# 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, モデルやパラメータの自動化は Automated Machine Learning(:AutoML)[1], データ拡張の自動化は AutoAugment[2] 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を行う．　この複数という点からアンサンブル学習によってさらなる精度の向上がみられるのではないかと考えられる．　そこで, 本実験では AutoAugment の研究のとりかかりとして, 簡単なモデルに対する有用性を確かめ, アンサンブル学習に適用できるかの検討を行う．

## 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．

### 2.3　アンサンブル学習

複数の弱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．個々の精度が低く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 6 万枚の画像からなるデータセットであり，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 はて画素値操作 (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，確率はそれぞれ整数値コーディング

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 段階の度合いとし整数値で保存する．また，順序は順列コーディングで表す．

$$
\begin{aligned}
\mathbf{G} &= (\mathbf{Ch}_{\mathrm{m}}, \mathbf{Ch}_{\mathrm{p}}, \mathbf{Ch}_{\mathrm{o}}) \\
\mathbf{Ch}_{\mathrm{m}} &= (m_0, m_1, ...m_{15}) \\
\mathbf{Ch}_{\mathrm{p}} &= (p_0, p_1, ...p_{15}) \\
\mathbf{Ch}_{\mathrm{o}} &= (o_0, o_1, ...o_{15})
\end{aligned}
$$

- $m_0, m_1, ...m_{15}$：強度の遺伝子
- $p_0, p_1, ...p_{15}$：確率の遺伝子
- $o_0, o_1, ...o_{15}$：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．

#### 3.1.6 GA の流れ

1. 個体の初期化
2. 各個体 (list 型) をもとにデータ拡張関数を生成
3. 関数を基に訓練データを 2 倍に拡張させつつ CNN を学習
4. 得られた CNN(弱学習器) ごとにテストデータを入力し各個体に対する prediction($10 \times \quad 10000$ 行列) を得る
5. prediction から適応度を得る
6. 適応度をもとに次世代の個体を生成
7. 以下 2-6 を繰り返す

#### 3.1.7 実験 1 の適応度

個体 $i$ に対する予測値を $\boldsymbol{pred}(i)$, 適応度を $fitness_i$ とし，また，テスト画像の正解ラベルの行列を $\mathbf{y}_{\mathrm{test}}$ としたとき，以下のようにした，

$$
\begin{aligned}
&fitness_i = f_{\mathrm{acc}}(\boldsymbol{pred}(i)) \\
&f_{\mathrm{acc}}(\boldsymbol{pred}) = \\
&\frac{1}{10000}\sum_{j=0}^{9999} f_{\mathrm{if}}(\mathrm{argmax}(\boldsymbol{pred}[j]), \quad \mathrm{argmax}(\mathbf{y}_{\mathrm{test}}[j])) \\
&f_{\mathrm{if}}(a,b) = \begin{cases} 1 & (\mathrm{if} a = b) \\ 0 & (\mathrm{otherwise}) \end{cases}
\end{aligned}
$$

#### 3.1.8 実験 2 の適応度

$\boldsymbol{pred}(i)$ を用いた全アンサンブル学習の精度の平均を個体 $i$ に対する適応度とし，数式化すると以下のようになる．

$$
\begin{aligned}
&fitness_i = \frac{1}{2^{n-1}}\sum_{Y \subset C_j^*} f_{\mathrm{ens\_acc}}(Y) \\
&f_{\mathrm{ens\_acc}}(Y) = f_{\mathrm{acc}}(\{\hat{Y} | \hat{Y}[j][k] = \boldsymbol{Y}_1[j][k], ..., \boldsymbol{Y}_m[j][k], \\
&0 \leq j \leq 9999, 0 \leq k \leq 9\})(m = \#Y)
\end{aligned}
$$

$$
\begin{aligned}
&U = \{\boldsymbol{pred}(1), \boldsymbol{pred}(2), ..., \boldsymbol{pred}(n)\} \\
&U_i^{-1} = \{U \setminus \boldsymbol{pred}(i)\} \\
&C_j^* = \{\boldsymbol{pred}(i) \cup U_i^* | U_i^* \subset U_i^{-1}\}
\end{aligned}
$$

## 3.2 実験の流れ

- 訓練データを 2000 枚，エポック数を 30epochs の設定で GA を行い，個体を得る．
- 得られた個体に対し 50000 枚の訓練データを用いて 100epochs で学習，テストデータの予測を行う．

## 3.3 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 |

## 3.4 実験 1　GA とアンサンブル学習の動作確認

### 3.4.1 GA のパラメータ

表 2 に実験のパラメータを示す．

表 2: 実験 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 |

### 3.4.2 実験 1 におけるアンサンブル学習

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．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．

### 3.4.3 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 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．

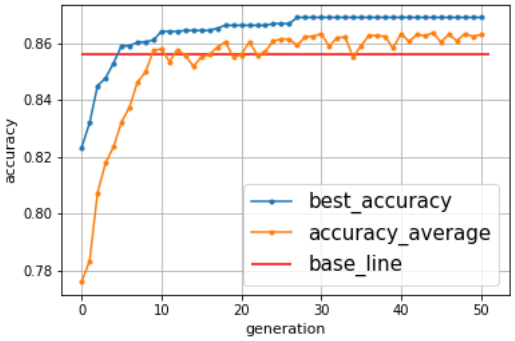


図 1: 実験 1-1 の accuracy の推移

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

## 3.5 実験 2:アンサンブル学習を見据えた GA

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．

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

アンサンブル学習において多様性のあるほうがよく，多様性を確保するために順序の染色体 $\mathbf{Ch}_o$ 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 実験 2　結果と考察

図 2 に accuracy の推移を示す．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 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 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: 最終結果

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

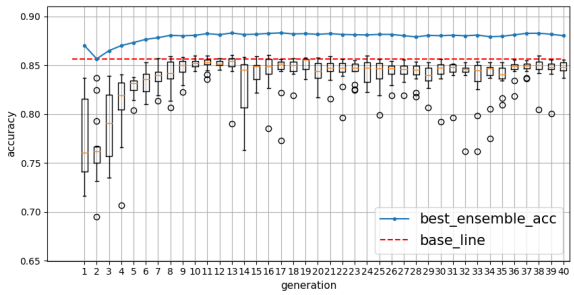


図 2: 実験 2 の accuracy の推移

# 4 まとめと今後の課題

本実験で AutoAugment は data_augment 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．

適応度関数をうまく設定したり，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るなどやりようによっては精度の良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．

# 参考文献

[1] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. Amc: Automl for model compression and acceleration on mobile devices. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 784–800, 2018.

[2] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.