2023 年度

卒　業　論　文

題　目　遺伝的プログラミングを用いた
最適化アルゴリズム探索に関する研究

学籍番号　20A3172

氏　名　山﨑　太郎

提出月日　1 月 31 日

指導教員　二宮　洋

湘南工科大学工学部情報工学科

概要

本研究では，多くのニューラルネットワークの学習に用いられる勾配降下法に基づく最適化アルゴリズムを進化計算の 1 つである遺伝的プログラミングを用いて自動設計する手法を提案する．進化計算を用いて最適化アルゴリズムを自動設計する従来研究として，Symbolic Discovery of Optimization Algorithms[1]がある．従来手法では出力先，数学演算子，引数で構成されるリストを複数用いて多次元リストとして最適化アルゴリズムを設計した．従来手法の実験では，既存の最適化アルゴリズムを初期個体に設定し，実験を始め，用意した視覚タスクと言語タスクを用いて設計された最適化アルゴリズムを評価する．最適化アルゴリズムを設計している多次元リストに対して，ランダムに選択したリストを削除，追加，要素の変更を行う突然変異を用いて進化させて，よりタスクの評価値が高くなる最適化アルゴリズムを設計する実験を行った．結果として，用意したタスクにおいて初期個体より評価の高い最適化アルゴリズムを設計することができた．しかしながら，最適化アルゴリズムを各次元が独立している多次元リストとして最適化アルゴリズムを設計する従来手法では，突然変異を行った際に，ニューラルネットワークの学習に使用されない計算式を作成してしまう問題があった．

本研究では，木構造を用いて最適化アルゴリズムを設計し，進化の際に突然変異，木の交叉や Semantic Crossover[3]を用いて新たな最適化アルゴリズムを設計する手法を提案する．すべての要素が繋がっている木構造を用いることで，従来研究の問題である進化を行うことでニューラルネットワークの学習に用いられない要素を作成することを防ぐことができると考えた．すべての要素が繋がっているため，進化の際に交叉を行うことが容易である．

実験より，提案手法で最適化アルゴリズムを設計させた結果，従来手法の使用されない計算式が作成されてしまう問題を解決した．また，突然変異と Semantic Crossover を用いて最適化アルゴリズムを進化させることで，他の進化条件で進化させた最適化アルゴリズムの評価を上回るアルゴリズムを設計することができた．実験にて得られた最適化アルゴリズムを実験の評価に使用していない問題を学習し，既存の最適化アルゴリズムと比較を行った．学習結果より，設計された最適化アルゴリズムは比較した最適化アルゴリズムと同程度の学習を行うことができ，実験の評価に使用した問題以外にも適応できることが分かった．

目次

第 1 章　序論

近年，情報通信技術の普及により，膨大かつ複雑なデータの取得，蓄積が可能となった．それに伴い，このようなデータを高精度かつ高速に処理する方法としてニューラルネットワークが必要とされる．この需要を満たすため，ニューラルネットワークの研究は盛んに行われている．ニューラルネットワークの活用分野は多岐に渡り，医療や証券取引，自動運転といった実益に繋がる分野のみならず，絵画や音楽といった芸術分野でも活用されている．ニューラルネットワークを実用化するためには学習が不可欠であり，多くの学習には勾配降下法に基づく最適化アルゴリズムが使用され，ニューラルネットワークをより高性能に，より早く学習する工夫を行った様々なアルゴリズムが提案されている．本研究では，多くのニューラルネットワークの学習に用いられる勾配降下法に基づく最適化アルゴリズムを進化計算の 1 つである遺伝的プログラミングを用いて自動設計する手法を提案する．

最適化アルゴリズムを自動設計する従来研究として，Symbolic Discovery of Optimization Algorithms[1]がある．従来手法では出力先，数学演算子，引数で構成されるリストを複数用いて多次元リストとして，最適化アルゴリズムを設計している．実験では，既存の最適化アルゴリズムを初期個体に設定し，実験を始め，用意した視覚タスクと言語タスクを用いて設計された最適化アルゴリズムを評価する．最適化アルゴリズムを設計している多次元リストに対して，ランダムに選択した次元のリストの削除，追加，要素の変更，この 3 つの突然変異を用いて進化させて，より評価値が高くなる最適化アルゴリズムの進化，設計を行った．結果として，用意したタスクにおいて初期個体と比べ，評価値の高い最適化アルゴリズムを設計することができた．しかしながら，最適化アルゴリズムを各次元が独立している多次元リストで最適化アルゴリズムの設計を行う従来手法では，進化を繰り返すと，ニューラルネットワークの学習に不要な計算式を作成する問題があった．プログラムを用いて自動削除もできたが，今後の進化でニューラルネットワークの学習に必要な計算式に変異する可能性があったため，その都度削除せず，実験終了後に削除，修正を行った．

本研究では，従来手法にあるこの問題を解決するために，木構造を用いて最適化アルゴリズムを設計し，遺伝的プログラミングの考えに基づき進化を行う手法を提案する．木構造に含まれる要素はすべてつながっているため，最適化アルゴリズムを設計する際にニューラルネットワークの学習に不要な計算式を作成し，修正が必要にならない．本研究では，木構造の一部を変更する突然変異と選択された 2 つの木構造から部分木を選択し，位置を入れ替えることで，2 つの木

構造の考え方を組み合せられる木の交叉を用いることで，より評価値の高い個体の作成を目指す.

第 2 章　基盤技術

本章では，本研究で実装する遺伝的プログラミングを用いて最適化アルゴリズムを設計するための基盤技術となる勾配降下法に基づく最適化アルゴリズム，ニューラルネットワーク，畳み込みニューラルネットワーク，遺伝的アルゴリズムについて説明する．

第 2 章 1 節　勾配降下法に基づく最適化アルゴリズム

ある入力$\mathbf{x}$が与えられ，パラメータ$\mathbf{w}$と積和演算することで予測値yを出力する線形モデルを式(1)に示す．

$$y = \mathbf{w}^{\mathrm{T}}\mathbf{x} \tag{1}$$

予測性能の良い線形モデルとは入力$\mathbf{x}$が与えられ，出力するyと真値であるtとの誤差を最小にするモデルである．ここで，yとtの二乗和誤差をコスト関数$E(\mathbf{w})$として定義し，式(2)に示す．

$$E_{mse}(\mathbf{w}) = \frac{1}{2}(t-y)^2 \tag{2}$$

勾配降下法は，コスト関数$E(\mathbf{w})$を最小化するためにコスト関数をパラメータで微分した勾配を用いてパラメータを更新していくアルゴリズムである．勾配降下法の更新式とイメージ図を図 1，式(3)に示す．

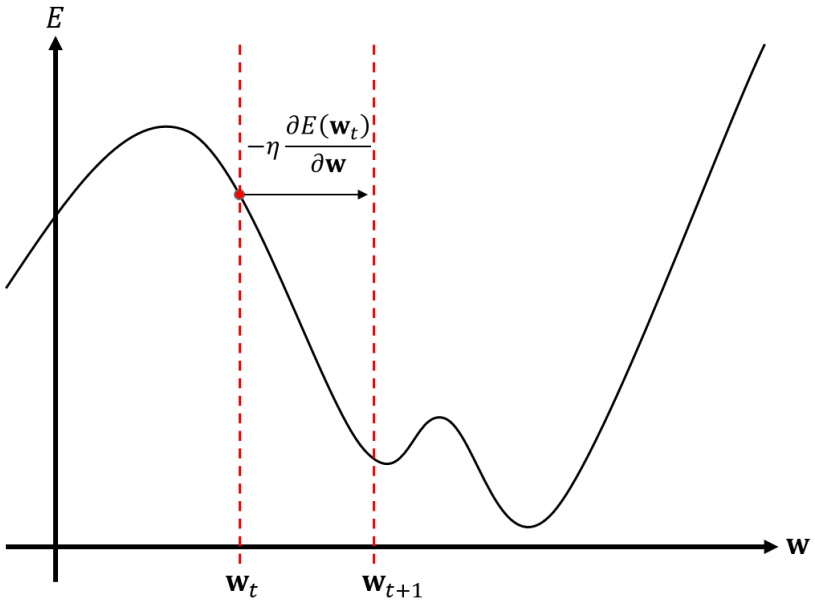

図 1　勾配降下法のイメージ

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}} \tag{3}$$

ここで，tはタイムステップ，$\mathbf{w}_t$はタイムステップtの時のパラメータ，ηは学習率，$\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$はコスト関数$E(\mathbf{w})$をパラメータ$\mathbf{w}$で微分した勾配である．一般的に勾配降下法では，$\eta$は 0.01 が使用される．パラメータを学習率と勾配を用いて，勾配の逆方向に更新していくことで，パラメータを最適化する．しかし，勾配降下法ではパラメータを最適化するまでに時間がかかることや勾配が 0 に近いが最小値ではなく，極小値に陥った場合抜け出せない問題があった．この問題を解決する策の 1 つとして慣性項を利用した Momentum がある．更新式を式(4)，(5)に示す．

$$\mathbf{v}_{t+1} = m\mathbf{v}_t + \eta \frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}} \tag{4}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{v}_{t+1} \tag{5}$$

ここで，mは慣性係数である．Momentum では一般的にηは 0.01，mは 0.9 が使用される．式(4)で勾配を蓄積することができ，蓄積した勾配をもとにパラメータを更新するため，Momentum は勾配降下法と比べ，一度の更新量が大きく，極小値を抜け出せることや早く最適化することができる．

一方で，慣性項を用いて学習を早くするのではなく，各パラメータの学習率を適応的に決定することで，パラメータを早く最適化するアルゴリズムとして RMSProp がある．更新式を式(6)，(7)に示す．

$$\mathbf{h}_{t+1} = \rho\mathbf{h}_t + (1-\rho)\frac{\partial E(\mathbf{w}_t)^2}{\partial \mathbf{w}} \tag{6}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \frac{1}{\sqrt{\mathbf{h}_{t+1} + \varepsilon}} \frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}} \tag{7}$$

ここで，ρは減衰率，εはゼロ除算を避けるための極めて小さい値，$\frac{\partial E(\mathbf{w}_t)^2}{\partial \mathbf{w}}$は$t$の時の勾配の各要素を 2 乗したものである．RMSProp では一般的にηは 0.001，ρは 0.9，εは 0.0000001 が使用される．式(6)で過去の勾配の影響も受けるが減衰率を用いることで，過去の勾配が徐々に忘却されるようになっている．そのため，勾配が小さくなることが続くと，式(7)よりパラメータの更新量が大きくなり，極小値を抜けやすくなり，パラメータを最適化することができる．

最後に Momentum の慣性を用いる考えと RMSProp の適応的にパラメータの学習率を決定する考えを組み合わせた最適化アルゴリズム Adam がある．更新式を式(8)，(9)，(10)，(11)，(12)に示す

$$\mathbf{v}_{t+1} = \beta_1 \mathbf{v}_t + (1-\beta_1)\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}} \tag{8}$$

$$\mathbf{h}_{t+1} = \beta_2 \mathbf{h}_t + (1-\beta_2)\frac{\partial E(\mathbf{w}_t)^2}{\partial \mathbf{w}} \tag{9}$$

$$\hat{\mathbf{m}}_{t+1} = \frac{\mathbf{m}_{t+1}}{1-\beta_1^t} \tag{10}$$

$$\hat{\mathbf{h}}_{t+1} = \frac{\mathbf{h}_{t+1}}{1-\beta_2^t} \tag{11}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \frac{1}{\sqrt{\hat{\mathbf{h}}_{t+1} + \varepsilon}} \hat{\mathbf{m}}_{t+1} \tag{12}$$

ここで，tはタイムステップ，β_1は慣性係数，β_2は減衰率である．Adam では一般的にηは 0.001，β_1は 0.9，β_2は 0.999，εは 0.0000001 が使用される．Adam は Momentum の慣性と RMSProp の適応的に決めた各パラメータの学習率を用いてパラメータを更新していくため，安定して高速にパラメータが最適化されていく．

第 2 章 2 節　ニューラルネットワーク

ニューラルネットワークは，人間の脳にある神経回路網やそれを構成する神経細胞のニューロンの働きを数式で表現し，人間の学習や記憶というメカニズムを再現した技術である．神経回路網を構成するニューロンは大きく分けて，他ニューロンからの信号を受け取る樹状突起，受け取った信号を評価する細胞体，他ニューロンに向けて細胞体の評価を送り出す軸索の 3 つの要素によって構成されている．軸索と樹状突起の結合部をシナプスと呼ぶ．細胞体の評価がある閾値を超えると発火し，軸索とつながっている他ニューロンへ向けて信号を送る．

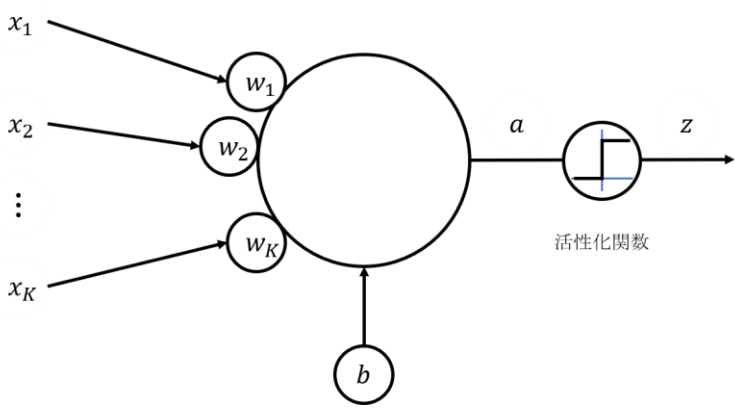

図 2　パーセプトロンの入出力関係

このような機能を持つニューロンを数式でモデル化したパーセプトロンの入出力関係を図 2，式(13)，(14)に示す.

$$a = \sum_{i=1}^{K} w_i x_i + b \tag{13}$$

$$z = \mathrm{step}(a) \tag{14}$$

ここで，Kは入力の数，x_iはi番目の入力，w_iはi番目の重み，bは閾値である．式(13)で得られたパーセプトロンの出力aをステップ関数に入力することで，シナプスの発火している，していないという 2 つの状態を表現していた．ステップ関数を図 3，式(15)に示す.

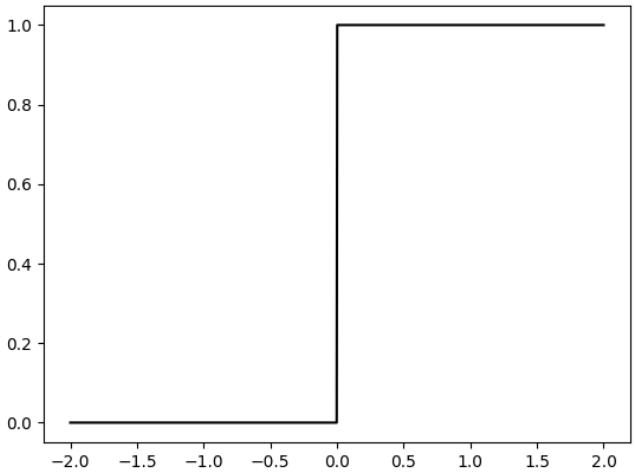

図 3　ステップ関数

$$\mathrm{step}(x) = \begin{cases} 1\ (x > 0) \\ 0\ (x \leq 0) \end{cases} \tag{15}$$

ステップ関数のような入力の値を任意に変形する関数を活性化関数と呼ぶ．パーセプトロンは線形分類器であり，非線形の問題に対応できない問題があった．複数個のパーセプトロンを階層化するように組み合わせることで，単一のパーセプトロンよりも分類能力が向上し，非線形の問題を解くことができる．複数のパーセプトロンを組み合わせた多層パーセプトロンの入出力関係を図 4，式(16)に示す.

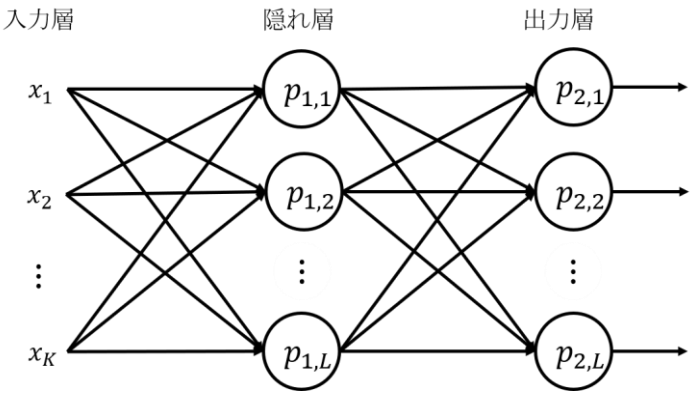

図 4　多層パーセプトロンの入出力関係

$$\mathbf{y} = \mathrm{h}(\mathbf{W}_2^{\mathbf{T}}\mathrm{h}(\mathbf{W}_1^{\mathbf{T}}\mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2) \tag{16}$$

ここで，Kは入力の数，x_iはi番目の入力，$\mathbf{x}$は入力のベクトル，$\mathbf{W}_i$はi層目にあるパーセプトロンの重み行列，$\mathbf{b}_\mathrm{i}$はi層目にあるパーセプトロンの閾値ベクトル，h()は活性化関数である．図 4 のように，複数のパーセプトロンを層のように縦に重ね，横につなげたものを多層パーセプトロン，または階層型ニューラルネットワークと呼ぶ．ニューラルネットワークは入力を受け取る入力層，ネットワークの最終出力を送る出力層，入力層と出力層の間にある層を隠れ層で構成されている．ニューラルネットワークは主に勾配降下法を用いて学習を行うため，連続的で微分可能な関数が活性化関数として使われる．ニューラルネットワークで使われる活性化関数の例として，ステップ関数を連続的で微分可能にするために近似したシグモイド関数を図 5，式(17)に示す．

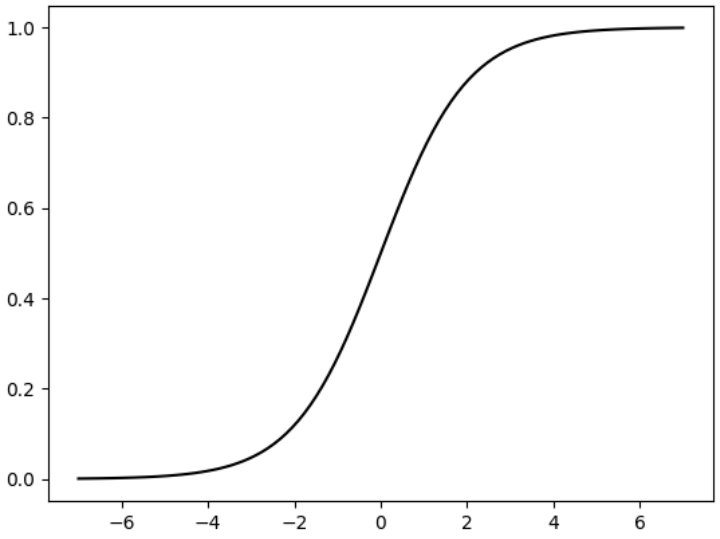

図 5　シグモイド関数

$$\mathrm{sigmoid}(x) = \frac{1}{1+e^{-x}} \tag{17}$$

ニューラルネットワークのように複数のパーセプトロンを組み合わせたネットワークの活性化関数に不連続で微分不可能なステップ関数を用いた場合，重みや閾値(パラメータ)を変化させても，出力に変化がない．対して，活性化関数にシグモイド関数の様な連続かつ微分可能な関数であれば，パラメータの小さな変化を出力の小さな変化として観測できる．設計したばかりのニューラルネットワークのパラメータは乱数により初期化され，ニューラルネットワークから望む出力を得るためには，重みや閾値を適切なパラメータに設定する必要がある．しかし，一般的に手動で設定することは現実的ではないため，主に勾配降下法に基づく最適化アルゴリズムを用いて学習を行い，ニューラルネットワークのパラメータを最適な値に設定する．

第 2 章 3 節　畳み込みニューラルネットワーク

ニューラルネットワークの入力は 1 次元である必要があり，画像を扱うためには 1 次元化しなければならない問題があった．画像を 1 次元化することで，ニューラルネットワークの入力に適した形にできるが，画像本来が持つ空間情報が損なわれてしまうことや，画像のサイズに比例して入力の数が大きくなり，隣り合う層のすべてのパーセプトロンとつながっている全結合のニューラルネットワークではパラメータ数が膨大になることが問題となり，ニューラルネットワークで画像を扱うことは困難であった．この問題を解決したのが畳み込み層とプーリング層を用いた畳み込みニューラルネットワークである．畳み込み層は，パーセプトロンを 2 次元に配置しフィルターとして扱うことで，畳み込み演算のように画像の特徴抽出を行い，フィルターにあるパラメータを学習によって最適化することができる．フィルターを通して畳み込み演算をすることで，画像の特徴を抽出した特徴マップが得られる．畳み込み層の入出力のイメージを式(18)と図 6，7，8 に示す．

$$z^{k}_{i,\ j} = \sum_{c}^{C}\sum_{u}^{U}\sum_{v}^{V} w^{k}_{u,\ v}\, x^{k}_{s\cdot i+u,\ \ s\cdot j+v} + b^{k} \tag{18}$$

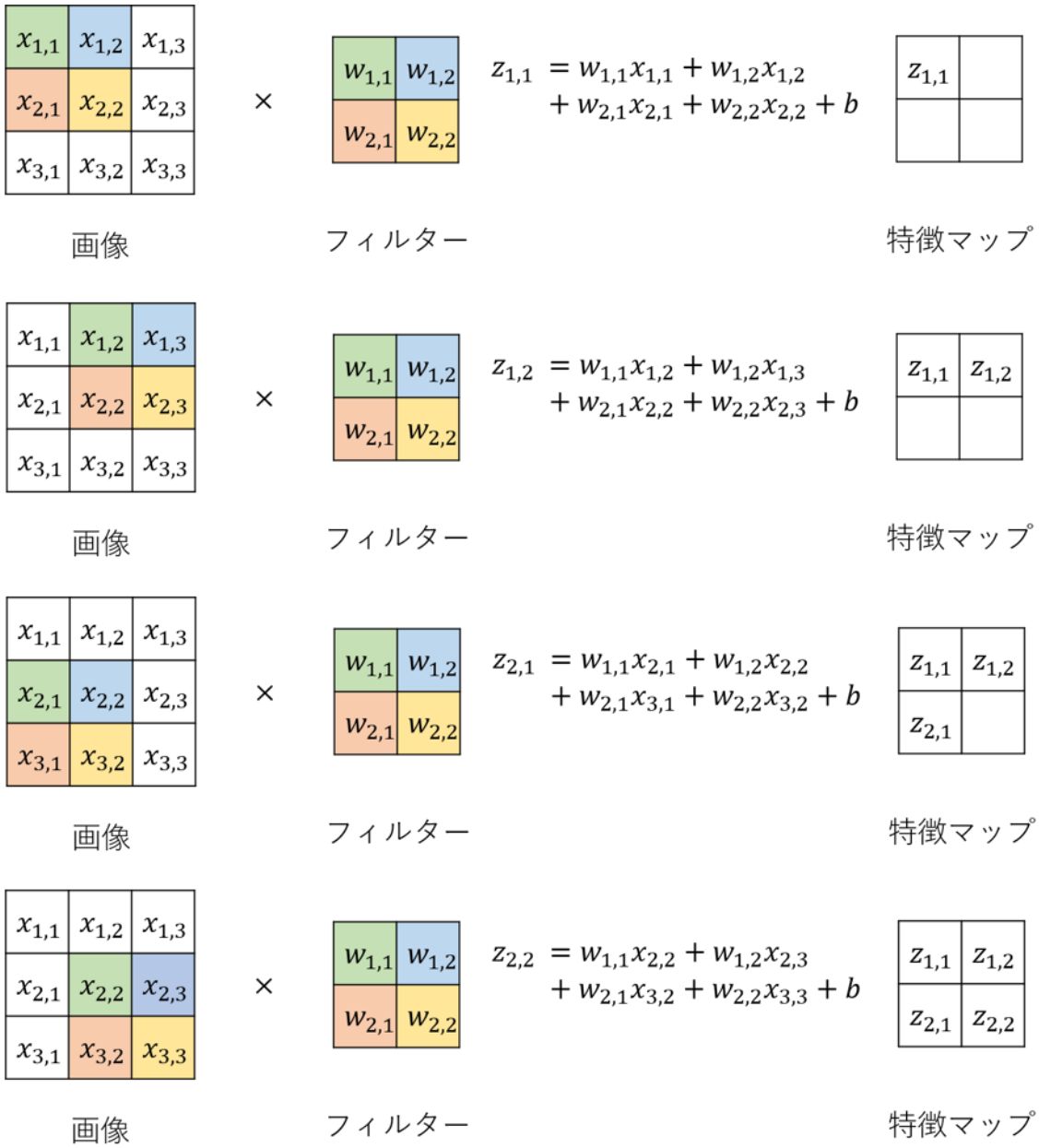

図 6　畳み込み層の入出力関係 1

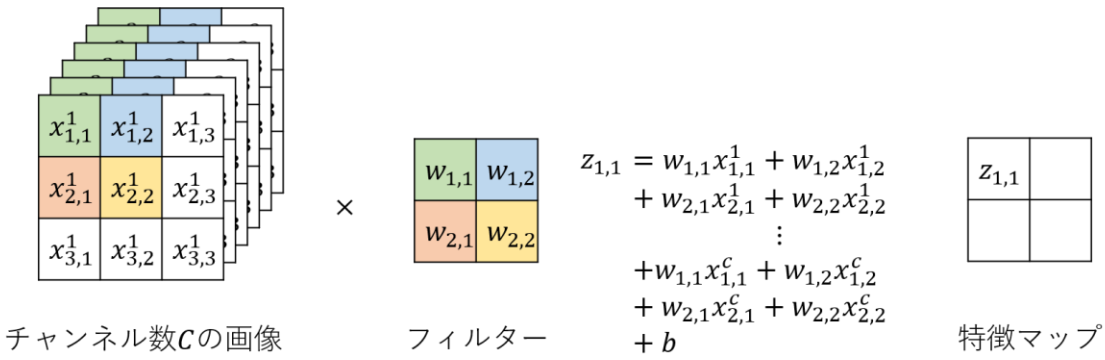

図 7　畳み込み層の入出力関係 2

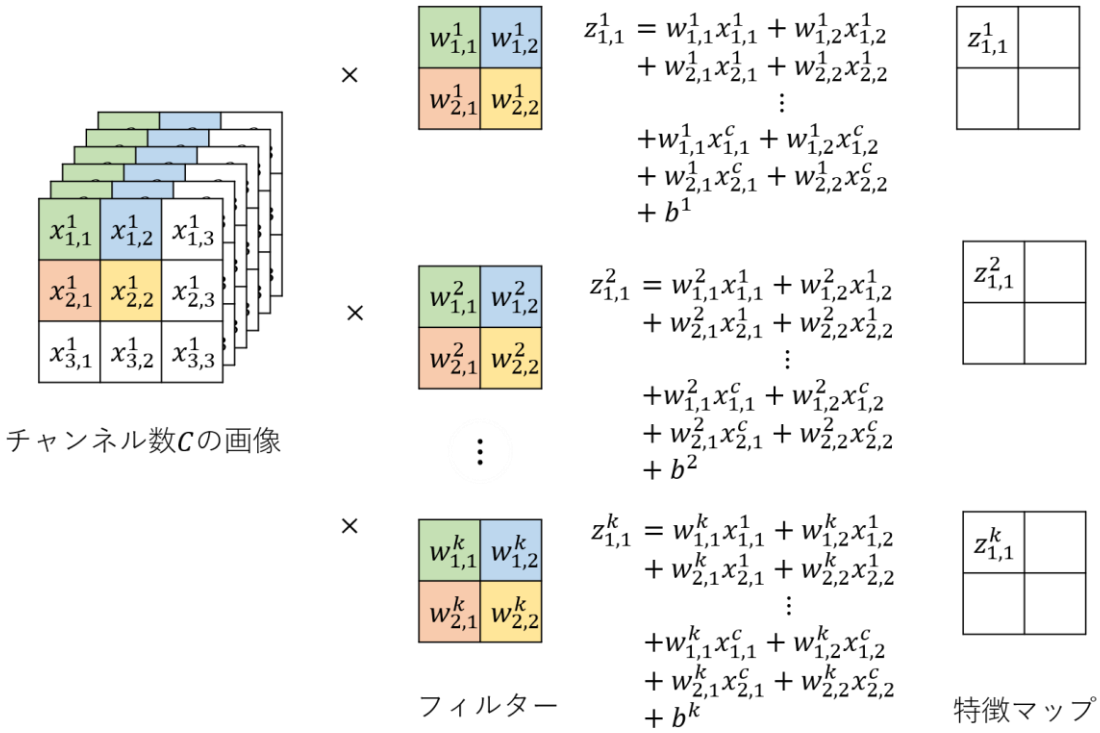

図 8　畳み込み層の入出力関係

ここで，Cは入力画像のチャンネル数，Uはフィルターの縦のピクセル数，Vはフィルターの横のピクセル数，sはフィルターの移動の幅を示すストライドの大きさ，$w^k_{i,\ j}$はk番目のフィルターの$(i,\ j)$ピクセルのパラメータ，$x^c_{i,\ j}$は入力画像のc番目のチャンネルにある$(i,\ j)$ピクセルの値，b^kはk番目のフィルターの閾値である．図 7 は$C=1,\ U=2,\ V=2,\ s=1$，フィルターが 1 枚である時の畳み込み層の入出力イメージである．図 8 は$C=C,\ U=2,\ V=2,\ s=1$，フィルターが 1 枚であるときの畳み込み層の入出力イメージ，図 9 は$C=C,\ U=2,\ V=2,\ s=1$，フィルターがk枚であるときの畳み込み層の入出力イメージである．複数枚のフィルターを用いることでより多様な特徴を抽出することができる．

画像を 1 ピクセルずらすことで，人間の目では変化を感じないがニューラルネットワークは全く別のパターンとして認識してしまう画像の位置ずれ問題があった．この問題を解決する方法としてプーリング演算を行うプーリング層の利用がある．プーリング演算には，最大プーリングと平均プーリングがある．最大プーリングを式(19)，図 9，平均プーリングを式(20)，図 10 に示す．

$$z_{m,\ n}^{k} = \max_{i,\ j \in R_{m,\ n}} x_{i,\ j}^{k} \tag{19}$$

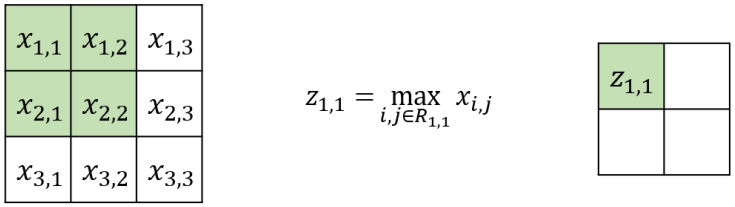

図 9　最大プーリング演算のイメージ

$$z_{m,\ n}^{k} = \frac{1}{\left|R_{m,\ n}\right|} \sum_{i,\ j \in R_{m,\ n}} x_{i,\ j}^{k} \tag{20}$$

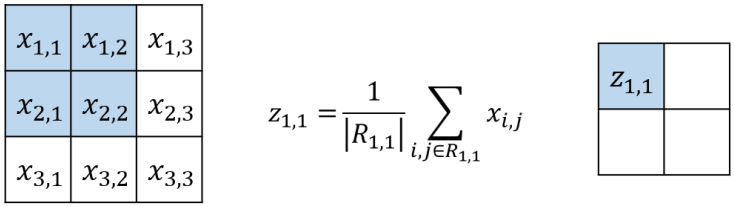

図 10　平均プーリング演算のイメージ

ここで，kは特徴マップの番号，$R_{m,\ n}$は演算後の座標$(m,\ n)$に対応する矩形領域，$x_{i,\ j}^{k}$はk番目の特徴マップの$(i,\ j)$にある値である．

ニューラルネットワークが畳み込み層やプーリング層を用いることで，扱える学習データの種類が増えた．より多くのデータを分類できるニューラルネットワークを作成するためには，層を深くすることや学習データの量を大きくすることが挙げられる．学習データが膨大であったとき，1 度のパラメータの更新に全ての学習データを用いるバッチ学習では，コンピュータのメモリ上に載らないという問題があった．この問題を解決するため，学習データをいくつかに分割し学習するミニバッチ学習や学習データを 1 つずつ学習するオンライン学習がある．バッチ学習，ミニバッチ学習，オンライン学習のイメージ図を図 11 に示す．

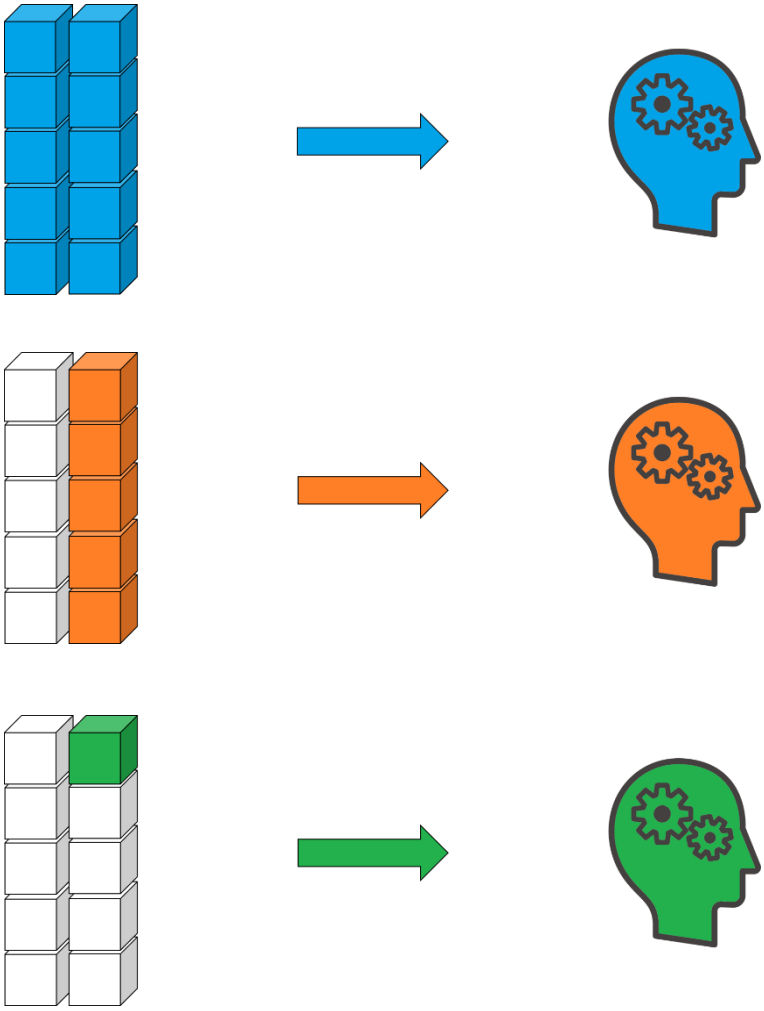

図 11　バッチ学習，ミニバッチ学習，オンライン学習のイメージ

第 2 章 4 節　遺伝的アルゴリズム

遺伝的アルゴリズムは, 与えられた問題に対して, 解候補となる個体を複数設計し, 評価と進化を繰り返し, 与えられた問題に対しての評価の高い個体を設計することが目的である. 解候補となる個体は多くの場合, 固定長 1 次元配列で設計され, 人間が設定した評価関数を基に評価し, 得られる評価値を用いて選択し, 個体を進化させる. これを繰り返し, 設定した回数や一定の評価値を超えるまで繰り返す.

第 2 章 4 節 1 項　選択

選択は与えられたある問題に対して, 現在の個体よりも評価の高い個体の設計するために遺伝子操作を行う個体を選ぶ処理である. 選択の種類として, 一定数の個体を取り出し, 取り出された各個体の中で最も評価値の高い個体を選択するトーナメント選択や, 各個体の評価値を確率に直して, 確率に従い個体を選択するルーレット選択などがある. トーナメント選択, ルーレット選択のイメージ図を図 12, 13 に示す.

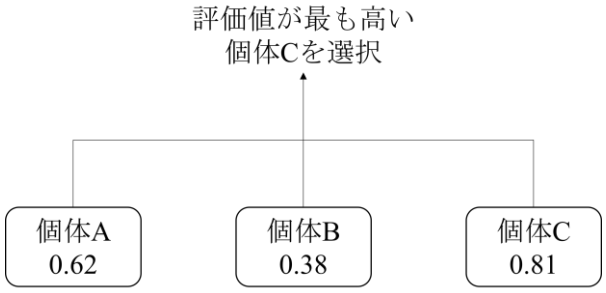

図 13　トーナメント選択のイメージ図

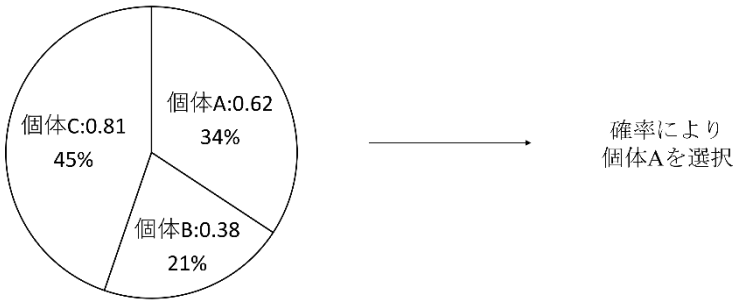

図 12　ルーレット選択のイメージ図

図 12 より, トーナメント選択はランダムに選出した個体群を比較し, 評価値が最大の個体を選択する. 図 13 より, ルーレット選択はランダムに選出した個体群の評価値を確率に直し, 確率で進化手法を行う個体を選択する.

第 2 章 4 節 2 項　突然変異

突然変異は個体の一部要素を変更することで，個体の再設計を行い，局所解を抜け出すことや，新しい解候補を設計することを目的としている遺伝子操作手法の一つである．突然変異のイメージを図 14 に示す．

図 14　突然変異のイメージ図

図 14 では，固定長 1 次元配列で設計された個体の 6 次元目の要素がランダムに選択され，0 から 1 に変異した例である．

第 2 章 4 節 3 項　交叉

交叉は選択された 2 個体を親とし，それぞれの一部要素を組み合わせることで子となる新しい個体を設計する遺伝子操作手法の 1 つである．親として選択された個体の要素が似通っているとき，交叉をすることで現在の解候補をより改善する局所探索を行う．個体の要素が似ていないとき，交叉をすることで広い空間を探索し，新しい解候補を見つける大域探索を行う．交叉のイメージを図 15 に示す．

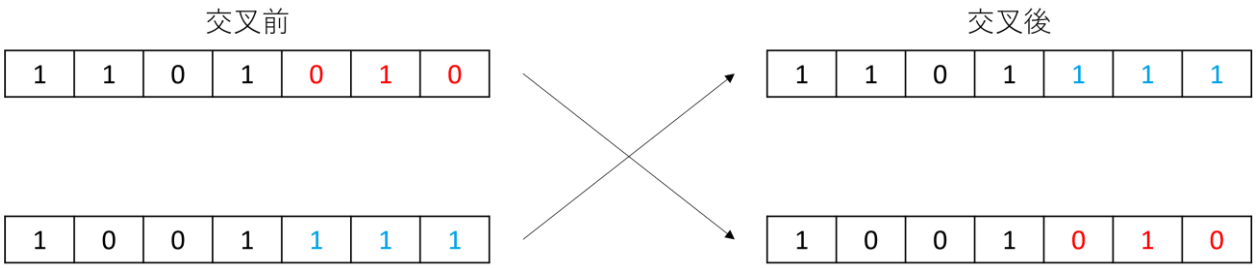

図 15　交叉のイメージ図

図 15 は親個体として選択された個体から，ランダムに選択された 4 次元目以降の要素を入れ替え，交叉を行う．交叉を行うことで親個体とは要素が大きく変わった個体が設計された例である．

第 3 章　先行研究

Symbolic Discovery of Optimization Algorithms[1]は，AutoML-Zero[2]を応用し，ニューラルネットワークを学習させるために必要な最適化アルゴリズムを出力先，数学演算子，引数で構成されるリストを複数個用いた多次元リストとして設計した．既存のアルゴリズムである Momentum を多次元リストで設計した例を表 1 に示す．

表 1　多次元リストで設計した Momentum

Momentum の更新式	多次元リストで設計
$\mathbf{v}_{t+1} = m\mathbf{v}_t + \eta\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$ $\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{v}_{t+1}$	[[v0, mul, [m, v2]], [v1, mul, [η, $\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$]], [v2, add, [v0, v1]]]

表 1 は，Momentum を多次元リストで設計した例である．[1]では，パラメータの更新量を求める式を設計した．表の Momentum では$\mathbf{v}_{t+1}$が該当する．[1]の実験では，既存のアルゴリズムを多次元リストで設計し初期個体とした．設計した最適化アルゴリズムを画像データセットである ImageNet の内 10%を Vision Transfomer に学習させる視覚タスクと言語データセット LM1B を用いて Transformer を学習させる言語タスクを基に評価を行い，評価値として，視覚タスクで得られる分類精度と言語タスクで得られるパープレキシティを用いた．最適化アルゴリズムを設計する多次元リストに対して，ある次元のリストの削除，追加，要素の変更を行う突然変異を用いて進化を行った．表 2 に多次元リストで設計した Momentum に対し，各種突然変異を行った例を示す．

表 2　多次元リストの Momentum に突然変異を行った例

Momentum	リストの削除	リストの追加	リストの要素変更
[[v0, mul, [m, v2]], [v1, mul, [η, $\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$]], [v2, add, [v0, v1]]]	[[v0, mul, [m, v2]], [v2, add, [v0, v1]]]	[[v0, mul, [m, v2]], [v1, mul, [η, $\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$]], [v3, add, [$\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$, 0.1]], [v2, add, [v0, v1]]]	[[v0, mul, [m, v2]], [v1, add, [$\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$, 0.1]], [v2, add, [v0, v1]]]

表 2 では，多次元リストで設計した Momentum に対して[1]で行われていた 3 つの突然変異を行い，空白部分はリストの削除，赤字はリストの追加や要素の変更を行った部分を示した例である．多次元のリストで設計された最適化アルゴリズムを 3 つの要素を用いて進化させていく中で，パラメータの更新に関わらない不要な計算式が作成されてしまう問題があった．プログラムを用いて不要

な計算式の自動削除も可能であったが，今後の進化でパラメータの更新に関わる可能性があったため削除せず，人間が最後に修正した.

第 4 章　提案手法

第 4 章 1 節　遺伝的プログラミング

遺伝的プログラミングとは遺伝的アルゴリズムを拡張した手法で，遺伝子によって表されるプログラムを進化的に生成させる手法である．ある問題が与えられ，1 つの遺伝子はそれに対応する 1 つの解候補を表していて，これを個体と呼ぶ．一般的に遺伝的アルゴリズムは固定長 1 次元配列を用いて個体を表現するが，遺伝的プログラミングでは木構造を用いて個体を設計する．木構造を用いることで遺伝的アルゴリズムでは扱うことが難しかった関数やプログラムなど，構造表現を扱うことができる．各個体を評価し，得られる評価値に応じて選択を行う．選択した個体に対して，突然変異や交叉という遺伝子操作を適用し，新たな個体を生成することで，与えられたある問題に対してより評価の高い個体の設計を行う．

第 4 章 1 節 1 項　突然変異

木構造における突然変異は，選択された個体を親個体とし，親個体の要素を一部変更することで親個体とは異なる新たな個体を生成する遺伝子操作の 1 つである．木構造における突然変異の例を図 16，17 に示す．

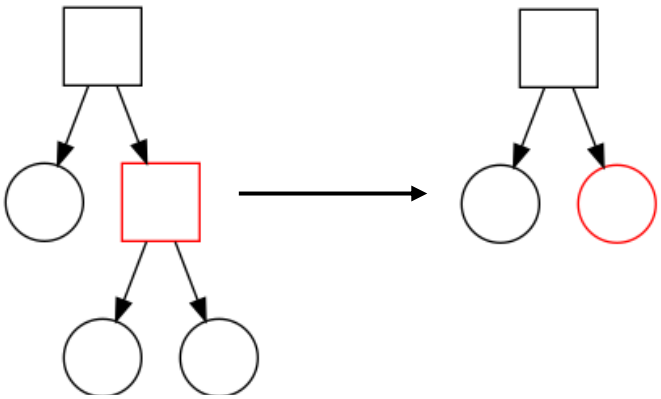

図 16　突然変異 1

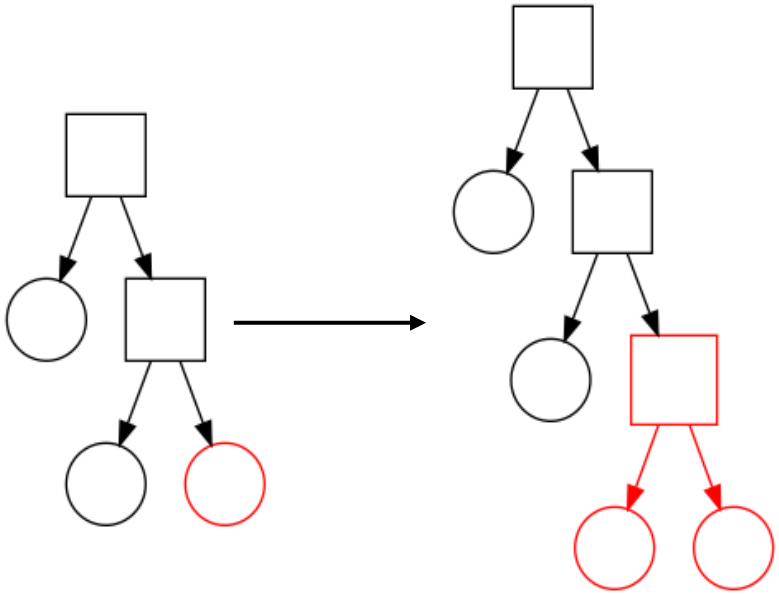

図 17　突然変異 2

図 16 は非終端ノードを終端ノードに変異させた例，図 17 は終端ノードを非終端ノードに変異させ，木構造を深くした例である．

第 4 章 1 節 2 項　交叉

木構造における交叉は，親個体となる 2 つの木構造を選択し，各木構造の一部分を切り取った部分木を入れ替え，異なる新たな木構造を生成する遺伝子操作の 1 つである．切り取る部分木はランダムに選ばれる．部分木の選択，交叉のイメージ図を図 18，19 に示す．

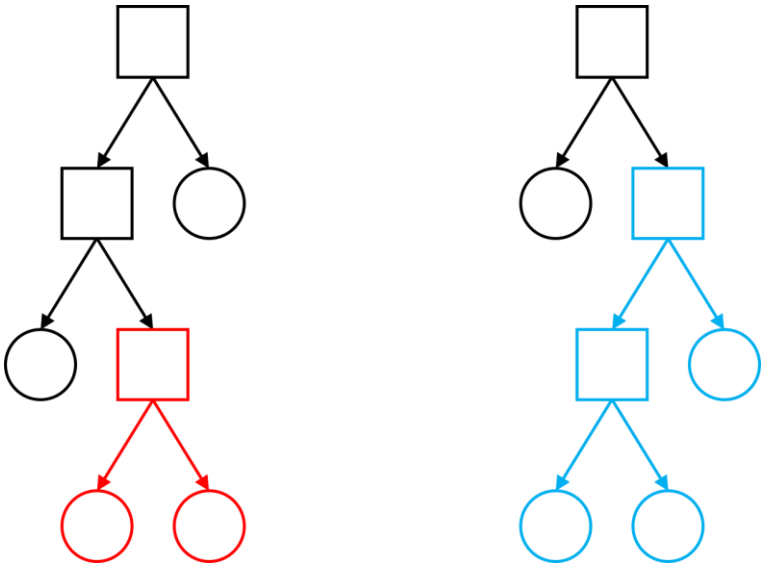

図 18　部分木の選択

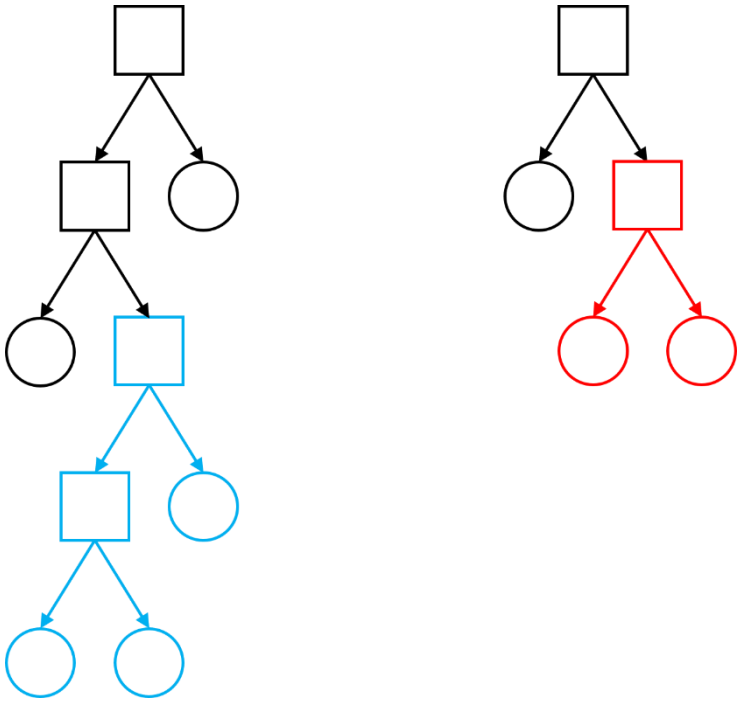

図 19　交叉の実行

ここで，図 18 は交叉を行う前に親個体として選択された 2 つの木構造からランダムに部分木を選択し，図 19 では，図 18 で選択された赤色と水色の部分木の位置を入れ替えることで交叉を行い，子個体となる新しい木構造を作成した例を示している．

第 4 章 2 節　Semantic Crossover

Semantic Crossover[3]は，遺伝的プログラミングで使用される木の交叉手法の 1 つである．通常の木の交叉は，交叉させる部分木をランダムに選択するため，局所解を抜け出すための大域探索や，解を微調整することでより良い解を探索する局所探索を意図的に行うことが難しい．対して，Semantic Crossover は交叉を行う際，個体間の部分木の意味や類似性を用いて大域探索と局所探索を意図的に制御することができる．部分木の意味とは，各個体の各部分木が出力する値のことであり，ベクトルとして考える．出力された各部分木の意味の距離を計算することで，部分木間の類似性を求めることができる．これにより，距離が近いほど類似性が高くなり，距離が遠いほど類似性が低くなる．この類似性を用いて，Semantic Crossover は交叉させる部分木を選択する．

Semantic Crossover では，親個体で選択された個体 A，B のから，個体 A の交差点をランダムに選択する．選択された個体 A の部分木の意味と個体 B で列挙

できるすべての部分木の意味との距離を計算する．距離の計算には式(21)に示すユークリッド距離を用いる．

$$d_{AB_j} = \sqrt{\sum_{i=1}^{n} \left(semantics_i^{PA} - semantics_i^{PB_j} \right)^2} \tag{21}$$

ここで，$semantics_i^{PA}$は個体 A で選択された部分木の意味のi次元目の値，$semantics_i^{PB_j}$は個体 B で列挙されたj番目の部分木の意味のi次元目の値，nは入力データの数である．個体 B で列挙されたすべての部分木との距離を求める．求めた部分木間の距離を確率に変換し，ルーレット選択を行うと，距離の大きい部分木を選択しやすく，大域探索を行ってしまう．現在の世代数に応じて，大域探索や局所探索を行うために式(22)に示すαを用いる．

$$\alpha = \alpha_{max} + (\alpha_{min} - \alpha_{max}) \frac{t}{G} \tag{22}$$

ここで，α_{min}はαの最小値，α_{max}はαの最大値，tは現在の世代数，Gは最大世代数である．このαを用いて，距離d_{AB_j}を$d_{AB_j}^{\alpha}$にしてルーレット選択を行う．世代数が小さいとき，αはα_{max}に近い値となり，ルーレット選択では類似性が低いものを選びやすく，大域探索を行う．現在の世代数が大きいときαはα_{min}に近い値となり，ルーレット選択では類似性の高いものを選びやすく，局所探索を行う．α_{min}，α_{max}の値を調整することによって，大域探索と局所探索を意図的に制御することができる．

第 4 章 3 節　提案手法

[1]は，多次元リストで設計した 1，000 個の初期個体から用意したタスクで評価し，突然変異を用いて進化させることで，初期個体よりも評価値の高い最適化アルゴリズムを設計することができた．しかし，突然変異させることで進化した最適化アルゴリズムを設計している多次元リストにパラメータの更新に不要な計算式が設計される問題があった．プログラムを用いて削除することも可能であったが，今後の進化で用いられる可能性があるため，最終的に人間が修正する必要があった．

本研究では，すべてのノードがつながっている木構造を用いて，最適化アルゴリズムの設計を行う．最適化アルゴリズム Momentum を木構造で設計したものを図 20 に示す．

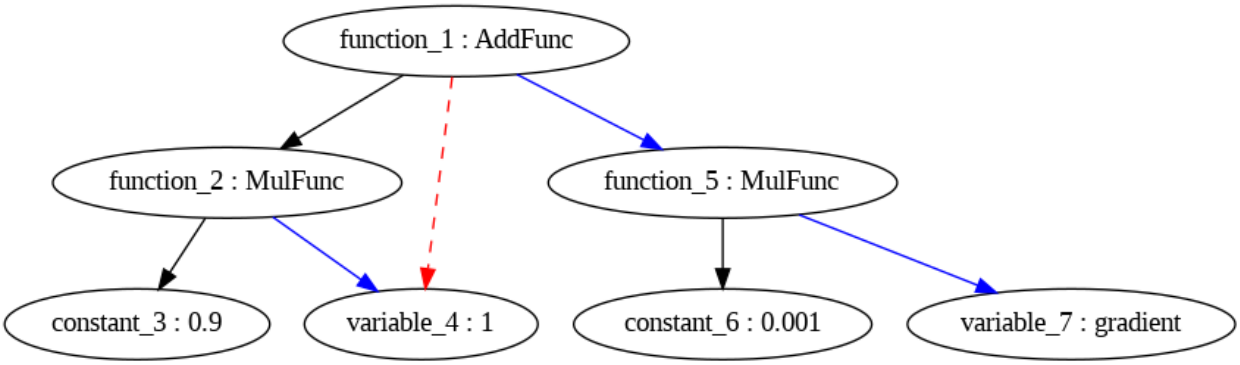

図 20　木構造で設計された Momentum

図 20 のように，1 つの木構造で複数の数学演算子を組み合わせた合成関数の式を表現することができ，[1]にあったパラメータの更新に不要な計算式を設計してしまう問題を解決できると考えた．本研究で設計する木構造の要素は，数学演算子を格納し引数となる子ノードを持っている関数ノードと定数や勾配などの変数を格納する葉ノードで構成される．赤破線で関数ノードと繋がっている葉ノードは始点にある関数ノードの演算結果を参照し保持することを意味する．本研究では木構造で設計した最適化アルゴリズムを一部変更する突然変異と部分木を他の木構造の部分木と位置を入れ替える交叉を用いて進化を行う．選択される部分木は関数ノードで切り取られ，葉ノードの参照関係が崩れない部分木が選択される．交叉と突然変異を行った Momentum の木構造を図 21 に示す．

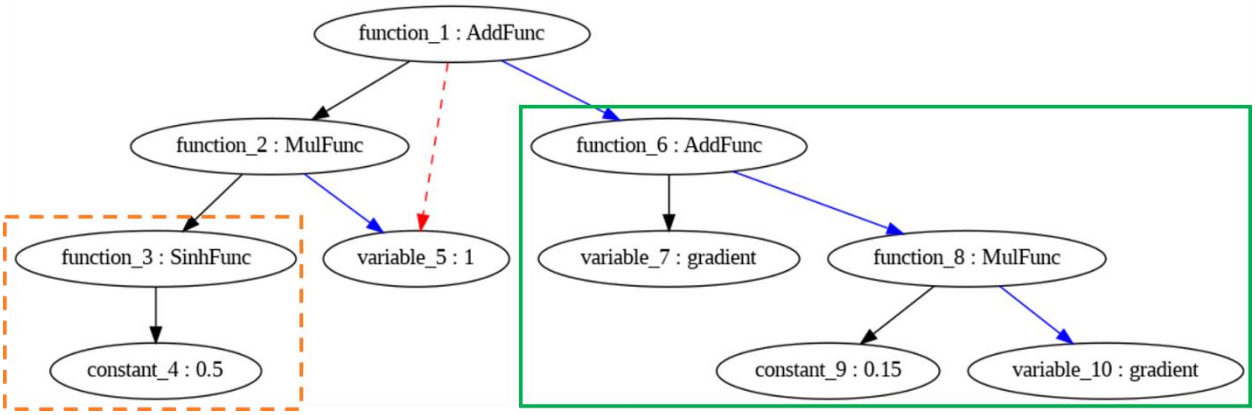

図 21　交叉・突然変異を行った Momentum

図 21 はオレンジ色の破線枠は突然変異，緑色の実践枠は交叉を行い，他の部分木と入れ替えたことを示している．全ての要素が繋がっているため，交叉や突然変異を行い新しい要素が追加されたが，[1]の問題点である不要な要素の設計をしないことがわかる．本研究では，この性質を持つ木構造を用いて設計を行い，突然変異と世代数に応じて，局所解を抜け出すことや新しい個体を見つける大域探索や現個体を改善する局所探索を制御することができる Semantic Crossover を用いて進化を行う．

第 5 章　実験

第 5 章 1 節　探索空間と評価タスクの作成

本節では，最適化アルゴリズムの設計をプログラムベースで行うために設計する必要がある探索空間と設計されたアルゴリズムを評価するタスクを作成する．探索空間は最適化アルゴリズムの設計に必要な数学演算子と引数に用いられる要素で構成される．本実験で使用する数学演算子の要素を表 3，引数の要素を表 4 に示す．

表 3　数学演算子の要素

四則演算	$+$，$-$，$\times$，$\div$
三角関数	$\sin(x)$，$\cos(x)$，$\tan(x)$
逆三角関数	$\sin^{-1}(x)$，$\cos^{-1}(x)$，$\tan^{-1}(x)$
双曲線関数	$\sinh(x)$，$\cosh(x)$，$\tanh(x)$
逆双曲線関数	$\sinh^{-1}(x)$，$\cosh^{-1}(x)$，$\tanh^{-1}(x)$
指数関数	$x^{\frac{1}{2}}$，x^2，x^y，e^x
対数関数	$\log(x)$，$\log_x(y)$
符号関数	$\mathrm{sign}(x)$
最大，最小	$\mathrm{maximum}(x, y)$，$\mathrm{minimum}(x, y)$

表 4　引数の要素

定数	平均 0，標準偏差 1 の正規乱数
変数	各ノードの出力，イテレーションt， パラメータ$\mathbf{w}_t$，勾配$\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$

人間の影響を多く与えないため，最適化アルゴリズムの設計に用いられる探索空間の要素に数学演算子を複数用いた合成関数や人間の手によって加工された引数を用意しない．CIFAR10 画像分類問題を評価タスクとして使用し，畳み込み層 1 層の CNN を設計された最適化アルゴリズムで学習し，分類精度を評価値とする．

第 5 章 2 節　従来の設計方法と比較

本節では，従来の設計と提案する従来の設計を用いて最適化アルゴリズムを設計する比較実験を行う．表 5 に示す条件と図 22 に示すフローチャートで実験を行い，実験結果を図 23，表 6 に示す．

表 5　実験条件

試行回数	5
探索空間	表 3，4 の各要素を使用
個体数	50
最大世代数	500
初期個体	Momentum，RMSProp：各 25 個体
個体の設計方法	従来 / 木構造
評価タスク	CIFAR10 画像分類問題
選択方法	トーナメント選択：サイズ 2
遺伝子操作	突然変異：確率 0.2

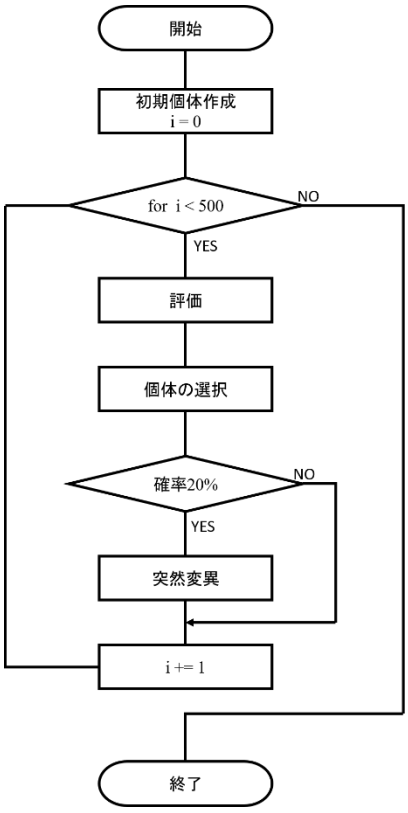

図 22　実験 1 のフローチャート

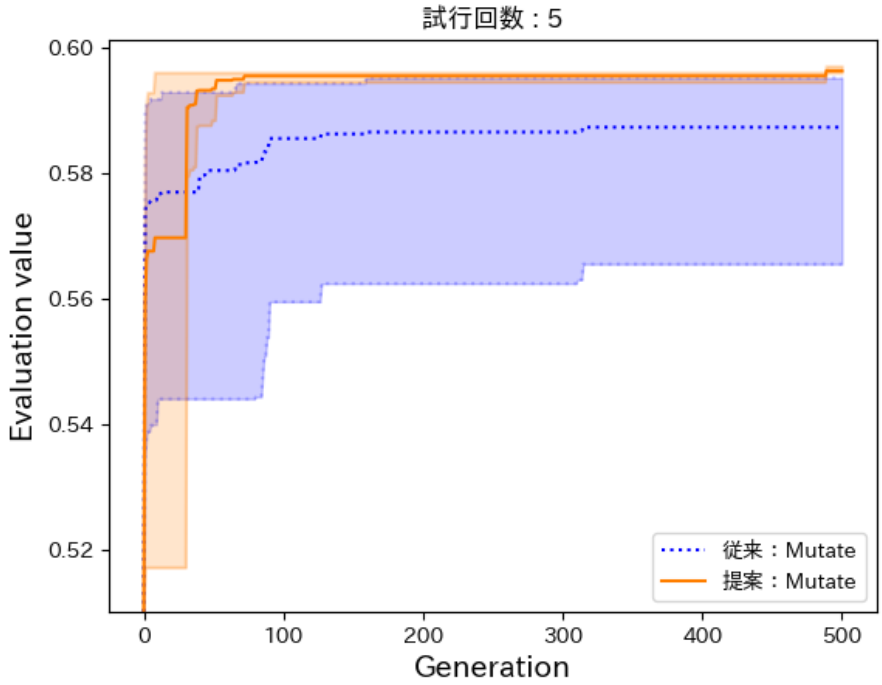

図 23　各世代の最優良個体の評価値の推移

表 6　500 世代目における最優良個体の最小値，平均値，最大値の評価値

	最小値	平均値	最大値
従来の設計方法	0.56550	0.58470	0.59530
提案する設計方法	0.59238	0.59614	0.59742

図 23 は横軸を世代数，縦軸を各世代の評価値の最大値を示したグラフ，表 6 は 500 世代目の最優良個体の評価値の最小値，平均値，最大値を示した表である．図 23，表 6 より，500 世代探索し，設計された最適化アルゴリズムの評価値は従来の設計方法より提案する設計方法の方が安定して高い評価値を得られていることがわかる．すべての要素がつながっている木構造を用いて最適化アルゴリズムを設計することで，突然変異で変更された要素が個体の評価値に影響し，有効な進化を行うことができたと考える．従来の設計方法で設計された最優良個体の更新式を表 7 に示し，定式化したものを式(23)，(24)に，提案する設計方法で設計された最優良個体の木構造を図 24，更新式を式(25)，(26)に示す．

表 7　従来手法で設計された最優良個体

削除前：18 次元	削除後：9 次元
[[v0, mul, [v1, v2]], [v1, sub, [1.0, 0.9]], [v2, square, [$\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$]], [v3, mul, [v1, v2]], [v4, add, [v0, v3]], [v5, sqrt, [v4]], [v6, add, [ε, v5]], [v7, div [0.001, v6]], [v8, sign, [-0.723]], [v9, arccos, [0.422]], [v10, square, [v14]], [v11, log, [v6]], [v12, cosh, [$\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$]], [v13, pow, [v10, 0.730]], [v14, sign, [-0.371]], [v15, arccos, [v11]], [v16, sign, [0.128]], [v17, mul, [v7, $\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$]]]	[[v0, mul, [v1, v2]], [v1, sub, [1.0, 0.9]], [v2, square, [$\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$]], [v3, mul, [v1, v2]], [v4, add, [v0, v3]], [v5, sqrt, [v4]], [v6, add, [ε, v5]], [v7, div [0.001, v6]], [v17, mul, [v7, $\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}$]]]

$$\mathbf{v1}_{t+1} = 0.1\left(\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}^2 + \frac{\partial E(\mathbf{w}_{t-1})}{\partial \mathbf{w}}^2\right) \tag{23}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - 0.001\frac{1}{\varepsilon + \sqrt{\mathbf{v1}_{t+1}}}\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}} \tag{24}$$

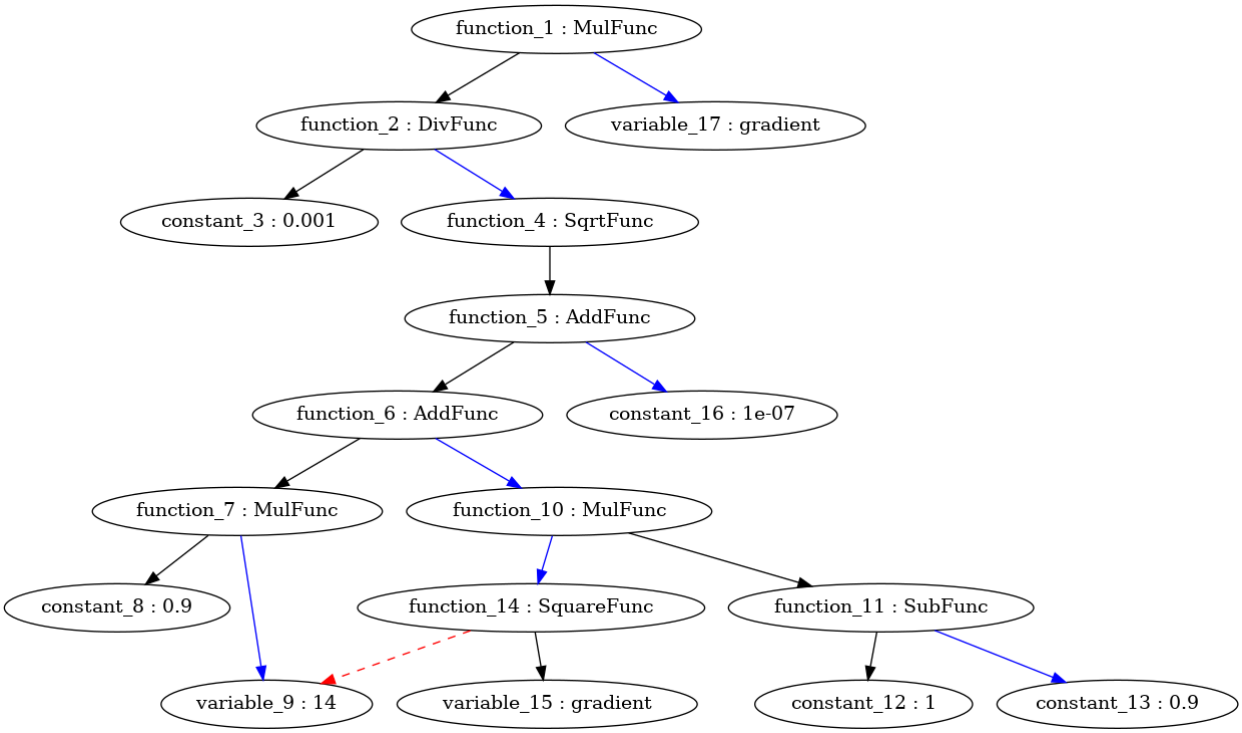

図 24　木構造で設計された最優良個体

$$\mathbf{v1}_{t+1} = \frac{\partial E(\mathbf{w}_t)^2}{\partial \mathbf{w}} \tag{25}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - 0.001 \frac{1}{\sqrt{\varepsilon + 0.9\mathbf{v1}_t + 0.1\frac{\partial E(\mathbf{w}_t)^2}{\partial \mathbf{w}}}} \frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}} \tag{26}$$

ここでεは 0.0000001 である．従来手法で設計された最優良個体は 18 次元のリストで設計されたが，9 次元のリストがパラメータの更新に関わらないと判断でき，削除したアルゴリズムが設計された．提案手法で設計された最優良個体は全ての要素が繋がっているため，従来手法のように人間が実験後修正する必要がない．各手法で設計された最優良個体は，tの時までの勾配を基に学習率を決めていた RMSProp と比べ，tと$t-1$の時の勾配を基に学習率を決める最適化アルゴリズムが設計された．

第 5 章 3 節　進化の条件を追加

実験 1 では，従来と比べ提案する設計方法を用いることでより評価値の高い最適化アルゴリズムを設計することができた．本節では，提案する木構造を用いて最適化アルゴリズムを設計し，進化の方法を変更することで最適化アルゴリ

ズムの設計への影響を確認する．表 8，9 に示す条件と図 25 に示すフローチャートに従い実験を行い，実験結果を図 26，27，表 10 に示す．

表 8　実験 2 条件

試行回数	5
探索空間	表 3，4 の各要素を使用
個体数	50
最大世代数	500
初期個体	Momentum，RMSProp：各 25 個体
個体の設計方法	木構造
評価タスク	CIFAR10 画像分類問題
選択方法	トーナメント選択：サイズ 2
遺伝子操作	Normal Crossover：確率 0.8 突然変異：確率 0.2

表 9　実験 3 条件

試行回数	5
探索空間	表 3，4 の各要素を使用
個体数	50
最大世代数	500
初期個体	Momentum，RMSProp：各 25 個体
個体の設計方法	木構造
評価タスク	CIFAR10 画像分類問題
選択方法	トーナメント選択：サイズ 2
遺伝子操作	Semantic Crossover：確率 0.8 突然変異：確率 0.2

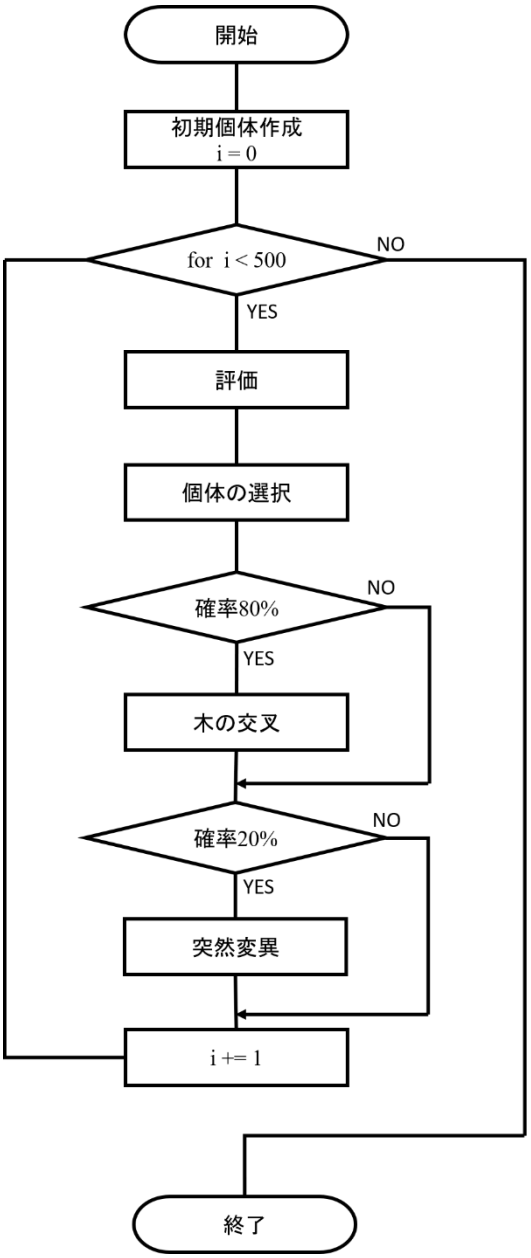

図 25　実験 2，3 のフローチャート

図 25 に示すフローチャートにおいて確率に従い行う木の交叉は，実験 2 では交叉させる部分木をランダムに選択する Normal Crossover を使用し，実験 3 では交叉させる部分木を現在の世代数 i と部分木の類似度を基に選択する Semantic Crossover を使用する．

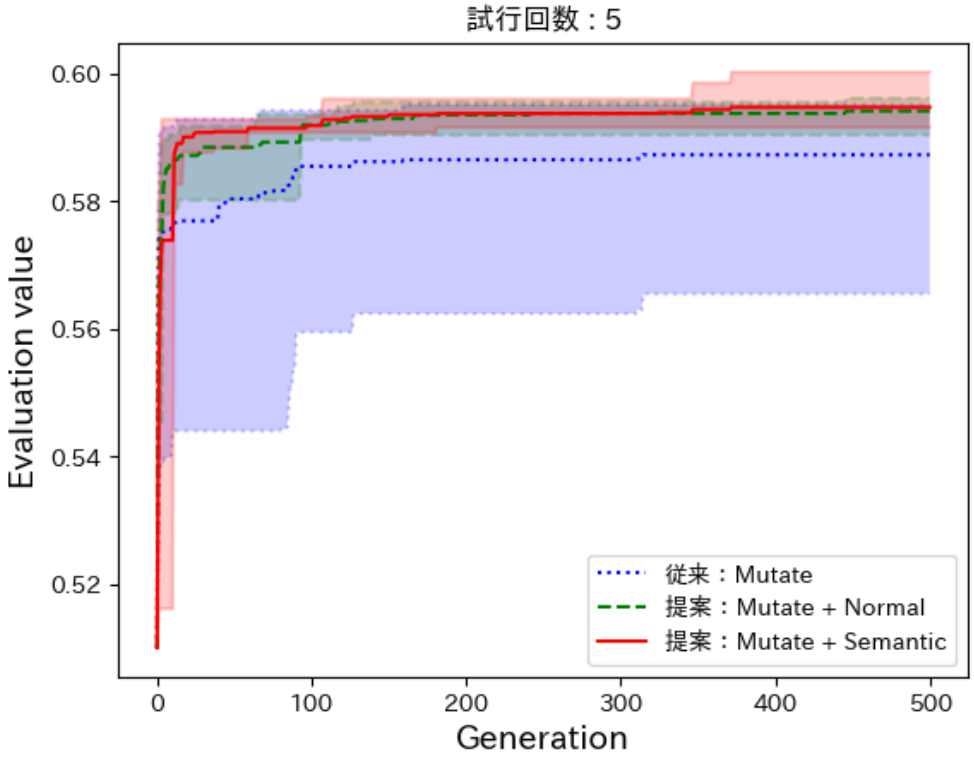

図 26　各実験条件の最優良個体の評価値の推移

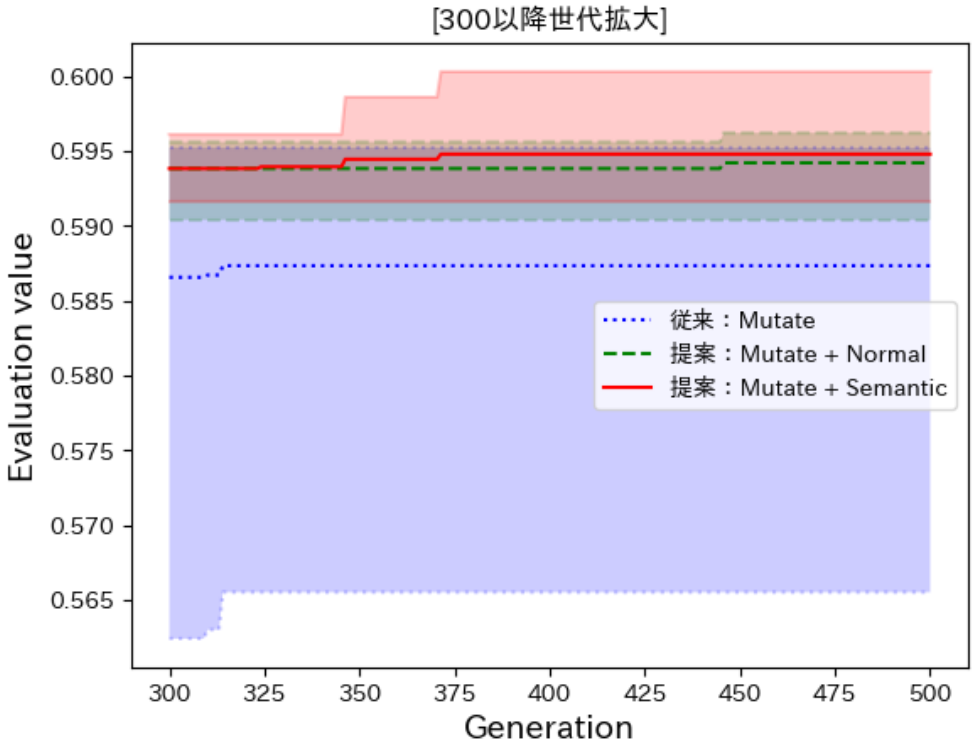

図 27　300 世代以降の評価を拡大

表 10　各実験の 500 世代目の評価値

進化の条件	最小値	平均値	最大値
突然変異	0.59360	0.59470	0.59600
突然変異 Normal Crossover	0.59560	0.59560	0.59560
突然変異 Semantic Crossover	0.59160	059525	0.60030

図 26，27 は横軸を世代数，縦軸を各世代の評価値の最大値を示したグラフ，図 27 は図 26 の 300 世代目以降を拡大したグラフである．表 10 は 500 世代目の最優良個体の評価値の最小値，平均値，最大値を示した表である．図 26，27 より，進化の条件が突然変異と Semantic Crossover を組み合わせた実験 3 の最優良個体の評価値の推移を示す赤線は，300 世代以降でも最優良個体が更新され，最適化アルゴリズムの探索においても局所探索が有効であることが分かった．実験 2 で得られた最優良個体の木構造を図 28，更新式を(27)，(28)に示し，実験 3 で得られた木構造を図 29，更新式を(29)，(30)，(31)，(32)，(33)に示す．

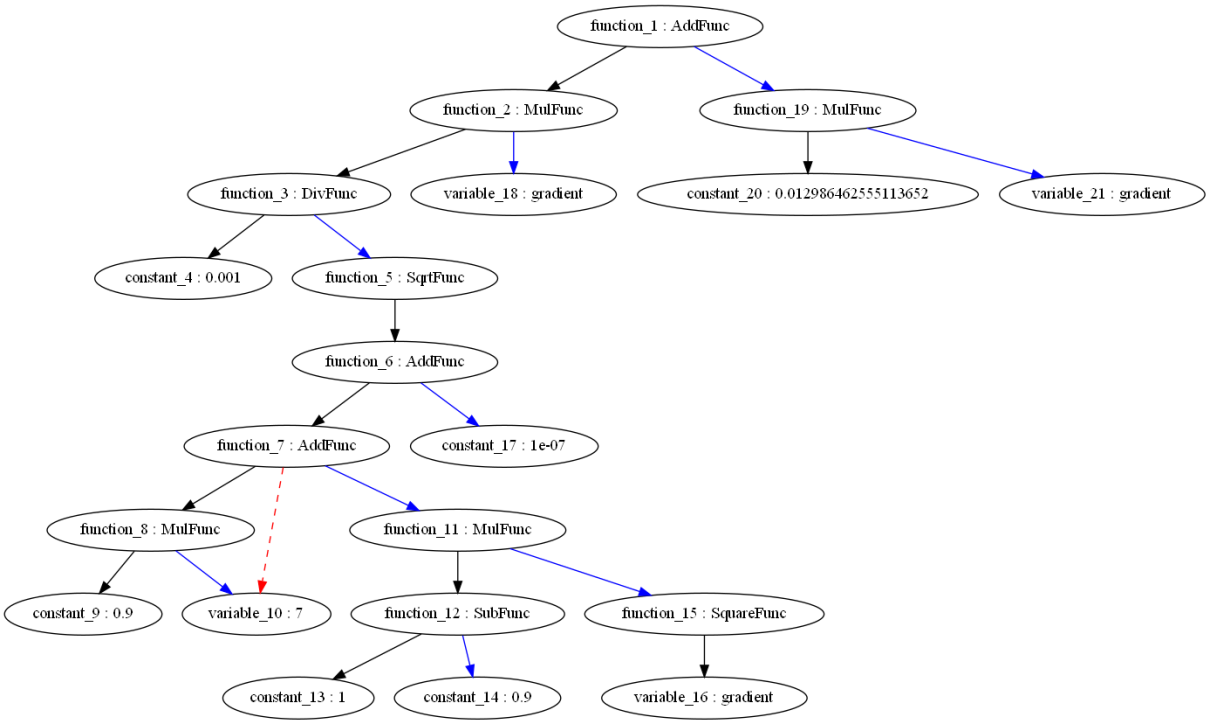

図 28　実験 2 で得られた最優良個体の木構造

$$\mathbf{v1}_{t+1} = 0.9\mathbf{v1}_t + 0.1\frac{\partial E(\mathbf{w}_t)^2}{\partial \mathbf{w}} \tag{27}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \left(0.01298 + \frac{0.001}{\varepsilon + \sqrt{\mathbf{v1}_{t+1}}}\right)\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}} \tag{28}$$

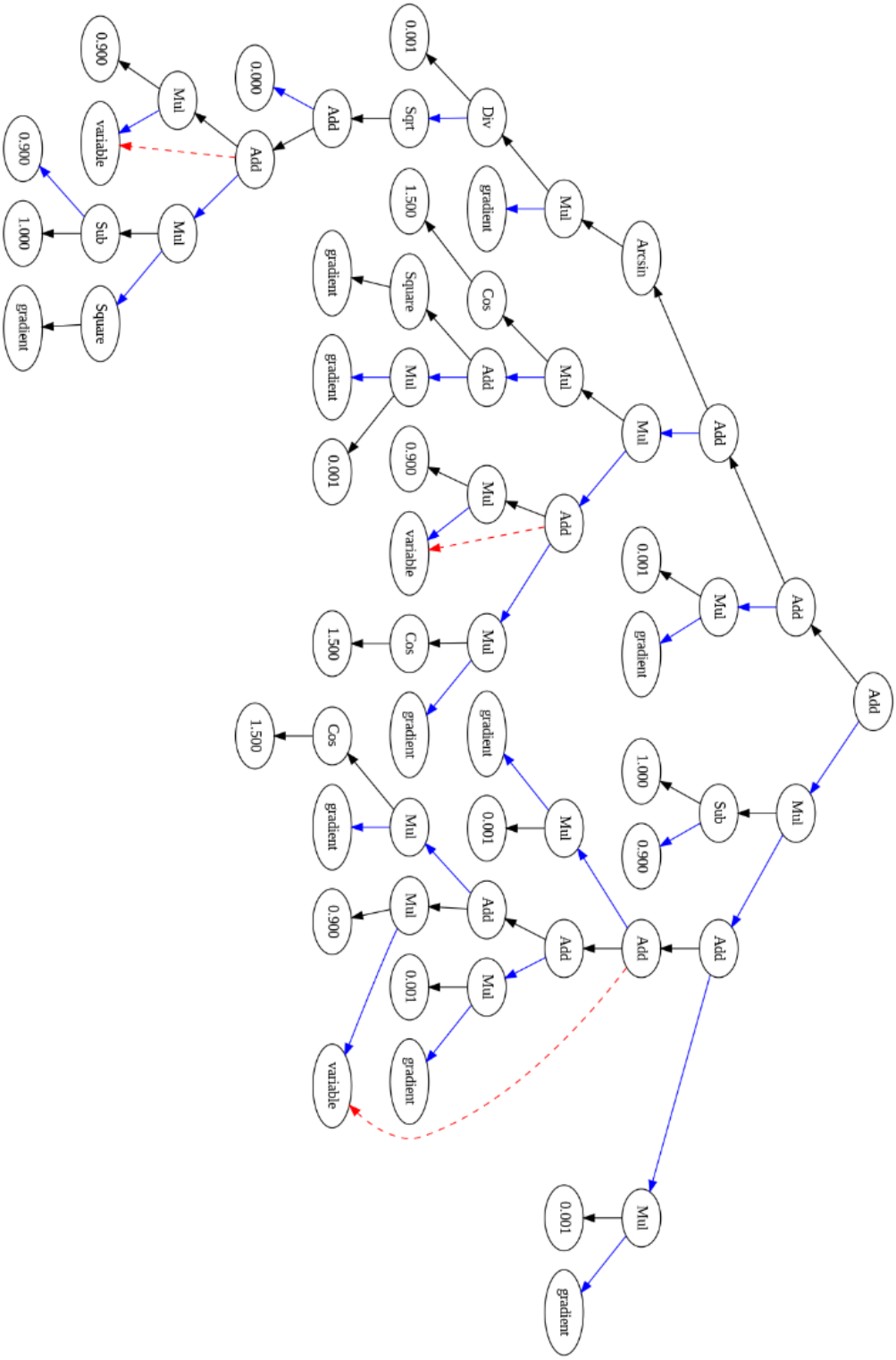

図 29　実験 3 で得られた最優良個体の木構造

$$\mathbf{v1}_{t+1} = 0.9\mathbf{v1}_t + 0.1\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}^2 \tag{29}$$

$$\mathbf{v2}_{t+1} = 0.9\mathbf{v2}_t + 0.07\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}} \tag{30}$$

$$\mathbf{v3}_{t+1} = 0.9\mathbf{v3}_t + 0.07\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}} \tag{31}$$

$$\mathbf{v4}_{t+1} = \left(\sin^{-1}\left(\frac{0.001}{\sqrt{\varepsilon + \mathbf{v1}_{t+1}}}\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}\right) + 0.07\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}}\left(\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}} + 0.01\right)\right)\mathbf{v2}_{t+1} \tag{32}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \left\{0.0011\frac{\partial E(\mathbf{w}_t)}{\partial \mathbf{w}} + 0.1\mathbf{v3}_{t+1} + \mathbf{v4}_{t+1}\right\} \tag{33}$$

ここでεは 0.0000001 である．図 28 に示す実験 2 で設計された最優良個体は RMSProp が適応的に決める学習率の下限を決めたものであった. 図 29 に示す実験 3 で設計された最優良個体は他の実験で得られた最優良個体と比べ，使用されている数学演算子の数が多く, 複雑である. 初期個体の RMSProp と Momentum の式が式(29)，(30)，(31)に確認できる．Adam のように RMSProp の学習率と Momentum の更新量を掛け合わせていないが，それぞれの考え方を足し合わせてパラメータを更新していくアルゴリズムが設計された.

第 5 章 4 節　最優良個体の汎用性の確認

前節の実験では, 設計された最適化アルゴリズムを畳み込み増 1 層の CNN に CIFAR-10 画像分類問題を学習させ，分類精度で評価した．しかし，設計された最適化アルゴリズムの評価指標が 1 つであるため，問題に依存している可能性がある．本節では，実験 3 で設計された最優良個体(Ours)の依存性を，VGG16 と呼ばれる畳み込み層 16 層の CNN と CIFAR-10 より分類カテゴリが多い CIFAR-100 を学習する分類問題，土地や部屋の広さなどの 13 個の情報から家賃を予測する回帰問題を用いて評価する．分類問題の学習条件を表 11，回帰問題の学習条件を表 12 に示し実験結果を図 30，31 に示す.

表 11　分類問題学習条件

学習エポック	50
最適化アルゴリズム	Mometum，RMSProp，Adam，最優良個体
データセット	CIFAR-100
ネットワーク構造	VGG16

表 12　回帰問題学習条件

学習エポック	500
最適化アルゴリズム	Mometum，　RMSProp，Adam，最優良個体
入力数	13

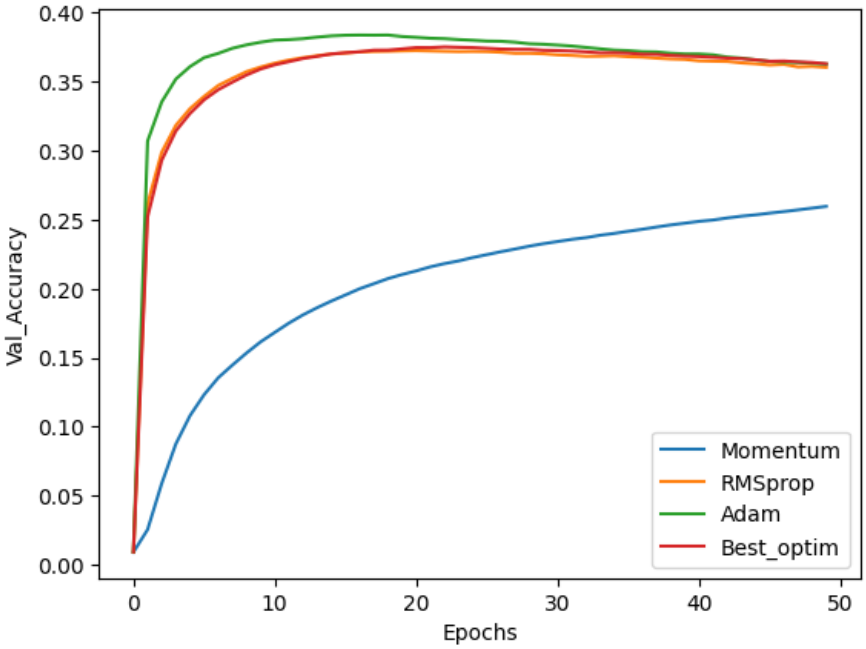

図 30　VGG16 の CIFAR-100 学習精度

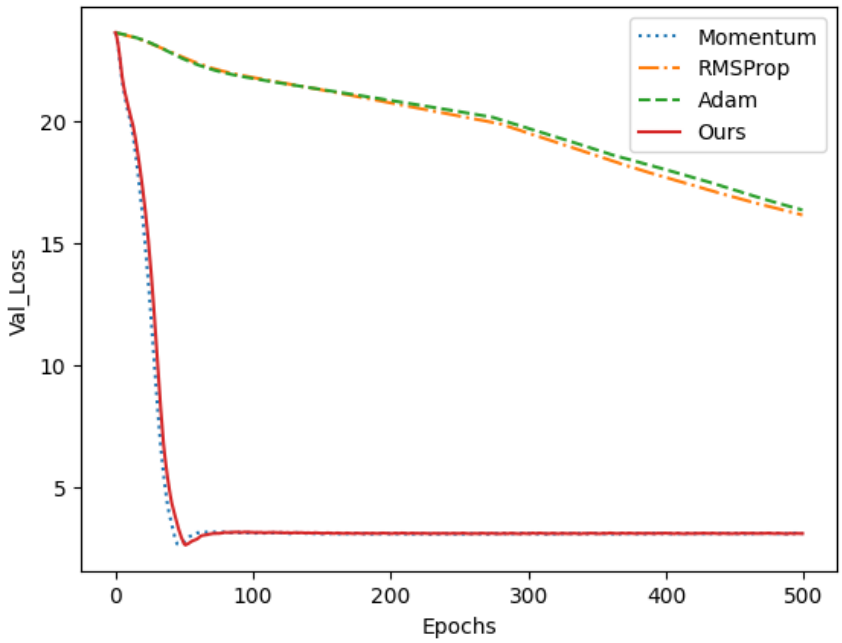

図 31　家賃予測誤差

図 30 は横軸は学習エポック，縦軸を検証用データの精度を示したグラフであり，図 31 は横軸は学習エポック，縦軸は予測誤差を示したグラフである．各グラフの青線は Momentum，オレンジ線は RMSProp，緑線は Adam，赤線が設計された最優良個体である．図 30 より，CIFAR-100 を学習する分類問題において，最優良個体は Adam に劣るものの，RMSProp と同程度の学習を行うことができた．図 31 より，家賃予測を行う回帰問題において最優良個体は Adam や RMSProp より早期に収束し，Momentum と同程度の学習を行うことができた．図 30，31 の学習結果より，実験の評価で使用した分類問題に依存していないことを確認した．

第 6 章　結論

本研究では，木構造を用いて最適化アルゴリズムを設計する手法を提案した．実験により，従来の設計手法の問題点であるパラメータの更新に不要な計算式を作成してしまう問題を全ての要素がつながっている木構造を用いて設計することで解決できた. 従来の設計方法と提案する設計方法を比較する実験 1 では，提案する設計方法で得られた最優良個体の評価値が従来の設計方法で得られた最適化アルゴリズムの評価値を上回った．これは，従来の設計方法に比べ，提案する設計方法は木構造を用いて最適化アルゴリズムを設計しているため，突然変異し変更した要素の影響を受けやすく有効な進化につながったと考える．また，木構造を用いて最適化アルゴリズムを設計し，進化の条件を組み合わせた実験 2，3 を行った．突然変異と Semantic Crossover を組み合わせた実験 3 では，探索終盤においても最優良個体が更新され，ほかの実験と比べ，最終世代で得られる最優良個体の評価値が高かった．これは，交叉させる部分木をランダムに選択するのではなく，探索終盤に類似性の高い部分木を選択し，現個体の改善を行う局所探索を行うように制御できる Semantic Crossover が大きく影響していると考える．最優良個体は 1 つの評価指標を基に設計されたため，実験の評価に依存している可能性があった．学習させる CNN の構造やデータセットを変更し，学習を行ったが，既存の最適化アルゴリズムと同等の学習を行うことができたため，依存していないことを確認した．

今後は，設計された最適化アルゴリズムの評価を複数の問題を用いて行うこと，探索要素や世代数を大きくこと，進化手法を追加し，最適化アルゴリズムの設計への影響を確認することが挙げられる．

謝辞

本研究を進めるにあたり，卒業研究を指導していただいた二宮　洋先生，渡辺重佳先生，佐々木　智志先生，齋藤　友彦先生，鎌塚　明先生，シェヘラザードマハブービ先生にはたくさんの助言をいただきました．ここに深謝いたします．

また，山富先輩や堀先輩をはじめ，二宮研究室のメンバーや学科横断 AI コースのメンバーには日頃の議論を通じ知識やアドバイスをいただき，精神的にも支えられました．ありがとうございました．

参考文献

[1]. X Chen et al.， “Symbolic Discovery of Optimization Algorithms”，*arXiv preprint arXiv:2302.06675(2023)*

[2]. Esteban Real， Chen Liang et al.， “AutoML-Zero: Evolving Machine Learning Algorithms From Scrath”，*arXiv preprint arXiv:2003.03384(2020)*

[3]. 上野祥昌，原章，高濱徹行，”遺伝的プログラミングにおける部分木の意味の類似性に基づく交叉の提案”， 2011 IEEE SMC Hiroshima Chapter Young Researchers， Workshop Proceeding，pp65-68， 2011

[4]. Chen, Qi et al., "Improving generalization of genetic programming for symbolic regression with angle-driven geometric semantic operators." *IEEE Transactions on Evolutionary Computation* 23.3, pp488-502, 2019