階層型ニューラルネットワークにおける マルチブランチ構造とその局所性

非会員 山下 貴志* 正員 平澤 宏太郎*
正員 古月 敬之*

Multi-Branch Structure and its Localized Property in Layered Neural Networks

Takashi Yamashita*, Non-member, Kotaro Hirasawa*, Member, Takayuki Furuzuki*, Member

Neural networks (NNs) can solve only a simple problem if the network size is too small, on the other hand, if the network size increases, it costs a lot in terms of memory space and calculation time. So, we have studied how to construct the network structure with high performances and low costs in space and time. A solution is a multi-branch structure. Conventional NNs use the single-branch for the connections, while the multi-branch structure has multi-branches between nodes. In this paper, a new method which enables the multi-branch NNs to have localized property is proposed. It is well known that RBF networks have localized property that makes it possible to approximate functions faster than sigmoidal NNs. By using the multi-branch structure having localized property of RBF networks, NNs could obtain high performances keeping the lower costs in space and time. Simulation results of function approximations and a classification problem illustrated the effectiveness of multi-branch NNs.

キーワード：ニューラルネットワーク，マルチブランチ構造，局所性

Keywords: Neural networks, Multi-branch structure, Localized property

1. はじめに

階層型ニューラルネットワークの学習に最も一般的に用いられる手法は、誤差逆伝播法⁽¹⁾である。誤差逆伝播法には幾つかの問題点もあるが、階層型ニューラルネットワークにとって、強力な学習アルゴリズムである。その問題点の1つに学習時間が長いということがある。これは、計算コストが大きいということを意味する。そこで、人工ニューラルネットワークを設計する際に、ネットワークの最適なサイズ（中間層ノード数）を選択する1つの基準は、計算コストとなる。より小規模サイズのネットワークがより少ない計算コストを必要とすることは明確である。しかし、小規模サイズのネットワークでは、単純な問題しか解くことは出来ない。いくら計算コストが小さくても与えられた問題を解くことが出来なければ無意味であるため、計算コストが増加しても、ネットワークサイズを大きくせざるを得

ないことがある。

このように、ネットワークのサイズは計算コストと強い相関関係にあり、また、ネットワークのサイズがその性能に大きな影響を与えている。ネットワークのサイズと計算コストの強い関係は変えられないが、ネットワークのサイズとその性能の関係は改善する余地がある。本論文では、その方法の1つとして、マルチブランチニューラルネットワークを提案している。

Universal Learning Networks (ULNs)⁽²⁾⁽³⁾は各種のニューラルネットワーク構造の一般的な枠組みを与えており、その中にマルチブランチ構造も含まれている。このマルチブランチ構造では、任意の数のブランチがノード間に接続しており、主に動的ネットワークにおいて時間遅れを設定するために用いられている⁽³⁾。本論文では、ノード数を削減した小さいサイズのネットワークを実現するためにマルチブランチ構造を用いる。従来の階層型のニューラルネットワーク (NNs) では、非線形処理の点で中間層ノードはかなり重要な役割を持っている。それに対し、マルチブランチニューラルネットワークではノード以外のブランチにも同様に非線形処理の役割を持たせてノードの非線形処理の負担を軽減している。

また、提案手法のマルチブランチニューラルネットワーク

* 早稲田大学大学院情報生産システム研究科
〒 808-0135 福岡県北九州市若松区ひびきの 2-7
Graduate School of Information, Production and Systems,
Waseda University,
2-7 Hibikino Wakamatsu-ku Kitakyushu-shi Fukuoka
808-0135

では、マルチブランチ構造において局所性を考慮している。局所性を利用している手法として、Radial Basis Function (RBF) ネットワーク^{(4)~(6)}がある。RBF ネットワークは入力空間に対して RBF を設定して、局所性を実現している。これにより、従来のニューラルネットワークよりも短い学習ステップで滑らかに関数を近似することが知られている。局所性は効率良く学習するための性質の 1 つであり、提案手法ではマルチブランチ構造にこの性質を利用している。

提案手法では、マルチブランチ構造と局所性を利用して、学習効率の高いニューラルネットワークを実現している。従来手法との比較のために、シミュレーションとして関数近似と分類問題について取り上げ、提案手法の有効性を示している。

本論文の構成は次の通りである。まず、第 2 章で誤差逆伝播法の計算コストについて説明し、第 3 章ではマルチブランチニューラルネットワークについて、第 4 章ではマルチブランチの局所性について説明する。そして、第 5 章でシミュレーションについて記述し、第 7 章で結論を述べる。

2. 誤差逆伝播法の計算コスト

誤差逆伝播法の計算コストについて、パラメータやノードの数がどのように関わっているのかを整理する。計算コストには、時間的コストに関わる計算量と空間的コストに関わるメモリ使用量を含むが、ここでは、主に計算量を示している。

勾配法を用いて重みなどのパラメータを更新する際、

$$\lambda_m \leftarrow \lambda_m - \gamma \frac{\partial^+ E}{\partial \lambda_m} \dots \dots \dots (1)$$

を用いる⁽²⁾。ここで、 E は評価関数、 λ_m はパラメータ、 γ は学習係数である。誤差逆伝播法においては、 $\frac{\partial^+ E}{\partial \lambda_m}$ を

$$\begin{aligned} \frac{\partial^+ E}{\partial \lambda_m} &= \frac{\partial^+ E}{\partial h_j} \frac{\partial h_j}{\partial \lambda_m} + \frac{\partial E}{\partial \lambda_m} \\ &= \delta(j) \frac{\partial h_j}{\partial \lambda_m} + \frac{\partial E}{\partial \lambda_m}, \dots \dots \dots (2) \end{aligned}$$

$$\delta(j) = \sum_k \left[\delta(k) \frac{\partial h_k}{\partial h_j} \right] + \frac{\partial E}{\partial h_j} \dots \dots \dots (3)$$

により δ を逆伝播して計算する。ただし、 h_j はノード j の出力値、 k はノード j から接続するノードの添字を表す。

つまり、パラメータ更新における全体の計算量は $\delta(j)$ ($j = 1, 2, \dots, J$) と $\frac{\partial h_j}{\partial \lambda_m}$ ($m = 1, 2, \dots, M$) の計算量に依存する (通常 $\frac{\partial E}{\partial \lambda_m}$ は零であり、また、 $\frac{\partial E}{\partial h_j}$ も出力ノードを除いて通常零である)。ここで、 J はノード数 (入力層を除く)、 M はパラメータ数である。また、 $\delta(j)$ と $\frac{\partial h_j}{\partial \lambda_m}$ の計算量は、ノード数とパラメータ数に依存している。このことから、 $\frac{\partial^+ E}{\partial \lambda_m}$ の計算量はノード数とパラメータ数に依存する。特に、ノード数を削減することにより、 $\delta(j)$ の個数が削減する。従って、ノード数を削減することは計算コスト削減に効果的である。

図 1 ノード間におけるマルチブランチ構造

Fig. 1. Multi-branch structure between nodes.

3. マルチブランチニューラルネットワーク

3.1 マルチブランチの構造 本節では、中間層ノード数を少なくできるマルチブランチニューラルネットワークについて述べる。

従来のニューラルネットワークはその接続にシングルブランチを用いており、また、それぞれのブランチには 1 つの結合重みが設定されている。これを拡張したものが図 1 に示したマルチブランチ構造である。マルチブランチ構造では、ノード間に任意の数のブランチを設定できる。

従来のニューラルネットワークでは、中間層のノード数を増加することにより、その表現能力の向上を図っている。しかしながら、第 2 章で示したように、多数のノードは学習に要する計算コストを大きくする。ニューラルネットワークの技術を実際に様々な問題に応用する際、計算コストは出来るだけ削減されることが望ましい。マルチブランチ構造は、ノード数を削減し、ブランチ数を増加させて計算コストを小さく抑えつつ、ネットワークの表現能力を向上させることに特徴がある。

また、マルチブランチニューラルネットワークでは、ブランチの信号処理を拡張する。式 (4) は従来のブランチの信号の処理を示す。ただし、 α_j はノード j への入力値を表す。

$$\alpha_j = \sum_{i \in JF(j)} w_{ij} h_i + \theta_j, \dots \dots \dots (4)$$

ここで、 $JF(j)$: ノード j へ接続するノードの添字の集合、 w_{ij} : ノード i からノード j へのブランチの結合重み、 h_i : ノード i の出力値、 θ_j : ノード j の閾値、である。

一方、マルチブランチニューラルネットワークの信号処理は、式 (5) で表される。

$$\alpha_j = \sum_{i \in JF(j)} \sum_{p \in B(i,j)} w_{ij}(p) g_{ij}(p, h_i) + \theta_j, \dots \dots (5)$$

ここで、 $B(i, j)$: ノード i からノード j へのブランチの添字の集合、 $w_{ij}(p)$: ノード i からノード j への p 番目のブランチの結合重み、 $g_{ij}(p, h_i)$: ノード i からノード j への p 番目のブランチにおけるノード出力 h_i の非線形変換値、である。

また、式 (5) の表現は式 (4) の形式を含んでおり、 $g_{ij}(p, h_i) = h_i, |B(i, j)| = 1$ のときに式 (5) は式 (4) と等しくなる。

図2 線形関数とシグモイド関数

Fig. 2. Shapes of linear and sigmoidal function.

3.2 マルチブランチ構造での計算コスト ここで、マルチブランチ構造による計算コスト削減について記述する。第2章で説明したように、 $\delta(j)$ ($j = 1, 2, \dots, J$) と $\frac{\partial h_i}{\partial \lambda_m}$ ($m = 1, 2, \dots, M$) について考える。

マルチブランチ構造のノード数をシングルブランチ構造と同じく J とすると、マルチブランチ構造のパラメータ数 M は当然ブランチ数の増加に依存して多くなる。逆に、同じパラメータ数 M で両構造を比較すると、マルチブランチ構造のノード数 J は少なくなる。つまり、同じ総数のパラメータについて $\frac{\partial^+ E}{\partial \lambda_m}$ を計算する場合、 $\delta(j)$ の総数が少なくなる分、マルチブランチ構造の全体の計算量は少なくて済む。これにより、ノード数を削減してブランチ数を増やすことは、計算コストの削減となる。

3.3 マルチブランチの非線形変換 ノード数を削減したマルチブランチニューラルネットワークが、従来のニューラルネットワークと同等あるいはそれ以上の性能を持つためのポイントはブランチ上の非線形変換の表現である。

多くのニューラルネットワークのノード関数にはシグモイド関数が用いられている。シグモイド関数には図2に示すように、ほぼ線形領域の部分がある。このため、非線形性を得られにくい場合もある。実際、汎化能力を強化するために学習を行うと、結合重みの絶対値は小さくなる傾向があるので、ノードの非線形変換にはこの線形領域が用いられるケースがよくある。また、ブランチにおける重み付けも線形スケールになっている。

つまり、これらのことから従来のシグモイド型ニューラルネットワークは多数の線形領域を使用する場合がある。このことは、ノード数の少ないニューラルネットワークでは、非線形性の強い問題を解くのが困難であることを意味する。このことを考慮すると、マルチブランチニューラルネットワークの各ブランチでの非線形変換は、非線形性を得るという点で大きな意味を持っている。実際、べき乗関数、正弦波関数などの非線形変換を用いたマルチブランチ構造でもその効果は確認されている⁽⁷⁾。

以上のように、マルチブランチでの非線形処理によって、マルチブランチニューラルネットワークではノードの数が少なくてもネットワークの表現能力は向上する。

4. マルチブランチの局所性

4.1 マルチブランチの非線形関数 提案手法のマル

チブランチニューラルネットワークでは、式(5)の関数 $g_{ij}(p, h_i)$ として式(6)に示すガウス関数を用いる。

$$g_{ij}(p, h_i) = \exp \left(-\frac{(h_i - c_{ij}(p))^2}{\sigma_{ij}^2(p) + L} \right) \dots \dots \dots (6)$$

$c_{ij}(p)$ はノード i からノード j への p 番目のブランチのガウス関数の中心を表すパラメータ、 $\sigma_{ij}(p)$ はノード i からノード j への p 番目のブランチのガウス関数のばらつきを表すパラメータである。 L は汎化能力を調整するための正の定数である。

このガウス関数を設定することにより、ノード間の非線形処理に局所性を設けることが出来る。つまり、マルチブランチへの入力信号に応じて、あるブランチを主に使用し、他のブランチは使用しない、という処理が出来る。これにより、各ブランチは全領域の入力信号に出力する必要がなく、それぞれ局所的な領域に出力すればよいことになる。これにより効率の良い学習となり、適切な表現能力を素早く獲得できる。

マルチブランチが持つ構造的な意味は、モジュール的なノード間の信号処理と考えることもできる。一般に、タスクは小さな部分タスクに分解されたほうが、より適切に処理ができる。それぞれの部分タスクは全体のタスクよりも見通しよく処理されるからである。このモジュール性は工学の広い分野で受け入れられている計算原理である。

マルチブランチにおいて、ノード間の信号処理をどのような部分タスクに分解するか、すなわち、式(5)の関数 $g_{ij}(p, h_i)$ をどのように設定するか検討する必要がある。一般性のある形式のひとつは、処理する信号の領域に応じて部分タスクを設定する形式である。ノード間の各ブランチが、それぞれ異なる信号領域で局所的処理をするようなモジュール的構造である。これは、RBF ネットワークと類似の構造になる。このような理由から、マルチブランチにガウス関数を設定することは適していると考えられる。

ただし、提案手法においては、ブランチの入力信号の範囲に注意する必要がある。ガウス関数の値がほぼ零となる入力に対しては信号は伝播されないからである。従って、訓練データの入力範囲を基にしてネットワークの入力値を正規化しなければならない。

4.2 学習パラメータ 提案手法では、学習パラメータとして、結合重み $w_{ij}(p)$ に加えてガウス関数の中心 $c_{ij}(p)$ 及びばらつき $\sigma_{ij}(p)$ のパラメータがある。

各種パラメータは、式(1)の勾配法により、更新される。従って、各々の学習係数が必要になる。ここで、注意すべきことは、 $c_{ij}(p)$ 及び $\sigma_{ij}(p)$ は $w_{ij}(p)$ に比較して小さな更新量にすべきであるという点である。ガウス関数の中心 $c_{ij}(p)$ 、ばらつき $\sigma_{ij}(p)$ を結合重みと同じ学習係数で変更すると入出力関係が大きく変わってしまうからである。また、 $\sigma_{ij}(p)$ の値は汎化能力に大きな影響を与えるので、より小さい学習係数で更新する必要がある。

以上のことから、 $w_{ij}(p)$ 、 $c_{ij}(p)$ 、 $\sigma_{ij}(p)$ の学習係数を

それぞれ $\gamma, \gamma_c, \gamma_\sigma$ とすると, $\gamma > \gamma_c > \gamma_\sigma$ となるように設定した。例えば, バッチモードで更新する場合, $w_{ij}(p), c_{ij}(p), \sigma_{ij}(p)$ のそれぞれの更新量の絶対値の最大値 ($\Delta w_{max}, \Delta c_{max}, \Delta \sigma_{max}$) の比が $1:r_c:r_\sigma$ となるように γ_c と γ_σ の値を適応的に決めることができる。第5章でのシミュレーションでは, 5・2 で $r_c = 0.5, r_\sigma = 0.2$, 5・3 で $r_c = 0.4, r_\sigma = 0.1$ を使用している。

4・3 RBF ネットワークとの相違点 提案手法のマルチブランチニューラルネットワークでは各ブランチにガウス関数を用いている。この点で, RBF ネットワークと同じ局所性の特徴を持つ。

両者の相違点は, RBF ネットワークでは, 入力空間に対して RBF を設定しているのに対し, マルチブランチニューラルネットワークでは, 各ブランチに対して RBF を設定している点である。従って, RBF ネットワークでは, 教師データのパターン数が増加すると RBF の個数は多くなる。一方, マルチブランチニューラルネットワークでは, 中間層ノードのシグモイド関数とブランチ上のガウス関数を組み合わせてネットワークの表現能力を実現するため, 教師データのパターン数が増えてもブランチの数は数個で十分である。

また, RBF ネットワークでは, 入力空間に対して RBF を設定しているので, 各 RBF では中心ベクトル, 共分散行列などの多次元情報を処理する必要がある。これに対し, マルチブランチニューラルネットワークでは, ブランチ上に RBF を設定しているので, スカラー処理で対応できる。

5. シミュレーション

5・1 シミュレーション条件 提案手法を従来手法と比較するために, 関数近似と Two Spirals 分類問題のシミュレーションを行った。ここで, シミュレーション条件を説明する。ここに示した条件は, 特に記述がない場合は提案手法と従来手法両方に対して当てはまる。

ネットワーク構成に関しては, 3 階層のニューラルネットワークを考え, ノード関数はシグモイド関数 $\frac{1-e^{-\alpha_j}}{1+e^{-\alpha_j}}$ を用いている。ただし, 関数近似問題を扱うときに限り出力層ノードは入力の重み付き総和のみであり, シグモイド関数ではない。また, 訓練データを線形スケーリングにより $[-1.0:1.0]$ の範囲に正規化し, テストデータもそれを基にして正規化して入力している。各シミュレーション結果はパラメータ数が同じ条件で比較したものである。

誤差逆伝播アルゴリズムにより学習し, 従来手法では, 評価関数を $E = \sum (h_o - h'_o)^2 + \eta \sum w_{ij}^2$ とした。 h_o はネットワークの出力値であり, h'_o は教師信号値を表す。また, 重み減衰⁽⁸⁾ のためのペナルティ項を用いており, η は正則化係数である。 η を調整することで汎化能力を強化することができる。提案手法では, ガウス関数のばらつき $\sigma_{ij}(p)$ が小さくなりすぎて汎化能力が低下しないように, $E = \sum (h_o - h'_o)^2 + \eta \sum \frac{w_{ij}^2(p)}{\sigma_{ij}^2(p)}$ とした。

図 3 教師関数

Fig. 3. Target functions for approximation.

パラメータ更新に関しては, 慣性項⁽⁹⁾ を利用した更新式: $\lambda_m(s) \leftarrow (1 - \alpha)\Delta\lambda_m(s) + \alpha\Delta\lambda_m(s - 1)$ によって更新した。 $\lambda_m(s)$ は s ステップの学習パラメータであり, $\Delta\lambda_m(s)$ は更新量を表している。係数 α は 0.95 を用いた。

パラメータの初期値に関しては, $w_{ij}(p)$ と θ_j は $[-1:1]$ の範囲でランダムに初期値を決め, $c_{ij}(p)$ はノードの出力範囲 $[-1:1]$ に各ブランチ上のガウス関数が均等に分布するように初期値を決めた。 $\sigma_{ij}(p)$ は $[0.4:0.6]$ の範囲でランダムに初期化される。

学習係数に関しては, $w_{ij}(p)$ と θ_j に対する学習係数 γ は 0.03 を初期値として, 適応的に変化する。すなわち, 更新の結果 E が減少の場合は, $\gamma \leftarrow 1.05\gamma$ とし, E が増加の場合は, $\gamma \leftarrow 0.95\gamma$ とした。また, $E(s)/E(s-1) > 1.05$ のときはパラメータの更新は行わないものとした。ここで, $E(s)$ は s ステップの評価関数である。

これらの条件の下で, 提案手法の性能を従来手法と比較した。マルチブランチニューラルネットワークでは, ブランチの数は任意に設定できるが, ここでは, 全てのノード間のブランチ数が p である構造のものを p -branch NNs と表している。

5・2 関数近似問題 関数近似問題では, 以下の2つの式で表される関数を近似対象とした(図3)。

$$f_1(x, y) = 0.5(0.475((1.35 + e^x \sin(13.0(x - 0.6)^2) e^{-y} \sin(7y)) - 2.5) - 0.2),$$

$$(0 \leq x, y \leq 1).$$

$$f_2(x, y) = 0.2 \frac{(x - 0.4)(2x + 0.2)}{0.04 + x^2} \frac{(y - 0.4)(2y + 0.2)}{0.04 + y^2},$$

$$(-1 \leq x, y \leq 1).$$

全体的に緩やかな関数 f_1 と, 緩やかな領域と急勾配な領域の両方をもつ関数 f_2 を取り上げ, それぞれに対して提案手法がどの程度有効なのかを調べる。

関数 f_1 の訓練データは入力空間を 16×16 点に分割した均等に分布する 256 個のサンプルを用いた。また, 汎化誤差を計算するための f_1 のテストデータは 441 個 (21×21) のサンプルである。関数 f_2 に関しては, 同様に訓練データ, テストデータをそれぞれ 441 個 (21×21) と 784 個 (28×28) のサンプルを用いた。

この関数近似問題では, 正則化係数 $\eta = 10^{-8}$ とし, 式(6)の定数 L を $L = 0.01$ とした。

図 4 関数 f_1 の学習誤差曲線Fig. 4. Training error curves of function f_1 . 図 5 関数 f_2 の学習誤差曲線Fig. 5. Training error curves of function f_2 .表 1 関数 f_1 のシミュレーション結果Table 1. Simulation results of function f_1 .

NNs [hidden node]	Training error (MSE)	Generalization error (MSE)	Execution time ratio
Conventional [36]	1.50×10^{-4}	1.29×10^{-4}	1.00
1-branch [12]	1.74×10^{-5}	1.31×10^{-5}	0.66
2-branch [6]	4.23×10^{-6}	4.29×10^{-6}	0.53
3-branch [4]	2.14×10^{-6}	3.42×10^{-6}	0.51
6-branch [2]	2.84×10^{-5}	1.01×10^{-3}	0.54

表 2 関数 f_2 のシミュレーション結果Table 2. Simulation results of function f_2 .

NNs [hidden node]	Training error (MSE)	Generalization error (MSE)	Execution time ratio
Conventional [36]	1.24×10^{-3}	1.33×10^{-3}	1.00
1-branch [12]	9.29×10^{-5}	1.56×10^{-4}	0.66
2-branch [6]	8.84×10^{-5}	1.12×10^{-4}	0.53
3-branch [4]	2.50×10^{-5}	3.93×10^{-5}	0.51
6-branch [2]	3.45×10^{-5}	5.83×10^{-3}	0.54

関数 f_1 及び関数 f_2 に対する学習誤差曲線をそれぞれ図 4, 図 5 に示す。これらの学習誤差曲線を見ると, 提案手法 (3-branch NNs) は従来手法と比較して, 早い学習ステップで誤差が小さくなっていることがわかる。

表 1, 表 2 にそれぞれ関数 f_1 , 関数 f_2 に対する学習誤差と汎化誤差を示す。提案手法のマルチブランチニューラルネットワークについては, マルチブランチのブランチ数を変えたものも幾つか示した。これらは, パラメータの初期値が異なる 10 回のシミュレーションの平均である。また, 各表

図 6 6-branch NNs の関数 f_1 の学習誤差曲線Fig. 6. Training error curves of function f_1 by 6-branch NNs.

には, ネットワークの中間層のノード数及び学習の計算時間比も示した。ここでは, パラメータ ($w_{ij}(p)$, $c_{ij}(p)$, $\sigma_{ij}(p)$) の総数は 108 個に設定した。パラメータ数が同じである条件では, 中間層ノードの数は提案手法の方が少なくなるために, 提案手法の計算時間は, 従来手法の約半分となっている。

1-branch NNs, 2-branch NNs よりも 3-branch NNs は良好な結果を示している。ノード間のブランチ数が多ければ, それだけ局所化も可能であり, ネットワークの表現能力が高くなる。ただし, 表 1, 表 2 の 6-branch NNs の結果のように, ブランチ数を増加することにより中間層ノード数が小さくなりすぎると, 平均して誤差が大きくなる。図 6 に 6-branch NNs の関数 f_1 での学習誤差曲線を示した。図はパラメータの初期値の異なる 10 回のシミュレーション全ての結果を示している。良い結果を示すものもあるが, 性能のばらつきが大きい。これは, 中間層ノード数が少なすぎると, ネットワークの表現能力が大きく制限されてしまうからである。2-branch NNs や 3-branch NNs では中間ノード数が過少ではないので, 性能のばらつきは小さい。6-branch NNs でも中間ノード数を増加することで性能のばらつきは小さくなることを確認している。

5.3 Two Spirals 分類問題 Two Spirals 分類問題⁽¹⁰⁾ は 2 種類の絡み合った螺旋データを分類することが目的である。訓練データは 192 個のサンプルであり, それぞれの螺旋に属する 96 個のサンプルがある。

ここでは, 汎化能力を十分に得るために, 正則化係数 $\eta = 10^{-5}$ とし, 定数 $L = 0.2$ とした。

図 7 及び図 8 は, 従来ニューラルネットワークとマルチブランチニューラルネットワーク (3-branch NNs) の訓練データに対する分類成功率の学習による変化を表している。これらは初期値の異なる 10 回のシミュレーション全ての結果である。従来手法 (図 7) では, 学習しても分類成功率が向上している様子が見られない。つまり, ネットワークの能力が不足していることがわかる。従来の 3 層構造のネットワークでは, Two Spirals 分類問題を解くことができない⁽¹⁰⁾ ということを確認できる。それに対して, 提案手法 (図 8) では, すべての初期値に対して 100% の分類

図 7 従来ネットワークの分類成功率

Fig. 7. Correct classification rate by conventional NNs.

図 8 マルチブランチネットワークの分類成功率

Fig. 8. Correct classification rate by multi-branch NNs.

成功率を示している。これらの分類成功率から提案手法では、3 層構造でも表現能力が高く、より早い学習ステップで安定した良好な結果が得られることがわかる。

さらに、汎化能力に関しても良好な結果が得られている。テストデータに対する分類の成功例として、図 9 に従来のニューラルネットワークの結果を、図 10 に 3-branch ニューラルネットワークによる結果を示す。成功した分類とは、学習データに対して 100% 正しく分類し、テストデータについても、それぞれの螺旋データの適切な境界線によって分類されることを意味する。

表 3 に従来手法とブランチ数の異なるマルチブランチニューラルネットワークで Two Spirals 分類問題を解いた場合の成功率等を示す。従来のニューラルネットワークでは、成功した分類は得られなかったが、提案のマルチブランチニューラルネットワークでは、3-branch NNs で、8/10 のシミュレーションが成功していることがわかる。つまり、汎化能力を十分に得ている。5・2 の関数近似問題の時と同様に、6-branch NNs には十分な数の中間層ノードがないために、成功した分類は得られなかった。3-branch NNs においても、中間層ノードを 8 個より削減すると図 10 のような成功した分類は得られにくくなった。表 3 には学習の計算時間比も示した。なお、パラメータ ($w_{ij}(p)$, $c_{ij}(p)$, $\sigma_{ij}(p)$) の総数は 216 個に設定した。

参考までに、4 層構造の従来手法では、20 個ずつの 2 層の中間層を用いて、図 10 のような成功した分類を得られるのは 3/10 程度である。

図 9 従来ネットワークによる分類結果

Fig. 9. A classification result by conventional NN.

図 10 マルチブランチニューラルネットワークによる分類の成功例

Fig. 10. A classification result by multi-branch NN.

表 3 Two Spirals 分類の成功率

Table 3. Simulation results of Two Spirals classification.

NNs [hidden node]	Successful results	Execution time ratio
Conventional [72]	0/10	1.00
1-branch [24]	7/10	0.57
2-branch [12]	9/10	0.49
3-branch [8]	8/10	0.45
6-branch [4]	0/10	0.48

6. 考 察

提案手法のマルチブランチニューラルネットワークでは、各ノード間のブランチを任意の数だけ設定できる。ブランチ数は増加したほうがよいが、極端に多くする必要はない。シミュレーションでは、中間層ノードに値域 $[-1:1]$ のシグモイド関数を使用し、入力層ノードにおいても訓練データを $[-1:1]$ にスケールしている。つまり、各ブランチの定義域となるのは $[-1:1]$ であり、それほど広い領域ではない。この領域を局所化する目的でマルチブランチを設定する場合、3-branch 程度であれば十分であると考えられる。各ブランチのガウス関数のばらつきを小さくして、マルチブランチ数を多くすると汎化能力が低下する可能性が高くなるからである。

図 11、図 12 はマルチブランチニューラルネットワークの学習後の結合重みを含めた各ブランチの入出力関係の例を示すために、訓練データを入力したときのブランチの入

図 11 各ブランチ（入力層中間層間）の入出力関係の例

Fig. 11. Examples of input-output relation of multi-branch from input layer to hidden layer.

図 12 各ブランチ（中間層出力層間）の入出力関係の例

Fig. 12. Examples of input-output relation of multi-branch from hidden layer to output layer.

出力値をプロットしたものである。これにより、ガウス関数による局所性の様子が分かる。

7. 結 論

本論文では、マルチブランチ構造のニューラルネットワークの中で、特に、マルチブランチにガウス関数を用いて非線形性及び局所性を有効に利用する方式を提案した。関数近似や分類問題を用いて従来のニューラルネットワークと提案方式を比較した結果、小さい計算コストでありながら、より高い性能を示した。従って、マルチブランチニューラルネットワークでは、ネットワークサイズが小さく、かつ、表現能力の優れた構造を効率良い学習により求めることができる。

(平成 16 年 3 月 24 日受付, 平成 16 年 12 月 2 日再受付)

文 献

- (1) D. E. Rumelhart, G. E. Hinton, and R. J. Williams: "Learning internal representations by error propagation", *Parallel distributed processing*, Vol.1, The MIT Press, pp. 318-362 (1986)
- (2) K. Hirasawa, X. Wang, J. Murata, J. Hu, and C. Jin: "Universal learning network and its application to chaos control", *Neural Networks*, Vol.13, pp. 239-253 (2000)
- (3) K. Hirasawa, M. Ohbayashi, H. Fujita, and M. Koga: "Universal Learning Network Theory", *T. IEE Japan*, Vol.116-C, No.7, pp. 794-801 (1996-7)
- (4) M. J. D. Powell: "Radial Basis Functions for Multivariable Interpolation: A Review", in *Algorithms for Approximation*,

- J. C. Mason and M. G. Cox, Eds., pp. 143-167, Oxford (1987)
- (5) M. T. Musavi, W. Ahmed, K. H. Chan, K. B. Faris, and D. M. Hummels: "On the Training of Radial Basis Function Classifiers", *Neural Networks*, Vol.5, No.4, pp. 595-603 (1992)
- (6) S. Chen, C. F. N. Cowan, and P. M. Grant: "Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks", *IEEE Trans. Neural Networks*, Vol.2, No.2, pp. 302-309 (1991)
- (7) T. Yamashita, K. Hirasawa, J. Hu, and J. Murata: "Multi-Branch Structure of Layered Neural Networks", *International Conference on Neural Information Processing*, pp. 243-247 (2002-11)
- (8) G. E. Hinton: "Connectionist Learning Procedure", *Machine Learning*, ed. J. Carbonell, The MIT Press (1990)
- (9) T. J. Sejnowski and C. R. Rosenberg: "Parallel networks that learn to pronounce English text", *Complex Systems 1*: pp. 145-168 (1987)
- (10) K. J. Lang and M. J. Witbrock: "Learning to Tell Two Spirals Apart", *Proc. of 1988 Connectionist Models Summer School*, pp. 52-59 (1988)

山下 貴志 (非会員) 1979 年 2 月 27 日生。2002 年 3 月九州大学工学部電気情報工学科卒業。2004 年 3 月同大学大学院システム情報科学府電気電子システム工学専攻修士課程修了。同年 4 月早稲田大学大学院情報生産システム研究科情報生産システム専攻博士後期課程入学、現在に至る。

平澤 宏太郎 (正員) 1942 年 1 月 26 日生。1966 年九州大学大学院工学研究科修士課程電気工学専攻修了。同年(株)日立製作所入社日立研究所勤務、同研究所副所長、大みか工場主管技師長を経て、92 年九州大学工学部教授、96 年同大学院システム情報科学研究科教授。02 年早稲田大学大学院情報生産システム研究科教授、現在に至る。工学博士。

古月 敬之 (正員) 1986 年中国中山大学大学院修士課程修了。同年、同大学電子工学科助手、88 年同講師。97 年九州工業大学情報工学研究科博士後期課程修了。同年、九州大学大学院システム情報科学研究科助手。03 年早稲田大学大学院情報生産システム研究科助教授、現在に至る。情報工学博士。