GA によるデータ拡張手法の探索と探索済み手法に基づくアンサンブル学習の検討

1　はじめに

近年, 機械学習の発展には目を見張るものがあり, 画像, 自然言語さらには動画など様々なタスクでよい性能を示している. しかし, それらの性能を出すまでにハイパーパラメータのチューニングが必須であり, それらは一般的に人間が手動で行っている. 一方で現在ではそれらのチューニングも自動化させるための研究が行われており, Autmated Machine Learning(:AutoML) と呼ばれている. また AutoML が成果を出し始めた事に触発され, データ拡張においても同様に自動化させる研究が　進められ, データ拡張の自動化は AutoAugment と呼ばれている.　AutoAugment はデータ拡張に関する複数のポリシーを探索し, 手動のデータ拡張　よりも~~制度~~のよいポリシーが探索された一方で, それらの探索に非常に莫大な時間をかけてしまっている　ことが問題点として挙げられる. // 以上の背景から本実験では AutoAugment の研究への取り掛かりとして, 簡単なモデルに対する有用性を確かめ, 複数のポリシーという点に着目しアンサンブル学習を行うことで制度を上げられないか検討を行う.

2　要素技術

2.1　畳み込み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 (Convoltional Neural Network:CNN) はニューラルネットワークに畳み込み層とプーリング層を追加したものである. 画像などの二次元データに対し特徴を抽出することができ, 画像の分類問題に有効である.

2.2　遺伝的アルゴリズム

遺伝的アルゴリズム (Genetic Algorithm:GA) は生物の進化を模倣して適切なデータを見つけるアルゴリズムである. 最小単位を遺伝子とし, 探索するデータを遺伝子の集合である個体として表現する. 各個体の適応度を計算し, 個体の集まりである集団に対し選択, 交叉, 突然変異の三種類の操作を適用させ次の集団を作る, という操作を繰り返して適応度の高い個体を探索する. 交叉の特性上, 他のアルゴリズムより局所探索になりにくいが, 一方で設定によっては初期収束を起こしてしまう.

2.3　アンサンブル学習

複数の学習器を融合させ一つの学習モデルを生成する方法である. 弱学習器であっても高精度を実現することができる.

2.4　データセット

データセットは cifar10 を用いた. cifar10 は Alex Krizhevsky によって整備された 6 万枚の画像からなるデータセットである. 各画像は 32×32pixel で, 10 種類のラベル (airplane,automobile,bird,cat,deer,dog,frog,horse,ship,truck) のいずれかが添付されていて, 各ラベルの枚数は一様である. このうち 5 万枚を train_data に, 1 万枚を test_data に用いる.

3　数値実験

本実験ではデータ拡張のポリシーとして次項で述べる transform を全てかけることとし,　それらについての適用する強度, 確率, 順序をまとめたものを一つの個体とした.　この個体でデータ拡張を行って CNN の学習を行い, 分類問題の精度が高い個体, および　アンサンブル学習を行った時の精度を向上させることを目的とする.

3.1　GA の設定

3.1.1　transform

今回用いる transform は~~と~~画素値操作 (Sharpness, Posterize, Brightness, Autoconstrast, Equalize, Solarize, Invert, Contrast, ColorBalance), 変形操作 (Mirror, Flip,Translate X/Y, Shear X/Y, Rotate) の 16 種類の操作を用いる.

3.1.2 個体表現

一個体 **G** は強度，確率，順序の 3 つの染色体を持ち，各染色体は transform の数，つまり 16 の遺伝子を持つ．強度，確率はそれぞれ実数値コーディングで，強度は-100%から 100%まで 25%ずつ分 11 段階の度合い，確率は-0%から 100%まで 10%ごと 11 段階の度合いとする．また，順序は順列コーディングで表す．

$$
\begin{aligned}
\mathbf{G} &= (\mathbf{Ch}_{\mathrm{mag}}, \mathbf{Ch}_{\mathrm{prob}}, \mathbf{Ch}_{\mathrm{ord}}) \\
\mathbf{Ch}_{\mathrm{mag}} &= (mag_0, mag_1, ...mag_{15}) \\
\mathbf{Ch}_{\mathrm{prob}} &= (prob_0, prob_1, ...prob_{15}) \\
\mathbf{Ch}_{\mathrm{ord}} &= (ord_0, ord_1, ...ord_{15})
\end{aligned}
$$

- $mag_0, mag_1, ...mag_{15}$：強度の遺伝子
- $prob_0, prob_1, ...prob_{15}$：確率の遺伝子
- $ord_0, ord_1, ...ord_{15}$：順序の遺伝子

3.1.3 選択

選択について，エリート選択によって最も適応度の高い 2 つの個体を選択する．なお，この二つは後述する交叉，突然変異は受けずに次の世代に追加する．残りの選択にはトーナメント選択 (トーナメントサイズ 2) を用いた．この選択は集団からランダムに 2 つの個体を取り出し，そのうち適応度の高いものを次の個体に加える操作である．

3.1.4 交叉

強度，確率を表す染色体については 2 点交叉，順序を表す染色体については部分写像交叉を用いた．2 点交叉は一対の親染色体をそれぞれ同じ場所で三分割し中央の染色体を入れ替えて交叉を行う．部分写像交叉は親遺伝子を二分割し入れ替える際重複をなくす交叉法で，重複のあった遺伝子について，それに該当した重複する遺伝子座を見つけ，それに対となっているもう一方の親の遺伝子を参照する．

3.1.5 突然変異

強度，確率を表す染色体について，対象となる遺伝子の値を各 50%の確率に 1 増減させ，　順序を表す染色体について，染色体の一部を逆順にする操作か，染色体を二つに分け前後を入れ替える操作のいずれかを行うものとした．

3.2 予備実験

各個体にたいし CNN の学習をはじめからやろうとすると非常に時間がかかってしまう．そこで予め学習済みのモデルを用意した．表 1 にパラメータを示す．epoch 数は 250epochs とした．CNN のモデルは 11 層の畳み込み層と一層の全結合層からなり，kernel_size は (3,3)，活性化関数は ReLU，三層ごとにフィルター数が 64，128，256，512 で，三層ごとに BatchNormalization, Maxpooling, ドロップ率 0.25 の Dropout を設けた．また，全結合層のユニット数は 1024 で，出力に softmax を用いた．また，これによって得られたモデルを用いた予測精度 0.8475 を以降 base_line とする．

表 1: CNN 学習パラメータ

optimizer	Adam
learning rate	0.001
loss function	categorical_crossentropy
batch size	128

3.3 実験 1　GA とアンサンブル学習の動作確認

3.3.1 実験 1-1

まず，GA としての精度の向上を調べる．
適応度を各個体の accuracy とした．表 2 に実験のパラメータを示す．また，時間削減のため train_data は 5 万枚のうち各ラベル 200 枚，計 2000 枚を用い，これを 2 倍の 4000 枚に拡張して学習する．epoch 数は 30epochs とした．

3.4 実験 1-2

実験 1-1 で得られた上位 10 個体について train_data を 5 万枚すべてを使い学習をし，予測値を出した．epoch 数は 100epoch とした．上位 3 個体をアンサンブル学習し予測精度を求めた．対照実験として 3 個体を等確率で適用し，6 倍に水増して学習させた．

表 2: 実験 1-1 における GA のパラメータ

個体数	20
世代数	50
交叉率	0.9
突然変異率	
強度，確率 (遺伝子ごと)	0.06
順序 (染色体ごと)	0.1

また上位 10 個体に対して最もアンサンブルの精度が良くなるものを全探索した．

3.4.1 実験 1 結果と考察

図 1 に accuracy の推移を示す．表 3 に上位 3 個体の精度とアンサンブル学習の精度を示す．表 4 に 10 個体のアンサンブル学習のうち最も制度が良かったものを示す．色付きの個体が用いられた個体である．

実験 1-1 の最終的な accuracy は 0.8691 となった．図 1 から世代を重ねるほど個体が収束している様子が伺えるが，最良個体の精度はあまり変化がない．表 3 から複数のポリシーを一度の学習で用いるより，アンサンブル学習を行った方が良いことが分かる．表 4 からデータを減らした状態での適応度の計算でも良い個体が得られていることが分かり，また，アンサンブルに使われる個体は多ければ良いと言うわけではないことが分かる．

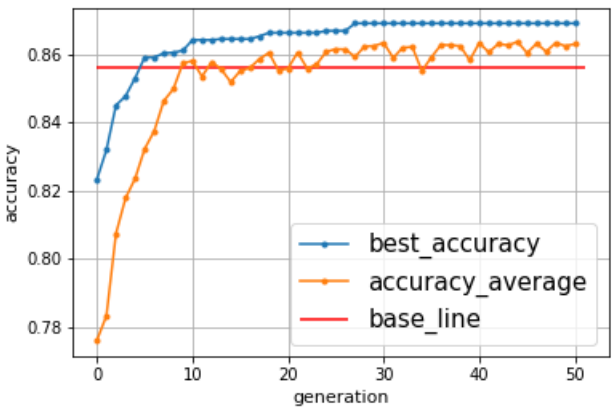

図 1: 実験 1-1 の accuracy の推移

表 3: 上位 3 個体のアンサンブル結果

1st	0.9044
2nd	0.9006
3rd	0.9037
ensemble	0.9141
control experiment	0.8963

表 4: 上位 10 個体のアンサンブル結果

1st	0.9044	6th	0.8624
2nd	0.9006	7th	0.8608
3rd	0.9037	8th	0.8565
4th	0.8386	9th	0.8740
5th	0.8541	10th	0.8596
ensemble		0.9205	

3.5 実験 2 アンサンブル学習を見据えた学習

実験 1 からアンサンブル学習の有用性が確かめられたので，それを見据えた設定を追加して実験を行った．表 5 にパラメータを示す．また，実験 1 と同様に得られた個体について train_data を 5 万枚すべてにして学習及び予測を行った．epoch 数は 30epochs,100epochs とした．

表 5: 実験 2 における GA のパラメータ

個体数	15
世代数	40
交叉率	0.9
突然変異率	
強度，確率 (遺伝子ごと)	0.06
順序 (染色体ごと)	0.1

3.5.1 多様性

アンサンブル学習でよい結果を出すためにはある個体が不正解である予測が別の個体では正解である，というような個体が必要となり，多様性が必要となる．そこで，多様性を確保するために順序の染色体 $\mathbf{Ch}_{\mathrm{ord}}$ が同じ個体が 3 つ以上ある時，強制的に突然変異させ 2 つ以下になるようにした．

3.5.2 適応度

個体 G_i の test_data10000 枚の予測値の集合を $\mathbf{pred}(i)$，accuracy を $f_{\mathrm{acc}}(\mathbf{pred}(i))$ とし，予測値の集合の集合 A に対するアンサンブルによる accuracy を $f_{\mathrm{ens\_acc}}(A)$ とする，また集団 1 のうち上位 3 個体の予測値の集合の集合を B としたとき，適応度 $fitness_i$ を

$$f_{\mathrm{ens\_acc}}(X) = f_{\mathrm{acc}}\left(\frac{1}{\#X}\sum_{\mathbf{a}\in X}\mathbf{a}\right)$$

$$fitness_i = \frac{1}{2^{n-1}}\sum_{A} f_{\mathrm{ens\_acc}}(A)$$

$$U = \{\mathbf{pred}(1), \mathbf{pred}(2), ..., \mathbf{pred}(n)\}$$

$$A = A \subset U \mid A \text{ have } \mathbf{pred}(i)$$

とした．

3.5.3 実験 2 の結果と考察

図 2 に accuracy の推移を示す．表 6 に最良値を示す．表 6 に train_data をすべて用いたものを示す．図 2 からはアンサンブル学習の精度向上はあまり見られない表 6，表 6 についてアンサンブル学習として 2%ほどの改善はみられるが実験 1-2 ほどの精度が出ていない．これはアンサンブルを目的としたために精度の良い個体を探索できなかったためであると考えらえる．また表から一部精度の低い個体は精度の向上に役立てる可能性がある．

表 6: 最良値 (13 世代目)

1st	0.8605	9th	0.8462
2nd	0.7899	10th	0.8574
3rd	0.8512	11st	0.8440
4th	0.8560	12nd	0.8543
5th	0.8547	13rd	0.8487
6th	0.8513	14th	0.8487
7th	0.8574	15th	0.8556
8th	0.8587	ensemble	0.8828

4 まとめと今後の課題

本実験では AutoAugment の有効性を確認したとともに，得られた個体からアンサンブル学習の有用性について検討した．AutoAugment は data_augment

表 7: 最終結果

1st	0.8723	9th	0.8521
2nd	0.8769	10th	0.8684
3rd	0.8787	11st	0.8738
4th	0.8778	12nd	0.8817
5th	0.8787	13rd	0.8809
6th	0.8727	14th	0.8755
7th	0.8782	15th	0.8755
8th	0.8719	ensemble	0.9048

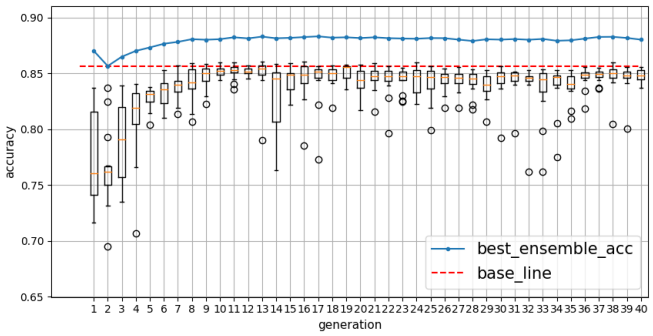

図 2: 実験 2 の accuracy の推移

なしの base_line よりも 5%ほど向上することができ，またアンサンブル学習によってさらに 2%の向上する結果となった．そのため，AutoAugment およびアンサンブル学習の有用性について確認できた．一方で，アンサンブル学習を行うための個体の選抜について今回用いた適応度では多様性をとることが難しいことが分かった．このことについて個体数や世代数が足りない，あるいは学習パラメータが原因であることも考えられる．本実験ではうまくいかなかったが，うまく適応度関数を設定したり，多様性を持つように複数の集団に分けその代表同士が多様性を持つように学習させたり，ある良個体をあらかじめ選択しそれに対し多様性を持つようにさせるなどやりようによってはより扱いやすいアンサンブル学習のための個体が得られそうではある．今後の課題は，多様性をもつ集団を作るためにアルゴリズムや適応度関数を改良することや，世代数や個体数を増やすための時間削減の工夫が挙げられる．