G-07

オートエンコーダの進化的学習における遺伝的アルゴリズムと進化戦略の比較

Experimental Comparison of Genetic Algorithm and Evolution Strategy applied to Evolutionary Training of Autoencoders

岡田 英彦
Hidehiko Okada

1. はじめに

ニューラルネットの学習は一般的に誤差関数の勾配を利用した方法によって行われているが，一方で，その勾配を利用しない方法として進化的アルゴリズムを用いた方法も研究されている．著者はこれまでに，遺伝的アルゴリズム（GA）を用いたオートエンコーダの学習について実験結果を報告している[1]．しかし，進化的アルゴリズムにはGA以外にも様々な手法があり，それらの比較が課題の1つである．本稿では進化戦略（ES）を用いた実験を行い，遺伝的アルゴリズムの場合の結果と比較する．

2. オートエンコーダ

オートエンコーダ（AE）とは階層型ニューラルネットの一種であり，出力が入力と一致するように学習させることを特徴としている[2]．オートエンコーダを積み重ねて構成される積層オートエンコーダはディープニューラルネットの一種である．本研究における AE の構造を図1に示す．中間層の数は 1 であり，出力層ユニット数は入力層ユニット数と同一である．また一般的に，中間層ユニット数 M は入力（出力）層ユニット数 N よりも小さい値に設定される．

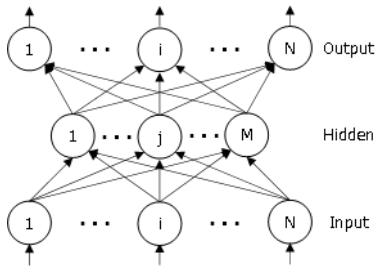


図 1: 本研究におけるオートエンコーダの構造

図 1 の AE の入出力関係を式(1)～式(5)に示す．この入出力関係は一般的な 3 階層パーセプトロンと同一である．

入力層（第 1 層）：

$$out_i^{(1)} = x_i \tag{1}$$

中間層（第 2 層）：

$$in_j^{(2)} = \theta_j^{(2)} + \sum_i w_{i,j}^{(2)}\, out_i^{(1)} \tag{2}$$

$$out_j^{(2)} = f(in_j^{(2)}) \tag{3}$$

出力層（第 3 層）：

$$in_i^{(3)} = \theta_i^{(3)} + \sum_j w_{j,i}^{(3)}\, out_j^{(2)} \tag{4}$$

$$out_i^{(3)} = f(in_i^{(3)}) \tag{5}$$

式(1)～式(5)における記号の意味は次の通りである．

x_i	i 番目の入力層ユニットへの入力値
$out_i^{(1)}$	i 番目の入力層ユニットからの出力値
$in_j^{(2)}$	j 番目の中間層ユニットへの入力値
$w_{i,j}^{(2)}$	i 番目の入力層ユニットから j 番目の中間層ユニットへの結合強度
$\theta_j^{(2)}$	j 番目の中間層ユニットのしきい値
$out_j^{(2)}$	j 番目の中間層ユニットからの出力値
$in_i^{(3)}$	i 番目の出力層ユニットへの入力値
$w_{j,i}^{(3)}$	j 番目の中間層ユニットから i 番目の出力層ユニットへの結合強度
$\theta_i^{(3)}$	i 番目の出力層ユニットのしきい値
$out_i^{(3)}$	i 番目の出力層ユニットからの出力値

式(3)および式(5)における関数fはユニット内部関数であり，本研究ではシグモイド関数$f(x) = 1/(1+e^{-x})$を用いる．

この AE の学習用データとして，N 次元実数ベクトルで表現されたデータが D 個得られているものとする．式(6)の$\boldsymbol{X}$が学習用データの集合を表しており，個々のデータ$\boldsymbol{x}_d$は式(7)のように N 個の実数$x_{d,1}, x_{d,2}, \ldots, x_{d,N}$で構成されるベクトルである．

$$\boldsymbol{X} = \{\boldsymbol{x}_d\}, d = 1,2,\ldots,\mathrm{D} \tag{6}$$

$$\boldsymbol{x}_d = (x_{d,1}, x_{d,2}, \ldots, x_{d,N}) \tag{7}$$

$x_{d,1}, x_{d,2}, \ldots, x_{d,N}$のそれぞれの値が入力層ユニット 1～N にそれぞれ入力されたとき，この入力に対する出力層ユニット 1～N の出力がそれぞれ$x_{d,1}, x_{d,2}, \ldots, x_{d,N}$とできるだけ一致するように，結合強度$w_{i,j}^{(2)}, w_{j,i}^{(3)}$およびしきい値$\theta_j^{(2)}, \theta_i^{(3)}$を学習させる．つまり，AE の学習は出力の N 次元ベクトルの値が入力の N 次元ベクトルの値を復元するように行われる．したがって，誤差逆伝搬法[3]を用いて AE を学習させる場合には，入力した$x_{d,i}$の値と出力された$out_i^{(3)}$の値のずれを誤差関数として定式化すればよい．この例を式(8), 式(9)に示す．個々のデータ$\boldsymbol{x}_d$に対する誤差の 1 次元あたりの平均がe_dであり，その誤差のデータ平均が e である．式(9)において，$x_{d,i}$は 0.0≦$x_{d,i}$≦1.0 を満たす実数値であり，$out_i^{(3)}$とe_dの値は 0.0<$out_i^{(3)}$<1.0，0.0<e_d<1.0 を満た

京都産業大学, Kyoto Sangyo University

す実数値である．e ≅ 0.0のとき，AE の出力はすべての$\boldsymbol{x}_d$の入力をほぼ完全に復元していることになる．

$$\mathrm{e} = \frac{1}{D}\sum_{d=1}^{D} e_d \tag{8}$$

$$e_d = \frac{1}{N}\sum_{i=1}^{N}(out_i^{(3)} - x_{d,i})^2 \tag{9}$$

3. ニューラルネットの進化的学習

進化的アルゴリズムを用いたニューラルネットの学習はニューロエボリューション（NE）と呼ばれている[4]．NE は大きく 2 種類に分けることができ，(A)ニューラルネットの構造（階層数やニューロン数など）は事前に決定しておいて結合強度としきい値の値だけを学習させる手法と，(B)ニューラルネットの構造自体も学習によって最適化する手法がある．本研究で用いる NE は前者(A)の手法である．図 1 の構造を持つ AE には，結合強度がMN + MN = 2MN個，しきい値がM + N個含まれており，合わせると2MN + M + N個含まれている．これを2MN + M + N次元の実数ベクトルとして表現し，進化的アルゴリズムにおける個体の遺伝子形（genotype）として扱う．個体の表現形（phenotype）は図 1 の AE である．つまり，AE に含まれる合計2MN + M + N個の実数パラメータに対して進化的操作を適用し，それらの実数パラメータの最適化を図る．各個体の評価値には式(8)に示した誤差関数を用い，式(6)に示した学習用データの集合$\boldsymbol{X}$を用いて評価値を求める．式(8)の誤差関数の値は小さいほど良いため，評価値が小さい個体ほどより良い個体と評価する．

3.1 ＧＡ

本研究における GA の手続きは次の通りである[1].

```
Step1: 初期化
Step2: 個体の評価
Step3: 終了判定
Step4: 子個体生成
Step5: Step2 に戻る
```

Step1 では，あらかじめ決められた数の個体のそれぞれについて genotype の値（2MN + M + N次元実数ベクトルの値）を初期化する．本研究では genotype の要素は AE の結合強度やしきい値であることを踏まえて，各要素の定義域をあらかじめ決めておく．

Step2 では，Step1 や Step4 で生成された新たな個体の適合度を評価する．これにより，集団に含まれるすべての個体の適合度が定まる．

Step3 では，あらかじめ決められた基準に基づいて終了判定を行う．本研究では世代数の最大値を決めておき，世代数がその値まで到達した時点で終了とした．

Step4 では，現世代の個体集団を親として用い，次世代の子個体の集団を生成する．Step4 の手続きを以下に示す．

```
Step4-1: エリート保存
Step4-2: 親 2 個体選択
Step4-3: 交叉
Step4-4: 突然変異
Step4-5: 終了判定
Step4-6: Step4-2 に戻る
```

Step4-1 では，現世代の個体集団のうち評価値が優れている（ランク上位の）個体を一定個数，次世代の子個体集団へとコピーする．エリートとして保存する個体の数は事前に決めておく（L とする）．集団内の個体数（集団サイズ）を P とすると，次世代の P 個体のうち L 個を現世代からのコピーによって獲得し，残りの(P-L)個の個体を交叉と突然変異によって生成する．

Step4-2 では，現世代の P 個体から，交叉に用いる親個体を 2 個体選択する．その選択方法として本研究ではトーナメント選択を用いる．

Step4-3 では，Step4-2 で選択された 2 個体の親を交叉させて子個体を生成する．本研究では交叉手法としてブレンド交叉法[5]を用いる．ブレンド交叉法では 1 回の交叉で出来る子個体は 1 つである．

Step4-4 では，Step4-3 で出来た子個体に突然変異を適用する．具体的には，genotype のベクトルに含まれる2MN + M + N個の要素のそれぞれを一定の確率でランダムな値に書き換える．このランダムな値の決め方は，その要素の値を Step1 において初期化した際の方法と同一である．

Step4-5 では，次世代の子個体が P 個すべて決定されたか判定し，Yes なら Step4 を終了する．

3.2 ＥＳ

本研究における ES はいわゆる(μ + λ)-ES であり，親個体数 μ，子個体数 λ，プラス戦略である．その手続きを以下に示す．

```
Step1: 初期化
Step2: 個体の評価
Step3: 終了判定
Step4: 淘汰
Step5: 子個体生成
Step6: Step2 に戻る
```

Step1 では，λ 個の個体のそれぞれについて genotype の値を初期化する．初期化の方法は GA の場合と同一である．

Step2 では，GA の場合と同様に，新たな個体の適合度を評価する．Step3 も GA の場合と同一である．

Step4 では，初回は Step1 で生成された λ 個の個体のなかから評価値が上位の μ 個を残してそれ以外を削除する．一方，2 回目以降は，後述する Step5 で生成された λ 個の個体とその生成に親として利用された μ 個の個体を合わせた μ + λ 個の個体のうち，評価値が上位の μ 個を残してそれ以外を削除する．

Step5 では，Step4 で生き残った μ 個の個体を親として用い，新しい子個体を λ 個生成する．Step5 の手続きを以下に示す．

```
Step5-1: 親 1 個体選択・複製
Step5-2: 摂動
Step5-3: 終了判定
Step5-4: Step5-1 に戻る
```

Step5-1 では，親個体 μ 個のなかからランダムに 1 個選択し，その複製を生成する．

Step5-2 では，Step5-1 で複製して出来た子個体の genotype のベクトルに含まれる2MN + M + N個の要素のそれぞれに摂動を加える．これにより，子個体の genotype の値は複製元の親個体の値から変化する．この摂動には問題に応じて適当な乱数が利用される．

Step5-3 では，子個体がλ個すべて生成されたか判定し，Yes なら Step5 を終了する．

4. 実験

4.1 方法

学習用データの集合$\boldsymbol{X}$として手書き数字のデータを用いて実験を行った．具体的には，UC Irvine Machine Learning Repository に掲載されている Optical Recognition of Handwritten Digits Data Set[1] を用いた．optdigits.tra ファイルに記録されたデータのなかから数字「0」～「9」の手書きデータをそれぞれランダムに 10 個ずつ抽出し，計 100 個のデータを用いた．さらに，学習後の AE の汎化能力を調べるためのテスト用データも学習用データと同じ個数だけランダムに抽出した．各データは 8×8＝64 個の要素で構成されており，各要素の値は 0～16 の整数である．本実験ではこれらの要素の値をすべて 16 で割り，区間[0.0, 1.0]内の実数値に変換して用いた．用いたデータを図 2 に示す．

データの次元数が 64 のため，AE の入力層および出力層のユニット数（図 1 の N の値）も 64 である．中間層のユニット数は，64 の 10%（7 個）から 90%（58 個）まで 10%刻みで計 9 通りとした．この入力層と中間層のユニット数の比を以下では「ユニット比」と表すことにする．

ユニット比 90%（中間ユニット数 58 個）の場合，AE に含まれる結合強度およびしきい値の総数は$2 * 58 * 64 + 58 + 64 = 7546$個である．したがって，進化的学習における genotype は 7546 次元実数ベクトルとなる．

本実験における GA および ES の設定を以下に示す．これらの設定は，著者のこれまでの経験と，試行錯誤的な予備実験に基づいて決定した．

・GA/ES 共通の設定
  ・genotype の各要素の定義域： [-5.0, 5.0]
  ・世代数：10,000

・GA の設定
  ・個体数：100
  ・エリート保存数：2
  ・トーナメントサイズ：10
  ・交叉方法：ブレンド交叉（α = 0.5）
  ・突然変異確率：1/(genotype の長さ)

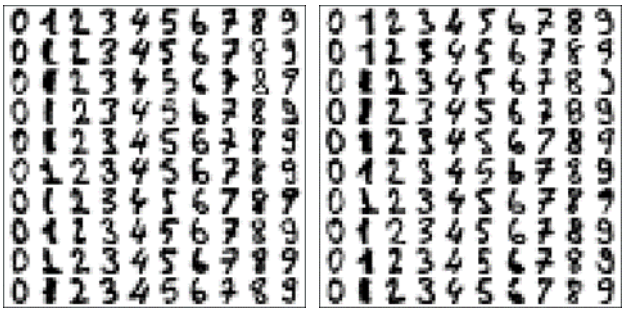
・ES の設定
  ・個体数および戦略：(10+100)-ES
  ・摂動量：0.1 *標準正規乱数

4.2 結果と考察

同じ設定の学習試行をそれぞれ 10 回実行し，学習用データに対する誤差をそれぞれ求めた．その 10 回分の平均と最良の値を表 1 および図 3 に示す．ユニット比が大きくなるほど誤差の値は概ね減少している．また，GA と ES の間で比較すると，誤差は概ね ES のほうが小さいことがわかる．つまり，GA よりも ES のほうが学習用データをより精度よく復元可能な AE を獲得できたことがわかる．

次に，学習によって得られた AE の汎化誤差を表 2 に示す．ただしこれらの値は，同じ設定で実施した 10 試行のうち学習誤差が最良であった 1 試行の学習済み AE を用いて求めた値である．ユニット比は 10%，50%，90%の 3 通りのみを示している．さらに，学習後の AE が学習用データ

[1] http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits



(a)学習用　　(b)テスト用

図 2: 実験で用いた学習用／テスト用データ

表 1: ユニット比と学習誤差(%)

	GA		ES	
Ratio	average	best	average	best
10%	18.27	18.20	13.80	13.10
20%	17.89	14.87	12.86	12.31
30%	17.78	12.18	11.63	11.38
40%	16.73	10.32	11.01	10.76
50%	17.92	10.90	10.34	9.99
60%	14.68	8.53	9.41	8.87
70%	17.02	8.44	8.74	8.50
80%	10.59	8.10	8.40	7.98
90%	12.38	8.00	7.90	7.33

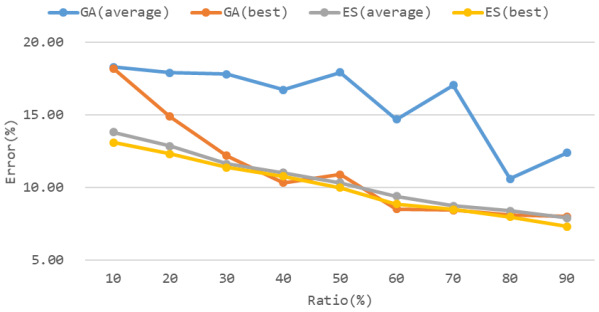


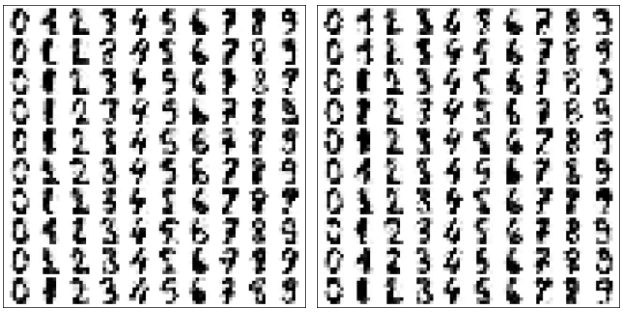
図 3: ユニット比と学習誤差(%)

表 2: 学習誤差とテスト誤差(%)

	Ratio	10%	50%	90%
GA	Train	18.20	10.90	8.00
	Test	18.61	12.40	9.30
ES	Train	13.10	9.99	7.33
	Test	14.34	10.99	8.07

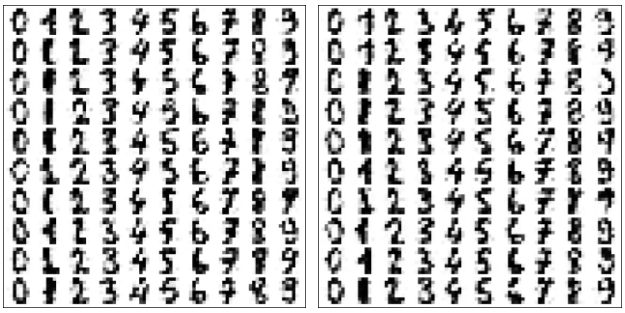
およびテスト用データに対してそれぞれ出力した値を図 4 および図 5 に示す．ただしこの AE のユニット比は 90%である．図 4(a)は，GA によって学習させた AE が図 2(a)の入力に対して出力した値を示しており，両者のずれが学習誤差である．また図 4(b)は，GA によって学習させた AE が図 2(b)の入力に対して出力した値を示しており，両者のずれがテスト誤差である．表 2 から，GA・ES いずれの場合においてもテスト誤差は学習誤差より大きいことがわかる．しかし図 4・5 から，学習していない図 2(b)のデータもある程度再現できており，過学習による汎化能力の低下は起こっていないと言える．

以上から，本研究における AE の進化的学習においては，GA よりも ES のほうがより適した手法であることがわかる．GA と ES を比べると，GA は広域的探索，ES は局所的探索



(a)学習用データに対する出力　(b)テスト用データに対する出力

図 4: GA で学習させた AE の出力（ユニット比 90%）



(a)学習用データに対する出力　(b)テスト用データに対する出力

図 5: ES で学習させた AE の出力（ユニット比 90%）

がより得意な手法である．したがって，今回の実験結果は，AE の進化的学習には局所的探索を重視したアルゴリズムがより適していることを示唆している．このことは今後さらに検証してゆく必要がある．

5. まとめ

遺伝的アルゴリズムおよび進化戦略を用いてオートエンコーダの進化的学習を実験し，その結果を比較した．様々な中間層ユニット数の設定で実験したところ，いずれの設定においても概ね進化戦略のほうが遺伝的アルゴリズムより学習誤差が小さく，かつ，一定の汎化能力もあることがわかった．この結果から，オートエンコーダの進化的学習には広域的探索よりも局所的探索を重視したアルゴリズムがより適していることがわかった．今後は他の進化的アルゴリズムのさらなる比較評価，ニューラルネットの学習に最適化された進化的アルゴリズムの改良などが課題である．

参考文献

[1] 岡田, 遺伝的アルゴリズムによるオートエンコーダの進化的学習, 情報処理学会関西支部 2016 年度支部大会, G-07, 2016.

[2] G.E. Hinton and R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, *Science*, 313(5786), 504-507, 2006.

[3] D.E. Rumelhart, G.E. Hinton and R.J. Williams, Learning representations by back-propagating errors, in *Neurocomputing: Foundations of Research*, J.A. Anderson and E. Rosenfeld (Eds.), MIT Press, 696-699, 1988.

[4] D. Floreano, P. Durr and C. Mattiussi, Neuroevolution: from architectures to learning, *Evolutionary Intelligence*, 1(1), 47-62, 2008.

[5] L.J. Eshelman and J.D. Schaffer, Real-coded genetic algorithms and interval-schemata, in D.L. Whitley (ed), *Foundation of Genetic Algorithms 2*, 187-202, 1993.