によってどの情報を保持し、どの情報を忘れるかを柔軟に制御することができるため、闘地主のような時間依存性のある連続的な意思決定タスクに適している. 闘地主では、地主と農民陣営が異なる情報（例えば、残りの手札の枚数やチームの目標など）を持つため、DouZero はそれぞれの陣営に対して異なる LSTM モデルを作成する. モデルの入力には、現在の対局情報、手札情報、および過去の行動データが含まれ、出力は現在の対局状況におけるすべての合法な行動と、それに対応する期待報酬（Q 値）となる.

次に、対局データの生成において、モデルを初期化した後、各役割に手札を配り、自己対戦を開始する. 対局中、モデルは現在の局面を入力として受け取り、合法な行動とそれに対応する Q 値を生成する. 行動選択には ϵ-グリーディ（ϵ-greedy）戦略を使用し、確率 ϵ で最大の Q 値を持つ行動（現在の戦略を利用）を選択し、確率 $1-\epsilon$ でランダムに合法な行動を選択する（新たな可能性を探索）. この戦略により、モデルは訓練中に既存の戦略を最適化しつつ、一定の探索能力を保持することで、局所最適解に陥るのを防ぐことができる.

その後、モデルの最適化プロセスにおいて、各対局終了後、各陣営のモデルに対して新しい Q 値を割り当てる. 勝利した側の Q 値を 1、敗北した側の Q 値を-1 とし、これらの新しい Q 値をターゲット値として、対局中にモデルが生成した各ステップの Q 値と比較することで損失関数を計算する. そして、誤差逆伝播（バックプロパゲーション）アルゴリズムを用いて、損失関数の結果に基づきモデルの重みを更新し、モデルが真の報酬値により近づくようにする.

最後に、性能向上のプロセスでは、継続的な自己対戦と、結果に基づいた重みの更新を通じて、DouZero は徐々に戦略の性能を向上させていく. 訓練が進むにつれて、モデルは闘地主の複雑な意思決定シナリオにより適応できるようになり、戦略の効率性と精度が向上する. 以上の方法は、DMC の軌跡全体の評価と自己対戦のメカニズムを組み合わせることで、DouZero の意思決定能力を最適化するだけでなく、闘地主環境における適応性と知能を大幅に強化することに成功している.

3.2.2 DQN (Deep Q-Network)

深層 Q ネットワーク（Deep Q-Network, DQN）は、深層学習と強化学習を組み合わせた手法であり、複雑な意思決定問題を解決するために使用される.DQN の主な目的は、エージェントが環境との相互作用を通じて、累積報酬を最大化する行動戦略を学習することである.

その動作原理は簡単に説明すると、エージェントが現在の環境の状態を観察し、ニューラルネットワークを通じて、各行動の「価値（Q 値）」を予測するというものだ. エージェントはこれらの価値に基づいて行動を選択する. 行動とその結果を観察していく中で、DQN はニューラルネットワークのパラメータを調整し、予測した価値が実際の結果により近づくようにすることで、最適な戦略を段階的に学習する.

DQN の大きな特徴の一つは「経験再生（Experience Replay）」技術を使用することだ. これは、エージェントが得た経験データを保存し、ランダムに抽出して学習に使用するという手法である. この技術はデータ利用効率を向上させるだけでなく、データ間の関連性を低減させることで学習をより安定させる効果がある. 同時に、DQN は「ターゲットネットワーク（Target Network）」技術を導入しており、安定した目標ネットワークを使用して Q 値の目標を計算することで、目標値が頻繁に更新されることによる不安定性を回避している.

DQN は最初に Atari ゲームの人工知能の訓練に使用された. エージェントはピクセル画像やスコア情報を観察するだけで、複数のゲームで人間プレイヤーをハイえる成果を上げた. この手法は、試行錯誤によって段階的に最適化できるタスクに適しており、例えばロボット制御、ゲーム AI、推薦システムなどで応用されている. また、DQN は高次元の生データ（ピクセルデータやセンサーデータなど）から直接学習するため、手作業による特徴設計への依存を効果的に減らすことができ、深層強化学習分野の重要な基盤となった. この手法はその後のアルゴリズム（Double DQN[16] や Prioritized Experience Replay[17] など）

の発展にも影響を与えた.

Leduc Hold'em は、2 人対戦のポーカーゲームで、K、Q、J のカード各 2 枚、合計 6 枚を使用して行う. ゲームは 2 ラウンドで構成される. 第 1 ラウンドの開始時に、各プレイヤーは 1 チップをアンティ（底注）として投入し、ランダムに 1 枚の手札を受け取る. その後、ベットまたはレイズを行い、レイズ額は固定で 2 とされる. 第 2 ラウンドでは、1 枚の共通カード（コミュニティカード）が公開され、再びベットまたはレイズが行われる. このラウンドではレイズ額は 4 に固定されている.

ゲーム終了時、プレイヤーの手札が公開カードと一致する場合、そのプレイヤーが勝利する. 一致しない場合は手札の強さを比較し、大きい方の手札を持つプレイヤーが勝利する. プレイヤーは任意のベット段階でフォールドを選択し、このラウンドを放棄することもできる.

3.2.3 ResNet50 (Residual Network 50)

畳み込みニューラルネットワーク（Convolutional Neural Networks, CNNs）は、画像処理や認識に特化したニューラルネットワークモデルであり、人間の視覚システムを模倣している.CNN は、画像の特徴を段階的に抽出し、エッジやテクスチャなどの単純な特徴から、複雑な形状やオブジェクトへと情報を統合し、最終的に画像分類や物体検出などのタスクを実現する.

CNN の中核をなすのが畳み込み層（Convolutional Layer）であり、これは畳み込みカーネル（フィルター）を用いて画像の局所領域をスキャンし、特徴を抽出する. 畳み込みカーネルは画像上をスライドしながら、ウィンドウ内のピクセルとフィルターの加重和を計算し、新たな特徴マップを生成する. この操作により、エッジや線などの低次特徴が抽出される.

次に、CNN はプーリング層（Pooling Layer）を導入し、データの次元を削減して計算効率を向上させる. プーリング層は最大値プーリング（Max Pooling）または平均値プーリング（Average Pooling）を用いて特徴マップのサイズを縮小しながら、主要な情報を保持する. これは、画像を圧縮するのと同じ効果を持ち、計算負荷を軽減すると同時に、モデルのノイズ耐性を向上させる.

特徴抽出が完了すると、CNN は通常全結合層（Fully Connected Layer）を用いて抽出された特徴を特定のクラスにマッピングする. 全結合層は画像の高次特徴を出力クラスと対応付け、softmax 関数を通じて最終的な確率を算出する. 例えば、猫と犬を分類するタスクでは、モデルの出力が「猫 80%、犬 20%」となることが考えられる.

CNN をより高度な画像処理に適応させるため、研究者たちはさまざまなネットワークアーキテクチャを開発してきた. たとえば、AlexNet[18] は、大規模な画像認識コンペティションで初めて優勝した深層ネットワークであり、より多くの畳み込み層と非線形活性化関数（ReLU）を導入することで、性能を大幅に向上させた. その後、ResNet50（Residual Network）は残差接続（Residual Connection）を導入することで、深層ネットワークにおける勾配消失問題を解決し、より深く、強力なネットワークの学習を可能にした.

ただし、CNN は高い性能を誇る一方で、大量のラベル付きデータと膨大な計算リソースを必要とする. また、回転やスケール変化に対しては依然として一定の制約がある. しかし、技術の進歩により、CNN は画像分類だけでなく、医療画像解析、自動運転、動画分析など幅広い分野で応用され、人工知能の基盤技術の一つとなっている.CNN の本質的なアイデアは、画像を階層的に処理し、重要な情報を段階的に抽出しながら、最終的にピクセルレベルの情報を分類ラベルに変換することである.

ResNet50[6] は、ResNet（深層残差ネットワーク）ファミリーの一つで、50 層の深さを持つ.ResNet の基本的なアイデアは、残差学習フレームワーク（Residual Learning Framework）を導入することで、深層ネットワークの退化問題（ネットワークの層が増えると逆に学習精度が低下する現象）を解決することにある.

ResNet50 は残差ブロック（Residual Block）という特殊な構造を採用している. 各残差ブロックでは、ショートカット接続（Shortcut Connection）を用いて、入力を直接出力に加算する. この設計により、ネットワークはより効率的に最適な特徴を学習できる. 具体的には、残差ブロックの目的は、入力と出力の間の差分（残差）を学習することであり、直接出力を学習するのではない. これにより、特定の層がタスクに対して貢献しない場合、その層はほぼゼロの残差を学習することで簡単にスキップされるため、深層ネットワークで頻発する勾配消失問題を回避できる.

ResNet50 は、計算量を抑えつつ性能を維持するために、ボトルネック設計（Bottleneck Design）を採用している. 各残差ブロックは、3 つの畳み込み層（1×1、3×3、1×1）から構成されており、最初の 1×1 畳み込みで次元削減を行い、中間の 3×3 畳み込みで特徴抽出を行い、最後の 1×1 畳み込みで次元を回復

する. この設計は計算コストを削減しながら、ネットワークの効率を向上させることができる.

実験結果によると、ResNet50 は ImageNet データセットにおける画像分類タスクで優れた性能を発揮し、Top-5 エラー率は 6.71% に達した. これは同時期の他のネットワークアーキテクチャ（VGG[19] や GoogLeNet[20] など）よりも優れた結果である. さらに、ResNet50-101 や ResNet-152 のように層を増やすことで、ネットワークの性能が向上し、退化問題が発生しないことも確認されている.

総括すると、ResNet50 の主な特徴は、残差学習フレームワークとボトルネック設計を導入することで、高い分類精度を維持しながら、深層ネットワークの学習効率と安定性を向上させた点にある. このアーキテクチャは、画像分類や物体検出などのコンピュータビジョンタスクに広く応用され、深層学習の発展における重要なブレイクスルーの一つとなっている.

4 DMC への遺伝的アルゴリズムの適用

ハイパフォーマンスな DMC モデルの訓練には長時間が必要であり、ハードウェアリソースの制約によって十分な数の高性能モデルを初期集団として準備することが困難である. そのため、本実験では 2 回の実験を行うことにした.

第 1 回目の実験では、3 日間訓練した比較的一般的な性能のモデルを初代親集団として使用した. この実験の主な目的は、初代親集団の多様性を確保し、早期収束を回避しつつ、遺伝的アルゴリズムがモデル性能の向上に与える影響を確認することである. 特に、性能の低いモデルを初代親集団とした場合、交叉や突然変異といった遺伝操作を通じて、遺伝的アルゴリズムが徐々にモデルの全体的な性能を向上させられるかどうかを観察する. この実験により、初期集団の性能が低い場合でも、遺伝的アルゴリズムが有効な最適化能力を持つかどうか、また、モデルの弱点をどの程度改善できるかを評価することができる.

第 2 回目の実験では、より長時間訓練した明らかに高性能なモデルを初代親集団として使用した. この実験の目的は、初代親集団の性能がすでに高い場合でも、遺伝的アルゴリズムがさらにモデルの性能を向上させられるかどうかを検証することである. この実験を通じて、すでに強い初代親集団に対しても、遺伝操作によってさらに最適化が可能かどうかを探ることができる. また、異なる初代親集団の性能条件下での比較実験を行うことで、遺伝的アルゴリズムの特性や可能性をより深く理解することを目指している.

これら 2 回の実験は、初代親集団の性能が遺伝的アルゴリズムの効果に与える影響を比較するだけでなく、将来の実験設計において貴重な参考となる. 特に、ハードウェアリソースが限られている環境において、どのように初代親集団を選択し、実験を設計すれば最適な結果を得られるかを考える上で、重要な示唆を与えてくれる.

4.1 低性能の DMC モデルに対する GA の適用

4.1.1 初期親集団の選択

DMC の訓練過程において、ϵ（epsilon）は重要なハイパーパラメータであり、モデルの学習プロセスや最終的な戦略の性能に直接影響を与える.ϵ は探索（exploration）と活用（exploitation）のバランスに密接に関係し、エージェントが学習過程でどのように意思決定を行うかを決定する.

具体的には、ϵ はエージェントが行動を選択する際に、新しい戦略を探索するか、すでに学習した戦略を活用するかのトレードオフを制御する.DMC では、エージェントの出力は現在の対局におけるすべての合法なアクションと、それに対するニューラルネットワークの報酬予測（Q 値）を含む. エージェントは、1-ϵ の確率で Q 値が最大の合法なアクションを選択し、これは最適な既存の戦略に従うことを意味する. 一方、ϵ の確率でランダムに合法なアクションを選択し、これは探索を行うことを意味する. 探索の目的は、新しい、まだ十分に学習されていないアクションを試すことであり、活用はすでに知っている情報に基づき、現在最適と考えられるアクションを選択することである.ϵ の設定は、エージェントが訓練過程で探索と活用をどのようにバランスさせるかを決定し、それによって学習全体のプロセスや最終的な戦略の品質に影響を与える.

もし ϵ が小さすぎると、エージェントの探索行動が極端に制限され、学習済みの戦略に過度に依存することになる. その結果、エージェントは早い段階で探索をやめてしまい、環境内に存在する可能性のある他の有効な戦略を無視してしまうことになる. 長期的には、現在の戦略に過剰に依存することで、エージェントが局所最適解に陥り、さらなる改善ができなくなる可能性がある. 結果として、エージェントが学習する戦略は必ずしも全体最適とはならず、モデルの改善の余地が制限されてしまう.

一方で、ϵ が大きすぎると、エージェントはランダムな探索を過剰に行い、現在の戦略との整合性が低くなる. これは訓練の安定性や収束速度に悪影響を与える可能性がある. この場合、エージェントは繰り返しランダムな行動を選択し続けるため、学習した知識を十分に活用して効果的に最適化する時間が得られなくなる. 結果として、訓練の不安定化や収束困難な状態を招く恐れがある. また、過度な探索によって、モデルが訓練過程で有益な経験を十分に蓄積できず、最終的な戦略の性能が低下する可能性もある.

したがって、DMC の訓練において適切な ϵ の設定は極めて重要である.ϵ の設定は、訓練過程の安定性や収束速度に影響を与えるだけでなく、最終的なモデルの性能にも関わる. 探索と活用のバランスを取るために、一般的には ϵ を徐々に減衰させる方法が採用される. これにより、訓練の初期段階ではより多くの探索を行い、新しい戦略を試す機会を増やし、後半では学習済みの戦略により多く依存し、安定した最適化を行うことができる. しかし、DMC では訓練が自動的に停止するわけではなく、終了のタイミングは手動で制御する必要があるため、ϵ は固定値として設定されることが多い. このような設定によって、エージェントは一定の探索能力を維持しつつも、過剰な探索による不安定化を防ぐことができる.

総じて、ϵ は DMC のような強化学習タスクにおいて極めて重要な役割を果たす. 適切な設定と調整を行うことで、モデルの性能を向上させ、学習プロセスの効率を高め、収束速度を向上させ、最終的な戦略の品質を向上させることができる. 強化学習モデルにとって、適切な ϵ の設定は最適化の重要な要素であり、モデルが複雑なタスクを成功裏に解決し、高品質な意思決定戦略を獲得できるかどうかを左右する.

初代親集団の多様性を確保するために、訓練ごとに異なる ϵ 値を設定するのは有効な選択肢である. 訓練では、4 種類の異なる ϵ 値（0.01、0.05、0.1、0.25）を使用した.NVIDIA RTX 3090 GPU 上で 3 日間の大規模訓練を行い、4 つの landlord モデル、4 つの landlord_up モデル、4 つの landlord_down モデルを生成した. これらのモデルは、それぞれ異なる ϵ 値の下での戦略の特徴を示しており、今後の遺伝的アルゴリズムの最適化に有用な基盤を提供する.

例えば、$\epsilon = 0.01$ のモデルは DMC の性能を最大限に発揮し、訓練中に学習済みの戦略をより多く活用する傾向がある. その結果、戦略の安定性が向上し、精度の高い意思決定が可能となる. しかし、訓練時間が限られているため、戦略の改善には十分な時間をかけられず、特定の複雑な局面に対応する柔軟性が不足する可能性がある. 小さい ϵ 値は戦略の安定性を高める一方で、探索の幅を制限するため、未知の環境への適応力が低下するリスクがある.

一方、ϵ=0.25 のモデルは、より多くの探索を行い、多様なアクションや戦略を試す傾向がある. その結果、より広範な戦略を学習でき、複雑な環境への適応力が向上する. しかし、過剰な探索は訓練の初期に戦略の安定性を損なうリスクを伴い、モデルの収束速度が遅くなる可能性もある. また、探索が多すぎると、学習した戦略の最適化が遅れ、最終的な性能に悪影響を及ぼすこともある.

ϵ=0.05 や ϵ=0.1 のモデルは、安定性と探索のバランスを取ることができる.ϵ=0.05 のモデルは適度な探索能力を維持しつつ、安定した戦略出力を実現できる.ϵ=0.1 のモデルは適度な探索を行い、戦略の多様性を向上させつつ、収束の過程で過剰なランダム性や変動を抑えることができる. これらのモデルは、遺伝的アルゴリズムにとって多様性のある有望な集団を提供し、局所最適解への収束を回避するのに役立つ.

同じ ϵ で訓練されたモデルでは 4 種類の戦略特性を両立できないが、異なる ϵ で訓練したモデルに遺伝的アルゴリズムを適用することで、それぞれの戦略の長所を組み合わせ、全体の性能を向上させることができる. 交叉と突然変異の操作を通じて、異なる ϵ 値のモデルを組み合わせることで、多様性を維持しながら、戦略の安定性と柔軟性を最適化し、アルゴリズムをグローバル最適解へと導くことができる.

初代親集団の獲得方法が決まったので、次はこの集団に遺伝的アルゴリズムを適用して最適化を行う. この段階では、交叉や突然変異といった遺伝的操作を設計し、適応値の評価と選択戦略を策定することで、モデルが世代を重ねるごとに最適化され、全体の性能が向上するようにする.

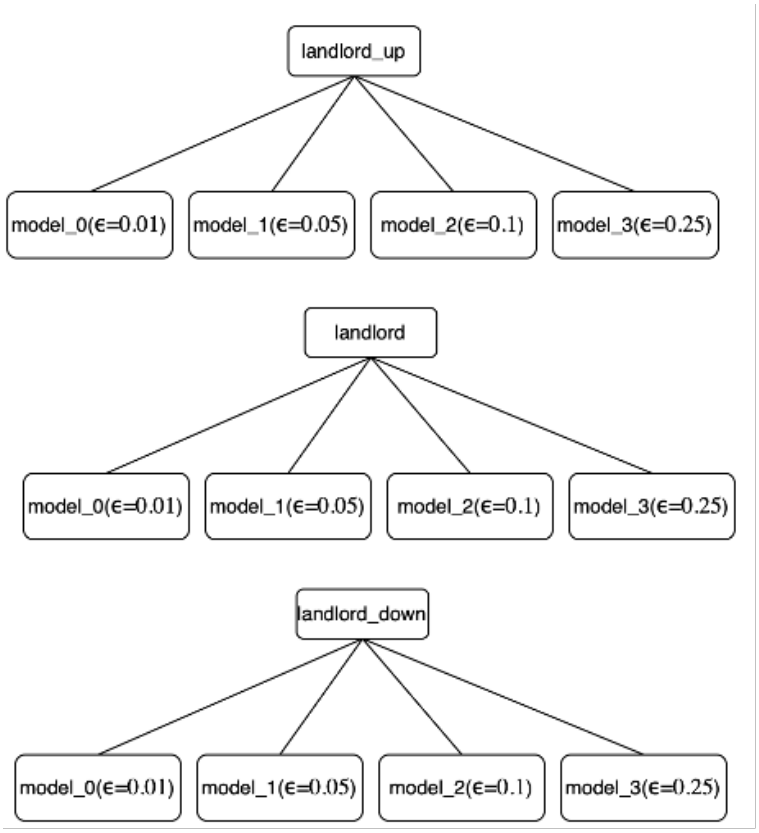

Figure 2: 初代親集団の構成

4.1.2 適応値計算

遺伝的アルゴリズムの最適化過程において、適応値（Fitness）は個体モデルの品質を測る重要な指標であり、次世代へ進むモデルの選択や交叉・突然変異の対象となるモデルを決定する. 本実験では、固定されたモデル（DouZero_30）との対戦時の勝率を適応値として採用し、評価の安定性と比較可能性を確保した.

例えば、landlord 位置の model_0 の適応値を評価する場合、landlord_up と landlord_down の位置には DouZero_30 の事前学習済みモデルを固定し、model_0 を landlord として対戦を行う. そして、1 万回の対局を実施し、その勝率を model_0 の landlord 位置での適応値とする. 同様に、landlord 位置の他のモデル（model_1, model_2, model_3...）の適応値を計算する場合も、landlord_up と landlord_down を DouZero_30 に固定し、landlord のみを変更して評価を行う.

また、landlord_up と landlord_down の適応値の計算方法も同じである.

landlord_up の適応値を計算する際は、landlord と landlord_down を DouZero_30 に固定し、landlord_up の勝率を統計する. landlord_down の適応値を計算する際は、landlord と landlord_up を DouZero_30 に固定し、landlord_down の勝率を統計する. なお、闘地主（DouDiZhu）のルールでは、二人の農民（landlord_up と landlord_down）が協力して地主（landlord）と対戦するため、適応値の評価時には以下の関係が成立する.

landlord_up の評価では、「landlord_up モデル＋ DouZero_30（landlord_down）」が DouZero_30（landlord）と対戦する. landlord_down の評価では、「landlord_down モデル＋ DouZero_30（landlord_up）」が DouZero_30（landlord）と対戦する. このようにして、各ポジションの適応値を一貫した基準で測定し、遺伝的アルゴリズムの最適化に利用する.

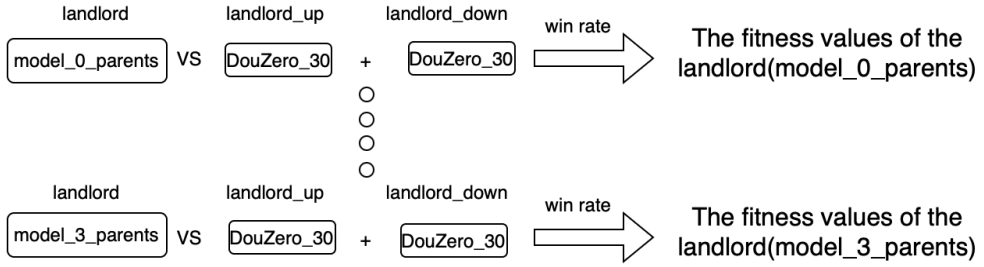

Figure 3: 親集団 landlord モデルの適応値計算

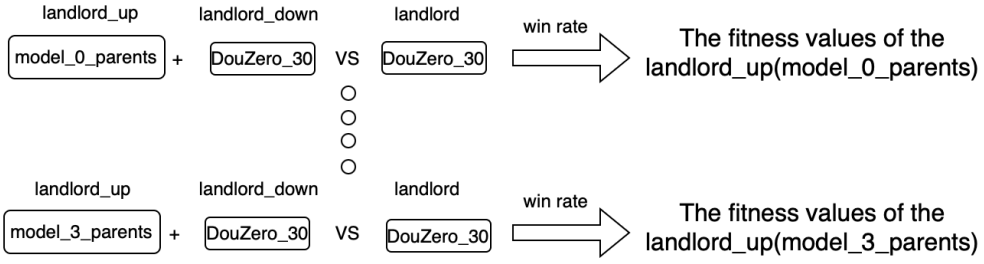

Figure 4: 親集団 landlord_up モデルの適応値計算

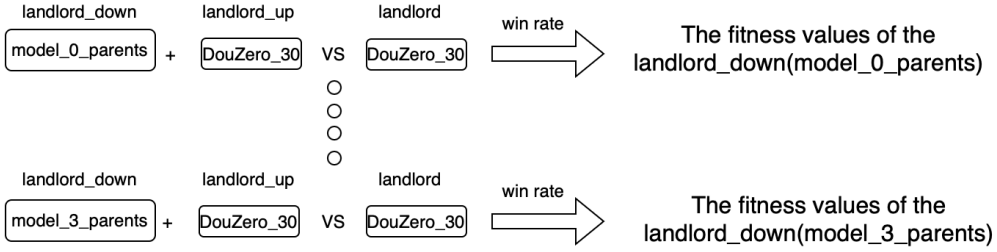

Figure 5: 親集団 landlord_down モデルの適応値計算

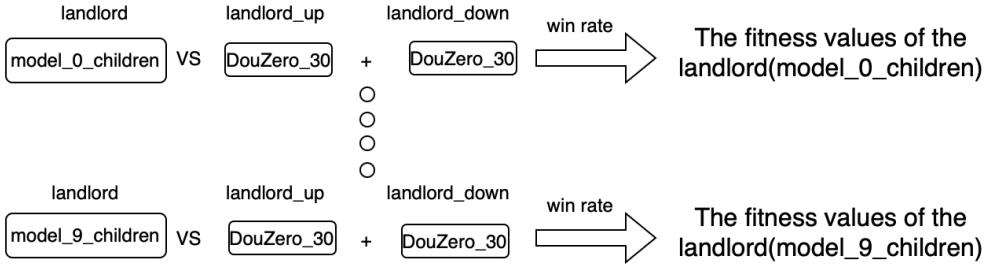

Figure 6: 子代集団 landlord モデルの適応値計算

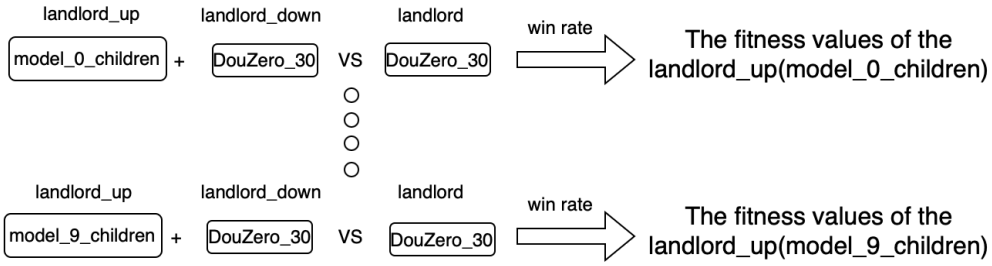

Figure 7: 子代集団 landlord_up モデルの適応値計算

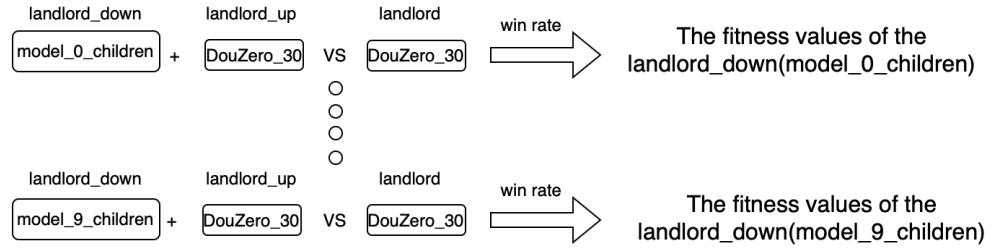

Figure 8: 子代集団 landlord_down モデルの適応値計算

4.1.3 交叉と突然変異

遺伝的アルゴリズムがニューラルネットワークを最適化する過程では、交叉（Crossover）操作は親個体から新たな子個体を生成し、異なる個体の優れた戦略を組み合わせることで、より良い解を探索する目的で使用される. 本実験では、親集団からランダムに 2 つのモデルを選択し、それらの全パラメータを走査して、各パラメータについて 50% の確率で一方のモデルの値を新しいモデルに割り当てる.

ニューラルネットワークの一部のパラメータはセットで使用されるため、交叉の過程では同じノードに関連するすべてのパラメータを一緒に交叉する必要がある. 例えば、layer3.5.bn1 という層には bias、running_var、running_mean など一連のパラメータがあり、これらは Batch Normalization（バッチ正規化）の計算に関与しているため、独立に交換してしまうとネットワークの安定性が崩れ、学習時に数値が異常になる可能性がある. そのため、交叉を実行する際には、すべてのパラメータを走査するだけでなく、パラメータの所属する層構造に従ってグループ単位で処理を行い、同じ層に属するすべての関連パラメータが同じ親から継承されるようにすることで、ネットワーク内部の整合性を保つ.

この改良された交叉戦略は、遺伝的多様性を確保しつつ、ニューラルネットワークの構造の安定性を最大限維持し、パラメータの不整合による学習の不安定化やモデルの性能低下を防ぐことができる.

変異（Mutation）の方法も同様にすべてのパラメータを走査する. 変異は元のパラメータ値を基に変更を加えるため、交叉のようにニューラルネットワークのパラメータ構造を変えてしまうことはない. そのため、同じ層のすべてのパラメータを統一して変異させる必要はなく、個々のパラメータ単位で変異を適用できる. 言い換えれば、変異操作は各パラメータに独立に作用し、特定の層のすべてのパラメータを同じように変異させる必要はない. これにより、変異の自由度が高まり、ネットワークの全体構造と既存の戦略の安定性を保ちながら最適化を行うことができる.

具体的な実行方法としては、各パラメータが 20% の確率で変異し、変異幅は 10% である. つまり、変異後のパラメータは元のパラメータの 0.9 倍から 1.1 倍の範囲でランダムに変化する.

4.1.4 淘汰

遺伝的アルゴリズムの最適化過程において、適応値の順位付けと選別は、どの個体が次世代に進み、進化を続けるかを直接決定する. 種群の継続的な最適化を確保するために、各世代で個体の適応値を順位付けし、低適応値の個体を淘汰し、最も優れた個体のみを新たな親集団として保持する必要がある.

遺伝的アルゴリズムの各世代の反復において、子代種群の適応値が親集団より低い可能性があるため、個体を選別する際には、新たに生成された子代個体のみを考慮するのではなく、親個体と子代個体を統一して順位付けし、選抜することで、選ばれた個体が常に現在の最適なものであることを保証する. 具体的には、各世代には合計 14 個体が含まれ、そのうち 4 個体は親集団のモデルであり、10 個体は交叉と突然変異によって生成された子代モデルである. 次世代の親集団を選択するために、この 14 個体の適応値を順位付けし、最も適応値の高い 4 個体を次世代の親個体として選抜する.

特に注意すべき点は、各世代の適応値の計算は、新たな 1 万回の対局データに基づいて統計を行うことであり、前世代の結果をそのまま使用しないという点である. これは、遺伝的アルゴリズムの適応値計算が環境の動的な変化に依存しているためである. 例えば、各ラウンドの対局で使用される手札情報や対戦相手の戦略などは異なるため、前世代で子代として適応値を計算済みの個体であっても、次世代で親個体として選抜された場合には、改めて適応値を再計算する必要がある. このようにすることで、環境の変化による適応値の歪みを回避し、モデルが新たな対局環境でも依然として高い競争力を維持できるようにする.

この選抜戦略を通じて、遺伝的アルゴリズムは各世代において最も性能の高い個体を選別し続け、新たに生成された子代個体の適応値が低い場合でも、種群全体の性能低下を防ぐことができる. 世代数が増えるにつれて、種群全体の適応値は徐々に向上し、最適化された個体はさまざまな対戦環境でより強力な競争力を発揮し、最終的なモデルの全体的な性能と汎化能力を向上させることが期待される.

4.1.5 モデルの命名

実験中の異なるモデルを管理しやすくするため、モデルの命名規則を標準化し、異なる実験条件や訓練プロセスを明確に区別できるようにした. 以下は、本実験で使用する主要なモデルの種類とその命名ルールである.

- DouZero3_0-3：初代親集団に含まれる、3 日間訓練された DouZero モデル.
- model_0-3_parents：親集団のモデル.
- model_0-9_children：子集団のモデル.
- DouZero_30：論文 [4] の GitHub に掲載されている、十分に学習された DouZero モデル.
- model_GA_0：最終世代の適応値が最も高いモデル

4.2 高性能の DMC モデルへの GA の適用

4.1では、遺伝的アルゴリズムが低性能モデルに与える影響を評価する実験方法について紹介した. 本節では、遺伝的アルゴリズムが高性能モデルに与える影響を探る. 低性能モデルとは異なり、高性能モデルは通常、訓練の過程で既に優れた戦略を獲得しており、特定の環境下で強い対戦能力を発揮する. そのため、遺伝的アルゴリズムが高性能モデルに与える影響は異なる可能性があり、最適化の目的もそれに応じて調整する必要がある.

高性能な DouZero の学習には長時間が必要であり、ローカルのハードウェアリソースには限りがあるため、本研究ではローカルでの長時間訓練を行わず、論文の GitHub に掲載されている既に学習済みの 3 つの DouZero モデル（それぞれ landlord、landlord_up、landlord_down に対応）を使用した. これらのモデルは十分に学習されており、各ポジションにおける最適な戦略を持っている. これらのモデルを活用することで、DMC が DouDiZhu（闘地主）の戦略学習にどのように貢献しているかを確認できる.

遺伝的アルゴリズムの初代親集団を構築する際、本研究ではモデルのコピーによる拡張を行い、3 つの高性能モデルをそれぞれ 4 つずつ複製した. これにより、合計 12 個の個体（4 つの landlord、4 つの landlord_up、4 つの landlord_down）からなる初代親集団を形成した. この方法により、初代親集団の品質を確保し、最初から競争力の高い個体で構成することができるが、一方で多様性の不足という課題がある.

交叉、変異、淘汰、および適応値計算の方法は 4.1 と同様である.

5 闘地主における DMC への GA 適用の実験結果

1 枚の RTX 3090 GPU 上で、親集団の数を 4、子集団の数を 10 の設定で、landlord、landlord_up、landlord_down を並列化して同時に訓練した.20 世代の訓練に合計 26 時間を要し、評価時間は 20 分であった.

5.1 低性能の DMC モデルの勝率

5.1.1 評価方法

最終的に訓練された第 20 世代の親集団から適応値（Fitness）が最も高いモデルを選択し、これを model_GA_0 と命名し、遺伝的アルゴリズムによって進化したモデルとして評価を行う. 評価対象は、このモデルと初代親集団に属する 4 つのモデルの性能の変化である.

遺伝訓練の過程では、各世代のモデルの適応値は DouZero_30 との 10,000 試合の対戦結果を基に算出される. 対戦の際、テスト対象のモデルが配置されるポジション以外の 2 つのポジションは DouZero_30 のモデルで固定される. これにより、遺伝的アルゴリズムの進化の方向は、DouZero_30 に対する勝率向上を目的とする最適化となる.

例えば、最適化対象が landlord（地主）の場合、その適応値は DouZero_30 の landlord_up と landlord_down を相手にした対戦結果に基づいて決定される. 勝率が高いほど適応値が高くなり、進化の方向は landlord モデルが DouZero_30 の landlord_up および landlord_down に対して競争力を高めることになる.

一方、最適化対象が landlord_up または landlord_down（農民）の場合、その適応値の計算方法は異なる. 適応値は、DouZero_30 の landlord_down（または landlord_up）と協力し、DouZero_30 の landlord を相手にした勝率を基に算出される. つまり、遺伝的アルゴリズムの最適化の方向は、単なる個体の戦略向上にとどまらず、チームワーク（協力プレイ）の強化も含まれることになる.

闘地主（DouDiZhu）は非対称な三人対戦型ゲームであり、landlord は単独で 2 人の農民（landlord_up と landlord_down）を相手に戦う一方、農民側は協力して landlord を打ち負かす必要がある. そのため、landlord_up または landlord_down を最適化する場合、その適応値はチームメイト（もう 1 人の農民）とどの程度連携できるかに依存する. 遺伝的アルゴリズムが個体戦略だけでなく、協力戦略の進化にも寄与することを確認することが重要である.

landlord、landlord_up、landlord_down の進化方向が異なるため、性能評価時には、遺伝的アルゴリズムによる最適化の効果を正確に反映できる方法を採用する必要がある. そのため、評価プロセスでは適応値計算時と同じ固定された対戦相手（または協力相手）として DouZero_30 を使用し、外部要因による実験結果のばらつきを排除する. また、対戦環境を変更（新たに 10,000 試合分のデータをリセット）し、異なるゲーム状況下でのモデルの適応能力をテストすることで、遺伝的アルゴリズムによる進化の有効性を多角的に検証する.

5.1.2 結果

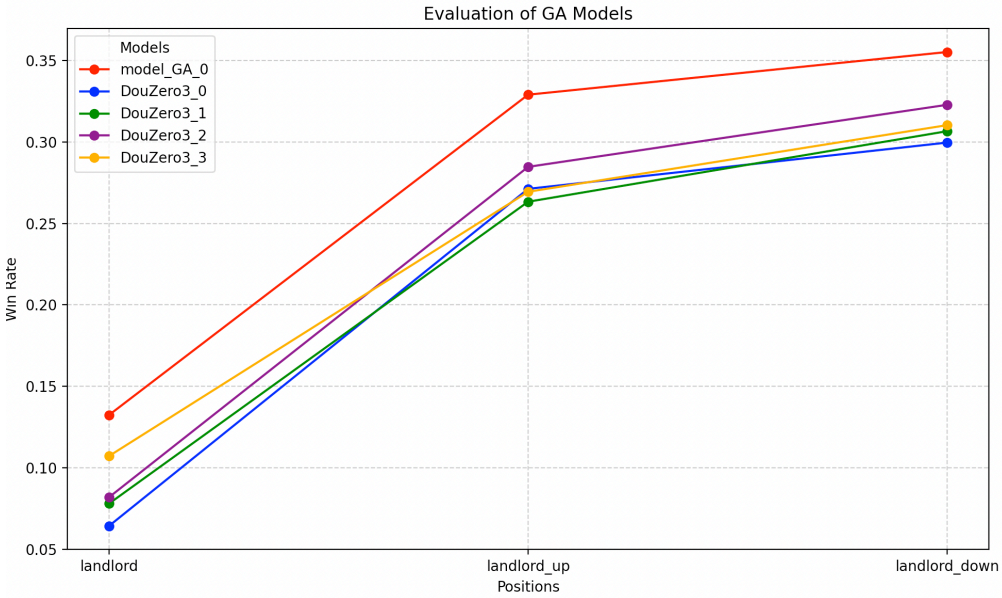

Figure 9: 各位置における遺伝的アルゴリズムモデル (model_GA_0) と DMC モデル (DouZero3_0-3) の勝率比較; landlord モデルが landlord_up(DouZero_30) と landlord_down(DouZero_30) に対して戦う際の勝率、landlord_up モデルが landlord_down(DouZero_30) と協力して landlord(DouZero_30) に対して戦う際の勝率、landlord_down モデルが landlord_up(DouZero_30) と協力して landlord(DouZero_30) に対して戦う際の勝率.

実験結果から見ると、model_GA_0 は初期の親集団に属する DouZero3_0-3 の 4 つのモデルと比較して、すべてのポジションで勝率が大幅に向上しており、遺伝的アルゴリズムがモデルの対戦能力を最適化する上で良好な効果を発揮したことが示された.

landlord ポジションでは、勝率の向上は遺伝的アルゴリズムが単独で 2 人の協力相手と戦う能力を効果的に強化したことを示しており、landlord_up（DouZero_30）と landlord_down（DouZero_30）による包囲戦に対して、より適切な出牌戦略を立て、自身の勝利確率を最大化できるようになった. この向上は、遺伝的アルゴリズムがより攻撃的な戦略を選択した可能性や、より効率的な手札管理方法を獲得したこと、またはより精密なリスク管理を強化したことに起因する可能性がある. その結果、landlord はさまざまな局面でより合理的な判断ができるようになり、勝率が向上したと考えられる.

同時に、landlord_up および landlord_down の勝率の向上は、遺伝的アルゴリズムが個々の対戦能力だけでなく、農民同士の協力能力も強化したことを示しており、固定された DouZero_30 の landlord と戦う際に、より優れたパフォーマンスを発揮できるようになったことがわかる. 闘地主は非対称なゲームであり、landlord は単独で 2 人の農民と戦う必要がある一方で、農民側は協力することで勝利を目指す. そのため、遺伝的アルゴリズムの進化の過程で、landlord_up と landlord_down の協力戦略が強化された可能性がある. 例えば、より合理的な譲り合い（パス）、積極的なカードの確保、または重要な局面でどのように味方と連携して landlord を封じるかなどの戦略が最適化されたと考えられる. これにより、個々のモデルの勝率が向上しただけでなく、遺伝的アルゴリズムが集団協力戦略の最適化にも有効であることがさらに検証された.

さらに、この最適化プロセスは、遺伝的アルゴリズムが異なる役割に適応する能力を持つことを示している.landlord ポジションのモデルに対しては、進化の方向が個々の単独対戦能力を強化する傾向があり、2 人の協力相手と対戦する能力を向上させることに重点が置かれた. 一方で、landlord_up および landlord_down ポジションのモデルに対しては、チームメイトとの協力を最適化し、共同で landlord を打ち負かす能力を強化することが主な目的となった. このように、異なる役割に応じた最適化が行われたこ

とは、遺伝的アルゴリズムが多エージェント対戦環境において適応性と柔軟性を備えていることを十分に示している.

5.2 高性能の DMC モデルの勝率

5.2.1 評価方法

評価方法は 7.1.1 と同じだが、今回の実験では初期集団が DouZero_30 をコピーした完全に同一の 4 つのモデルで構成されているため、評価時には遺伝的アルゴリズムモデル model_GA_0 と DouZero_30 の 2 つのモデルのみを比較すればよい.

5.2.2 結果

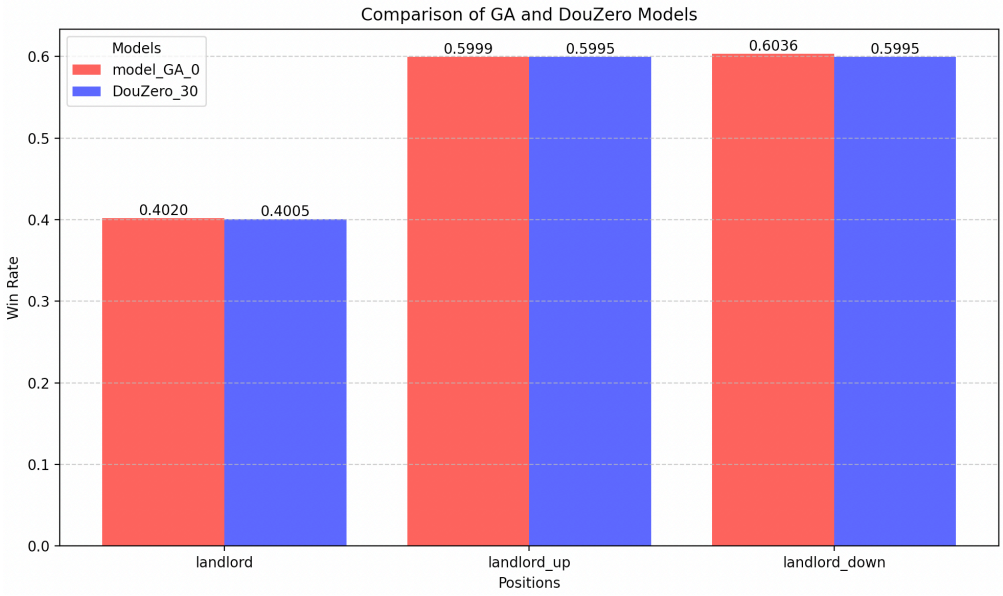

Figure 10: 各ポジションにおける遺伝的アルゴリズムモデル（model_GA_0）と DMC モデル（DouZero_30）の勝率比較；landlord モデルが landlord_up（DouZero_30）と landlord_down（DouZero_30）を相手にしたときの勝率、landlord_up モデルと landlord_down（DouZero_30）が協力して landlord（DouZero_-30）を相手にしたときの勝率、landlord_down モデルと landlord_up（DouZero_30）が協力して landlord（DouZero_30）を相手にしたときの勝率.

まず、7.1 節で使用した低性能モデルと比較すると、本節で採用した高性能モデルはすでに長時間の強化学習訓練を経ており、戦略が成熟しているため、最適化の余地が限られている.7.1 節の実験では、初期集団が低性能な DMC モデルを基にしていたため、遺伝的アルゴリズムの進化過程において戦略の大幅な改良が可能であり、最適化の効果がより顕著に表れた. 一方、本節の実験では、遺伝的アルゴリズムの最適化方向は主に高性能モデルの微調整に焦点を当てており、低レベルの戦略から大きく改善するのではなく、すでに高度に最適化されたモデルに対する微細な調整を行う形となる. そのため、性能向上の幅は自然と制限されることになる.

次に、本節の実験における初代親集団の多様性が限られていたことも、遺伝的アルゴリズムの最適化効果に影響を与えた可能性がある.7.1 節の実験では、初期集団の個体が異なる ϵ 値を用いて訓練された DMC モデルであり、多様な戦略を持つ個体が含まれていた. そのため、遺伝的アルゴリズムの進化過程において、DMC アルゴリズムの潜在的な可能性を引き出し、環境に適応した優れた戦略を選択することができた. しかし、本節の実験では、単一の高性能モデルを初代親集団として採用し、すべての個体が基本的に同じ構造と訓練方法で生成された. このため、遺伝的アルゴリズムの進化は、既存の最適化空間内で

の局所的な微調整にとどまり、新たな戦略の探索というよりも、既存モデルのポテンシャルを引き出すことに重点が置かれた. このような最適化方式では、根本的な戦略の進化ではなく、すでに完成度の高い戦略の調整にとどまるため、最終的な性能向上の幅も相対的に小さくなる.

6 DQN への遺伝的アルゴリズムの適用

6.1 初期親集団の選択

Leduc Hold'em モデルを DQN で訓練する際、ランダム戦略の対戦相手（Random Agent）を用いた学習方法を採用した. 具体的には、各訓練過程でプレイヤー 1（Player 1）を DQN エージェント（DQN Agent）とし、プレイヤー 2（Player 2）をランダムに行動を選択するモデル（Random Agent）に設定した.

訓練の多様性を確保し、異なるランダム初期化がモデルの収束に与える影響を観察するために、乱数シード（random seed）を変更しながら実験を行った. 具体的には、シード値 0 から 15 を順に適用し、それぞれのシードで学習を実行することで、合計 16 個の異なる DQN モデルを生成した.

DQN による Leduc Hold'em モデルの訓練において、乱数シードは重要な役割を果たす. これは主に、モデルの初期化、環境内のランダムイベント、および学習過程での確率的な選択に影響を与えるためである.

ニューラルネットワークの初期重みは通常ランダムに生成されるため、異なる初期化を適用すると、モデルが学習する戦略にも違いが生じる可能性がある. そのため、乱数シードを固定することで、訓練ごとに重みの分布を一定に保ち、初期化の違いによる学習の不安定性を低減することができる.

また、Leduc Hold'em のようなポーカー環境では、カードの配布、対戦順序などがランダムに決定される. 乱数シードを固定しない場合、訓練ごとにゲーム環境の初期状態が異なり、モデルが異なる戦略を学習してしまう可能性がある. そこで、乱数シードを設定することで、訓練プロセスの再現性を確保し、異なる訓練条件においても一貫した比較が可能となる.

本実験では、シード値（0〜15）を変更しながら 16 個の異なる DQN モデルを訓練し、それぞれのモデルが同じ訓練フレームワークの下で異なる初期化と学習過程を経験するようにした. これにより、異なるランダム初期化が DQN の学習結果に与える影響を分析できるだけでなく、進化アルゴリズム（遺伝的アルゴリズムなど）における多様な初期集団の形成にも活用できる. これにより、最適化の可能性をより高めることができる.

6.2 適応値計算

初始父母种群の 16 個の DQN モデルは、ランダムアクションモデルとの対戦を通じて訓練されたため、これらの DQN モデルは自然にランダムアクションモデルに対して高い勝率を示す. これは、訓練中に DQN が主にランダム戦略に対する最適な対応方法を学習したためであり、より複雑または知的な対戦相手の戦略を学習したわけではない. 本実験では、適応値の計算基準として、ランダムアクションモデルとの対戦における報酬（利益）を指標として使用する.

報酬の計算方法は、各ゲーム終了時のプレイヤーのチップ変動量（獲得チップ - ベットしたチップ）を計算し、全対局の純利益の総和をゲーム総数とビッグブラインド（Big Blind）で正規化するものである.

$$\text{Reward} = \frac{\sum_{i=1}^{N}(\text{プレイヤーの純利益}_i)}{N \times \text{Big Blind}} \tag{1}$$

ここで、N は対戦の総局数、Big Blind はゲームにおけるビッグブラインドの値である.

最適化の過程では、依然としてランダムな対戦相手に対するモデルのパフォーマンス向上を重視する傾向がある. この設定により、モデルがより複雑な環境での汎化能力を制限される可能性はあるが、一方で、遺伝的アルゴリズムが DQN の学習済み戦略をどこまで強化できるかをテストするのに適した環境となる. また、DQN がランダム戦略相手に対する勝率をどこまで向上させられるかを評価することも可能である.

この最適化フレームワークのもとで、遺伝的アルゴリズムは DQN の既存の戦略構造を微調整し、訓練中に獲得した有効な戦略を強化することに重点を置く. これにより、モデルはランダムな対戦相手に対

してより安定した戦略を維持し、戦略の不安定性や探索不足による予期せぬ敗北を減少させることが期待される.

この手法を通じて、遺伝的アルゴリズムがモデルの最適化にどの程度寄与できるかを検証する. つまり、DQN のコア戦略ロジックを変えずに、交叉と変異によって個体のパフォーマンスをさらに向上させることが可能かどうかを評価する. また、遺伝的アルゴリズムの集団進化メカニズムによって、個体間の競争が促進され、ランダム戦略相手に対して最も適応した個体が選別されることで、より堅牢なランダム戦略対策モデル群が形成される可能性がある.

6.3 交叉と突然変異

交叉と変異の方法は 4.1.3 と同じである.

6.4 淘汰

淘汰方法は、各世代の 16 個の親集団と 30 個の子集団、計 46 個のモデルの中から、適応度が最も高い 16 個のモデルを次世代の親集団として選択する方式である.

6.5 モデルの命名

- DQN_0-15：乱数シード 0-15 を使用して訓練された 16 個の DQN モデル.
- model_0-15_parents：親集団に属する 16 個のモデル.
- model_0-29_children：子集団に属する 30 個のモデル.
- Random：ランダム戦略モデル.
- GA：訓練の最終世代において、親集団の中から適応値が最も高いモデルを選出し、最終的な遺伝的アルゴリズムによるモデル（GA）として確定する.
- GA_1-3：初代の親集団、交叉・変異・淘汰の方法、および適応度の計算方法を同一に保った状態で、遺伝的アルゴリズムを 3 回実行し、それぞれの実験で最終的に選択されたモデル（GA）は、接尾辞を付けることでどの実験のモデルであるかを示す

7 ブラックジャックにおける DQN の実験結果

CPU 上で、親集団の数を 16、子集団の数を 30 の設定で 20 世代を訓練した結果、合計約 1 時間を要し、評価時間は約 4 時間であった.

7.1 評価方法

DQN の遺伝的アルゴリズム最適化の過程において、適応値の計算方法は、ランダム戦略モデル（Random）との 10,000 試合の対戦における平均報酬を基準とする. さらに、各世代の訓練時には、100 から 10,000 の範囲内でランダムに選択されたシード値を使用することで、進化過程においてモデルが異なる対局環境に適応できるようにしている. したがって、最適化の評価を行う際にも、訓練フェーズと同じ方法を採用し、公平性と結果の比較可能性を確保する.

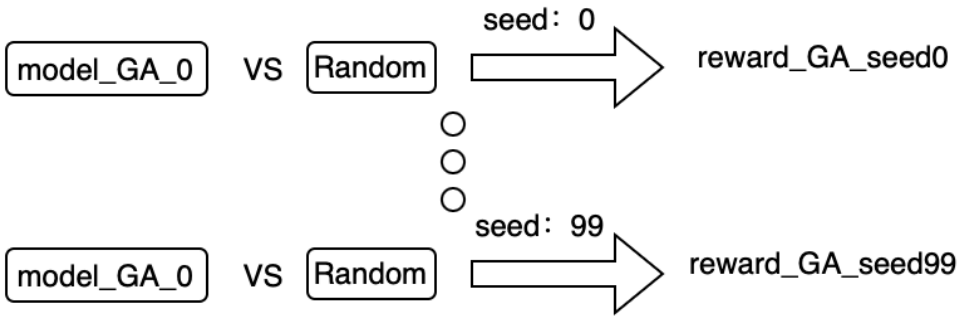

Figure 11: model_GA_0 の 0〜99 の乱数シードにおける対局報酬を計算する

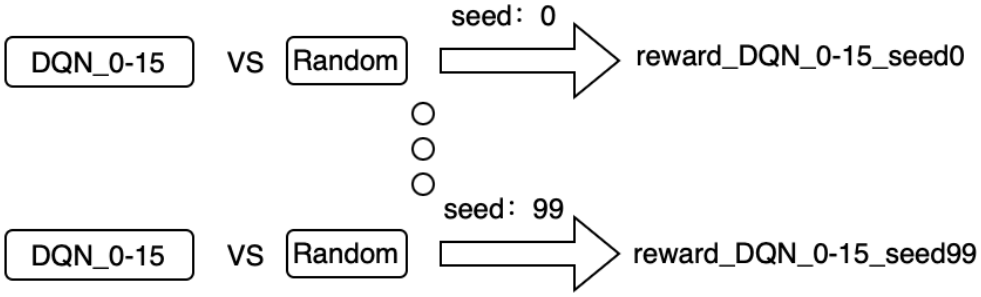

Figure 12: DQN_0-15 の 0〜99 の乱数シードにおける対局報酬を計算する

具体的には、model_GA_0 と初代の親集団に含まれる DQN_0 から DQN_15 を、それぞれ Random 戦略と対戦させる. 乱数シード 0-99 を使用し、異なる 100 個の環境で対戦を行い、それぞれの乱数シードごとに 1 万試合を実施する. そして、異なる乱数シード下で各モデルが Random 戦略と対戦した際の報酬を記録する.

model_GA_0 と DQN_0 のモデル性能を評価する際に、遺伝的アルゴリズムによる最適化後のモデルの改善状況を総合的に測るため、2 つの指標を計算する.

まず、同じ乱数シード（0-99）の下で、model_GA_0 と DQN_0 をそれぞれ Random 戦略モデルと 1 万試合ずつ対戦させ、その結果 reward_GA の値が reward_DQN_0 を上回った乱数シードの数を数える. この指標は、model_GA_0 がより多くの環境で優位性を持っているかどうかを測るものであり、異なる対戦条件の下で遺伝的アルゴリズムで最適化されたモデルが DQN_0 より競争力を持っているかどうかを示す.

次に、すべての乱数シード（0-99）における reward_GA の合計と reward_DQN_0 の合計を計算し、全体的に model_GA_0 が DQN_0 を累積報酬の面で上回っているかを評価する. すべてのシードの報酬を合計することで、遺伝的アルゴリズムが特定のシードだけでなく、より広範囲な対戦環境においてモデルの長期的な収益を向上させているかどうかを測ることができる.DQN_1 から DQN_15 の評価プロセスも、上記の方法と同様に行う.

7.2 結果

初代の親集団を同じに保った上で、同じ交叉、突然変異、淘汰の方法を用いて、3 回の遺伝的アルゴリズム実験を実施した. そして、それぞれの実験で最後に得られた model_GA_0 を、同じ評価方法を用いて評価した.

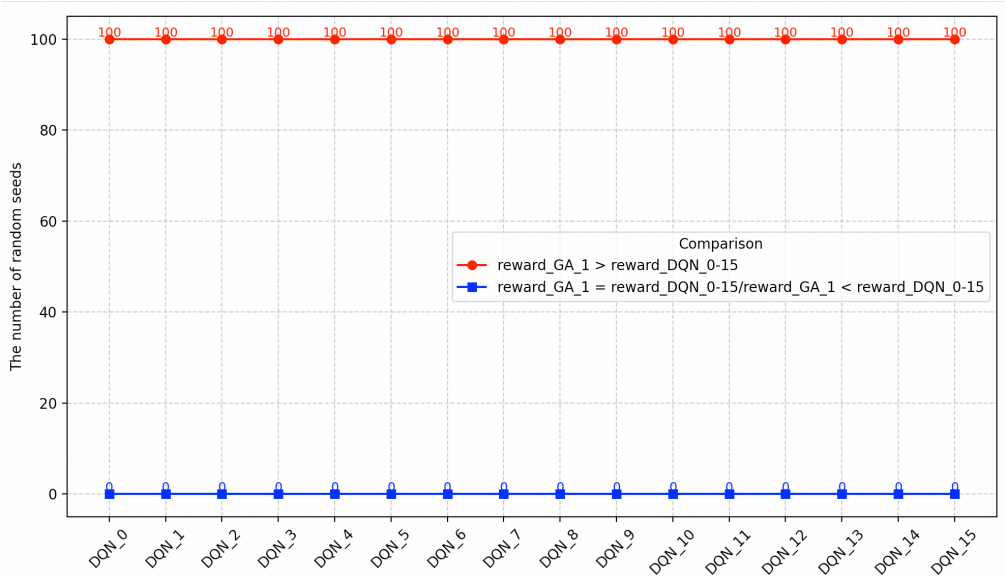

Figure 13: 第 1 回の実験において、乱数シード 0-99 の 100 回の対戦（各対戦は 1 万局）で、”GA_-1” 対”Random” の報酬は reward_GA_1、”DQN_0-15” 対”Random” の報酬は reward_DQN_0-15 である.reward_GA_1 と reward_DQN_0-15 の大小関係を比較し、横軸には比較対象の DQN モデル、縦軸には乱数シードの数を示す

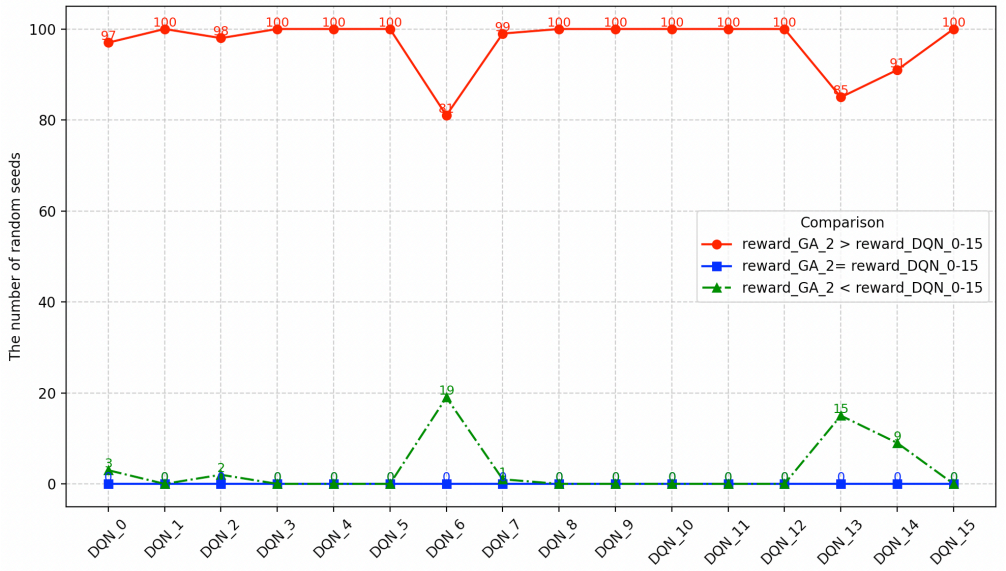

Figure 14: 第 2 回の実験において、乱数シード 0-99 の 100 回の対戦（各対戦は 1 万局）で、”GA_-2” 対”Random” の報酬は reward_GA_2、”DQN_0-15” 対”Random” の報酬は reward_DQN_0-15 である.reward_GA_2 と reward_DQN_0-15 の大小関係を比較し、横軸には比較対象の DQN モデル、縦軸には乱数シードの数を示す

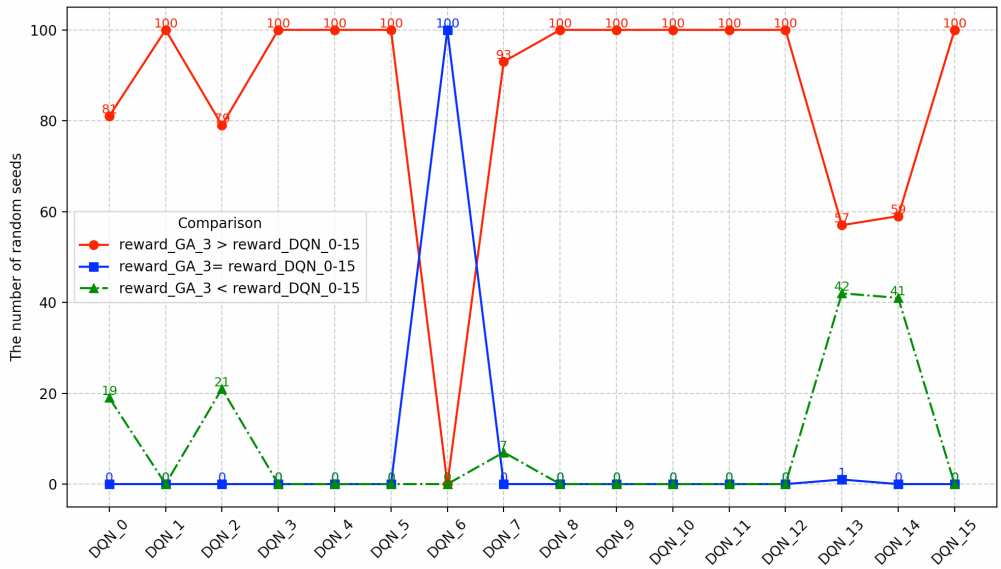

Figure 15: 第 3 回の実験において、乱数シード 0-99 の 100 回の対戦（各対戦は 1 万局）で、”GA_-3” 対”Random” の報酬は reward_GA_3、”DQN_0-15” 対”Random” の報酬は reward_DQN_0-15 である.reward_GA_3 と reward_DQN_0-15 の大小関係を比較し、横軸には比較対象の DQN モデル、縦軸には乱数シードの数を示す

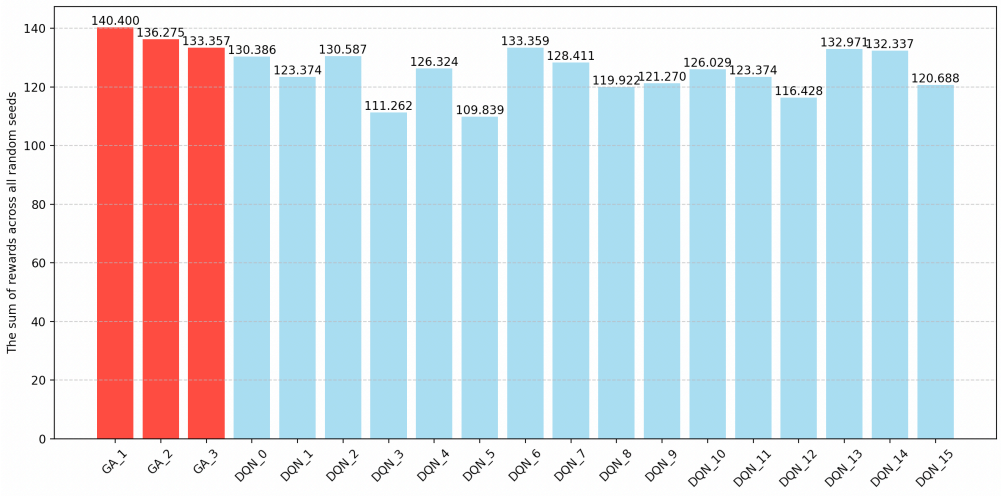

Figure 16: 三回の実験における GA_0-3 および DQN_0-15 の報酬総和は、乱数シード 0-99 の 100 回の対局結果に基づいて計算される. 対局環境（乱数シード 0-99）は一定であり、また DQN_0-15 も変わらず（DQN による初代親集団から得られたもの）、そのため DQN_0-15 の報酬総和は 3 回の実験を通じて一定である. 一方で、遺伝的アルゴリズムのランダム性により、各実験で得られる GA_0-3 のモデルは異なり、その結果 GA_0-3 の報酬総和も異なる.

3 回の実験結果には一定のばらつきが見られた. その中で、最初の実験が最も優れた最適化効果を示し、報酬が高い乱数シードの数、累積総報酬の両方において、model_GA_0 が DQN_0-15 を全面的に上回った. これにより、この実験環境下で遺伝的アルゴリズムが強力な最適化能力を持つことが確認された. 一方、3 回目の実験では最適化効果が相対的に低かったものの、それでも最終的に得られた model_GA_0 は DQN_6 と同等の性能に達した. これは、遺伝的アルゴリズムが最適化のパフォーマンスが低い場合で

も、ある程度の改善を維持し、大きな性能劣化を回避できることを示している.

実験結果にばらつきが生じた要因はいくつか考えられる. まず、各世代で使用する乱数シードの範囲が 100-1000 の間で変動するため、進化過程において各個体が経験する対戦環境に大きな違いが生じる可能性がある. その結果、一部の実験では適応値評価が特定の戦略に有利に働く一方で、他の実験では局所最適に陥る可能性がある. 次に、3 回の実験で交叉、変異、淘汰の手法は完全に同一だったものの、遺伝的アルゴリズムは本質的にランダム性に依存するため、ある実験ではより優れた遺伝子が初期段階で淘汰されてしまうことがあり、それが最終的な進化の成果に影響を与えた可能性がある.

全体的に見ると、実験結果にばらつきはあったものの、遺伝的アルゴリズムは複数の実験において安定した最適化能力を示した. 特に最も良好な実験では、model_GA_0 が DQN_0-15 を大きく上回る成果を達成し、この手法が特定の環境下で有効であることを証明した. また、最適化効果が最も低かった実験においても、最終結果が DQN_6 に匹敵する水準であったことから、遺伝的アルゴリズムには一定の安定性があり、大幅な性能低下を招かないことが確認された.

8 ResNet50 への遺伝的アルゴリズムの適用

8.1 画像データの取得

画像認識タスクにおいて、十分な数の画像は高性能な深層学習モデルを訓練する上で極めて重要である. 画像データの量はモデルの収束速度に影響を与えるだけでなく、モデルの汎化能力や最終的な分類精度を直接決定する. もし訓練データセットの画像が少なすぎると、モデルは過学習（overfitting）を引き起こしやすくなり、訓練セットでは良好な結果を示しても、テストセットでは性能が低下し、未見のデータに対する適用力が十分でなくなる. 一方で、十分な数の画像データを確保することで、モデルはより安定した特徴を学習し、多様な入力に対する適応力を向上させることができる.

実際の応用では、必要な画像数はタスクの複雑さ、クラス数、モデルの規模、データの品質といった要因によって決まる. 例えば、ResNet50 のような大規模な深層学習モデルの場合、データ量が不足すると、モデルは汎化能力の高い特徴を学習できず、安定した認識性能を得ることが難しくなるため、通常は数万枚以上の画像が必要となる. 本実験では CIFAR-10 データセットから 10 クラス分類タスク用のデータを抽出した.CIFAR-10 は画像分類研究に広く利用される標準的なデータセットであり、飛行機、自動車、鳥、猫、鹿、犬、カエル、馬、船、トラックの 10 クラスを含み、それぞれ 6,000 枚の 32×32 ピクセルのカラー画像で構成されている. データセット全体の画像数は 60,000 枚であり、そのうち 50,000 枚が訓練用、10,000 枚がテスト用に分割されている. 本実験では、モデルの学習と評価をより効果的に行うために、さらにデータセットを訓練セット（train）、検証セット（val）、テストセット（test）の 3 つに分割した.

訓練セットには各クラス 4,000 枚、合計 40,000 枚の画像を含み、主に初代親集団の個体の学習に使用され、遺伝的アルゴリズムの最適化過程において個体の適応値（Fitness）を計算するためにも利用される. このデータセットは ResNet50 の初期モデルの学習を担当し、勾配降下法によりモデルパラメータを最適化しながら、異なるクラスの特徴を段階的に学習する. また、遺伝的アルゴリズムの世代交代の中で、適応値の計算にこのデータセットを利用することで、異なるモデルが分類タスクにおいてどのように性能を発揮しているかを評価し、次世代に進む個体を決定することで、進化の方向性を最適化することができる.

検証セットには各クラス 1,000 枚、合計 10,000 枚の画像を含み、初代親集団の訓練過程における中間評価のために使用される. 訓練過程において、検証セットの役割は、未学習データに対するモデルのパフォーマンスを監視し、過学習の兆候を検出することにある. さらに、学習率や正則化強度などのハイパーパラメータを調整するために利用される. モデルの検証セットにおけるパフォーマンスは、訓練戦略の調整の基準として機能し、例えばアーリーストッピング（Early Stopping）や学習率スケジューリング（Learning Rate Scheduling）の適用を判断することで、訓練プロセスを最適化し、最終的なモデルの安定性と汎化能力を向上させる.

テストセットも 10,000 枚の画像で構成され、各クラス 1,000 枚を含むが、訓練セットや検証セットとは異なる役割を持つ. テストセットのデータは訓練や適応値の計算には一切使用されず、遺伝的アルゴリズムによる最適化が完了した後の最終評価段階のみに使用される. 進化の最終世代において、適応値が最も高い個体を選出し、そのモデルをテストセット上で評価することで、遺伝的アルゴリズムによる最適化が最終的にどの程度の分類性能をもたらしたかを測定する. テストセットのデータは学習および遺伝進化の過程で一度も使用されていないため、モデルの性能評価において客観的で公正な指標となり、実際の応

用環境における汎化能力を正確に反映する.

本実験では、訓練セット、検証セット、テストセットの適切な分割がモデルの最適化において極めて重要な役割を果たす. 訓練セットはモデルの学習と適応値の計算に使用され、検証セットは学習中のハイパーパラメータ調整と汎化能力の評価に使用される. 最終的に、テストセットは独立した評価データとして、モデルの最適化が成功したかどうかを測定する. このデータ分割戦略により、遺伝的アルゴリズムの最適化が合理的に進行し、最終的に高い性能と汎化能力を持つ画像分類モデルを進化させることが可能となる.

8.2 初期親集団の選択

ResNet50 の訓練における学習率の影響は、主に収束速度、安定性、および最終的なモデルの性能に関係する. 学習率はパラメータの更新ステップの大きさを決定する重要なハイパーパラメータであり、その設定は最適化プロセスの効率と効果を直接左右する. 例えば、学習率が 0.0001 のように小さすぎる場合、モデルは安定して収束し、パラメータの急激な変動を抑えることができるが、その分収束速度が遅くなり、最適なパフォーマンスに到達するまでの訓練時間が長くなる. また、学習率が小さすぎると、訓練初期のパラメータ更新幅が十分でなく、局所最適解に陥りやすくなるため、モデルの性能向上が難しくなる. 一方で、学習率が 0.008 のように大きすぎる場合、訓練の初期収束は速くなるが、パラメータの更新幅が大きくなりすぎることで損失関数の値が大きく変動し、最悪の場合、勾配爆発を引き起こして収束しないリスクがある. さらに、高すぎる学習率では最適解付近でパラメータが振動し続け、精密な調整ができなくなるため、最終的なモデルの性能が低下する可能性がある.

実際の訓練では、適切な学習率を設定することで、収束速度と安定性のバランスを取ることが重要となる.ResNet50 の場合、一般的に使用される学習率の範囲は 0.001 から 0.005 の間であり、この範囲内では適度な収束速度を維持しながら、訓練の安定性を確保できる. 例えば、訓練の初期に比較的大きな学習率（0.005 など）を使用すると、モデルはより高速に適切な解を探索できる. 一方で、学習が進むにつれて学習率を減少させることで、パラメータの微調整を行い、最終的なモデルの精度を向上させることができる. また、深層学習では動的に学習率を調整する手法が広く用いられており、学習率の減衰（Learning Rate Decay）を導入することで、訓練後半の収束をスムーズにし、最適解を見逃すことを防ぐことができる. さらに、Warm-up 機構を使用すれば、訓練初期に低い学習率から始めることで、パラメータの急激な変動を防ぎ、より安定した訓練を実現できる.

本実験では、16 種類の異なる学習率（0.0001、0.0003、0.0005、0.0007、0.001、0.003、0.005、0.007、0.0002、0.0004、0.0006、0.0008、0.002、0.004、0.006、0.008）を選択し、それぞれの学習率で ResNet50 を訓練した. 訓練の評価には、訓練セット（train）と検証セット（val）を用い、最終的に 16 個の異なる初期モデルを得た. これらのモデルは遺伝的アルゴリズムの初代親集団として使用され、多様な初期個体を確保することで、最適化プロセスにおける探索範囲を広げ、進化の可能性を高めることを目的としている.

8.3 適応値計算

本実験では、モデルの訓練セットにおける識別精度を、遺伝的アルゴリズムの最適化過程における適応値（Fitness）として使用する. ここで注意すべき点は、DMC や DQN の遺伝的アルゴリズムによる訓練過程では、各世代において適応値を計算するためのデータが動的に変化することである. 例えば、DMC の訓練では、各世代ごとに新たに 10,000 試合分の対局データを生成し、それを適応値の計算に使用する. 一方、DQN の訓練では、各世代ごとに異なる乱数シードを使用することで、適応値評価時のデータ多様性を確保し、モデルが特定のデータパターンに過適応（overfitting）するのを防ぐ.

しかし、画像認識タスクにおいては、高品質な画像データの収集が難しく、適応値の計算には通常、大規模な画像データセットが必要となるため、各世代ごとに異なるデータを用いることは実際の運用上、大きな課題となる. 強化学習タスクでは、環境との相互作用を通じて常に新しい訓練データを生成できるが、画像分類タスクではデータは通常静的であり、データセットの規模にも制約がある. そのため、本実験における遺伝的アルゴリズムの最適化過程では、適応値の計算には固定された訓練データセットを使用する.

8.4 交叉と突然変異

交叉と変異の方法は 4.1 節で DMC に対して適用した遺伝的アルゴリズムの手法と類似しているが、子集団の個体数が 30 に変更され、交叉および変異の対象となるパラメータの種類が調整されている.

4.1 節では、交叉の方法として、モデルのすべてのパラメータを共通のプレフィックス名ごとにグループ化し、グループ単位で交叉を行う手法を採用した. この方法により、同じ層や相互に関連するパラメータが交叉の過程で一貫性を保ち、パラメータの不整合による不安定性を回避することができる.

ResNet50 の実験では、画像認識におけるニューラルネットワークの構造が強化学習のニューラルネットワークと異なり、主に「畳み込み層（Convolutional Layers)」と「全結合層（Fully Connected Layers, FC 層)」で構成されている. この構造上の違いにより、ニューラルネットワークは情報処理と特徴抽出において異なる役割を担う重要な部分に分かれるため、遺伝的アルゴリズムの最適化過程においても、それぞれの構造に適した処理を行う必要がある.

畳み込み層は ResNet50 の中核部分であり、入力画像から特徴を抽出する役割を果たす. モデルは多層の畳み込み演算を通じて、低レベルの特徴（エッジやテクスチャ）から高レベルの特徴（物体の形状やカテゴリ情報）まで、段階的に有効な情報を抽出する. 各畳み込み層は複数のフィルタ（Filters）で構成され、これらのフィルタが入力画像と畳み込み演算を行うことで局所的な特徴を捉え、それを次の層へと伝達する. さらに、各畳み込み層の後にはバッチ正規化（Batch Normalization, BN）および活性化関数（ReLU）が広く用いられており、学習の安定性を向上させ、収束を加速させる役割を担っている.

一方で、全結合層（FC 層）は ResNet50 の最後のステージに位置し、グローバル平均プーリング（Global Average Pooling, GAP）層からの特徴ベクトルを受け取り、最終的な分類出力へとマッピングする役割を担っている. 畳み込み層とは異なり、全結合層は高次元の特徴を統合し、重み行列を通じて異なるカテゴリの確率を算出する. 全結合層の重みは通常大量に存在するため、遺伝的アルゴリズムの最適化においては、これらの重みの調整方法を慎重に設計しなければ、計算資源の消費が過剰になるリスクがある.

異なる層が異なる役割を持つため、交叉と変異の対象となるパラメータの選択方法も調整した. 本実験では、以下の 4 つの方法を比較する. （1）すべてのパラメータを共通のプレフィックス名ごとにグループ化し、グループ単位で交叉を行い、さらにすべてのパラメータを個別に変異する. この方法は 4.1節の DMC の手法とほぼ同じであり、全体的な遺伝情報の組み合わせを考慮しながら、局所的な適応を図る. (2）全結合層のパラメータのみを共通のプレフィックス名ごとにグループ化し、グループ単位で交叉を行い、さらに全結合層のパラメータのみを個別に変異する. この方法では、全結合層に特化して進化を促進し、特徴抽出部分（畳み込み層）には影響を与えずに最適化を行う. （3）畳み込み層のパラメータのみを共通のプレフィックス名ごとにグループ化し、グループ単位で交叉を行い、さらに畳み込み層のパラメータのみを個別に変異する. この方法では、特徴抽出の強化を重視し、分類段階（全結合層）は変更せずに保持する. （4）畳み込み層以外のパラメータを共通のプレフィックス名ごとにグループ化し、グループ単位で交叉を行い、さらに畳み込み層以外のパラメータを個別に変異する. この方法は、畳み込み層の特徴抽出を固定しつつ、バッチ正規化層や全結合層などの学習済みパラメータの最適化に重点を置く.

この 4 種類の方法を比較することで、ResNet50 における遺伝的アルゴリズムの最適な適用方法を探り、異なる層構造に対する影響を定量的に評価する.

8.5 淘汰

淘汰方法は 4.1 節で DMC に適用した遺伝的アルゴリズムの手法と類似している.

8.6 モデルの命名

- ResNet50_0-15：乱数シード 0-15 を使用して訓練された 16 個の DQN モデル. 初代親集団とする.
- model_0-15_parents：親集団に属する 16 個のモデル.
- model_0-29_children：子集団に属する 30 個のモデル.
- GA：訓練の最終世代において、親集団の中から適応値が最も高いモデル.
- GA_1-4：同じ初代の親集団を保持し、交叉・変異・淘汰の方法は 8.4 で述べた異なる交叉方法を用いて 4 回実行し、得られた GA モデル.

9　ResNet50 への GA 適用の実験結果

1 枚の RTX 3090 GPU 上で、親集団の数を 16、子集団の数を 30 の設定で 20 世代を訓練した結果、2 日弱の時間を要し、評価時間は約 20 分であった.

9.1　評価方法

訓練されたモデル model_GA_0 と、初代親集団に含まれる 16 個の ResNet50 モデル（ResNet_0-15)、計 17 個のモデルをテストセット（test）上で評価し、識別精度（認識率）を測定する.

本実験の主な目的は、model_GA_0 の識別精度が ResNet_0-15 のすべての個体を上回るかどうかを検証し、遺伝的アルゴリズムが ResNet50 モデルの最適化に寄与しているかを評価することである.

また、8.4 で言及されている 4 種類の交叉方法に基づき、4 回の実験を実施する.

9.2　結果

4 種類の方法において、GA_0-15 のテストセット（test）上での識別精度は完全に同一である.

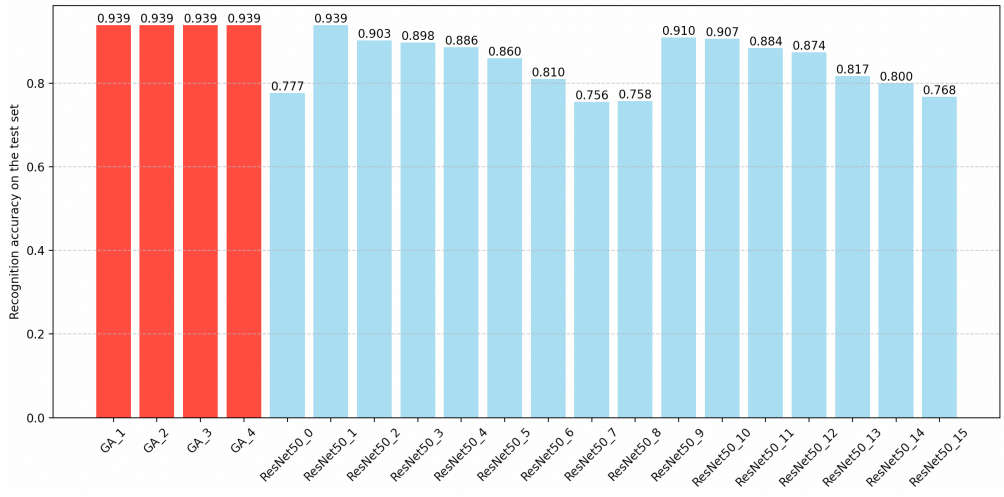

Figure 17: 8.4 で言及されている 4 種類の交叉方法を用いて得られた 4 つの遺伝的アルゴリズムモデル（GA_1-4）および初代親集団の ResNet50_0-15 のテストセット（test）上での識別精度（認識率）

4 つの遺伝的アルゴリズムによって生成されたモデル（GA_1-4）は、テストセット（test）上での識別精度がすべて 0.939 となり、初代親集団のモデル ResNet50_1 の精度と一致した. この結果から、テストセット（test）環境において、遺伝的アルゴリズムは ResNet50 の性能をさらに向上させることができなかったことが示された.

10　まとめと今後の課題

10.1　DMC

10.1.1　まとめ

本研究では、DMC に適用した遺伝的アルゴリズムの実験を通じて、多人数ゲーム環境における遺伝的アルゴリズムの実用性と最適化能力を証明した. 実験結果から、遺伝的アルゴリズムは単なる個体の戦略最適化だけでなく、協力関係の向上にも寄与することが示された. 具体的には、landlord の対戦戦略を最適化することで landlord_up や landlord_down に対してより効果的な対抗策を獲得できると同時に、landlord_up と landlord_down の協力戦略を向上させることで、全体の勝率を高めることができた. この

ことから、遺伝的アルゴリズムは三者ゲーム環境において、単体の競争力を強化するだけでなく、エージェント間の協力を促進し、より複雑な戦術の最適化を実現できることが確認された.

さらに、本研究は集団の多様性が遺伝的アルゴリズムの最適化効果に大きく影響を与えることを明らかにした. 異なる集団の構成方法を比較した結果、初期集団の多様性が高いほど、進化過程で探索できる戦略空間が広がり、より優れた戦略を見つけやすくなることが分かった. 一方、初期集団の構造が単一的である場合、遺伝最適化の効果が低下し、局所最適に陥りやすい傾向が見られた. 特に、異なる ϵ 値で訓練された DMC 初期モデルを用いた実験では、多様な初期個体を導入することで遺伝的アルゴリズムの探索能力が向上し、早期収束を回避できることが確認された.

また、本研究では、高性能モデルに対しても遺伝的アルゴリズムが一定の最適化能力を持つことも証明された. すでに長期間の訓練を経た高性能モデルは最適化の余地が少ないと考えられるが、実験結果によると、遺伝的アルゴリズムは既存の戦略のポテンシャルをさらに引き出し、交叉や変異を通じて勝率を向上させることが可能であることが分かった. これは、強化学習モデルが十分に訓練された後でも、遺伝的アルゴリズムを用いることでさらなる性能向上が期待できることを示唆しており、特定の環境下での適応能力を強化するための有効な手法となり得ることを示している.

10.1.2 今後の課題

本研究では、遺伝的アルゴリズムが多人数ゲームにおいて有効であることを示したが、さらなる最適化や汎化能力の向上のために改善できる点がいくつかある.

まず、適応値計算の方法を改良することができる. 現在の適応値は、固定された対戦相手との勝率に基づいて評価されているが、この方法では特定の相手の戦略に過剰適応する可能性がある. その結果、より多様な対戦環境では適応できず、性能が低下する可能性がある. 今後は、DQN、CFR、AlphaZero など、異なるタイプの強化学習モデルを含む多様な対戦相手の戦略ライブラリを導入することで、遺伝的アルゴリズムがより複雑な対戦相手に適応できるようにすることが考えられる. また、対戦相手の戦略分布を動的に調整することで、遺伝的アルゴリズムがより幅広いゲーム環境に適応できるようにすることも有効な手段となる.

さらに、初代の親集団として多様な高性能モデルを使用することで、遺伝的アルゴリズムの最適化効果をさらに向上させることができる. 本研究では、モデルの性能や初代親集団の多様性が最適化過程に与える影響を分析した. その結果、遺伝的アルゴリズムは高性能モデルを基盤とすることで性能向上を実現できるだけでなく、集団の多様性を高めることで最適化効果が強化され、全体的なパフォーマンスが向上することが示された. しかし、高性能な個体を維持しながら多様性を導入することで、遺伝的アルゴリズムの進化能力を最大限に引き出せるのか、さらなる最適化が可能なのかについては、今後の研究課題として検討する必要がある.

10.2 DQN

10.2.1 まとめ

本実験では、遺伝的アルゴリズムのランダム性の特徴を検証した. 交叉や変異のプロセス自体がランダムであることに加え、適応値の計算環境にも不確定性があるため、異なる実行ごとに実験結果にばらつきが生じた. 具体的には、3 回の独立した実験で得られた最終的な最適化結果は完全には一致せず、遺伝的アルゴリズムの進化経路がランダム要因の影響を強く受けることを示している. 同じ初期条件でも、最適解は毎回異なる可能性がある. このような変動性は、遺伝的アルゴリズムの探索特性を反映しており、局所最適解から抜け出すことを可能にする一方で、最適化結果の不確実性も高めている.

10.2.2 今後の課題

本研究では、遺伝的アルゴリズムが最適化の過程で一定のランダム性と変動性を示した. 今後は、その安定性や最適化効率を向上させる方法をさらに探求できる. 一つのアプローチとして、交叉や変異の戦略を調整し、例えば適応型の変異率を導入したり、経験に基づく交叉手法を採用したりすることで、ランダム性が最適化結果に与える影響を抑えることが考えられる. また、実験回数を増やす、複数の最適化結果を統合する、あるいは勾配降下法や強化学習などの他の最適化手法と組み合わせることで、最終的なモデルの収束性を向上させることも可能である.

10.3 Resnet50

10.3.1 まとめ

本研究の実験では、遺伝的アルゴリズムのランダム性の特徴が検証された. 交叉や変異のプロセス自体がランダムであることに加え、適応度の計算環境にも不確定要素が含まれるため、実験結果には実行ごとにばらつきが見られた. 具体的には、3 回の独立した実験の最終的な最適化結果が完全に一致しなかったことから、遺伝的アルゴリズムの進化経路がランダム要素の影響を大きく受けることが示された. 同じ初期条件のもとでも、各実験における最適解が異なる可能性がある. このような変動性は、遺伝的アルゴリズムが局所最適解から抜け出すための探索能力を持つことを示す一方で、最適化結果の不確実性を高める要因にもなっている.

10.3.2 今後の課題

今後の研究では、以下の点を改良・探求することが考えられる. まず、適応度の計算方法を調整することで、評価環境の多様性を高め、モデルが特定のデータパターンに過度に適応することを防ぐことができる. 例えば、動的データ拡張（Dynamic Data Augmentation）を導入したり、異なるサブデータセットを用いて適応度を計算することで、より汎化性能の高い評価手法を確立することが可能である.

11 総合的なまとめ

実験結果は、遺伝的アルゴリズムが既に学習済みのモデルの性能を向上させるだけでなく、アルゴリズムの潜在能力を引き出し、発展させることができることを示している. 本研究では、遺伝的アルゴリズムを用いてモデルのパラメータを最適化することで、すでに学習が完了した高性能モデルであっても、遺伝的進化を通じて対戦能力をさらに向上させることが可能であることを確認した. これは、遺伝的アルゴリズムが強化学習によって訓練されたモデルに対し、後期最適化の手段として追加の改良を施すことができることを示唆している. また、実験では、適応度関数（Fitness Function）を適切に設計することで、進化の方向性を誘導し、特定の側面でアルゴリズムの潜在能力を引き出すことができることも示された. 例えば、適応度計算の方法を調整することで、特定の相手に対する対戦能力を重視したり、協調能力を向上させたりすることが可能となり、遺伝的アルゴリズムを用いて環境に適応した戦略を進化させることができる.

さらに、本研究では、遺伝プロセスそのものが最終的な最適化結果に大きな影響を与えることが明らかになった. 遺伝的アルゴリズムは交叉・変異・淘汰といったランダム性を含む操作に依存しており、異なる遺伝戦略によって最終的なモデルの性能が変化する可能性がある. 実験結果によると、適切な交叉・変異の戦略を採用することでモデルの性能を効果的に向上させることができる一方、不適切な遺伝戦略ではモデルの進化が制限される、あるいは性能が低下する場合がある. そのため、実際の応用においては、交叉方法の調整、変異率の最適化、より適切な適応度関数の設計などを慎重に行う必要がある. これにより、進化の方向性を合理的に制御し、局所最適に陥ることや性能の劣化を防ぐことができる.

また、遺伝的アルゴリズムの訓練環境も最適化結果に大きな影響を与えることが確認された. 例えば、DMC の訓練において、各世代の遺伝的アルゴリズムが同じ対戦データ（固定された手札の分布など）を使用する場合、モデルは特定の対局パターンに適応しすぎてしまい、より広範な環境への適応能力を欠いてしまう可能性がある. その結果、最適化後のモデルは、未見の手札の組み合わせに対して適切に対応できず、汎化性能が制限される（ResNet50 の結果はこの点を示している）. 一方、世代ごとに異なる手札の組み合わせを用いることで、より多様な対局環境で学習が進み、モデルの汎化能力を向上させることができる.

一方で、画像認識タスクでは各世代の遺伝的アルゴリズムが常に同じ訓練データを使用すると、モデルは既存のデータに対して微調整されるだけで、本質的に汎化に役立つ特徴を学習できない可能性がある. この場合、たとえ遺伝的アルゴリズムによって訓練セット上での性能が向上しても、新しいデータに対しては大きな改善が見られない可能性がある. 一方、進化プロセスの各世代で適応度計算に使用する画像データを動的に変更したり、データ拡張（Data Augmentation）技術を導入したりすることで、モデルを新しいデータ分布に適応させることが可能となる. このように、遺伝的アルゴリズムの進化過程で訓練データの多様性を確保することで、最適化されたモデルはより頑健な特徴を学習し、未知のデータに対する適応能力を高めることができる.

References

[1] O. Simeone, "A brief introduction to machine learning for engineers," *Foundations and Trends® in Signal Processing*, vol. 12, no. 3-4, pp. 200–431, 2018. [Online]. Available: http://dx.doi.org/10.1561/2000000102

[2] C. Gershenson, "Artificial neural networks for beginners," 2003. [Online]. Available: https://arxiv.org/abs/cs/0308031

[3] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimedia Tools and Applications*, vol. 80, no. 5, pp. 8091–8126, 2021. [Online]. Available: https://doi.org/10.1007/s11042-020-10139-6

[4] D. Zha, J. Xie, W. Ma, S. Zhang, X. Lian, X. Hu, and J. Liu, "Douzero: Mastering doudizhu with self-play deep reinforcement learning," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 12 333–12 344. [Online]. Available: https://proceedings.mlr.press/v139/zha21a.html

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013. [Online]. Available: https://arxiv.org/abs/1312.5602

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[7] R. Sun, "Optimization for deep learning: theory and algorithms," 2019. [Online]. Available: https://arxiv.org/abs/1912.08957

[8] C. Darwin, "Origin of the species," in *British Politics and the environment in the long nineteenth century*. Routledge, 2023, pp. 47–55.

[9] M.-T. Wu, H.-I. Lin, and C.-W. Tsai, "A training-free genetic neural architecture search," in *Proceedings of the 2021 ACM International Conference on Intelligent Computing and Its Emerging Applications*, ser. ACM ICEA '21. New York, NY, USA: Association for Computing Machinery, 2022, p. 65–70. [Online]. Available: https://doi.org/10.1145/3491396.3506510

[10] K. Ganapathy, "A study of genetic algorithms for hyperparameter optimization of neural networks in machine translation," 2020. [Online]. Available: https://arxiv.org/abs/2009.08928

[11] X. Dong and Y. Yang, "Nas-bench-201: Extending the scope of reproducible neural architecture search," 2020. [Online]. Available: https://arxiv.org/abs/2001.00326

[12] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," 2019. [Online]. Available: https://arxiv.org/abs/1806.09055

[13] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 4780–4789, Jul. 2019. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/4405

[14] M. Mohammadi, M. Lakestani, and M. Mohamed, "Intelligent parameter optimization of savonius rotor using artificial neural network and genetic algorithm," *Energy*, vol. 143, pp. 56–68, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360544217318273

[15] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 2002, pp. 311–318.

[16] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, Mar. 2016. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/10295

[17] D. Horgan, J. Quan, D. Budden, G. Barth-Maron, M. Hessel, H. van Hasselt, and D. Silver, "Distributed prioritized experience replay," 2018. [Online]. Available: https://arxiv.org/abs/1803.00933

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[19] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1409.1556

[20] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.