

要旨

本研究は、ニューラルネットワーク（バックプロパゲーション法）を最適化問題として定式化し、確率的学習オートマトンをニューラルネットワークに対して適用し、その収束を調べる。そのため、シナプシス荷重をオートマトンの環境と見なし、環境によってシナプシス荷重の修正量を行動として定めるオートマトンを作成した。数値計算では本方法と、GA等その他の定式化との収束を比較する。

1. はじめに

設計の自動化に於ける知的支援として、AI、ニューラルネットワーク等が利用され始めており、これまでにAIとニューラルネットワークを協調させた概念設計の設計モデルが開発されている。しかし、ニューラルネットワークの学習においては、温度の設定、勾配ベクトルの係数などのパラメータ設定によって学習時間に大きな差が生じることがあり、より効率的な学習法の開発が必要とされている。

本研究は、ニューラルネットワークの学習を最適化問題であると見なし、確率的学習オートマトン(SLA: Stochastic Learning Automata)を用いたニューラルネットワークの学習法を示す。又、数値計算実験を行い、他の方法との収束の比較を報告する。

2. バックプロパゲーション法の最適問題としての定式化

ニューラルネットワークの一方法である三層バックプロパゲーション法(Fig. 1)は、最適問題として以下に定式化される。

$$\text{minimize } E(W, V, \theta, \gamma) = 1/2(\mathbf{O}-\mathbf{T})^T(\mathbf{O}-\mathbf{T})$$

$$W, V, \theta, \gamma \quad (1)$$

$$\text{subj. to } \mathbf{H} = \phi(\mathbf{W}\mathbf{I}), \quad \mathbf{O} = \phi(\mathbf{V}\mathbf{H})$$

ここで、 \mathbf{W} は入力層と中間層のシナプシス荷重の行列、 \mathbf{V} は中間層と出力層のシナプシス荷重の行列、 $\mathbf{I}, \mathbf{H}, \mathbf{O}$ はそれぞれ入力層、中間層、出力層のユニットベクトルであり、 θ, γ は中間層、出力層のスレッシュホールドベクトルである。また、関数 ϕ はシグモイド関数とする。バックプロパゲーション法は式(1)を満たすような $\mathbf{W}, \mathbf{V}, \theta, \gamma$ を最大傾斜法によって求められていると考えられる。このとき、これらの値の更新量は、

$$\Delta W = \alpha \frac{\delta E}{\delta W} \quad \Delta V = \alpha \frac{\delta E}{\delta V} \quad (2)$$

$$\Delta \theta = \beta \frac{\delta E}{\delta \theta} \quad \Delta \gamma = \beta \frac{\delta E}{\delta \gamma} \quad (3)$$

で求められる。

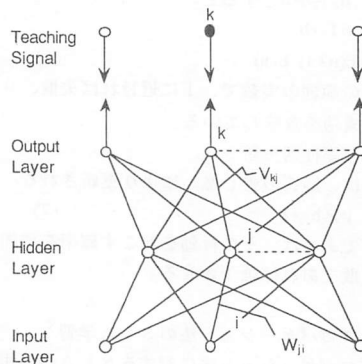


Fig. 1 Three layer back-propagation model

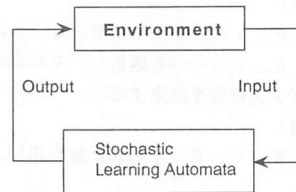


Fig. 2 Automata model.

3. 確率的学習オートマトン (SLA)

一般に学習オートマトンとは、環境と相互関係を持ち、行動集合から確率的にある行動を選択し、環境からその応答を受け取るモデルである。学習オートマトンは環境からの応答に基づき、行動確率行列をHebb則に従い更新する。この動作を繰り返し進めていくことにより、最終的に環境から失敗の返答を受けとりにくい動作を選択するようになる。以下に確率的学習オートマトンの定式化を行なう。

Φ : 状態決定

ニューロンのシナプシス結合係数またはスレッシ

ヨウルド値から離散状態Sを決定する。本実験では4種類のオートマトンを用いて、W, V, θ , γ の計4種類の係数を同時に最適化する。

$$\Phi : W, V, \theta, \gamma \rightarrow S \quad (4)$$

Ψ : 行動決定

オートマトンの行動を確率的に決定する。行動確率行列をPとすると、

$$\Psi : S \times P \rightarrow O \quad (5)$$

ここで、Oはオートマトンの行動でありシナップシス荷重の更新量を意味する。

λ : 応答

シナップシス荷重を更新した後、教師信号との誤差Eを調べる。応答をbとすると

$$\lambda : \delta I \rightarrow b \quad (6)$$

応答は[0,1]の範囲の実数で、1に近ければ失敗、0に近ければ成功を意味している。

Γ : 行動確率行列更新

行動確率は、応答の善し悪しにより更新される。

$$\Gamma : P \times b \rightarrow P \quad (7)$$

成功の応答であれば、その行動を起こす確率を増加させ、逆に失敗であれば低下させる。

4. バックプロパゲーション法のSLA学習

バックプロパゲーション法に対するSLAの適用を以下のように行う。

1) 環境 (入力)

環境として、W、V、 θ 、 γ を使用する。入力としては、W、V、 θ 、 γ の一つを選択し、それらの中の一つのシナップシス荷重を決定する。

2) 行動 (出力)

行動として、W、V、 θ 、 γ の修正量を用い、以下のようにする。

$$a1 = \text{Random}[-0.001, 0.001]$$

$$a2 = \text{Random}[-0.01, 0.01]$$

$$a3 = \text{Random}[-0.1, 0.1]$$

$$a4 = \text{Random}[-1.0, 1.0]$$

$$a5 = \text{Random}[-10.0, 10.0]$$

ここで、Random[a, b]は、a, bの中で正規乱数を取る関数である。

3) 応答

SLAの評価は、ニューラルネットワークの誤差E(t)で行う。すなわち、 $\Delta E(t+1) = E(t+1) - E(t)$ とし、 $\Delta E(t+1) \leq 0$ ならば成功、 $\Delta E(t+1) > 0$ ならば失敗とする。ここで、tは時間を示す。この評価により応答bを以下のように定める。

if $\Delta E(t+1) > 0$, then $b = 1$,

if $\Delta E(t+1) \leq 0$, then

$$b = 1 - \frac{\Delta E(t+1) - \Delta E(t)}{\Delta E(t+1) + \Delta E(t)}$$

(4) 確率行列の更新

確率行列の更新は以下のように行う。

if $s(t+1) = s_u$ and $a(t+1) \neq a_k$, then

$$P_{uk}(t+1) = P_{uk}(t) + \mu b P_{uk}(t) - \lambda(1-b)P_{uk}$$

if $s(t+1) = s_u$ and $a(t+1) = a_k$, then

$$P_{uk}(t+1) = P_{uk}(t) - \mu b(1 - P_{uk}(t)) + \lambda(1-b)(1 - P_{uk}(t))$$

otherwise, $P_{uk}(t+1) = P_{uk}(t)$

(5) 数値計算実験

バックプロパゲーションにおけるEXOR学習実験の収束実験を行った。又、他の学習法との比較をFig.3に示す。

(6) おわりに

SLAをバックプロパゲーション法に適用し、その収束を調べた。その収束は μ 、 λ によって学習が行われるときと、行われなるときがある。他の方法と比較すると、GAによる改善の方が良いことが分かり、今後GAと組み合わせた収束を調べたい。又、階層化オートマトンによる収束も試みる必要がある。

参考文献

- 1) 桐谷 滋他; 入門と実習ニューロコンピュータ (1989)
- 2) K.S.Narendra and M.A.L.Thathachar; Learning Automata An Introduction, Prentice Hall(1989)
- 3) 古川正志; GAによるニューラルネットワークの学習、日本機械学会第2回設計工学・システム工学部門講演会講演論文集

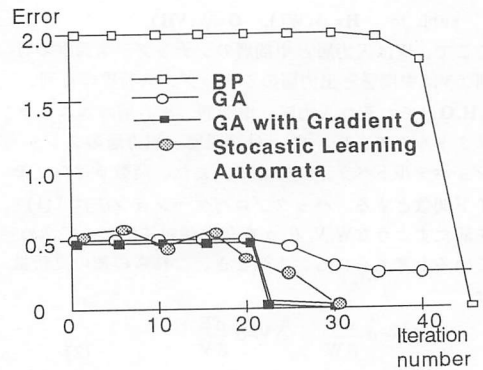


Fig.3 Numerical calculation result.