卒 業 研 究 報 告 書

題　　目

TDGA を導入した DARTS による深層学習の構造探索

研 究 グ ル ー プ　　第 1 研究グループ

指 導 教 員　　森直樹 教授

令 和　3　年（　　2021　年 ）度 卒 業

（No.　　1171201092　　）　　杉山 竜弥

大阪府立大学工学域電気電子系学類情報工学課程

目 次

図 目 次

表 目 次

1　はじめに

以下に本論文の構成を示す．まず，2 章では本研究で用いる要素技術について概説する．3 章で深層学習の構造の設定と探索手法を提案する. そして 4 章において，数値実験により手法の性能を検証し, 本研究で提案する手法の考察をする．5 章で本研究の成果をまとめたうえで，今後の課題について述べる．

2　要素技術

本章では，本研究の提案手法に用いた技術について説明する．

2.1　Neural Architecture Search

本章では本研究で用いた Differentiable Architecture Search(DARTS) をはじめとした Neural Architecture Search(NAS) について説明する．

従来の機械学習では手作業によって設計されたモデルをデータセットで学習し重みを最適化するが, ニューラルネットワークの設計は直感的でなく, チューニングに人による労力を多く必要とするため, ニューラルネットワークの設計は非常に困難である. NAS は機械学習の分野で使用されているニューラルネットワークの設計を自動化する手法である．

2.1.1　Neural Architecture Search with Reinforcement Learning

Neural Architecture Search with Reinforcement Learning(NAS with RL)[1] は, ニューラルネットワークが構造に関する設定の文字列で表現できることを利用して, この文字列を生成する Recurrent Neural Network(RNN) を強化学習 Reinforcement Learning(RL) によって学習する．

RNN はレイヤーごとにフィルタの高さ・幅, ストライドの高さ・幅, フィルタ数を決定し, RNN によって生成された構造は, ニューラルネットワークとしてその重みが学習されテストの正答率によって性能が評価される．その性能から得られた報酬で, 方策勾配法 (Policy gradient method) による RNN の更新を行い, アーキテクチャが最適化される．

NAS with RL は高い性能を達成した一方で, 計算に数千 GPU 日かかるという問題もある．

2.1.2　Differentiable Architecture Search

Differentiable Architecture Search(DARTS)[2] は, 離散的なアーキテクチャ探索空間に強化学習を適用した NAS with RL とは異なり, 微分可能な方法で定

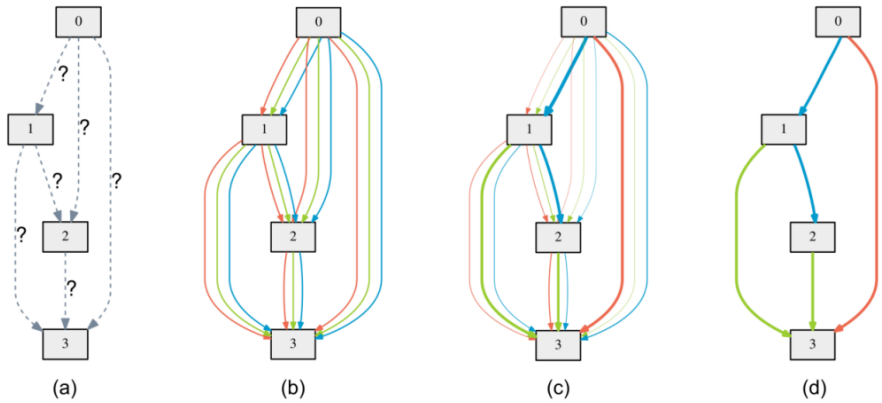

図 2.1: DARTS の概念図
(a) はじめ辺上の演算子は不明. (b) 各辺の演算子候補の混合で置換することで探索空間を連続性緩和. (c) 混合確率とネットワークの重みを最適化. (d) 学習した混合確率から最終的なアーキテクチャを導出.

式化し, 偏微分による勾配降下法を使用してアーキテクチャを効率的に探索する手法である.

探索空間を連続にするため, カテゴリカルな演算子の選択の代わりに, 候補全ての可能性をもつ混合演算子を (2.1) 式で定義する. アーキテクチャを有向非巡回グラフで表したとき, ノードを潜在的な特徴表現 $x^{(i)}$, エッジを特徴 $x^{(i)}$ が適用される関数 $o(\cdot)$ とすると,

$$\bar{o}^{(i,j)}(x) = \sum_{o \in O} \frac{\exp(\alpha_o^{(i,j)})}{\sum_{o' \in O} \exp(\alpha_{o'}^{(i,j)})} o(x) \tag{2.1}$$

となる. ここで O は探索する演算子の候補集合, $\alpha^{(i,j)}$ はエッジ (i, j) の混合演算子の重みベクトルである. DARTS は勾配降下法によって連続変数集合 α を学習する.

DARTS では次元を統一するためセルと呼ぶ小さなネットワーク構造を重ねたモデルを利用する. セルを構成するノードは 2 つのノードからの演算子エッジを持ち, どのノードからの演算子を選ぶのかをアーキテクチャを示す重み α によって決定する. DARTS の問題点として位置と演算子の種類は探索できるが, 大局的な構造やノードの持つエッジ数など固定されたアーキテクチャにしか適用できない点が挙げられる.

2.2 Genetic Algorithm

遺伝的アルゴリズム (Genetic Algorithm : GA) は生物の進化の仕組みを模倣した最適化手法である. 問題の解候補を遺伝子の持つ個体として表現し, 適応度によって個体を評価・選択する. 交叉・突然変異などの操作によって解候補の多様性を保ちつつ, 近傍を探索しながら世代を重ねて近似的な最適解を求める.

選択は現世代から次世代の個体群を選ぶ操作である. トーナメント選択

交叉は現世代から子を生成する操作である. 遺伝子型が整数型, 小数型, 順序型かによってそれぞれ交叉方法が存在する. 整数型では,

- 1 点交叉 : 染色体中の 1 箇所でランダムに切断し, 親の遺伝子を交叉する
- 多点交叉 : 染色体中の複数箇所でランダムに切断し, 親の遺伝子を互い違いに交叉する
- 一様交叉 : 遺伝子座ごとの交叉確率によって, 各遺伝子座でランダムに親の遺伝子を交叉する

などがあり, 少数型では加えて親の遺伝子をランダムにブレンドする平均化交叉もある.

突然変異は個体をランダムに変化させる操作である. 解が収束した場合, 交叉にはない局所解からの脱出という効果を持つが, 突然変異率が高すぎるとランダム探索になるため十分に小さな値を用いる. 変異方法には, 各遺伝子座でランダムに対立遺伝子へ置き換える方法, 少数値に対して摂動を与える方法などがある.

また GA には初期収束という, 偶然適応度の高くなった個体だけが選択され続け, 個体群を同じ個体が占める問題がある. この多様性が失われた状態になると単純なランダム探索と変わらない効率になるため, パラメータ調整などの方法で回避する必要がある.

交叉・突然変異の手法やパラメータは問題によって異なるため, 適切なものを設定するのは自明ではない.

2.2.1 Thermodynamical Genetic Algorithm

Thermodynamical Genetic Algorithm (TDGA) は熱力学における自由エネルギー最小化をモデルにした, GA で個体群の多様性を維持する手法である. 選択に温度とエントロピーの概念を導入し, 初期収束問題を解決した.

シミュレーテッドアニーリング法 (Simulated Annealing: SA) は次のエネルギー関数 (2.2) 式を用いて最適化問題を解く一般的な最適化手法である.

$$\min_{x} E(x), \quad x \in \mathcal{F} \tag{2.2}$$

ここで $\mathcal{F}$ は有限集合であることを仮定する. SA 法では系の状態 x に対して摂動を加え, 新しい状態 x' を得る. そして新しい状態でのエネルギー値 $E(x')$ が旧状態のエネルギー値 $E(x)$ より小さければ高い確率で, 大きければ温度パラメータ T に基づいた低い確率で新状態 $E(x')$ への遷移を行う. SA はこのアプローチを使用して最小状態を見つける.

T が定数のとき, SA の典型的な遷移規則であるメトロポリス法の分布はギブス分布となり, そのとき (2.3) 式で定義される自由エネルギー $\mathcal{F}$ を最小化することが知られており, これは自由エネルギーの最小化原理と呼ばれている.

$$F = \langle E \rangle - HT \tag{2.3}$$

ここで $\langle E \rangle$ は系の平均エネルギー, H はエントロピーである.

TDGA におけるエントロピーの計算と自由エネルギーの最小化　SA 法において最小化される自由エネルギー (2.3) 式の右辺第 1 項は, 系がエネルギー最小化という目的を追求する項, 第 2 項は系の状態の多様性を維持する項と解釈でき, これら両者を温度 T をパラメータとして調和させたものと考えられる. そこで TDGA では単純 GA(Simple GA: SGA) で用いられている適応度比例戦略に代えて, 自由エネルギーを最小化するように次世代の個体群を選択することを基本方針とする.

個体群中の個体の種を区別した系の多様性を表すエントロピー H^{ALL} は (2.4) 式で表される.

$$H^{\mathrm{ALL}} = -\sum_i p_i \log p_i \tag{2.4}$$

ここで p_i は種 i の存在確率である．GA では可能な状態 $x \in \mathcal{F}$ のうち各個体がとり得る値は個体数 N_{p} 程度であり, $|\mathcal{F}|$ に比べて極めて小さい．よって TDGA は代わりに (2.5) 式を用いて各対立遺伝子座から個体群のエントロピー H^1 を計算する．

$$H^1 = \sum_{k=1}^{M} H_k^1, \quad H_k^1 = -\sum_{j\in\{0,1\}} P_j^k \log P_j^k \tag{2.5}$$

ここで H_k^1 は個体群の遺伝子座 k の遺伝子に関するエントロピーを, P_j^k は遺伝子座 k における対立遺伝子 j の存在確率を表している．TDGA においては H^1 は (2.3) 式の第 2 項の H とみなされる．これにより世代のエントロピーが計算できるが, 自由エネルギーを厳密に最小化する個体群を選ぶことは, それ自体困難な組み合わせ最適化問題であり, 多大な計算量を要する．しかしながら, 自由エネルギーの最小化は, GA における次世代の形成の評価基準にすぎないので, 厳密な最小化は必要ではない．そこで TDGA では近似的な最小化手法として欲張り法を用いる．すなわち, 次世代の個体群を逐次的に形成する際にその時点で自由エネルギーを最小にする個体を現世代の個体群から選び, 次世代の個体群に付加するという方法を用いる．

3　ショートカット探索

DARTS で柔軟なアーキテクチャを探索するため, 深層畳み込みネットワークの VGG19[3] のショートカット接続を探索する. VGG19 は分岐がない単純なネットワーク構造であるため, ベースモデルに適しているとして選択した.

VGG19 は 16 層の畳み込み層と 3 層の線形結合層を持つ. 表 3.1 は, VGG19 の畳み込みニューラルネットワーク (Convolutional Neural Network: CNN) 部分の構造を示している. 構成する関数は, フィルターサイズが 3 x 3 の畳み込み層 (Conv2d), Batch Normalization(BN), 活性化関数 (Rectified Linear Unit: ReLU), ストライドが 2 の Max Pooling (MaxPool) である. この VGG19 に対し層を飛ばして接続するショートカットの数と位置を求め, 性能を向上させることを目的とする.

モデル中の潜在的特徴は高さ・幅・チャンネル数を持つデータであるが, 特徴の次元は場所によって異なるため, ショートカットは次元を変換する必要がある. したがってショートカット関数は以下のように設定した.

1. 次元が同じ場合：恒等関数
2. チャンネル数が違う場合：Pointwise Convolution
3. 高さと幅が半分の場合：Factorized Reduce
4. それ以外の場合：ショートカットを定義しない

ショートカットに使用する関数の制限によってショートカット位置の候補は 61 であるため, 探索空間は 2^{61} である. 演算子の種類は固定することで, アーキテクチャ α は畳み込み部に相当するグラフの重みをもつ隣接行列と定義した.

3.1　提案手法:DARTS

DARTS でネットワーク構造を探索するときの, 学習の手順は

1. 探索：アーキテクチャ α の訓練
2. 構成：α からネットワークを構成

表 3.1: VGG19 の構造
例として入力する画像を (32, 32, 3) 次元としている.

index	image size	channels	applied function
input	32 x 32	3	-
1	32 x 32	64	3x3_Conv2d, BN, ReLU
2	16 x 16	64	3x3_Conv2d, BN, ReLU, MaxPool
3	16 x 16	128	3x3_Conv2d, BN, ReLU
4	8 x 8	128	3x3_Conv2d, BN, ReLU, MaxPool
5	8 x 8	256	3x3_Conv2d, BN, ReLU
6	8 x 8	256	3x3_Conv2d, BN, ReLU
7	8 x 8	256	3x3_Conv2d, BN, ReLU
8	4 x 4	256	3x3_Conv2d, BN, ReLU, MaxPool
9	4 x 4	512	3x3_Conv2d, BN, ReLU
10	4 x 4	512	3x3_Conv2d, BN, ReLU
11	4 x 4	512	3x3_Conv2d, BN, ReLU
12	2 x 2	512	3x3_Conv2d, BN, ReLU, MaxPool
13	2 x 2	512	3x3_Conv2d, BN, ReLU
14	2 x 2	512	3x3_Conv2d, BN, ReLU
15	2 x 2	512	3x3_Conv2d, BN, ReLU
16	1 x 1	512	3x3_Conv2d, BN, ReLU, MaxPool

3. 評価：得られたネットワークをバックプロパゲーションにより訓練し, テストデータで性能を評価

の 3 段階から成る.

3.1.1　探索

探索段階では, 勾配降下法によって α の更新を行う. このとき探索用のネットワークは, ショートカットの本数も探索するため, α に対する重み補正 β を

(3.1) 式 で定義する.

$$x_i = f^{\mathrm{c}}_{i-1,i}(x_{i-1}) + \beta_i \sum_{j \in S_i} \alpha_{ij} f^{\mathrm{s}}_{j,i}(x_j) \tag{3.1}$$

ここで $f^{\mathrm{c}}(\cdot)$, $f^{\mathrm{s}}(\cdot)$ は, VGG の畳み込み関数とショートカット関数, S_i はノード i とショートカットで接続する先行 (predecessor) ノードのインデックス集合である.

ただし $\beta = 0$ で勾配の更新ができなくなるので,

$$\hat{\beta} = \begin{cases} \exp(\beta - 1) & (\beta \leq 1) \\ \log(\beta) + 1 & (\text{otherwise}) \end{cases} \tag{3.2}$$

で 0 とならないように補正した $\hat{\beta}$ を用いた.

3.1.2　構成

構成段階では, 探索段階で得られた α から具体的なネットワークをサンプリングする. α の値に対する, ネットワークの構成手法はいくつか考えられるため,

- 構成手法 A : predecessors の中で大きい順に採択
- 構成手法 B : 閾値以上のエッジを採択

の 2 通りの手法を設定した.

3.1.3　評価

評価段階では, 構成段階で得られたネットワークを学習し, 最大の正答率をネットワークの性能とする. このときのパラメータは, グリッドサーチによる最適化 API の optuna で事前学習した結果から設定する.

表 3.2: 実験 1 : 探索段階の設定

Step	Architecture Search
Loss	Cross Entropy Loss
batch size	64
Optimizer(w)	SGD(lr=0.001, momentum=0.9)
Optimizer(α)	Adam(lr=0.003, β=(0.5, 0.999))
epoch	150
train data	25000
valid data	25000
test data	10000

3.1.4　実験概要

表 3.2, 3.3 に探索段階と評価段階の実験設定を示す. 探索段階は DARTS を参考に, 評価段階は optuna で最適化した値を使用した. データセットは, 訓練画像が 32 pixel 四方で訓練データを 50000 枚持つ CIFAR-10[4] を利用して, 10 クラス分類問題を解いた.

探索時間は, 150 epoch とし, 50 epoch ごとにその時点の α の性能を評価した. 構成段階では手法 A, B に加えて比較のため, ショートカット数が同じとなる条件でランダムに選択する手法でも実験する. 各手法において 10 回試行して統計的な性能を比較した.

表 3.3: 実験 1 : 評価段階の設定

Step	Evaluation
Loss	Cross Entropy Loss
batch size	64
Optimizer(w)	SGD(lr=0.0090131, momentum=0.9)
Scheduler(w)	Step(γ=0.23440, stepsize=100)
epoch	150
train data	50000
valid data	0
test data	10000

3.2　提案手法:DARTS+TDGA

実験 1 では α の学習程度によって重み w の学習しやすさに偏りがあったため, 収束するグラフ構造にばらつきが見られた.

そこで実験 1 から得られた最適設定をもとに, 個体表現を α とした遺伝的アルゴリズムによって, アーキテクチャの多様性を維持しつつ, 安定的なネットワーク構造の学習を図った. 単純に個体数を増やすと計算コストが定数倍されるため, 重み w は全体で共有する One-Shot モデルを利用することで高速化した.

以下に TDGA をベースとして, DARTS の学習ステップの追加と多様性項の実数値拡張をした提案手法のアルゴリズムを示す.

1. DARTS で事前学習したモデルの重みを引き継いだ初期個体を生成
2. エリート個体選択
3. 個体 α_i を $\nabla_\alpha \mathcal{L}_{\text{valid}}(w^*, \alpha_i)$ で更新
4. 適応度 $\mathcal{L}_{\text{test}}(w^*, \alpha_i)$ で個体 α_i を評価
5. 交叉で子個体群生成
6. 親個体群と子個体群の突然変異
7. エリート個体と親個体, 子個体に熱力学的選択をして次世代とする
8. 収束するまで 2. に戻る

学習後最終世代の個体の性能を実験 1 と同じ条件で評価した.

3.2.1　実験概要

表 3.4, 3.5 にモデルと GA の実験設定を示した. モデルの重み w は Image Net で訓練された事前学習の重みを畳み込み層の部分に適用した. 初期収束を防ぐため, 交叉にはエッジに相当する遺伝子座ごとに 0.5 の確率で操作する一様交叉を使用した. 突然変異には遺伝子座ごとに 0.1 の確率で $\mu = 0, \gamma = 0.2$ となるガウス分布からの摂動を与えた.

表 3.4: 実験 2 : DARTS の設定

Optimizer(w)	SGD(lr=0.001, momentum=0.9)
Optimizer(α)	Adam(lr=0.001, β=(0.5, 0.999))
Loss	Cross Entropy Loss
batch size	64
train data	0
valid data	25000
test data	10000

表 3.5: 実験 2 : TDGA の設定

Population	10
Generation	150
Selection	TD Select
Temperature	$1 \rightarrow 0.01$
Crossover	Uniform Crossover
Crossover Rate	0.8
Mutation	Gaussian Mutation
Mutation Rate	0.2

表 4.1: 各アーキテクチャの精度

architecture		test accuracy (%)	param (M)	number of shortcuts	random architect accuracy (%)
method A	50 epoch	93.70 ± 0.22	21.06 ± 0.07	12.7 ± 1.4	93.60 ± 0.15
	100 epoch	94.02 ± 0.12	21.50 ± 0.11	18.2 ± 0.9	93.67 ± 0.14
	150 epoch	93.90 ± 0.17	21.57 ± 0.25	18.9 ± 0.6	93.64 ± 0.09
method B	50 epoch	93.57 ± 0.19	20.45 ± 0.09	5.8 ± 1.2	93.36 ± 0.19
	100 epoch	93.93 ± 0.08	20.73 ± 0.10	9.8 ± 1.0	93.47 ± 0.17
	150 epoch	93.92 ± 0.12	20.76 ± 0.15	10.6 ± 1.0	93.48 ± 0.15
baseline (VGG19)		93.03 ± 0.10	20.04	0	-

4　数値実験

4.1　提案手法:DARTS

表 4.1 に各構成手法におけるテストデータの精度を示す. 図 4.1, 4.2 には表 4.1 の精度に対するショートカット数とパラメータ数の関係を図示する. 最も性能が高かったのは 100 epoch 時点の手法 A で 94.02 %(baseline+0.99%) となり, 100 epoch 時点の手法 B は 93.93 %(baseline+0.90%) となった. しかしランダム手法と比較すると, 手法 A は+0.35%, 手法 B は+0.46%となり, 図 4.2 を参照しても少ないパラメータ数でより有効に探索できているのは手法 B と言える.

また 100 epoch 時点と 150 epoch 時点を比較すると, 学習によって性能が悪化している. 問題に対して過度に適合していることが原因であると考えられる.

探索時間は 150 epoch でおよそ 5 GPU hours を要したが, DARTS と比べ演算子を探索していないことや. 最適な重み w^* の近似を 1 次下げていることで高速になったと思われる.

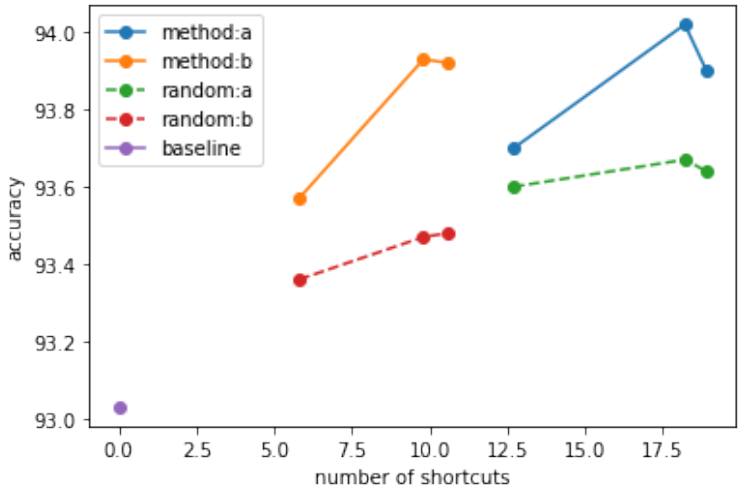

図 4.1: ショートカット数に対する精度

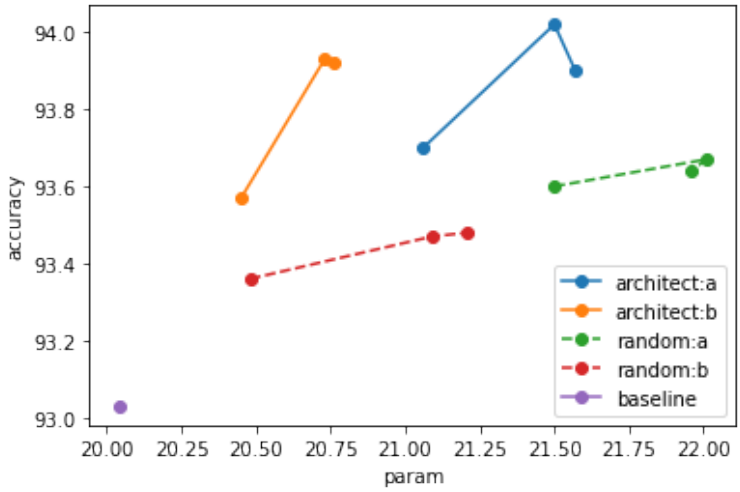

図 4.2: パラメータ数に対する精度

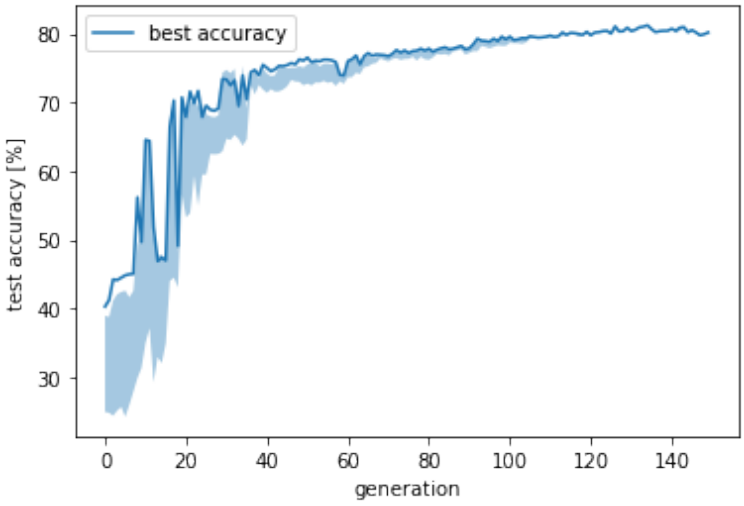

図 4.3: TDGA 精度

4.2　提案手法:DARTS+TDGA

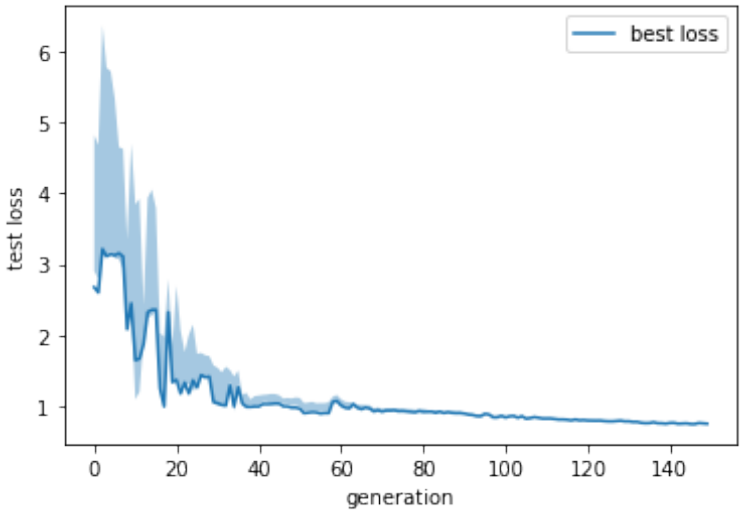

図 4.4: TDGA loss

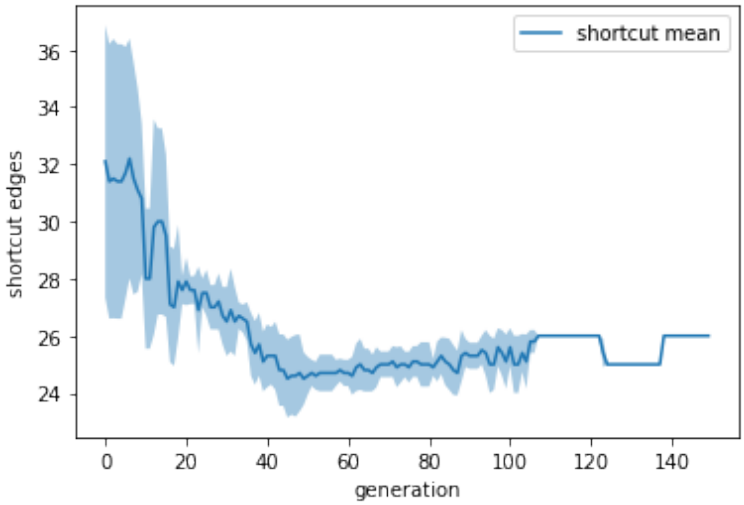

図 4.5: TDGA edge

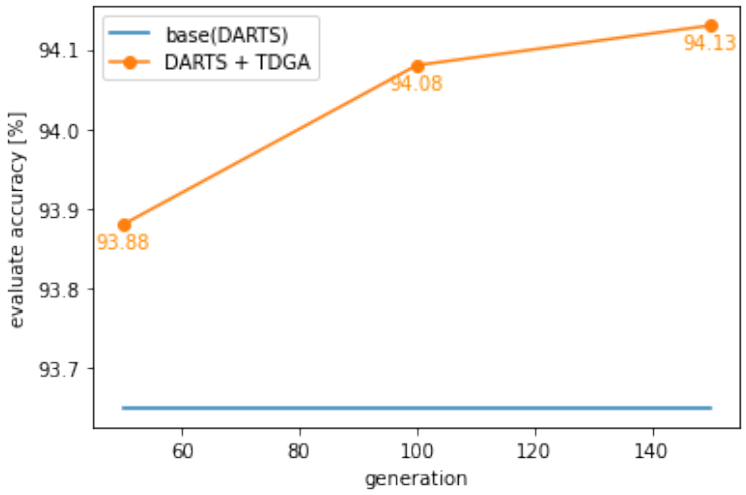

図 4.6: TDGA eval

5　まとめと今後の課題

謝辞

2021 年 3 月 11 日

参考文献

[1] CIFAR-10 (Canadian Institute for Advanced Research). Neural architecture search with reinforcement learning. abs/1611.01578, 2016.

[2] year, year. DARTS: differentiable architecture search. abs/1806.09055, 2018.

[3] year. Very deep convolutional networks for large-scale image recognition. 2015.

[4] year, year. Cifar-10 (canadian institute for advanced research).