VHDLによる学習可能な階層型ニューラルネットワークのハードウェア実装

山森　一人[1)]・石田二郎[2)]・吉原　郁夫[3)]

Hardware Implementation of Trainable Multi-Layer Neural Network by VHDL

Kunihito YAMAMORI[1)], Jiro ISHIDA [2)], Ikuo YOSHIHARA[3)]

Abstract

Neural network has been used many applications, such as pattern or image recognition, robot control, optimization problem and so on. However, real-world problems need large scale neural networks, and they lead enormous computation time for training process of neural network. To reduce the computation time, we try to implement neural network in a FPGA device. In this paper, we discuss on the performance of hardware neural network from the viewpoint of the processing speed and the scale of the circuit. The trainable hardware neural network used 117,876 cells in FPGA, and it could train the four training patterns in 800ns on the XOR problem.

Key Words:
Neural network, VHDL, Hardware implementation, Training

1. はじめに

ニューラルネットワーク（Neural Network）は，パターン認識，ロボット制御，最適化問題，画像認識など様々な分野に応用されている．しかし，大規模なニューラルネットワークでは，学習に膨大な計算時間が必要となる問題が指摘されている．計算時間を短縮するため，演算を並列処理で行う方法や，ニューラルネットワークをハードウェア上に直接実装する方法が挙げられる[1] [2]．しかし，並列処理では複数の計算機を組み合わせるため，マシンコストやプロセッサ間の通信時にオーバーヘッドが生じる点に問題がある．

本研究では，ハードウェア上に学習可能なニューラルネットワークを実装し，処理性能と回路規模を明らかにすることを目的とする．

2. 階層型ニューラルネットワーク

2.1 ニューロン

ニューラルネットワークとは，多数のニューロンと呼ばれる神経細胞が組み合わさって構成されている生物の脳神経系の機能をモデル化したものである．生物のニューロンが持つ情報伝達の特性をモデル化したものを図1に示す．図1で，$X_j(j=1,\cdots,n)$ をニューロンへの入力信号，$W_j(j=1,\cdots n)$ をニューロン間の結合係数，S を入力信号と結合係数の加重和，ニューロンの出力を Z と表すとき，ニューロンの入出力関係は(1)式，及び(2)式で表される．

1) 工学部情報システム工学科助教授
2) 工学部情報システム工学科学部学生（現職，株式会社テクシア）
3) 工学部情報システム工学科教授

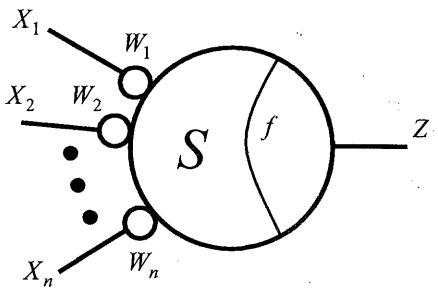

図1 ニューロンモデル

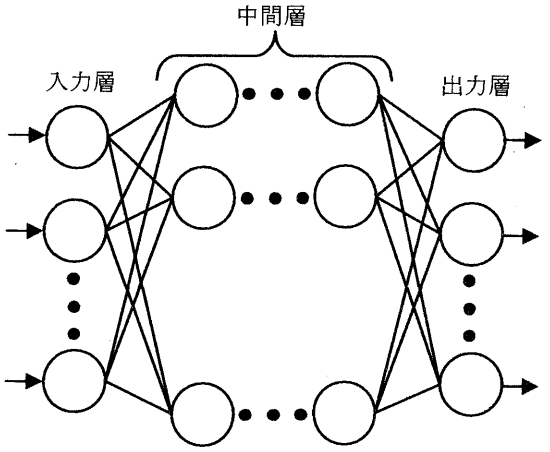

図2 ニューラルネットワークの構造

$$S = \sum_{j=1}^{n} W_j X_j, \tag{1}$$

$$Z = f(S) = \frac{1}{1+e^{-s}}. \tag{2}$$

2.2 階層型ニューラルネットワークの構造

階層型ニューラルネットワークは，図2に示すようにいくつかのニューロンが入力層から出力層に向かって順方向に結合しているネットワークである．階層型ニューラルネットワークは入力層，中間層，出力層の3層をもち，各層のニューロン数及び中間層数は任意である．各層間のニューロンは全結合しており，層内のニューロン間には結合はない．

2.3 誤差逆伝搬法

階層型ニューラルネットワークの代表的な学習法として誤差逆伝搬法（Back Propagation BP法）がある[3]．BP法は，ニューラルネットワークにある入力データを与えたときに得られる出力と教師信号との二乗誤差が小さくなるよう，最急降下法により結合係数の修正を繰り返し最適な結合係数を決定する方法である．

3. ニューラルネットワークのハードウェア設計

3.1 FPGA

設計したい論理回路を，設計現場で即座に実装できるハードウェアを PLD（Programmable Logic Device）と呼ぶ．ハードウェアで機能を実現する理由として，高速処理，小型化，低消費電力の三つが挙げられる．

FPGA（Field Programmable Gate Array）は，PLDの一種である．一般に LSI 中の回路は製造後に変更することはできないが，FPGA は容易に内部の回路を書き換えることができる．そのため，近年では LSI の試作・検証や，携帯電話などの規格変更が早い製品などに多く使用されている．

FPGA 上の回路を設計する場合，回路図を直接記述するスケマチック設計とハードウェア記述言語（HDL）でハードウェアの動作を記述する設計の2種類がある．

3.2 VHDL

ハードウェアを設計する際に，現在ではハードウェア記述言語（HDL）が多く用いられる．スケマチック設計に対して HDL による設計の利点として次のようなものが挙げられる．

- テキストで簡単に入力できる
- 複雑な論理式を求める必要がない
- 回路変更が容易である
- 回路の動作を理解しやすい

本研究では回路設計に HDL の一つである VHDL (VHSIC HDL) を用いた．VHDL は，アメリカ合衆国防省の VHSIC（Very High Speed Integrated Circuit）委員会で 1981 年に提唱された HDL であり，1987 年に IEEE（The Institute of Electrical and Electronics Engineers , Inc）で承認され，世界標準の HDL として広く普及している．高度なディジタル回路の設計や解析，複雑な制御回路などをモデル化し，システム全体を抽象度の高いレベルで記述できるなど柔軟な設計を可能としている[4].

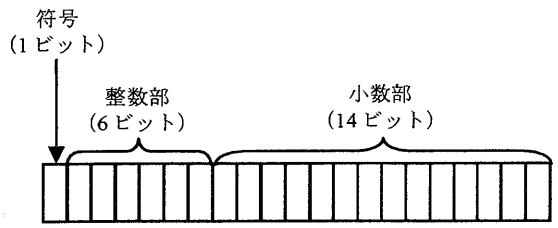

図4 数値の表現方法

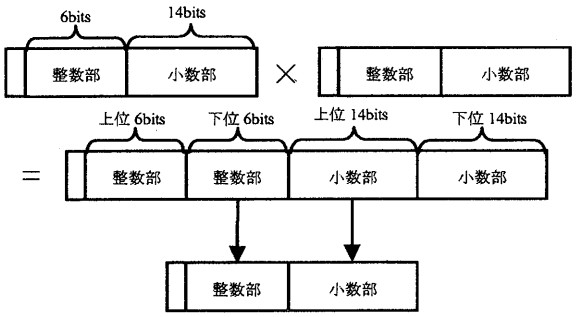

図5 ニューロンでの乗算

3.3 VHDLによるBP法の実装

VHDLでは通常，符号なしの整数値しか扱うことができない．BP法では実数値を用いるため，本研究では入力や結合係数を21ビットの固定小数点形式で表すこととし，図4に示すようにMSBを符号ビット，整数部に6ビット，小数部に14ビットを割り当てた．符号ビットはプラスなら0，マイナスなら1をとり，負数は2の補数で表現する．

21ビットで表現される数値の乗算を行うと，VHDLの仕様上符号ビットを除き40ビットの乗算結果が出力される．40ビットの乗算結果をそのまま以降の計算に用いると，回路規模が大きくなり実装が困難になる．そこで，図5に示した操作を行い，演算結果を21ビットに収めることにした．

まず，符号はそれぞれの入力値の符号ビットのXORをとることにより求めた．次に，乗算結果の整数部12ビットのうち上位6ビット全てが0のとき，整数部下位6ビットの値を乗算結果の整数部としてとる．乗算結果の小数部下位14ビットは後の演算に微小な影響しか及ぼさないと考えて切り捨て，小数部上位14ビットの値をとることとした．

乗算結果の整数部上位6ビットのうち1ビットでも1となっていたときには，最終的な乗算結果となる20ビットの全てのビットを1とし，表せる数値の最大値とした．

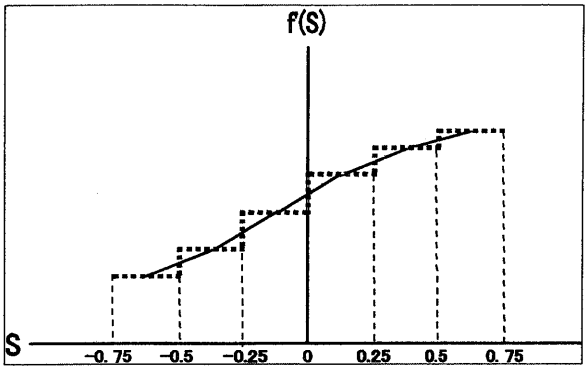

図6 ステップ関数とその微分

これらの操作により，符号1ビット，整数部下位の6ビット，少数部上位の14ビットの21ビットが乗算結果として出力される．

3.3.1 活性化関数

(2)式のシグモイド関数は非線形関数であり，そのままハードウェア上で実現すると回路が複雑になり回路規模が大きくなる．そこで，シグモイド関数の演算結果をあらかじめ適当な刻み幅で値を求め，入力Sに応じて次式に示す近似値をとるステップ関数を作成し，(2)式の代替とした．

$$f(S)=\begin{cases}0.0 & (S<-5)\\ \dfrac{1}{1+\exp(-S)} & (-5\le S\le 5,0.25\text{刻み})\\ 1.0 & (S>5)\end{cases}$$

BP法では，結合係数の更新量を求めるとき上式の導関数が必要となるが，ステップ関数は不連続関数であり微分不可能である．そこで，図6のようにステップ幅の中点と中点を結ぶ直線の傾きをステップ関数の微分値として次式で求め，結合係数の更新量計算に用いることとした．

$$f'(S)=\begin{cases}0.0 & (S<-5,S>5)\\ \dfrac{f(S+0.25)-f(S)}{0.25} & (-5\le S\le 5,0.25\text{刻み})\end{cases}$$

4. 実験

4.1 動作環境

回路合成，及びシミュレーションは以下の環境で行った．

- PC：Pentium III 700MH z 主記憶640MB
- OS：Windows XP

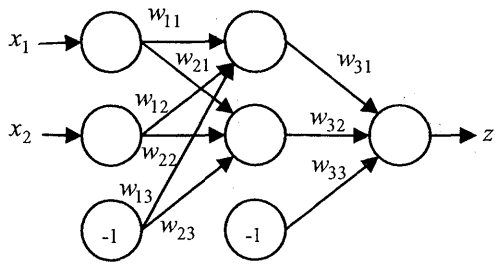

図7 XOR問題用ニューラルネットワーク

- 言語：VHDL
  C言語(比較シミュレーション用)
- 回路合成：Xilinx Foundation ISE
- 回路シミュレーション：Model Techonology
  Modelsim SE

4.2 シミュレーションによる動作検証

2入力の排他的論理和（XOR）について，図7のような階層型ニューラルネットワークモデルを構成した．図7で入力を持たないニューロンはしきい値ニューロンであり，常に-1を出力する．初期結合係数としてランダムな値を与えた上で，XOR問題を学習する回路をVHDLにより設計し，シミュレーションにより動作検証を行った．

ＢＰ法のパラメータは，以下のように設定した．

- 慣性係数：0.9
- 学習係数：0.3
- 学習回数：1000回

検証に用いた初期結合係数を表1，学習により得られた結合係数を表2に示す．表1，表2中のw_{11}～w_{33}は，図7の結合係数w_{11}～w_{33}に対応している．

表2の結合係数を用いて2入力XOR問題の4パターン｛(0,0), (0,1), (1,0), (1,0)｝を与えたときのニューラルネットワークの出力値を表3に示す．また，3.3.1節で述べた活性化関数を使ってPC上でシミュレーションを行ったときの，各パターンでの出力も表3に示した．PC上でのシミュレーション時の初期結合係数は，VHDLでのシミュレーション時と同じ値とした．与えられた入力に対して，VHDL，C言語でのシミュレーションそれぞれで正しい出力が得られていることが分かる．一方，同じパラメータを用いて学習しているにも関わらず，VHDLとC言語によるシミュレーションで学習後の出力値に若干の誤差が生じた．これは，VHDLでは固定小数点形式を用いているのに対して，C言語によるシミュレーションでは浮動小数点形式を用いており，演算精度が異なっているためと考えられる．

初期荷重係数を変化させて行った5回のシミュレーションでの，各入力パターンに対するニューラルネットワークの出力を表4に示す．表4から分かるように，初期荷重係数を変化させた5回のシミュレーションすべての場合で正しい出力を出すような結合係数の組を得ることができた．

表1 初期結合係数

$W11$=−0.6404	$W21$=−0.0286	$W31$= −0.6506
$W12$= 0.8502	$W22$=−0.5678	$W32$= 0.9648
$W13$= 0.4042	$W23$= 0.6488	$W33$= −0.9576

表2 学習後の結合係数

$W11$=−5.3385	$W21$=−3.7570	$W31$=−6.9875
$W12$=−5.3792	$W22$=−3.7587	$W32$= 6.5798
$W13$=1.7963	$W23$=5.2610	$W33$=−2.8774

表3 学習後の出力

	(0,0)	(0,1)	(1,0)	(1,1)
VHDL	0.0879	0.8859	0.8948	0.0957
C言語	0.0945	0.8831	0.8602	0.0986

表4 初期結合係数を変化させた時の学習後の出力

No		(0,0)	(0,1)	(1,0)	(1,1)
1	VHDL	0.0875	0.9249	0.9248	0.0804
	C言語	0.0936	0.8832	0.8801	0.1183
2	VHDL	0.0818	0.9076	0.9062	0.0818
	C言語	0.0890	0.8859	0.8675	0.0968
3	VHDL	0.0843	0.9293	0.9294	0.0879
	C言語	0.1163	0.8916	0.8898	0.1089
4	VHDL	0.0876	0.9076	0.9048	0.0878
	C言語	0.0879	0.8592	0.8697	0.0903
5	VHDL	0.0910	0.8960	0.9029	0.1073
	C言語	0.0926	0.8678	0.8637	0.0942

5 おわりに

本研究では，学習可能なハードウェアニューラルネットワークを VHDL により設計した．Xilinx 社製合成ツール Foundation ISE を用いて，2 入力 XOR 問題を学習するニューラルネットワークについて回路合成を行ったところ，使用したセル数は 117,876 となり，Xilinx 社製の FPGA である Virtex XCV3200 などに実装可能な規模であることが分かった．シミュレーションを行ったところ，4 つの学習パターンをそれぞれ 1 回ずつ学習するのに要した時間は 800ns となり，これは 1 秒間に 125 万回の学習処理に相当する．PC 上で C 言語を用いたシミュレーションでは，同様の処理に 230ms を要しており，ハードウェア化により大幅な高速化が期待できる．また，双方の学習シミュレーション後に得られた結合係数により，正しい出力が得られることを確認できた．

今後の課題としては設計したニューラルネットワークを FPGA 上に実装し，動作検証を行うことが挙げられる．

参考文献

[1] 平井有三，落合辰男，安永守利：“1000 ニューロン 100 万シナプスで構成されたニューラルネットワークハードウェアシステム”，電子情報通信学会論文誌，Vol.J84-D- Ⅱ，No.6,pp.1185-1193(2001)

[2] 川島毅，石黒章夫，大熊繁：“小規模回路で実現可能なニューラルネットワークのハードウェア化手法”，信学技報(NC-99-90)，電子情報通信学会(2000)

[3] 船橋誠壽, 吉原郁夫著：システム制御のための知的情報処理，朝倉書店（1999）

[4] 長谷川裕恭：VHDL によるハードウェア設計入門，CQ 出版(1995)