シグモイド関数付きニューロンモデルの反復演算法

正員 中 條 直 也 (豊田中央研究所)

非会員 黒 柳 奨 (名古屋工業大学)

正員 道 木 慎 二 (三重大学)

正員 大 熊 繁 (名古屋大学大学院)

An Iterative Calculation Method of Neuron Model with Sigmoid Function

Naoya CHUJO, Member (Toyota Central R&D Labs., Inc.), Susumu KUROYANAGI, Non-member (Nagoya Institute of Technology), Shinji DOKI, Member (Mie University), Shigeru OKUMA, Member (Nagoya University)

Neural network hardware is necessary for demanding real-time applications such as pattern recognition. Over one million gates of the latest FPGA (Field Programmable Gate Array) are available for the dedicated neuro hardware. In this paper, an improved calculation algorithm of the neuron model with sigmoid function is proposed, which is suitable for hardware implementation. The proposed algorithm is based on the multi dimensional binary search. The neuron circuits implemented on FPGA by the proposed algorithm have shown the excellence in size and circuit frequency compared with the conventional circuits with sum of product operation.

キーワード：ニューロン，ハードウェア，二分反復法，FPGA

1. はじめに

ニューラルネットワークを汎用の計算機上のソフトウェアとして実現することは、柔軟性や汎用性やアルゴリズム開発などの点で優れている。しかしながら、アルゴリズムを実装する場合にはより高速なハードウェアを使用したいという要求も高い。このため、高速な並列計算機上を利用した高速処理も行われている⁽¹⁾。また、組込機器などへの実装を考えた場合には、集積回路を利用してハードウェア化を行なうことが必要となり、このような試みも数多く行なわれている⁽²⁾⁽³⁾。一方、専用のハードウェアとして利用可能で、使用時に回路設定できるFPGA(Field Programmable Gate Array)などのデバイスがある。近年、その高集積化が著しく⁽⁴⁾⁽⁵⁾、1チップで100万ゲート以上が利用できる状況である。このため、これらのデバイスの特性を生かす計算法が求められている⁽⁶⁾⁽⁷⁾⁽⁸⁾。

ニューロンモデル⁽⁹⁾の基本演算は、ニューロンへの入力と結合係数の積和演算とシグモイド演算である⁽¹⁰⁾。なかでも多数の入力を扱う積和演算は重要な要素である。そのため、ニューラルネットワークのハードウェア化に関しては、できるだけ小さい回路でこの積和演算を実装することが必要である。しかしながら、ハードウェア乗算器を使用

した積和演算回路による実装方法は回路規模が大きくなるという問題がある⁽¹¹⁾。特にFPGAなどを利用して実装した場合には回路規模が大きかつ高速でないという問題点がある。

そこで、前報⁽¹²⁾においてパーセプトロン型のニューロンモデルに対する反復計算法の基本的な提案を行った。この方法は多次元の二分反復法に基づくニューロンの計算法である。従来の積和演算を利用した手法に比べて、演算器として加算器とシフト演算器のみを使用し除算器を必要としないため、ハードウェアを小型化できる。また収束に必要な平均的な反復回数は小さく、演算時間も短くできる。このような微分不能なニューロンモデルのための専用の学習法も提案されているが⁽¹³⁾、シグモイド関数を扱うことができないため、一般的なBP学習⁽¹⁴⁾などの結果をハードウェア化して利用することはできなかった。

そこで本論文では、シグモイド関数を扱えるように拡張した二分反復法に基づく計算法を提案する。以下、第2章で対象とするニューロンモデルについて簡単に述べる。第3章で提案手法を述べ、第4章ではその平均反復回数を評価し、第5章ではFPGAを利用した提案手法のハードウェア化について述べる。第6章では考察を述べ、第7章でまとめとする。

2. ニューロンモデルと演算

対象とするニューロンモデルについて簡単に述べる。本論文では図1に示すニューロンモデルを扱う。このモデルではニューロンは n 個の入力と1つの出力を持っている。このニューロンへの各入力 x_i と結合係数 w_i の積和演算の結果に対して図2のシグモイド関数を適用する。本論文では0から1へ遷移する領域をシグモイド領域 (Sigmoid Region) と定義する。またシグモイド領域以外も出力に応じて0出力領域 (0-Region), 1出力領域 (1-Region) と定義する。

図1 ニューロンモデル
Fig. 1. Neuron model.

図2 シグモイド関数
Fig. 2. Sigmoid function.

2入力のニューロン演算は、図3に示すように2次元平面に表すことができる。結合係数 w_i と閾値 w_0 によってニューロン直線 $g(x) = \sum_{i=1}^2 w_i x_i - w_0 = 0$ (以下、 $g(x)$ と表記) が決定される。この $g(x)$ の周りにはシグモイド領域が帯状に存在する。入力 d からニューロン直線へ垂線をひき、その垂線上のニューロン出力を示したのが右図である。入力 d がシグモイド領域に含まれていれば、出力は $g(x)$ との距離に応じて0と1の間の連続値となる。それ以外の場合は、出力は0または1となる。多次元の場合はニューロン直線の代わりに超平面 $g(x) = \sum_{i=1}^n w_i x_i - w_0 = 0$ との距離を計算することになるが考え方は同じである。なお本論文では一般性を失うことなく $w_i > 0$ と仮定する。

3. 提案手法-多次元二分反復に基づく計算法

本章では提案アルゴリズムについて述べる。前章で述べたように、ニューロン演算は入力データとニューロンの作る超平面の距離を求め、この距離に応じてシグモイド関数の出力を計算する。提案手法ではこの距離の計算を多次元の二分反復法を使用して行う。以下では、まず2次元の場合を例に提案手法の説明を行い、次に n 次元の場合を述べる。

図3 2次元のニューロンモデル
Fig. 3. Two-dimensional neuron model.

〈3・1〉2次元のアルゴリズム 2次元の場合の提案手法を図4,5を用いて説明する。提案アルゴリズムでは、入力 d を含む長方形領域を二分反復法によって小さくしていくことで、ニューロン平面 $g(x)$ と入力 d の距離を推定する。この際、入力 d の存在領域がシグモイド領域外と判明した場合には計算を停止することで計算を加速する。シグモイド領域の場合には必要な精度だけ反復を繰り返して $g(x)$ と d の距離を評価し、シグモイド関数の値を求める。

まず図4に示すように、参照点 p_r をとり、そこから x_1 軸、 x_2 軸に平行に直線をひくことで、 $g(x)$ 上の2点 p_1, p_2 をとることができる。そして p_r, p_1, p_2 を頂点とする長方形を定義する。参照点 p_r を直線から十分離すことにより、任意の入力 d がこの長方形の内部にあると仮定できる。線分 $p_1 p_2$ の中点を c として、 c を通り、 x_1 軸、 x_2 軸に平行な線を引く。すると長方形を等しい4つの小さな長方形領域に分けることができる。入力 d がこれらの長方形領域のどこに含まれるかは、 $d_k (k=1,2)$ と $c_k (k=1,2)$ の位置関係だけで決まるため、以下のように次元ごとの独立した減算の結果として判断することが可能である。

- $d_1 - c_1 < 0, d_2 - c_2 \geq 0$ なら左上領域
- $d_1 - c_1 \geq 0, d_2 - c_2 \geq 0$ なら右上領域
- $d_1 - c_1 \geq 0, d_2 - c_2 < 0$ なら右下領域
- $d_1 - c_1 < 0, d_2 - c_2 < 0$ なら左下領域

ここで $g(x)$ に垂直な直線 w を引き、入力 d が含まれる長方形の w への射影を求める。射影された線分領域を $R_1 R_2$ とする。4分割された長方形のうち入力 d を含む長方形を同様に射影することにより、 $R_1 R_2$ の半分の長さを持つ領域 $R'_1 R'_2$ を求めることができる。

図4の場合、領域 $R'_1 R'_2$ 全体がシグモイド領域外となり、1出力領域に含まれる。このため、この領域に含まれている入力 d に対する出力も1と確定でき、この時点で反復は終了する。

一方、図5の場合、領域 $R'_1 R'_2$ は、シグモイド領域 $S_1 S_2$ と重なりを持つ。このため、この領域に含まれている入力 d に対する出力は確定できない。このような場合には、必要精度まで領域分割と判定の過程を反復する。1回の反復により入力 d を含む領域は2分の1の大きさになり、1bitの精度向上が可能である。このため、例えば要求精度8ビット

図 4 入力 d がシグモイド領域外の場合

Fig. 4. Input d in non-sigmoid region.

図 5 入力 d がシグモイド領域内の場合

Fig. 5. Input d in sigmoid region.

トの場合、最大 8 回の反復を行うことが必要となる。

提案アルゴリズムの l 回目の反復を図 6 を用いて説明する。 d を含む長方形 $Rect^l$ の w 方向への射影を $R_1^l R_2^l$ とする。その長さは次元数 2 に正規化している。 i はニューロン平面 $g(x)$ の位置を示し、ここでは $i = 1$ として説明を行う。このときシグモイド領域 $S_1 S_2$ の範囲は $i - b2^{l-1}, i + b2^{l-1}$ で表わすことができる。ここでシグモイド領域幅の 2 分の 1 を b としている。これに対して分割された小さな長方形は 4 つ定義できる。図では入力 d を含む長方形 $Rect^{l+1}$ の射影 $R_1^{l+1} R_2^{l+1}$ のみを実線で示し、その他を点線で示す。このとき、 $d_k - c_k \geq 0 (k = 1, 2)$ が成り立つ場合の数 j を用いて、 R_1^{l+1}, R_2^{l+1} の位置は、それぞれ $j/2, j/2 + 1$ と表現することができる。この例では $j = 1$ としている。領域 $R_1^{l+1} R_2^{l+1}$ がシグモイド領域 $S_1 S_2$ と重ならない場合には、ニューロンの出力は 0 または 1 と決定できる。そのための条件は以下の二つである。

- $j/2 + 1 \leq i - b2^{l-1}$ となる場合、0 出力 (0-region)
- $i + b2^{l-1} \leq j/2$ となる場合、1 出力 (1-region)

図 6 で示した場合のようにシグモイド領域と重なっている場合には、図 7 のように $Rect^{l+1}$ を対象として $l + 1$ 回目の反復を行う。 $l + 1$ 回目の反復ではニューロン平面 $g(x)$ の位置 i は l 回目の i_j を用いて $i \leftarrow 2i - j$ で計算される。この式に関しては文献⁽¹²⁾に説明があるので本論文ではこ

図 6 l 回目の反復

Fig. 6. l -th search.

図 7 $l + 1$ 回目の反復

Fig. 7. $l + 1$ -th search.

れ以上述べない。

反復回数が上限に達した場合には、入力 d の直線 $g(x)$ からの距離計算を行って、この距離からシグモイド関数値を出力する。まず距離計算について述べる。 $R_1 R_2$ の長さを L とすると、例えば図 4 で R_1 からの R_1', R_1'' の距離はそれぞれ、 $\frac{L}{2}, \frac{L}{2}(2^2 + 1)$ と表すことができる。一般的に l 回目の反復時に R_1^l の R_1 からの距離 $L_{R_1^l}$ は $\frac{L}{2^{l+1}}(2^{l-1}j_1 + 2^{l-2}j_2 + \dots + j_l)$ となる。ここで j_l とは l 回目の反復時に $d_k - c_k \geq 0 (k = 1, 2)$ が成立する場合の数である。この $L_{R_1^l}$ を d の R_1 からの距離 L_d の推定値とする。従って反復回数が最大値 l_{max} に達した場合、 d の R_1 からの距離 L_d は以下のように計算する。

$$L_d = \frac{L}{2^{l_{max}+1}}(2^{l_{max}-1}j_1 + 2^{l_{max}-2}j_2 + \dots + j_{l_{max}}) \quad (1)$$

ここで (2) 式で j_{sum} を定義することにより、 L_d は以下のように整理される。

$$j_{sum} = 2^{l_{max}-1}j_1 + 2^{l_{max}-2}j_2 + \dots + j_{l_{max}} \quad (2)$$

$$L_d = \frac{R_1 R_2}{2^{l_{max}+1}} j_{sum} \dots \dots \dots (3)$$

なお j_{sum} は $j_{sum} \leftarrow 2j_{sum} + j_l$ という漸化式を用いて

アルゴリズム内で計算できる。この L_d をシグモイド関数の入力とすることによりニューロンの最終的な出力を求めることができる。シグモイド関数の計算については、関数テーブルを用いる方法⁽¹¹⁾、補間を用いる方法⁽¹⁵⁾などが適用できるが、本論文ではこれ以上述べない。

2次元の概略アルゴリズムをまとめると以下のように記述できる。

初期化 d を含む長方形 $Rect$ を決定する。この長方形を w へ射影してできる線分を R_1R_2 とする。反復回数を $l = 1$ と設定する。

二分割 第 l 回目の反復において、 $R_1^{l+1}R_2^{l+1}$ の長さを2と正規化する。長方形 $Rect^l$ の各辺を垂直2等分することで、4個の等しい長方形を作る。4つの長方形が共有する点を中心点 c とする。このうち d が含まれる $Rect^{l+1}$ を、 d と c の座標を比較することで計算する。また $d_k - c_k \geq 0 (k = 1, 2)$ の成立する場合の数 j を計算する。

射影 $Rect^{l+1}$ を w へ射影し、その領域を線分 $R_1^{l+1}R_2^{l+1}$ とする。その長さは、1であり、 R_1^{l+1} と R_2^{l+1} の座標はそれぞれ、 $j/2$ 、 $j/2 + 1$ となる。シグモイド領域を表す S_1 、 S_2 の座標はそれぞれ $i - b2^{l-1}$ 、 $i + b2^{l-1}$ である。

収束判定 線分 $R_1^{l+1}R_2^{l+1}$ がシグモイド領域 S_1S_2 と重なっているかどうかを下式で判定する。

重ならない場合

- $j/2 + 1 \leq i - b2^{l-1}$ ならばシグモイド関数値0を出力して終了
- $i + b2^{l-1} \leq j/2$ ならばシグモイド関数値1を出力して終了

重なる場合

- $j_{sum} \leftarrow 2j_{sum} + j$ を計算する
- 反復回数 l が上限 l_{max} を越えた場合、 j_{sum} を (3) 式に代入してシグモイド関数値を計算して終了
- 反復回数 l が上限 l_{max} を越えない場合、次回の $g(x)$ の位置 i を $i \leftarrow 2i - j$ で計算し、 $l \leftarrow l + 1$ として再度二分割を行う

〈3・2〉 n 次元のアルゴリズム

2次元での議論を一般化した n 次元のアルゴリズムについて説明する。2次元の場合と異なるのは、 n 次元の場合には長方形の代わりに超直方体を使用する点である。入力 d を含む超直方体を二分反復法によって小さくしていくことで、ニューロン平面 $g(x)$ と入力 d の距離を推定する。この際、入力 d の存在領域がシグモイド領域外と判明した場合には計算を停止することで計算を加速する。シグモイド領域内の場合には必要な精度だけ反復を繰り返して $g(x)$ と d の距離を評価し、シグモイド関数の値を求める。

2次元の場合と同様に、 l 回目の反復を図8を用いて説明する。 d を含む超直方体 HRP^l の w への射影を $R_1^lR_2^l$ とする。その長さは次元数 n に正規化している。 i はニューロン平面 $g(x)$ の位置を示す。このときシグモイド領域 S_1S_2 の範囲は

図8 n 次元における l 回目の反復
Fig. 8. l -th search in n -dimensional case.

$i - b2^{l-1}$ 、 $i + b2^{l-1}$ で表わすことができる。これに対して各次元後とに二分割された小さな超直方体は 2^n 個定義できるが、ここでは入力 d を含む長方形 HRP^{l+1} の射影 $R_1^{l+1}R_2^{l+1}$ のみを示している。このとき、 $d_k - c_k \geq 0 (k = 1 \dots n)$ が成り立つ場合の数 j を用いて、 $R_1^{l+1}R_2^{l+1}$ の位置は、それぞれ、 $j/2$ 、 $(j+n)/2$ と表現することができる。 S_1 、 S_2 と $R_1^{l+1}R_2^{l+1}$ を比較することにより、シグモイド領域が HRP^{l+1} と重なるかどうかを判定することができる。2次元の場合と同様に、領域 $R_1^{l+1}R_2^{l+1}$ がシグモイド領域 S_1S_2 と重ならない条件は以下の二つである。これらの場合には、ニューロンの出力は0または1と決定できる。

- $(j+n)/2 \leq i - b2^{l-1}$ となる場合、0出力 (0-region)
- $i + b2^{l-1} \leq j/2$ となる場合、1出力 (1-region)

図8で示した場合のようにシグモイド領域と重なっている場合には、 HRP^{l+1} を対象として $l+1$ 回目の反復を行う。 $l+1$ 回目の反復ではニューロン平面 $g(x)$ の位置 i は l 回目の i, j を用いて $i \leftarrow 2i - j$ で計算される。

2次元の場合と同様に、反復回数が上限に達した場合には、入力 d の直線 $g(x)$ からの距離計算を行って、この距離からシグモイド関数値を出力する。

R_1R_2 の長さを L とする。 l 回目の反復時の d を含む超直方体 HRP の射影を $R_1^lR_2^l$ として、 R_1^l の R_1 からの距離 $L_{R_1^l}$ は $\frac{L}{2^{ln}}(2^{l-1}j_1 + 2^{l-2}j_2 + \dots + j_{lmax})$ となる。ここで j_l とは l 回目の反復時に $d_k - c_k \geq 0 (k = 1 \dots n)$ が成立する場合の数である。反復回数が最大値 l_{max} に達した場合、この $L_{R_1^{lmax}}$ を d の R_1 からの距離 L_d の推定値とする。 d の R_1 からの距離 L_d は下式で計算する。

$$L_d = \frac{L}{2^{lmaxn}}(2^{lmax-1}j_1 + 2^{lmax-2}j_2 + \dots + j_{lmax}) \quad (4)$$

この L_d をシグモイド関数の入力とすることによりニューロンの最終的な出力を求めることができる。ここで (5) 式で j_{sum} を定義することにより、 L_d は以下のように整理さ

れる。

$$j_{sum} = 2^{l_{max}-1}j_1 + 2^{l_{max}-2}j_2 + \cdots + j_{l_{max}} \quad (5)$$

$$L_d = \frac{R_1 R_2}{2^{l_{max}n}} j_{sum} \cdots \cdots \cdots (6)$$

なお j_{sum} は $j_{sum} \leftarrow 2j_{sum} + j_l$ という漸化式を用いてアルゴリズムで計算できる。の L_d をシグモイド関数の入力とすることによりニューロンの最終的な出力を求めることができる。

n 次元の概略アルゴリズムをまとめると以下のように記述できる。

初期化 d を含む超直方体 HRP を決定する。この超直方体を w へ射影してできる線分を $R_1 R_2$ とする。反復回数を $l = 1$ と設定する。

二分割 第 l 回目の反復において、 $R_1^{l+1} R_2^{l+1}$ の長さを n と正規化する。超直方体 HRP^l の各辺を垂直 2 等分することで、 2^n 個の等しい超直方体を作る。 2^n 個の超直方体が共有する点を中心点 c とする。このうち d が含まれる HRP^{l+1} を、 d と c の座標を比較することで計算する。また $d_k - c_k \geq 0 (k = 1 \dots n)$ の成立する場合の数 j を計算する。

射影 HRP^{l+1} を w へ射影し、その領域を線分 $R_1^{l+1} R_2^{l+1}$ とする。 $R_1^{l+1} R_2^{l+1}$ の長さは、 $n/2$ であり、 R_1^{l+1} と R_2^{l+1} の座標はそれぞれ、 $j/2$ 、 $(j+n)/2$ となる。シグモイド領域を表す S_1 、 S_2 の座標はそれぞれ $i - b2^{l-1}$ 、 $i + b2^{l-1}$ である。

収束判定 線分 $R_1^{l+1} R_2^{l+1}$ がシグモイド領域 $S_1 S_2$ と重なっているかどうかを下式で判定する。

重ならない場合

- $(j+n)/2 \leq i - b2^{l-1}$ ならばシグモイド関数値 0 を出力して終了
- $i + b2^{l-1} \leq j/2$ ならばシグモイド関数値 1 を出力して終了

重なる場合

- $j_{sum} \leftarrow 2j_{sum} + j$ を計算する
- 反復回数 l が上限 l_{max} を越えた場合、 j_{sum} を (6) 式に代入してシグモイド関数値を計算して終了
- 反復回数 l が上限 l_{max} を越えない場合、次回の $g(x)$ の位置 i を $i \leftarrow 2i - j$ で計算し、 $l \leftarrow l + 1$ として再度二分割を行う

4. 反復回数の評価

ここでは、提案手法の平均的な反復回数について評価を行なう。シグモイド領域が存在せず、入力データはランダムに分布していると仮定すると、理論的な平均反復回数を与えることができる⁽¹²⁾。2次元の図4でシグモイド領域がない場合を考えると、1回の反復で $g(x)$ との関係が決まるのは、4分割された長方形のうち右上と左下の長方形であり、面積比では $1/2$ である。同様に2回の反復では $1/2^2$ である。このため平均反復回数 m は下式で表現され、この

式を計算すると $m = 2$ となる。

$$m = \lim_{l \rightarrow \infty} \sum_{i=1}^l i \left(\frac{1}{2}\right)^i \cdots \cdots \cdots (7)$$

3次元以上 ($n \geq 3$) の場合も同様のことがいえる⁽¹²⁾。1回の探索において、分割された 2^n 個の直方体のうち $g(x)$ との関係が決まるのは、次元によらず全体の $1/2$ の直方体である。このため1回の反復で超平面との関係が決まるのは、直方体 (4次元以上の場合は超直方体) 全体の体積を1とするとやはり $1/2$ の体積である。このため、2次元の場合と同様となり、次元によらず平均的な反復回数 $m = 2$ となる。

次にシグモイド領域が全領域の場合は、要求精度 (bit) に応じた反復回数が必要である。8bit 精度の場合は8回の反復が必要である。

最後にシグモイド領域が部分的である場合は、シグモイド領域の面積比によって反復回数が変動し理論的な評価式を一般的に与えることは難しい。またデータの分布がランダムでない場合には反復回数を理論的に与えることはできない。そこでシミュレーションを用いて評価を行った結果を次節で述べる。

〈4・1〉シミュレーションによる評価 簡単な例として、2次元平面の1000点の入力データを使用して、反復回数の評価をシミュレーションで行った。

〈4・1・1〉ランダム分布 まず入力データを正方形領域にランダムに分布させて計算を行った例を示す。シグモイド領域の割合を0%、10%、25%、50%と変化させて平均反復回数をそれぞれ計算した。ここでシグモイド領域の割合とは、 $R_1 R_2$ に対するシグモイド領域幅 $2b$ の割合で定義している。なお、2次元ではシグモイド領域25%、50%の場合についても、簡単な作図によって反復回数の評価式を与えることができ、それぞれ、 $\frac{7}{16} \text{bit} + \frac{3}{2}$ 、 $\frac{3}{4} \text{bit} + \frac{3}{4}$ となる。ここで bit は要求精度を示す。

シミュレーション結果を表1に示す。8bit 演算では、シグモイド領域が0%の場合は平均反復回数 $m = 2.01$ でほぼ理論値 $m = 2$ でありであるといえる。シグモイド領域が10%の場合、平均反復回数は $m = 3.91$ と増加した。シグモイド領域が25%、50%の場合には、平均反復回数はそれぞれ $m = 4.95$ 、 $m = 6.73$ とさらに増加する。これらの値は上記の評価式で計算した値 $m = 5.00$ 、 $m = 6.75$ に十分近く、ほぼ理論通りといえる。16bit 演算でもシグモイド領域が0%の場合は8bit 演算と同じであるが、シグモイド領域の割合が10%、25%、50%の場合には、その割合が増加するにつれて8bit 演算との差が増大した。24bit 演算でも同様の傾向が見られた。このようにシグモイド領域が大きい場合、また、演算精度が高い場合には平均反復回数は大きくなる。

〈4・1・2〉EXOR 型分布 ここでは入力データを図9に示すような EXOR 型に分布させて計算を行った例を示す。左上と右下の◇で示される分布と、右上と左下の◆で

表 1 反復回数のシミュレーション結果
Table 1. Average number of search by simulation.

Sigmoid Region	0%	10%	25%	50%
m (8-bit)	2.01	3.91	4.95	6.73
m (16-bit)	2.03	5.46	8.36	12.8
m (24-bit)	2.02	6.82	12.2	18.7

示される分布の 2 種類のデータが存在する。

図 9 EXOR 型分布例
Fig. 9. EXOR type distribution.

これに対して図 10 に示すような 3 層のニューラルネットワークを作成し、ソフトウェアを使用してこれら 2 種類の分布を分類するように学習を行った。学習後の図 10 の中間層ニューロン (Hidden Layer) を図 11 に示す。

図 10 使用した 3 層ネットワーク
Fig. 10. Three-layered neural networks.

図 11 の分布と中間層ニューロン g_1 を使って平均反復回数をシミュレーションで評価した。結果を表 2 に示す。演算精度は 8bit,16bit,24bit の 3 つのケースを計算した。なお表中の Data Rate はシグモイド領域に含まれた入力データの割合を示す。シグモイド領域の割合が 0% の場合、精度に関わらず平均 1.51 回とランダム分布の場合の理論値より小さい反復回数でアルゴリズムは終了できている。これはデータ分布の偏りのためであると考えられる。この例ではデータ分布がニューロン直線 $g(x)$ によって明確に分離されているため反復回数は小さくなる。逆に平均反復回数が多くなるのは、データ分布がニューロン直線 $g(x)$ の周りに接近して存在するような場合である。シグモイド領域の割合が 10% の場合でも、平均反復回数の演算精度への依存性

は小さい。これはシグモイド領域に含まれる入力データの割合が 0.7% と小さいためと考えられる。一方、シグモイド領域の割合が 25%, 50% の場合にはこれはランダム分布の場合と同様な傾向が見られた。つまり、シグモイド領域が大きい場合、また、演算精度が高い場合に平均反復回数は大きくなった。これはシグモイド領域に含まれる入力データの割合が高いためと考えられる。また g_2 についても同様の数値が得られた。

図 11 学習後の中間層ニューロン
Fig. 11. Learned hidden layer.

表 2 EXOR 型分布の場合の平均反復回数
Table 2. Average number of search of EXOR-type distribution.

Sigmoid Region	0%	10%	25%	50%
Data Rate	0%	0.7%	36.4%	77.9%
m (8-bit)	1.52	2.99	4.90	6.67
m (16-bit)	1.53	3.07	7.83	12.8
m (24-bit)	1.51	3.12	10.8	18.8

5. 回路設計

提案手法をハードウェアとして設計する場合、使用する数値演算は、加減算と 2 での除算である。デジタル回路においては 2 での除算はシフト演算器で実現することが可能である。このため、提案手法で使用する数値演算回路は加算器とシフト演算器のみである。ニューロン面の遷移表については、アルゴリズムの節で述べた式に基づいて計算するが、メモリテーブルに記憶させる方式も可能である。

例として ALTERA 社の APEX20K シリーズの FPGA を対象に提案手法の回路設計を行ってみた。2 次元データを逐次的に実行する構成で設計した。ブロック構成を図 12 に示す。提案回路は、超直方体の座標計算回路 (Hyper Cube Calculator, Hyper Cube Data Table), 超直方体の中心 c と入力データ d を比較して条件成立数を数える回路 (J-Counter), ニューロン面の遷移用テーブル (Transition Table), ラッチ回路 (Latch, Latch for Distance Data), 制御回路 (Controller) からなる。比較のために従来の積和演

算回路を用いた回路を設計した。ブロック構成を図 13 に示す。この回路は、積和演算回路 (Multiplier, Adder, Accumulator), 重みデータ回路 (Weight Table), ラッチ回路 (Latch), 制御回路 (Controller) からなる。なお、積和演算回路で用いた乗算器は、ALTERA 社のライブラリを用いたパラレル演算型の乗算器である。演算精度として 8bit, 16bit, 24bit の 3 種類を使用した。

図 12 提案回路のブロック図

Fig. 12. Block diagram of proposed circuit.

図 13 従来回路のブロック図

Fig. 13. Block diagram of conventional circuit.

〈5・1〉 設計結果 両手法の回路設計を行って結果を比較した。なお、比較しているのは入力 d とニューロン直線 $g(x)$ の距離計算回路のみであり、図 12, 13 の Distance Unit の部分である。シグモイド関数については、提案手法と従来手法とで同じ回路が適用できるため比較していない。

比較結果を表 3 に示す。論理回路 (Logic) の単位は FPGA での設計単位である Logic Cell (LC) である。1LC は約 20gate 相当と言われているが回路によってかなり変動す

ることがある。Memory は数値データ用の記憶回路の使用量を示す。最近の FPGA では記憶回路は Logic Cell とは別に専用回路が用意されており、それを使用した。この表に示すように、全ての場合で、提案回路は従来回路に比べて論理回路が小さくできた。特に 16bit 精度では従来回路の 35% で、24bit 精度では 21% で実現できた。反復中の長方形の座標値を記憶する必要があるため、記憶回路は提案手法の方が 2 倍大きいのが、1bit を約 2gate 相当と考えて 0.1LC に換算すると回路規模全体に与える影響は小さい。高精度の場合に提案手法と従来手法の差が大きくなるのは、提案手法ではほぼ精度に比例した回路規模の増加であるのに対し、従来手法では精度が高いと回路規模の増加が著しくなるからである。最高動作周波数 (Max. Frequency) についても、全ての場合で提案手法が高い値を示している。高精度の場合にはその差は大きくなり、24bit 精度では 4 倍以上の差となった。これは乗算器の回路規模が精度の自乗に比例して大きくなり、回路内部の信号伝達時間もそれに付随して増大するためと考えられる。実行時間 (Execution Time) は、データ入力から距離計算回路の出力までの時間を示す。提案手法では出力までの時間は反復回数に依存するため確定しないので、ここでは平均反復回数 m をパラメータとして与えた。これらの結果から提案手法が従来手法に比べて高速に演算できるのは、演算精度が 8bit, 16bit, 24bit の場合、それぞれ m が 2.5, 4.85, 6.16 以下の場合となる。

表 3 回路設計結果

Table 3. Designed circuits.

	Proposed	Conventional
8-bit Circuit		
Logic	120 LC	148 LC
Memory	64 bit	32 bit
Max. Frequency	54 MHz	27 MHz
Execution Time	37.0m+18.5 ns	111 ns
16-bit Circuit		
Logic	191 LC	550 LC
Memory	128 bit	64 bit
Max. Frequency	50 MHz	14 MHz
Execution Time	40.0m+20.0 ns	214 ns
24-bit Circuit		
Logic	269 LC	1228 LC
Memory	192 bit	96 bit
Max. Frequency	40 MHz	9 MHz
Execution Time	50.0m+25.0 ns	333 ns

(LC: Logic Cell)

6. 考 察

提案手法は多次元の二分法を利用した反復によって、入力データとニューロンの作る超平面の距離を計算する。入力データがシグモイド領域に含まれる場合は、距離を正確に計算し、シグモイド関数の値を求める。一方、シグモイド領域外の場合は、距離を計算する必要がなく、領域に応じて 0 または 1 を出力する。このようにしてシグモイド関数を含むニューロンモデルを実現する。提案手法はシグモ

イド領域が小さい場合には乗算がかなり省略できることを利用している。このため、通常の BP 学習に必要な乗算を高速に行うことは難しく、現在の計算方法を利用してオンラインでの学習を行うことは適切ではない。しかしながら、ホストコンピュータ上の BP などの学習結果を利用した認識処理を高速に行うことができる。

提案手法で必要な反復回数は、シグモイド領域外では次元数 n にかかわらず平均的に 2 回で収束するという良好な性質を持つ。一方、シグモイド領域内では精度に応じた反復回数が必要である。このため、提案手法を用いて高速な演算を行うためには、シグモイド領域を小さく抑えることが必要である。この条件が適用できるかどうかは学習方法とデータ分布に依存する。データ領域が相互に明確に分離できるような分布の場合に、このような条件が成り立つと考えられる。なお、ここでの平均反復回数の評価は、ニューロン数に対してニューロン回路が少ない場合を想定して、一つのニューロンの平均反復回数を計算した。これはコスト面での制約の大きな組込装置などに当てはまる。しかしながら、ニューロン一つ一つに専用のニューロン回路を割り付けるような大規模なネットワークの場合には、ニューロンの最大反復回数によって演算速度が律速されることになるため、ネットワーク全体のシミュレーション評価を行うことが必要である。

提案手法において使用する算術演算は加減算とビットシフト演算であり乗算を使用しない。このため実装においてもハードウェア乗算器をつかった積和演算回路を必要としないため、積和演算回路を使用する従来手法に比べて回路を小型化できる。提案手法の回路規模はほぼ精度に比例する。一方、従来手法では乗算回路がほぼ精度の自乗に比例して回路規模が大きくなる。このため、高精度の回路においては提案手法がより有利である。

以上の議論より、提案手法はデータ領域が相互に分離できるような問題で高精度の演算を必要とする場合に、従来手法に比べて演算速度、回路規模の両方で有利であると考えられる。

7. ま と め

シグモイド関数付きのニューロンモデルに対する多次元の二分反復法に基づく計算法を提案した。この提案手法で必要な平均反復回数はシグモイド領域の大きさと精度に依存する。提案手法では数値演算として加減算とビットシフト演算のみを用いるため、本手法のハードウェア化を行った場合、乗算器を使用して積和演算を行う従来手法と比べ、小型化が可能である。FPGA を使用した回路設計を行った結果、従来手法に比べて、小さな回路規模と高い最大動作周波数を実現できた。

今後の課題としては学習方法も含めたハードウェア化、並列実装による高速化が挙げられる。

なお、この研究は（財）名古屋産業科学研究所の主催する創発型ソフトコンピュータ開発プロジェクトの一部として

行われた。

（平成 13 年 7 月 24 日受付，同 13 年 12 月 3 日再受付）

文 献

- (1) 安永守利: “超並列計算機を用いた人工ニューラルネットワーク計算とその応用”, 情報処理学会誌, Vol. 42, No. 2, pp. 161-166(2001).
- (2) 岩田穆, 雨宮好仁: ニューラルネットワーク LSI, 電子情報通信学会版 (1995).
- (3) 阿江忠: VLSI ニューロコンピュータ, 共立出版 (1991).
- (4) 末吉 敏則, 飯田 全広: リコンフィギャラブル・コンピューティング, 情報処理学会誌, Vol. 40, No. 9, pp. 777-782(1999).
- (5) Shiratsuchi, S.: FPGA as a Key Component for Reconfigurable System, Int. Conf. on Evolvable Systems from Biology to Hardware, ICES 96(1996).
- (6) Perez-Urive, A., Sanches E.: The Fast Architecture: A Neural Network with Flexible Adaptable-Size Topology, Int. Conf. on Microelectronics for Neural Networks and Systems, MicroNeuro'96 (1996).
- (7) 三浦洋一, 吉野敏樹, 赤松則男, 熊岸 正夫: 機能メモリによるニューラルネットワークとその実現方法, 電子情報通信学会論文誌, D-II, Vol. J81-D-II, No. 11, pp. 2670-2679 (1998).
- (8) 肥川宏臣: ハードウェア化に適した学習機能付き 3 値多層ニューラルネットワーク, 電子情報通信学会論文誌, D-II, Vol. J81-D-II, No. 12, pp. 2811-2818(1998).
- (9) McCulloch, W. S., Pitts, W. H.: A Logical Calculus of the Idea Immanent in Nervous Activity, Bulletin of Mathematical Biophysics, 5, pp. 115-133, 1943.
- (10) Rosenblatt, F.: Principles of Neurodynamics, Spartan(1961).
- (11) 安藤道則, 太田則一, 渡邊章弘: 高集積化をめざした簡略型ニューロンの設計, 豊田中央研究所 R&D レビュー, Vol. 31, No. 4, pp. 13-21(1996).
- (12) 中條直也, 黒柳奨, 道木慎二, 大熊 繁: 多次元二分探索に基づくニューロン演算法, 電気学会 C 部門誌, Vol. 121-C, No. 2, pp. 315-322(2001).
- (13) 大堀隆文, 鎌田修平, 小林靖宏, 小西儀紀, 渡辺一央: 微分不能素子をもつ階層型ニューラルネットに対する誤差逆伝搬法の提案, 電子情報通信学会技術報告, NC99-85, pp. 41-48(2000).
- (14) Rumelhart, D. E., Hinton, G. E., and Williams, R. J.: “Learning Representations by Back-propagation Errors,” Nature, vol. 323-9, pp. 533-536(1986).
- (15) Myers D. J., Vincent J.M. and Orray D. A.: HANNIBAL, A VLSI Building Block for Neural Networks with On-chip Backpropagation Learning, Proc. The 2nd Int. Conf. on Microelectronics for Neural Networks, Vol. 1. 171-181 (1991).

中 條 直 也 (正員) 1982 年名古屋大学大学院工学研究科博士前期課程修了。現在豊田中央研究所勤務。自動車用電子機器, 創発型ソフトコンピュータに関する研究に従事。情報処理学会, 電子情報通信学会, IEEE 各会員。

黒 柳 奨 (非会員) 1996 年名古屋工業大学大学院工学研究科電気情報工学専攻にて博士 (工学) 学位取得。現在名古屋工業大学電気情報工学科助手。ニューラルネットワークおよび聴覚系神経情報処理システムに関する研究に従事。電子情報通信学会, 日本神経回路学会, 日本音響学会各会員。

道 木 慎 二（正員） 1995 年名古屋大学大学院工学研究科博士課程後期課程電子機械工学専攻修了。同年 4 月同助手，2000 年 6 月三重大学工学部電気電子工学科助教授，現在に至る。主としてモータドライブシステム、パターン情報処理技術の制御への応用に関する研究に従事。IEEE，計測自動制御学会，電子情報通信学会各会員（工学博士）。

大 熊 繁（正員） 1972 年名古屋大学大学院修士課程電気工学専攻修了。74 年ケース・ウェスタン・リザーブ大学大学院修士課程システム工学専攻修了。77 年名古屋大学大学院博士課程電気工学専攻修了。同年名古屋大学工学部電気工学科助手，講師，助教授を経て 90 年同教授，現在に至る。ロボティクス，パワーエレクトロニクス，創発型ソフトコンピュータに関する研究に従事。計測制御学会，日本ロボット学会，システム制御情報学会，日本機械学会，IEEE 各会員（工学博士）。