学科横断型学修プログラム　AI コース

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

20A3172　　山﨑　太郎
指導教員　　二宮　洋

1. はじめに

近年，様々な分野で活用されているニューラルネットワーク(NN)は学習が不可欠であり，多くは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う．学習に用いられる最適化アルゴリズムは学習の高速化や，高性能化のために工夫され，日々提案されている．最適化アルゴリズムは人間が設計する数学ベースとアルゴリズムが自動設計するプログラムベースがある[1]．プログラムベースは人間が最適化アルゴリズムに使用する数学演算子や引数を決め，進化計算により定められた要素を用いて最適化アルゴリズムを設計する．プログラムベースは人間の労力が少なく，最適化アルゴリズムの設計に人間が関わらないことで新しい考え方が見つかる可能性がある．[1]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，最適化アルゴリズムを多次元リストで設計する手法を提案した．設計された最適化アルゴリズムを評価し，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させた．しかし，進化を繰り返していく中で，NN の学習に用いられないリストが設計される問題があった．そこで本研究では，この問題を解決するため，全ての要素が繋がっている木構造で最適化アルゴリズムを設計する手法を提案する．また，実験として提案手法を突然変異や木の交叉などを用いて進化させ，[1]の手法との比較を行う．

2. Symbolic Discovery of Optimization Algorithms

Symbolic Discovery of Optimization Algorithms[1]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た多次元リストで最適化アルゴリズムを設計した．多次元リストで設計した既存の最適化アルゴリズムである Momentum を表 1 に示す．

表 1　Momentum の多次元リスト設計

Momentum の更新式	多次元リスト設計
$\Delta \mathbf{w}_t = 0.9\Delta \mathbf{w}_{t-1} + \eta \mathbf{g}_t$	[[v0, ×, [0.9, $\Delta \mathbf{w}_{t-1}$]], [v1, ×, [η, $\mathbf{g}_t$]], [$\Delta \mathbf{w}_t$, +, [v0, v1]]]

ここで，tはタイムステップ，ηは学習率，$\mathbf{g}_t$はtの時の勾配，$\mathbf{w}_t$はtの時のパラメータである．[1]は，現在の$\mathbf{w}_t$の更新量$\Delta \mathbf{w}_t$を計算する式を設計している．多次元リストで設計された最適化アルゴリズムの進化を繰り返していく中で，多次元リストにNNの学習に不要な計算式が設計される問題があった．この問題の例を表 2 に示す．表 2 は 2 次元目に新しく作成したリストを追加されたが，この式の出力は他の式で使用されず，不要な計算式であるといえる．

表 2　[1]の進化における問題例

進化前	進化後
[[v0, ×, [0.9, $\Delta \mathbf{w}_{t-1}$]], [v1, ×, [η, $\mathbf{g}_t$]], [$\Delta \mathbf{w}_t$, +, [v0, v1]]]	[[v0, ×, [0.9, $\Delta \mathbf{w}_{t-1}$]], [v1, ×, [η, $\mathbf{g}_t$]], [v3, ×, [0.1, $\mathbf{g}_t$]], [$\Delta \mathbf{w}_t$, +, [v0, v1]]]

3. 提案手法

本研究では，勾配降下法に基づく最適化アルゴリズムの設計をすべての要素が繋がっている木構造を用いて行う．木構造で設計した Momentum を図 1 に示す．

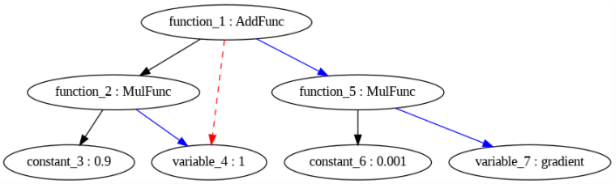

図 1　木構造で設計された Momentum

設計する木構造は数学演算子を格納し引数となる子ノードを持つ関数ノードと，子ノードを持たず定数や勾配などの変数を保持する葉ノードで構成される．破線と繋がっている葉ノードは破線の始点にある関数ノードの演算結果を保持することを意味する．提案手法では，木構造で設計した最適化アルゴリズムの一部要素を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．入れ替える部分木は関数ノードで切り取られ，葉ノードの変数の参照関係が崩れない部分木が選択される．図 1 の木構造に突然変異と交叉を行った例を図 2 に示す．図 2 の木構造は，破線の枠が突然変異，実線の枠が交叉を行った例を示している．図 2 より，追加された要素は他の要素と繋がっていることがわかる．この性質を持つ木構造を用いることで，不要な計算式が設計される[1]の問題を解決できると考えた．また，木の交叉に世代数に応じて局所解を抜け出すために行う大域探索や現個体を改善する局所探索を制御できる Semantic Crossover(SC)[2]を用いる．進化は，個体の選択，交叉，突然変異の順で行い，設定した各確率を満たした場合，交叉や突然変異を行う．

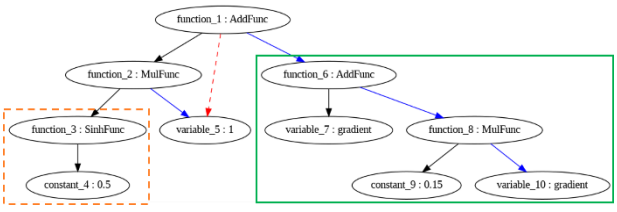

図 2　交叉・突然変異を行った Momentum

4. 実験

本実験では，従来手法と提案手法で 5 回実験を行い，各手法を評価する．四則演算や三角関数などの 22 種類の数学演算子とt, $\mathbf{w}_t$, $\mathbf{g}_t$などの変数や正規乱数を用いて生成した定数を用いて最適化アルゴリズムを設計する．設計された最適化アルゴリズムを畳み込み 1 層の CNN に CIFAR-10 を学習させた分類精度を評価値とする．実験では，突然変異する確率を 0.2, 提案手法において SC を行う確率を 0.8 とした．初期個体を Momentum と RMSProp 各 25 個体として実験を始め，500 世代になるまで評価と進化を繰り返す．表 3 に各手法の最優良個体の評価値を示す．

表 3　各手法の最優良個体の評価値

設計	進化手法	Min	Ave	Max
従来	突然変異	0.566	0.587	0.595
提案	突然変異	0.596	0.596	0.597
	突然変異と SC	0.592	0.594	**0.600**

表 3 の Min，Max，Ave は各手法で得られた評価値の最小値，最大値，平均値を示している．表 3 より，提案手法により設計された 500 世代目の最優良個体の評価値が従来手法で設計された最優良個体の評価値を上回っている．これは，提案手法によって従来の不要な計算式を作成される問題を解決し，進化によって変更された要素が個体の評価値に影響することで，有効な進化につながったと考える．また，500 世代目の最優良個体で最も高い評価値は提案手法で設計され，突然変異と SC で進化を行った最適化アルゴリズムである．これは，世代数に応じて，新しい範囲を探索し個体を設計す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で，他の進化手法の組み合わせで探索しきれなかったアルゴリズムの設計を行うことができたと考える．従来手法で得られた最優良個体のアルゴリズムを表 4，提案手法で得られた最優良個体を図 3 に示す．

表 4　従来手法で設計された最優良個体

従来：修正前(18 次元)	従来：修正後(9 次元)
[[v0, mul, [v1, v2]], [v1, sub, [1.0, 0.9]], [v2, square, [$\mathbf{g}_t$]], [v3, mul, [v1, v2]], [v4, add, [v0, v3]], [v5, sqrt, [v4]], [v6, add, [0.0000001, v5]], [v7, div, [0.001, v6]], [v8, sign, [-0.723]], [v9, arccos, [0.422]], [v10, square, [v14]], [v11, log, [v6]], [v12, cosh, [$\mathbf{g}_t$]], [v13, pow, [v10, 0.730]], [v14, sign, [-0.371]], [v15, arccos[v11]], [v16, sign, [0.128]], [$\Delta\mathbf{w}_t$, mul, [v7, $\mathbf{g}_t$]]]	[[v0, mul, [v1, v2]], [v1, sub, [1.0, 0.9]], [v2, square, [$\mathbf{g}_t$]], [v3, mul, [v1, v2]], [v4, add, [v0, v3]], [v5, sqrt, [v4]], [v6, add, [0.0000001, v5]], [v7, div, [0.001, v6]], [$\Delta\mathbf{w}_t$, mul, [v7, $\mathbf{g}_t$]]]

表 4 に示す従来手法で得られた最優良個体は，18 次元のリストの内 9 次元のリストを不要と判断し修正できる最適化アルゴリズムだった．

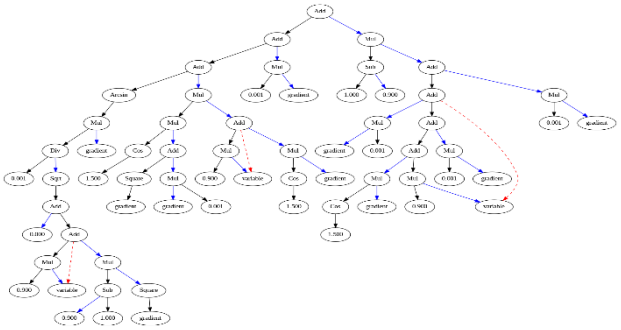

図 3　提案手法で設計された最優良個体

図 3 より，提案手法で得られた最優良個体(Ours)は SC を用いることで初期個体の Momentum と RMSProp を足し合わせ，不要な要素を含まない最適化アルゴリズムを設計することができた．

次に，設計された最優良個体と既存の最適化アルゴリズムである Momentum，RMSProp，Adam で実験に使用した分類問題と異なる問題を学習した結果を比較する．学習には家賃を予測する The Boston house-price data を用いた回帰問題を使用した．各アルゴリズムの予測誤差を図 4 に示す．

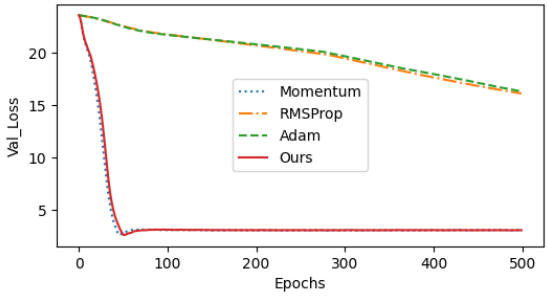

図 4　家賃予測誤差グラフ

図 4 の横軸は学習反復回数，縦軸は予測誤差である．図 4 から設計された最優良個体のアルゴリズムは回帰問題において，Adam と RMSProp と比べ，学習が早期に収束していることがわかる．このため，提案手法で設計された最優良個体は実験に使用した分類問題に依存していないことを確認した．

5. まとめ

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ことで，[1]の不要な計算式が設計される問題の解決を行った．実験では，設計方法と進化手法で比較実験を行い，得られた最優良個体のアルゴリズムの汎用性を確認した．

参考文献

[1] X.Chen, et al., *arXiv preprint arXiv:2302.06675(2023)*

[2] 上野祥昌，他, Proc. 2011 IEEE SMC Hiroshima Chapter Young Researchers, *Workshop*, pp65-68