バイナリーニューラルネットワークの進化的学習と応用

中山, 雄太 / NAKAYAMA, Yuuta

(発行年 / Year)
2013-03-24

(学位授与年月日 / Date of Granted)
2013-03-24

(学位名 / Degree Name)
修士(工学)

(学位授与機関 / Degree Grantor)
法政大学 (Hosei University)

2012年度　修士論文

論文題名　バイナリーニューラルネットワークの進化的学習と応用

指導教授　斎藤 利通 教授

法政大学大学院工学研究科
電気工学専攻修士課程
学生証番号： 11R3128
ナカヤマ　ユウタ
氏名 **中山　雄太**

あらまし

本論文では、バイナリーニューラルネットワーク (Binary Neural Networks:BNN) について研究し、論理合成法との関係を考察していく。BNN はシグナム活性化関数をもち、適切なパラメータ選択を行えば所望のブール関数の近似が可能となる。あるパラメータのとき、BNN はブール関数の論理和標準形と等価となる。また、あるパラメータのとき、BNN は中間層ニューロン数を見ると論理和標準形よりも簡単な構造となる。さらに、効果的なパラメータを設定するために、我々は遺伝的アルゴリズムに基づく学習アルゴリズムを提案する。基本的な数値実験を通して、アルゴリズムの有効性を確認する。

また、本論文では動的バイナリーニューラルネットワーク (Dynamic Binary Neural Networks:DBNN) についても研究し、2 値周期系列の学習について考察していく。DBNN は 3 層構造のネットワークに遅延フィードバックを適用することで構築される。この DBNN の応用として、我々は代表的ないくつかのパワーエレクトロニクスの回路について考えていく。我々はインバータやマトリックスコンバータのスイッチ信号に対応した教師信号の学習について提案する。その際に分析ツールとしてグレーコードに基づくリターンマップ (Gray-code-based return map:Gmap) を導入する。Gmap は周期解の数などの DBNN の基本的な特徴を表わすのに用いることができる。我々は、教師信号の銘記を確認し、周期軌道の自動安定化について考察していく。

Abstract

This paper studies learning of the binary neural networks (BNN) and its relation the logical synthesis. The network has the sigum activation and can approximate a desired Boolean function if parameters are selected suitably. In a parameter subspace the network is equivalent to the disjoint canonical form of the Boolean functions. Outside of the subspace, the network can have simpler structure than the canonical form where the simplicity is measured by the number of hidden neurons. In order to realize effective parameter setting, we present a learning algorithm based on the genetic algorithm. Performing basic numerical experiments, the algorithm efficiency is confirmed.

Also, this paper studies the dynamic binary neural networks (DBNN) and its learning of periodic binary sequence. The DBNN is constructed by applying the delayed feedback to a three-layer network. As an application of object, we study representative power electronics circuits. We propose the learning of teacher signal corresponding to switching signals of the inverter and matrix converter. We give a systematic analysis tool. The Gray-code-based return map that is useful to grasp basic characteristics of the DBNN such as the number of periodic orbits and their domain of attraction. We have confirmed storage of the teacher signal and automatic stabilization of the periodic orbits.

目 次

図 目 次

表目次

第1章　まえがき

　ニューラルネットワークは、脳神経系をモデルにした情報処理モデルである。人間の大脳は、140 億個のニューロンから構成されている。それらのニューロンは相互に結合し、巨大なネットワークを築いている。このネットワークをソフトウェアで再現しようとニューラルネットワークの研究は行われてきた。現在普及しているコンピュータである記憶装置に格納したデータを順番に読み込んで処理するノイマン型コンピュータは、メモリの読み込み速度など物理的な限界がある。さらに、判断基準が変化するパターン認識などの処理の難しさという観点から、別の情報処理のアプローチが考えられた。その一つが非ノイマン型であるニューラルネットワークに代表される情報処理である。近年、コンピュータの飛躍的な性能向上により、膨大な計算量を必要とする処理が可能となってきた。膨大な計算量を必要とする処理では、人間が実際に行うことが難しい。つまり、計算処理では、コンピュータは人間を超えている。その一方で、コンピュータが苦手な分野がある。例として、意味のある情報を選別するパターン認識やデータマイニングなどがある。このような分野では、判断基準が変化するので定式化が困難であり、プログラムによる処理が難しい。しかし、人間の脳は判断基準が変化しても、処理を行うことができる。このような理由から、人間の脳をモデルとしたニューラルネットワークが注目されている。

　ニューラルネットワークは、まず、1943 年にマカロ (McCulloch) とピッツ (Pitts) によるニューロンの数式モデルを提案され、1949 年にニューロン間の結合重み係数の調整に関するヘブ (Hebb) の学習則の提案、1958 年ローゼンブラット (Rosenblatt) によるパーセプトロンの提案、1969 年にミンスキー (Minsky) とパパート (Papert) によるパーセプトロンの限界の指摘、1986 年にラメルハート (Rumelhart) らにより多層パーセプト

ロン (Multi-Layer Perceptron:MLP) が提案され、誤差逆伝播法を学習に用いている。

MLP の誤差逆伝播法アルゴリズムは最急降下法を基本としており、教師信号が適切であれば汎化能力を持つネットワークを構成できる。しかし、誤差逆伝播法アルゴリズムは滑らかな非線形関数を有するニューラルネットワークを対象としているので、入出力が 2 値であるブール関数の近似には適していない。また、近似を行うとしても学習に時間がかかってしまう。そこで、簡素な MLP として入出力信号を 2 値に限定するバイナリーニューラルネットワーク (Binary Neural Networks:BNN) が提案された。

BNN は、N 入力 1 出力であり、MLP と同様な 3 層構造のフィードフォワードネットワークである。それぞれのニューロンは、シグナム活性化関数をもつことで、入力値により 2 値の信号を出力する [1]-[7]。中間層には、3 値をとる荷重パラメータと整数値をとるしきい値パラメータがある。また、出力層の荷重パラメータは 2 値をとる。このようなネットワーク構造により、BNN は所望のブール関数を実現・近似できる。この BNN の応用として、パターン分類、誤り訂正コード、時系列予測などがある [8]-[10]。BNN は、十分な数の中間層ニューロンを用いることで所望の関数を実現できる。しかしながら、取り扱う問題の規模に伴い中間層ニューロン数の増加を引き起こし、同時に学習時間の増加も問題になっている。この問題を解決するために、多くの学習アルゴリズムが研究されてきた。その中の一つに、ヒューリスティックなアルゴリズムである遺伝的アルゴリズム (Genetic Algorithm:GA) があり、これを用いた学習方法は多く提案され、その有効性が示されてきた [11]-[15]。

2 章では、BNN の定義と GA に基づいた学習アルゴリズムについて提案する。BNN は、所望のブール関数を実現できる。また、BNN のネットワークの最適化を行うことで、冗長な部分を取り除いたブール関数を表現することができる。同様にブール関数の論理式を簡単化する方法としてクワインマクラスキー法 (Quine-McCluskey algorithm:QMC) に代表される論理合成法がある。この 2 つの方法を比較し、BNN による論理合成の有効性について考察する [16]。

3 章では、BNN の発展系である動的バイナリーニューラルネットワーク (Dynamic

Binary Neural Networks:DBNN) について紹介する [8][10]。DBNN は、BNN の N 入力 1 出力のフィードフォワードネットワークを発展させ、N 入力 N 出力にし、遅延フィードバックを適用した構造である。また、パラメータは BNN と同様に、入出力信号は 2 値をとり、中間層ニューロンの荷重パラメータは 3 値、出力層ニューロンの荷重パラメータは 2 値、それぞれのしきい値パラメータは整数値をとる。DBNN では、出力信号を入力信号へフィードバックさせる。したがって、DBNN は動的なデジタルシステムであり、様々な周期的な 2 値系列を生成することができる。しかしながら、BNN に対して DBNN についてはあまり研究されていない。そのため、DBNN の GA に基づいた学習アルゴリズムを提案し、いくつかの数値実験によりその有効性を示す。その実験では、いくつかの例題を用いる。例題として、主にパワーエレクトロニクスの実際の回路 [17]-[19] で用いられるスイッチングパターンを学習させ、実験を行う。また、DBNN のダイナミクスの視覚化を行うために、我々は、グレーコードを用いたリターンマップ (Gray-code based return map:Gmap) を導入する。この Gmap を用いて、DBNN のダイナミクスについての考察を行う。

4 章では、本論文のまとめと今後の課題について示す。

第2章　バイナリーニューラルネットワークと論理合成

2.1　まえがき

バイナリーニューラルネットワーク (Binary Neural Networks:BNN)[1]-[6] は 3 層のフィードフォワードネットワークであり、2 値の入力ベクトルを 2 値のスカラー出力へ変換する。また、中間層と出力層のニューロンはシグナム活性化関数をもつ。ニューラルネットワークの中でも代表的なマルチレイヤーパーセプトロン (Multi-Layer Perceptron:MLP) は滑らかな関数のシグモイド関数であるのに対して、BNN のもつ不連続な関数であるシグナム関数はブール関数を取り扱うことに適している。BNN には中間層と出力層にいくつかのパラメータがある。この BNN の最適なパラメータ設定を実現するために、いくつかの学習アルゴリズムが研究されてきた。しかし、いまだに学習アルゴリズムは確立されていない。これは、取り扱う問題の規模に伴う中間層ニューロンの増加や計算コストの上昇など、学習アルゴリズムを考えるにはいくつかの問題点が挙げられる。

問題の 1 つとして、より簡素なネットワーク構造になる最適なパラメータを探索することにある。この問題を考える際に、中間層ニューロン数 (N_H) を用いて BNN の簡素度合を示す。また、N_H はブール関数の論理和標準形 (Disjunctive Canonical Form:DCF) の項数 (N_D) に対応する。この DCF の N_D を最小にするような最適解を探索するために論理合成法を用いることはよく知られている。そのいくつかある古典的な論理合成法の 1 つにクワインマクラスキー法 (Quine-McCluskey method:QMC) がある。本章では、この QMC と BNN の関係について考察していく。もう 1 つの問題として、大抵の場合、入力ベクトルの次元が増加すると、入力パターン数が指数関数的に増加する。そのため、所望のブール関数の実現が困難になる。これらの問題を解決するための学習アル

ゴリズムとして、遺伝的アルゴリズム (Genetic Algorithm:GA) が研究されている。このGA はヒューリスティックなアルゴリズムとして、近似解探索を行う。この問題は、BNN だけでなく QMC でも同様である。さらに、我々は BNN 学習の効率化を図るために、エラー許容率 (Error Tolerance Rate:ETR) について考える。この ETR を考えることで、BNN の学習に曖昧さを取り入れることができる。

本章で提案する学習アルゴリズムは、遺伝的アルゴリズムに基づいている (Learning Algorithm is based on the Genetic Algorithm:GALA)[16]。この GALA を用いた BNN と論理合成法である QMC の特性を考え、BNN の有効性について検証する。

2.2 バイナリーニューロン

本論文では、入出力信号が “-1”, “+1” の 2 値：$\boldsymbol{B} \equiv \{-1, +1\}$ の非線形素子であるバイナリーニューロンを単一の素子として考える。単一のニューロンは N 入力 1 出力で構成される。また、結合荷重パラメータ w は $\{-1, 0, +1\}$ の 3 値であり、しきい値パラメータ T は整数値として扱う。その活性化関数にシグナム関数を持つ。以下にバイナリーニューロンの動作式、そのときに用いるシグナム活性化関数を示す。また、バイナリーニューロンのモデル図を図 2.1、動作図を図 2.2 に示す。

$$
\begin{aligned}
y &= \mathrm{sgn}\left(\sum_{i=1}^{N} w_i x_i - T\right) \\
\mathrm{sgn}(x) &= \begin{cases} +1 & \text{for } x \geq 0 \\ -1 & \text{for } x < 0 \end{cases}
\end{aligned}
\tag{2.1}
$$

ただし、$x_i \in \boldsymbol{B}$ は i 番目の入力、$y \in \boldsymbol{B}$ は出力である。また、w_i は i 番目の入力とニューロンとを結びつける結合荷重パラメータ、T はニューロンのしきい値パラメータである。ニューロン内では、それぞれ N 個の入力信号と結合荷重パラメータを線形結合させる。結合度合により、式 (2.1) や図 2.2 に示すように入力ベクトルと結合荷重ベクトルの内積の和がしきい値以上であれば、“+1” の信号を出力し、しきい値未満であれば “-1” の信号を出力する。式 (2.1) のダイナミクスには、N 次元のユークリッド空間を $N-1$ 次元の分離超平面 (Separating Hyper Plane:SHP) で分離し、結合荷重ベクトル w の指す側の入力に対して “+1” の値を出力し、反対側の入力に対しては “-1” の値を出力する線形分

離の性質がある。そのため、SHP による出力の分類が幾何学的に可能である。ニューロンの結合荷重としきい値が求まればSHP が一意に決まることになる。しかし、このニューロンでは線形分離不可能なものに対して対処できない。次項では、線形分離不可能な問題にも対処可能な多層パーセプトロンの構造をもつニューラルネットワークについて議論していく。

2.3　バイナリーニューラルネットワーク

BNN は図 2.3 に示すような 3 層のフィードフォワードネットワークである。N 次元の 2 値入力ベクトルはシグナム活性化関数をもつニューロンを用いて、2 値のスカラー出力に変換される。

$$
\begin{aligned}
y &= \mathrm{sgn}\left(\sum_{j=1}^{N_H} w_j^o z_j - T^o\right) \\
z_j &= \mathrm{sgn}\left(\sum_{i=1}^{N} w_{ji} x_i - T_j\right) \\
\mathrm{sgn}(X) &= \begin{cases} +1 & \text{for } X \geq 0 \\ -1 & \text{for } X < 0 \end{cases}
\end{aligned}
\tag{2.2}
$$

ただし、出力は $y \in \boldsymbol{B} \equiv \{-1, +1\}$、中間層出力ベクトルは $\boldsymbol{z} \equiv (z_1, \cdots, z_{N_H})$、$j$ 番目の中間層ニューロンの出力は $z_j \in \boldsymbol{B}$、N_H は中間層ニューロン数、N 次元の入力ベクトルは $\boldsymbol{x} \equiv (x_1, \cdots, x_N)$、$i$ 番目の入力は $x_i \in \boldsymbol{B}$ である。j 番目の中間層ニューロンでは、荷重パラメータは 3 値、しきい値パラメータは整数値をとる。

$$
\begin{aligned}
&w_{ji} \in \{-1, 0, 1\},\ i = 1 \sim N \\
&T_j = \sum_{i=1}^{N} |w_{ji}| - \beta_j,\ \beta_j \in \{1, 3, \cdots, B_j\}
\end{aligned}
\tag{2.3}
$$

ただし、B_j は奇数整数値の最大値を意味し、$\sum_{i=1}^{N} |w_{ji}| + 1$ を超えない。出力ニューロンの荷重パラメータとしきい値パラメータは以下のように与えられる。また、出力ニューロンは論理和演算と等価となる。

$$
\begin{aligned}
&w_j^o = +1,\ T^o = -N_H + 1 \\
&\mathrm{sgn}\left(\sum_{j=1}^{N_H} z_j + T^o\right) = \begin{cases} -1 & \text{if } z_j = -1 \text{ for all } j \\ +1 & \text{if } z_j = +1 \text{ for some } j \end{cases}
\end{aligned}
\tag{2.4}
$$

BNN の学習アルゴリズムである GALA は、所望のブール関数 $F_B : \boldsymbol{B}^N \to \boldsymbol{B}$ の実現・近似を行うために w_{ji} と β_j の最適値を探索する。基本的な例として、真理値表によって

定義されるブール関数を考える。

$$\begin{aligned} y = 1 \text{ for } \boldsymbol{x} \in \{000, 001, 010, 100\} \\ y = 0 \text{ for } \boldsymbol{x} \in \{111, 011, 101, 110\} \end{aligned} \tag{2.5}$$

ただし、“0” は “-1” を示し、今後この表記を用いる。また、この真理値表は DCF と等価となる。

$$y = \overline{x}_1\overline{x}_2\overline{x}_3 + \overline{x}_1\overline{x}_2x_3 + \overline{x}_1x_2\overline{x}_3 + x_1\overline{x}_2\overline{x}_3 \tag{2.6}$$

QMC を適用すると、次のような最小形を得られる。

$$y = \overline{x}_2\overline{x}_3 + \overline{x}_1\overline{x}_3 + \overline{x}_1\overline{x}_2 \tag{2.7}$$

この DCF の最小形は BNN と等価となる。

$$\begin{aligned} y = \mathrm{sgn}(z_1 + z_2 + z_3 + 2), \quad z_1 = \mathrm{sgn}(-x_2 - x_3 - 1) \\ z_2 = \mathrm{sgn}(-x_1 - x_3 - 1), \quad z_3 = \mathrm{sgn}(-x_1 - x_2 - 1) \end{aligned} \tag{2.8}$$

ただし、3 つの中間層ニューロンのしきい値 T_j は $\beta_1 = \beta_2 = \beta_3 = 1$ で与えられる。式 (2.8) のネットワーク図を図 2.4 の (a) に示す。しかしながら、このブール関数は中間層ニューロン 1 個のみの BNN で実現することができる。

$$y = \mathrm{sgn}(z_1 + 0),\ z_1 = \mathrm{sgn}(-x_1 - x_2 - x_3 - 0) \tag{2.9}$$

ただし、中間層のしきい値 T_1 は $\beta_1 = 3$ によって与えられる。式 (2.9) のネットワーク図を図 2.4 の (b) に示す。このような BNN と DCF の構造を評価するために、我々はそれぞれ N_H と N_D を用いる。N_H は BNN の中間層数、N_D は DCF の項数を表わす。これらを用いると、式 (2.9) では $N_H = 1$ となり 1 個の中間層ニューロンをもつ BNN と表現でき、式 (2.7) で $N_D = 3$ となる 3 個の項をもつ DCF よりも簡単であると表わせる。一般に、式 (2.3) のすべての j で $\beta_j = 1$ となるならば、BNN は DCF と等価となる。つまり、N_H の最小数は N_D と同数になる。もし、式 (2.3) のいくつかの j について $\beta_j \geq 3$ となるならば、N_H は式 (2.9) で示されているように N_D よりも小さくなることがある。

2.4　バイナリーニューラルネットワーク学習アルゴリズム

我々は所望のブール関数 F_B を実現・近似するために GALA を提案する。準備として、F_B についてのいくつかの定義を与える。$F_B(\boldsymbol{u}_i) = +1$ のとき、$\boldsymbol{B}^N$ の要素 $\boldsymbol{u}_i$ は真

頂点とする。$F_B(\boldsymbol{v}_i) = -1$ のとき、$\boldsymbol{B}^N$ の要素 $\boldsymbol{v}_i$ は偽頂点とする。真頂点と偽頂点の集合はそれぞれ $\boldsymbol{U} = \{\boldsymbol{u}_1, \cdots, \boldsymbol{u}_{N_u}\}$ $\boldsymbol{V} = \{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_{N_v}\}$ によって定義する。もし、$\boldsymbol{U}$ でも $\boldsymbol{V}$ でもない $\boldsymbol{B}^N$ の要素 N_d が存在するならば、それらの要素を “don’t care” とする。これは、$N_u + N_v + N_d = 2^N$ の関係にある。真頂点の集合 $\boldsymbol{U}$ は教師信号として学習で用いる。GALA は、教師信号により中間層ニューロンのパラメータ w_{ji} と β_j を決定する。以下に学習アルゴリズムを示し、フローチャートを図 2.5 に示す。

Step 1 (初期化): j をある中間層ニューロンの番号とし、$j = 1$ とする。

Step 2: j 番目の中間層ニューロンは j 番目の SHP に対応する。

$$\sum_{j=1}^{N} w_{ji}x_i - T_j = 0,\ T_j = \sum_{i=1}^{N} |w_{ji}| - \beta_j \tag{2.10}$$

ただし、w_{ji} と β_j は式 (2.3) を満たす。GALA は図 2.6 に示すように SHP を決定していく。そして、SHPs の数は中間層ニューロン数 N_H に等しくなる。その SHP は GA サブルーチンを適用し、決定される。

Step 3 (判定): 最も評価値の高い染色体を j 番目の SHP とする。また、全ての分離された真頂点を “don’t care” とする。

Step 4: $j = j + 1$ とし、Step 2 へ戻り。これを全ての真頂点が分離されるまで繰り返す。

GA サブルーチン

染色体と評価関数: GA は Mg 個の染色体 $\{\boldsymbol{C_1}, \cdots, \boldsymbol{C_{M_g}}\}$ を用いる。それらの染色体はそれぞれ 3 値の荷重パラメータの候補となる。また、その染色体は次式で構成される。

$$\boldsymbol{C}_k = (w_{j1}^k, \cdots, w_{jN}^k) \tag{2.11}$$

初期染色体には教師信号の $\boldsymbol{U}$ が用いられる。また、評価関数は分離された真頂点数とする。さらに、それぞれの分離の際に ETR を適用する。ETR の適用方法について説明する。j 番目の SHP により分離される真頂点と一緒に偽頂点も分離された場合、その偽頂点の数の割合を求める。その割合を ETR と比較しエラーを許容するかどうかを決定する。そのときの ETR を P_e とする。それぞれの $\boldsymbol{C}_k$ について、パラメータ β_j は $\beta_j = 1$ から B_j まで増加させ最も評価値の高い値が選択される。

ランキング選択とエリート戦略: M_g 個の染色体はランキング選択とエリート戦略を G_{max} 回繰り返し次の世代へ選択される。ランキング選択は評価値をもとに順序をつける。エリート戦略は次の世代に保存されるときに、評価値の最も高い染色体を保存する。その一方で、最も評価値の低い染色体は淘汰される。

交叉と突然変異: M_g 個の染色体に対して、確率 P_c で 2 点交叉を行う。突然変異では、確率 P_m である遺伝子 w^l_{ji} が選択され、その値を $\{-1, 0, +1\}$ のいずれかにセットする。

パラメータ: GA では、次の 4 つのパラメータで特徴づけられる。交叉確率 P_c、突然変異確率 P_m、最大世代数 G_{max}、染色体数 M_g。そして、ETR P_e。ただし、P_c と P_m は乱数によって与えられる。

もし全ての j について $\beta_j = 1$ として固定するなら、このアルゴリズムは DCF と等価な BNN を得るものとして用いられる。この場合のアルゴリズムについて GALA1 と呼ぶ。また、式 (2.8) と (2.9) での BNNs に対応させた SHPs を図 2.6 に示す。このように、与えられた教師信号によりアルゴリズムはネットワークの構造を決定し、目的の関数を近似する。

2.5　数値実験

ここで、いくつかの例を用いてアルゴリズムの有効性を検証する。まず、1 つ目の例として、表 2.1 に示す 8 次元関数を用いる。我々は、全ての j について $\beta_j = 1$ となる

GALA1 の $P_e = 0$ にして適用する。また、GA パラメータは以下のように設定する。

$$(P_c, P_m, G_{max}, M_g) = (0.8, 0.1, 30, \#\text{真頂点})$$

学習後に得られるネットワークを図 2.7、パラメータを表 2.2 に示す。学習後には、表 2.2 に示すような 12 個の中間層ニューロンが得られる。また、BNN は、QMC によって得られる $N_D = 12$ の DCF と等価となる。

一方で、β_j を固定しない GALA の $P_e = 0$ として適用する。学習後に得られるネットワークを図 2.8、パラメータを表 2.3 に示す。表より、$\beta_j = 3$ が存在し、$N_H = 6$(=#SHP) となる。この BNN は、QMC によって得られる $N_D = 12$ である最小値の DCF よりも簡単に表現することができる。この場合の $\beta_j \geq 3$ は、図 2.6(b) に示すような近傍 (あるハミング距離内) に真頂点が存在していることを示している。

次に、P_e を可変にして GALA を適用する。その結果を図 2.9 に示す。図 2.9 では、#SHP (N_H) は P_c と P_m について異なる乱数を用い 10 回試行した平均を示している。この図から、P_e の増加に伴い#SHP (N_H) が減少することがわかる。この場合では、$5 < P_e < 10$ の範囲が実験から適しているといえる。つまり、$P_e = 0$ の場合よりも簡単な構造のネットワークを構成することができる。この例では、QMC を用いた場合、GALA よりも計算コストが高くなってしまう。

さらに、我々は高次元の例題に対して GALA を適用する。この時、教師信号の数は N^2 とする。16 次元のブール関数では 16^2 の教師信号、32 次元のブール関数では 32^2 の教師信号がある。ただし、教師信号は一様乱数を用いて生成した。$N = 16$ についての GA パラメータは以下のように設定する。

$$(P_c, P_m, G_{max}, M_g) = (0.8, 0.1, 60, \#\text{真頂点})$$

$N = 32$ についての GA パラメータは以下のように設定する。

$$(P_c, P_m, G_{max}, M_g) = (0.8, 0.1, 120, \#\text{真頂点})$$

図 2.10 と 2.11 では、P_c と P_m について異なる乱数を用い 10 回試行した平均を示している。図より、図 2.9 と同様の結果が得られた。GALA は $P_e = 0$ ならば、ブール関数

を実現することができる。また、P_e を増加させるにつれて、中間層ニューロン数 (N_H、#SHP) が減少していく。GALA は $5 < P_e < 10$ のとき、比較的簡単な構造の BNN によりブール関数を近似することができる。このような近似はエラーを誤り訂正符号だと考えれば、実際のアプリケーションでも有効であると考えられる。

QMC はこれらの例題で DCF を見つけることは難しい。なぜなら、QMC は真理値表にあるすべての真頂点と “don’t care” を用いる必要があるからである。さらに、エラー許容をもつ論理合成は多く議論されていない。それは、古典的な論理合成は大抵の場合エラーを好まないからである。GALA は頂点の部分集合を用い、あるレベルのエラーを許容することも可能であり、DCF のシステムと等価なものを含んだ BNN を実現することができる。また、BNN は MLP のシステムを含んだ並列システムであるので、一部分が壊れてもブール関数を実現できることも注目すべき点である。

2.6 むすび

本章では、BNN の中間層ニューロンの荷重パラメータを 3 値にし、学習アルゴリズム GALA を用いた。BNN がパラメータ $\beta_j = 1$ としたとき、DCF と等価になることを明らかにした。さらに、GALA は DCF の論理合成に用いることができる。また、$\beta_j \geq 3$ のとき、GALA を用いた BNN は QMC により論理合成を行った DCF より簡単な構造を実現できる。典型的な数値実験によって、GALA が比較的高次元のブール関数でも近似することができることを示した。その上、中間層ニューロン数は ETR が増加するにつれて減少していくことも示した。

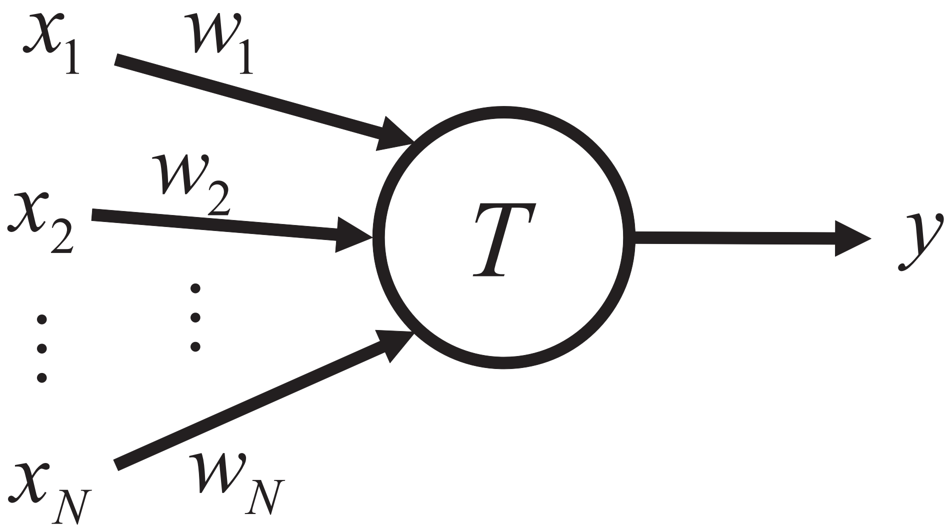

図 2.1: 単一ニューロンのモデル図

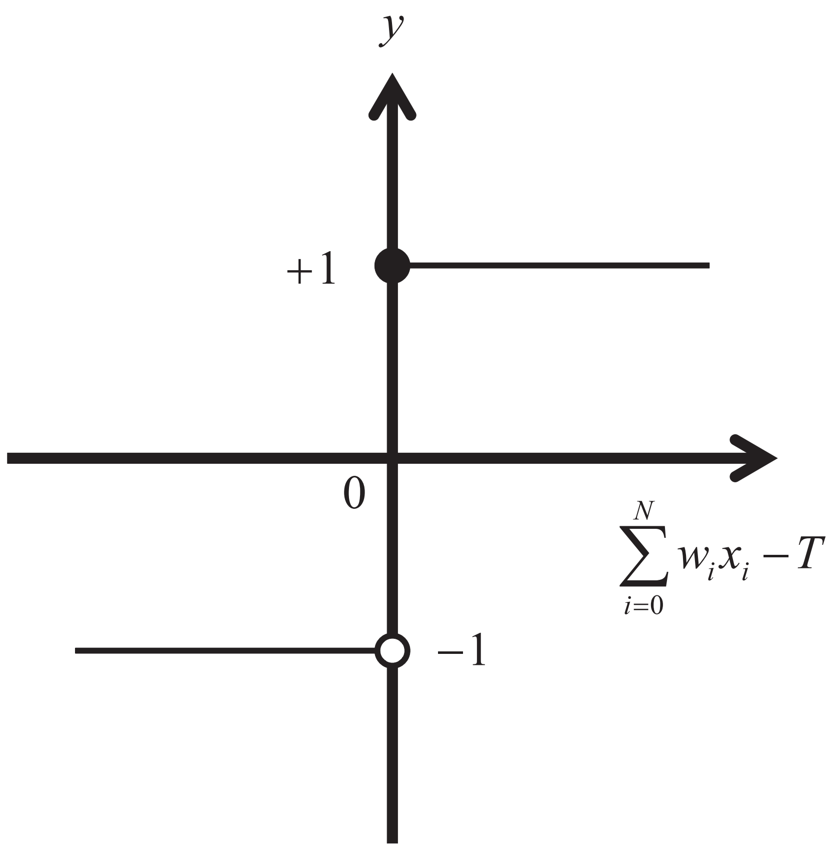

図 2.2: シグナム活性化関数

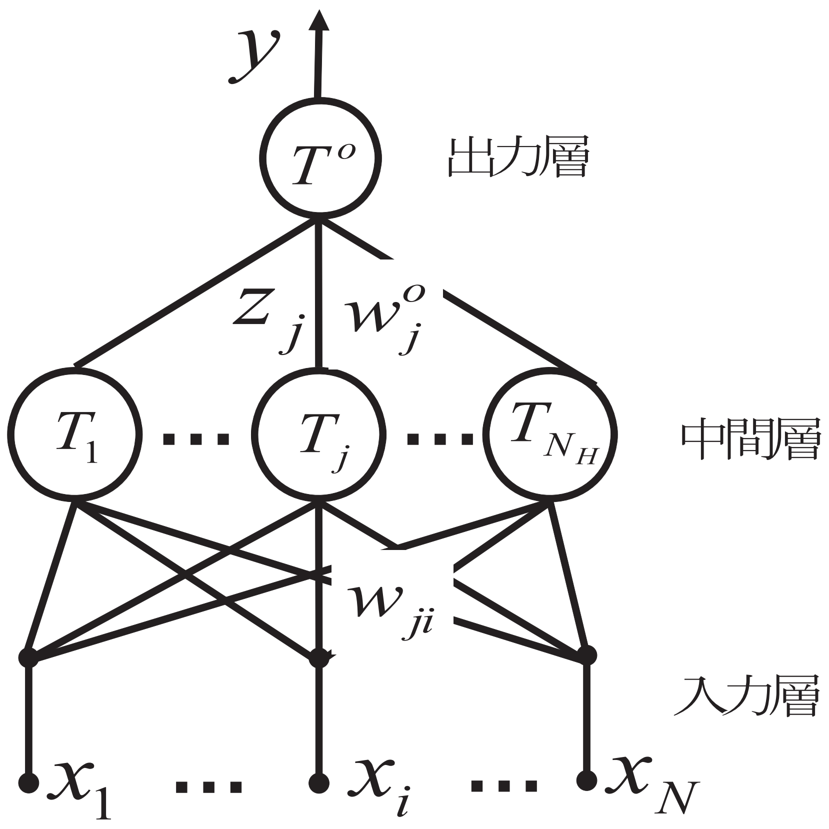

図 2.3: BNN の構造図

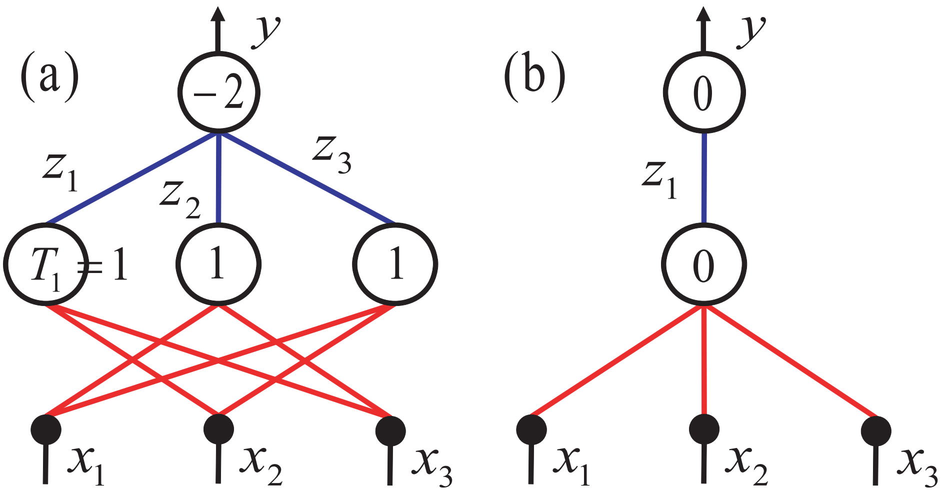

図 2.4: 式 (2.8) と式 (2.9) に対応するネットワーク図
(a) 式 (2.8) のネットワーク。(b) 式 (2.9) のネットワーク。
青い線を $w = +1$、赤い線を $w = -1$、$w = 0$ は接続なしとする。

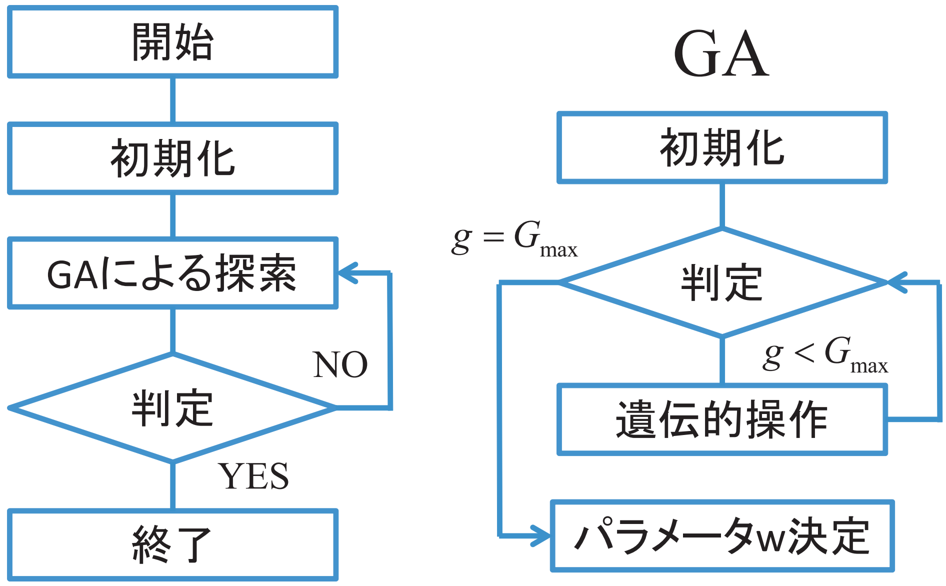

図 2.5: フローチャート (BNN)

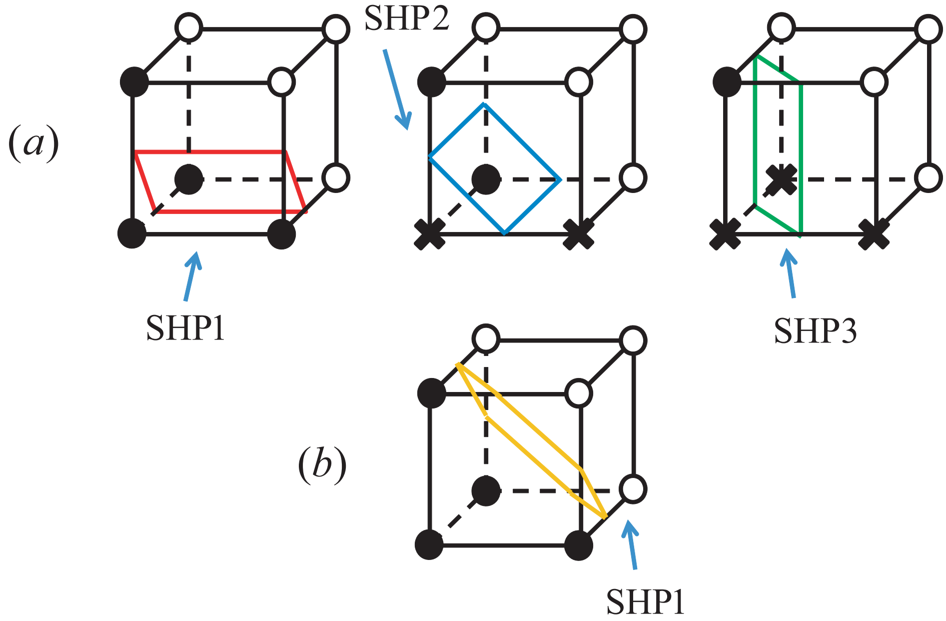

図 2.6: SHP による真頂点 (●) の分離図
(a)3 個の SHPs は式 (2.8) の中間層ニューロンに対応する。
(b)1 個の SHP は式 (2.9) の中間層ニューロンに対応する。
図の●、○、×はそれぞれ教師信号の出力 “+1”、“-1”、“don’t care” を意味する

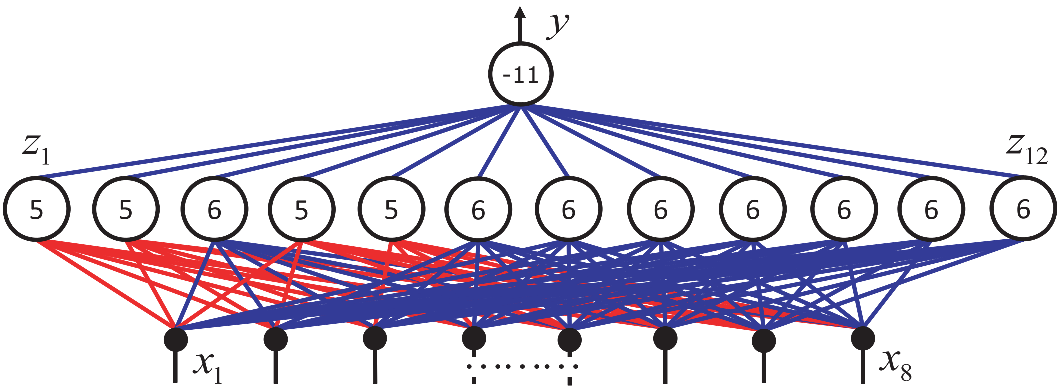

図 2.7: 表 2.2 に対応する BNN

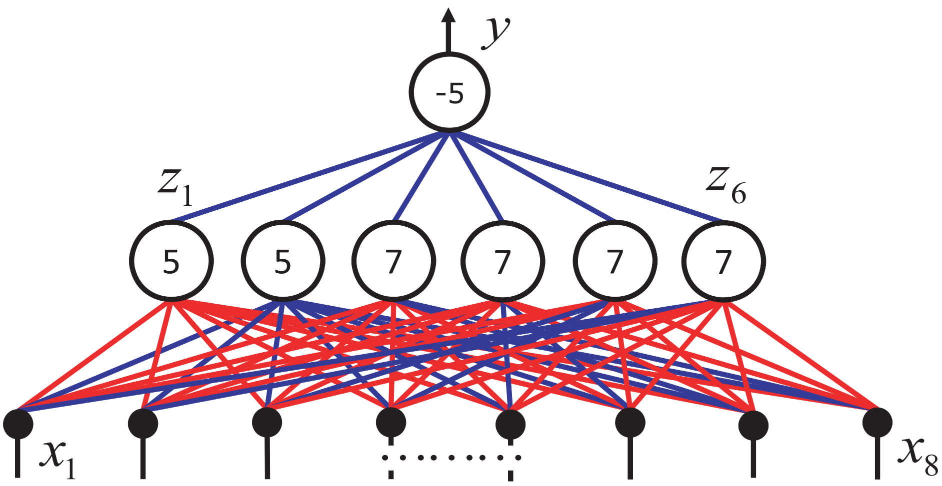

図 2.8: 表 2.3 に対応する BNN

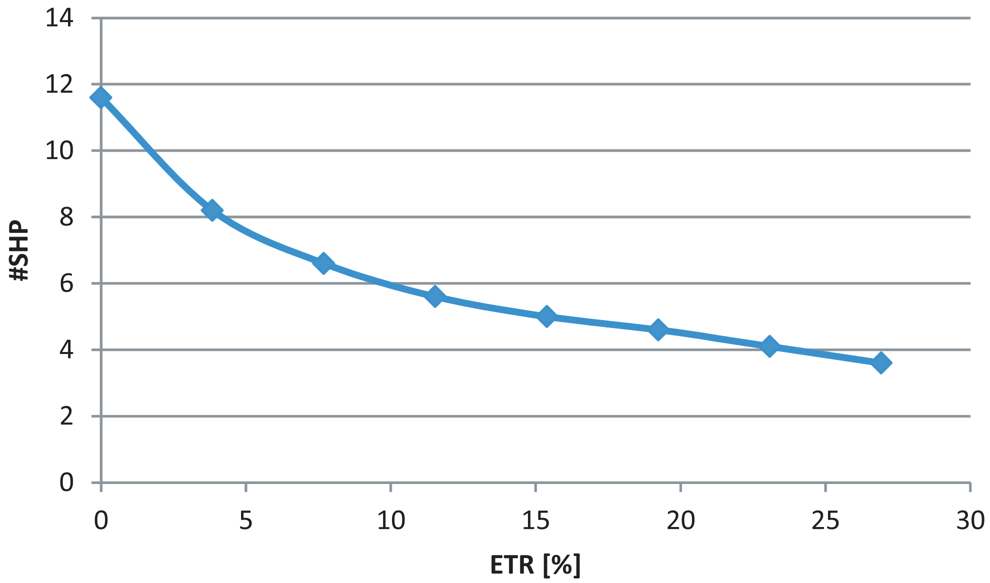

図 2.9: $N = 8$ での中間層ニューロン数 (N_H=#SHP) とエラー許容率 (ETR)

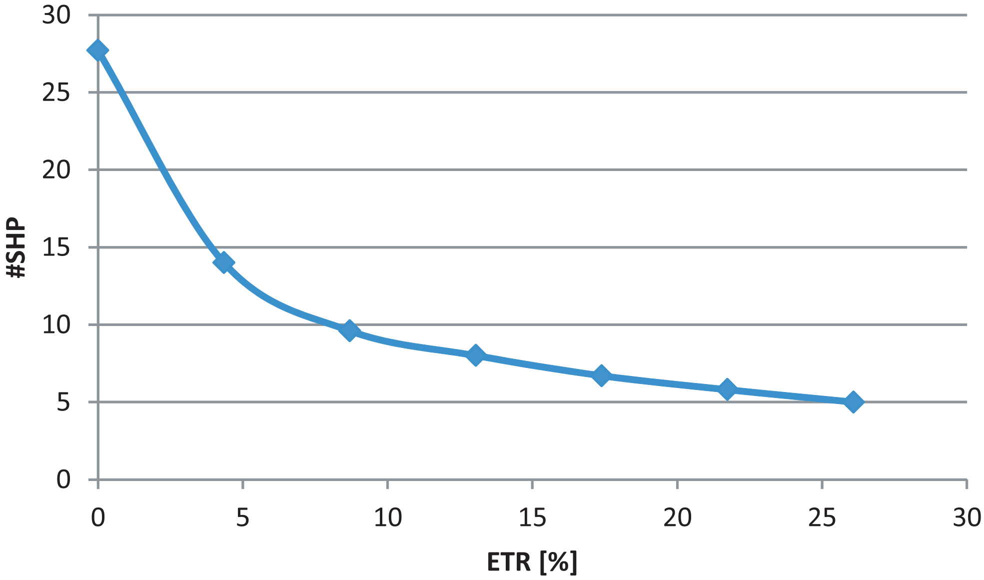

図 2.10: N = 16 での中間層ニューロン数 (N_H=#SHP) とエラー許容率 (ETR)

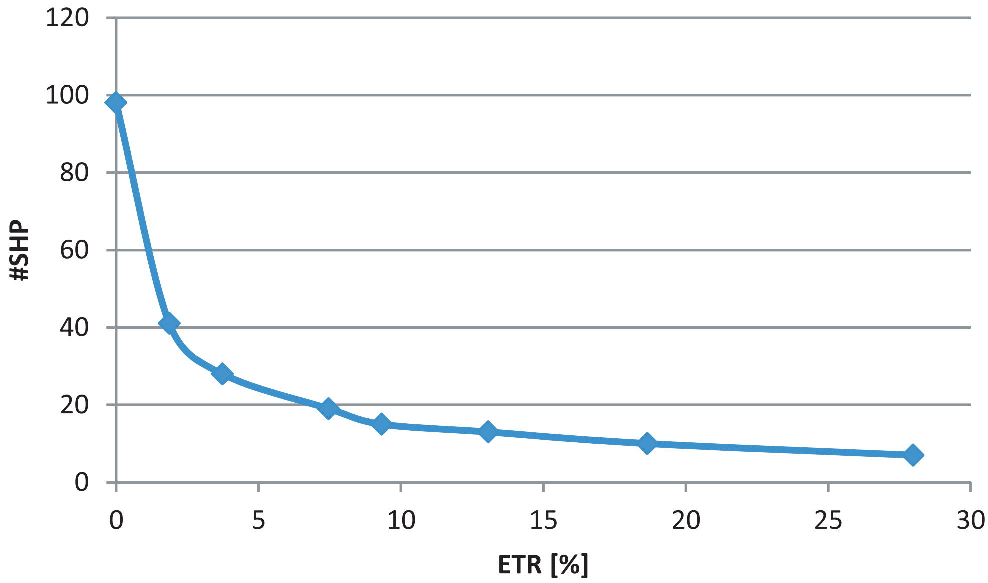

図 2.11: N = 32 での中間層ニューロン数 (N_H=#SHP) とエラー許容率 (ETR)

表 2.1: $N = 8$ の教師信号
“0” は “-1” とする。

$y = 1$	00000000, 00000001, 00000010, 00000011 00000100, 00001000, 00001100, 00010000 00100000, 00110000, 01000000, 01111111 10000000, 10111111, 11000000, 11011111 11101111, 11110111, 11111011, 11111101 11111110, 11111111
$y = 0$	other patterns

表 2.2: 表 2.1 の教師信号について $\beta = 1$ とした GALA1 適用後のパラメータ

j	w_{j1}	w_{j2}	w_{j3}	w_{j4}	w_{j5}	w_{j6}	w_{j7}	w_{j8}	β_j
1	-1	-1	-1	-1	-1	-1	0	0	1
2	-1	-1	-1	-1	0	0	-1	-1	1
3	+1	+1	+1	+1	+1	+1	0	+1	1
4	-1	-1	0	0	-1	-1	-1	-1	1
5	0	0	-1	-1	-1	-1	-1	-1	1
6	0	+1	+1	+1	+1	+1	+1	+1	1
7	+1	0	+1	+1	+1	+1	+1	+1	1
8	+1	+1	0	+1	+1	+1	+1	+1	1
9	+1	+1	+1	0	+1	+1	+1	+1	1
10	+1	+1	+1	+1	0	+1	+1	+1	1
11	+1	+1	+1	+1	+1	0	+1	+1	1
12	+1	+1	+1	+1	+1	+1	+1	0	1

表 2.3: 表 2.1 の教師信号について$\beta \geq 1$とした GALA 適用後のパラメータ

j	w_{j1}	w_{j2}	w_{j3}	w_{j4}	w_{j5}	w_{j6}	w_{j7}	w_{j8}	β_j
1	-1	-1	-1	-1	-1	-1	-1	-1	3
2	+1	+1	+1	+1	+1	+1	+1	+1	3
3	-1	-1	-1	-1	-1	-1	+1	+1	1
4	-1	-1	-1	-1	+1	+1	-1	-1	1
5	-1	-1	+1	+1	-1	-1	-1	-1	1
6	+1	+1	-1	-1	-1	-1	-1	-1	1

第3章　動的バイナリーニューラルネットワークとスイッチ信号の学習

3.1　まえがき

前章で、BNN は N 次元の 2 値入力を 1 次元の 2 値出力とする 3 層構造のフィードフォワードネットワークであることを説明した。本章では、この BNN に遅延フィードバックを施し、2 値系列を呈することができる動的バイナリーニューラルネットワーク (Dynamic Binary Neural Networks:DBNN) を構築した。DBNN では、十分な数の中間層ニューロンを用いることにより、様々な 2 値周期軌道 (Binary Periodic Orbit:BPO) を生成できる。この特性により、DBNN は信号処理や回路のスイッチング制御などへの応用が考えられる。しかしながら、BNN と同様に DBNN も取り扱う問題の規模に伴う中間層ニューロン数の増加や、計算コストの上昇などの問題点がある [8] [10] [11]。

これらの問題を解決するために、いくつかの学習法が提案されてきた。我々は DBNN にも、前章と同様に GA に基づく学習を提案する。DBNN に所望の BPO を銘記させる学習を考察し、アルゴリズムの有効性を確認してきた。

本章では、様々な BPO を生成できる DBNN の学習とその応用について考察する。この DBNN は、シグナム活性化関数を有し、荷重パラメータを整数としている。この DBNN は 2 値パターンやデジタル回路による実装の実現に適している。

この DBNN の新しい応用として、主にマトリックスコンバータの制御スイッチ信号に対応する BPO の学習を考察する。マトリックスコンバータは、AC/AC コンバータの一種であり、パワーエレクトロニクスの代表的な回路の一つとして、盛んに研究されている [18] [19]。与えられた AC 信号を所望の振幅と周波数の AC 信号に効率よく変換することを目的とし、様々な手法が研究されている。この数値実験だけではなく、2 ス

パイラル問題 [3] の一部を BPO として対応させた教師信号やパワーエレクトロニクスの DC/AC インバータの制御スイッチ信号 [17] に対応する BPO の学習も行い、様々な BPO に対応できることを示す。本章では、最も基本的な制御信号に対応する BPO を教師信号とし、GA に基づく方法で学習させる。そして、この BPO が学習でき、さらに、自動的に安定化できることを示す。

3.2　動的バイナリーニューラルネットワーク

DBNN は図 3.1 に示す 3 層構造のネットワークに遅延フィードバックを適用することで構成される。各ニューロンはシグナム活性化関数を有し、DBNN のダイナミクスは以下のように記述される。

$$
\begin{aligned}
x_i(t+1) &= \mathrm{sgn}\left(\sum_{j=1}^{M} w_{ij}^{o}\xi_j(t) - T_i^{o}\right),\ i = 1 \sim N \\
\xi_j(t) &= \mathrm{sgn}\left(\sum_{i=1}^{N} w_{ji}x_i(t) - T_j\right),\ j = 1 \sim M \\
\mathrm{sgn}(X) &= \begin{cases} +1 & \text{for } X \geq 0 \\ -1 & \text{for } X < 0 \end{cases}
\end{aligned}
\tag{3.1}
$$

ただし、$\boldsymbol{x}(t) = (x_1(t), \cdots, x_N(t)),\ x_i(t) \in \{-1, +1\} \equiv \boldsymbol{B},$ は離散時間 t での N 次元状態ベクトルである。また、$\boldsymbol{\xi}(t) \equiv (\xi_1(t), \cdots, \xi_M(t)),\ \xi_j(t) \in \boldsymbol{B},$ は t での M 次元中間出力ベクトルである。また、現在の状態 $\boldsymbol{x}(t)$ は入力ベクトルとして適用され、DBNN は中間層ニューロンを経由して次の状態 $\boldsymbol{x}(t+1)$ を出力する。DBNN は N 次元 2 値ベクトルの写像を実現し、その繰り返しは様々な時空間パターンを生成することができる。簡潔にするために、式 (3.1) を次式で略記する。

$$
\boldsymbol{x}(t+1) = F_D(\boldsymbol{x}(t)),\ F_D : \boldsymbol{B}^N \to \boldsymbol{B}^N \tag{3.2}
$$

中間層ニューロンは 3 値の荷重パラメータ w_{ji} と整数のしきい値パラメータ T_j で特徴づけられる。

$$
w_{ji} \in \{-1, 0, +1\},\ T_j = \sum_{i=1}^{N} |w_{ji}| - \beta_j,\ \beta_j \in \{1, 3, 5, \cdots\} \tag{3.3}
$$

ただし、$i = 1 \sim N,\ j = 1 \sim M$ と β_j は中間層ニューロン数 M を制御できる内部パラメータである。出力層ニューロンは 2 値の荷重パラメータ w_{ij}^{o} と整数のしきい値パラメータ

T_i^o で特徴づけられる。

$$w_{ij}^o \in \{0, +1\},\ T_i^o = 1 - \sum_{j=1}^{M} w_{ij}^o \tag{3.4}$$

出力層のしきい値 T_i^o は w_{ij}^o により一意に決定する。それぞれの出力層ニューロンは $w_{ij}^o = +1$ を用いた中間層出力 $\xi_j(t)$ の論理和演算と等価となる。つまり、DBNN のダイナミクスは 3 種類のパラメータ $w_{ji}, \beta_j, w_{ij}^o$ により制御される。

3.2.1　解析ツール

DBNN のダイナミクスを解析するために、我々はグレーコードに基づくリターンマップ (Gray-code-based return map:Gmap) を導入する。まず、格子点について定義する。

$$I_N \equiv \{l_0, l_1, \cdots, l_{2^N-1}\} \subset [0, +1),\ l_i = i/2^N$$

ただし、$i = 0 \sim 2^N - 1$ となる。この集合は N 次元の 2 値ベクトル $\boldsymbol{B}^N$ の集合に対応する。便宜のために、ハミング距離に基づいたグレーコードで $\boldsymbol{B}^N$ を表記する。

$$G_N : I_N \to \boldsymbol{B}^N$$

マップ G_N は格子点をグレーコードによって表記される $\boldsymbol{B}^N$ に変換する。例えば、$N = 3$ では

$$\begin{array}{ll}
G_3(0/8) = (-1, -1, -1) & G_3(4/8) = (+1, +1, -1) \\
G_3(1/8) = (-1, -1, +1) & G_3(5/8) = (+1, +1, +1) \\
G_3(2/8) = (-1, +1, +1) & G_3(6/8) = (+1, -1, +1) \\
G_3(3/8) = (-1, +1, -1) & G_3(7/8) = (+1, -1, -1)
\end{array}$$

式 (3.2) の DBNN F_D に G_N を適用すると、Gmap が得られる

$$F_g : I_N \to I_N,\ F_g = G_N^{-1} \circ F_D \circ G_N$$

つまり、DBNN のダイナミクスは次のようにまとめられる。

$$\theta(t+1) = F_g(\theta(t)),\ \theta(t) = G_N^{-1}(\boldsymbol{x}(t)) \in I_N$$

ただし、$\theta(t)$ はグレイコードによって表記される $\boldsymbol{x}(t)$ に対応する格子点である。図 3.2 に Gmap の例を示す。図では $2^3 = 8$ の格子点がグレイコードによって表記される 3 次

元の 2 値ベクトル $\boldsymbol{x}(t) \in \boldsymbol{B}^3$ に対応する。この Gmap は 6 周期の周期解をもつ。また、Gmap には膨大な数のパターンがある。例えば、$N = 3$ の簡素な場合でさえ 8^8 パターンもある。

ここで、いくつかの基本的な定義をする。点 $\theta_p \in I_N$ は $F_g^p(\theta_p) = \theta_p$, $F_g^k(\theta_p) \neq \theta_p$, $0 < k < p$ であるならば、p 周期の 2 値周期点 (Binary Periodic Point:BPP) という。ただし、F_g^p は F_g の p 回合成写像である。BPPs の数列 $\{F_g(\theta_p), \cdots, F_g^p(\theta_p)\}$ は p 周期の BPO という。図 3.2 にある赤い軌道は表 3.1 の 6 周期の BPO を示す。

3.3 動的バイナリーニューラルネットワーク学習アルゴリズム

我々は教師信号が T 周期をもつ BPO であると考える。

$$\begin{aligned} &\boldsymbol{z}(1), \boldsymbol{z}(2), \cdots, \boldsymbol{z}(T), \boldsymbol{z}(T+1) = \boldsymbol{z}(1) \\ &\boldsymbol{z}(t_k) \neq \boldsymbol{z}(1) \text{ for } 1 < t_k \leq T \end{aligned} \tag{3.5}$$

ただし、教師信号は未知関数 $\boldsymbol{F}_u$: $\boldsymbol{z}(t+1) = \boldsymbol{F}_u(\boldsymbol{z}(t))$ のサンプルと仮定する。学習では、中間層ニューロンの数などの構造が可能な限り簡単になる DBNN で教師信号を実現しようとする。

学習において、全探索はパラメータを見つけるのにとても困難である。$N = 6$ と $M = 6$ でさえも、w_{ji} の探索空間は $3^{6\times6}$ の候補で構成される。また、論理合成 [1] を繰り返すことでも教師信号を実現できるが、この場合は $\beta_j = 1$ とした $\boldsymbol{B}^N$ から $\boldsymbol{B}$ への BNN の N 個の並列処理に対応し、冗長な中間層ニューロンが発生する。また、活性化関数が不連続であるので傾きの情報は使えない。よって、いくつかのヒューリステックなアルゴリズムが要求される。

本論文では、学習に GA を適用する。準備として、いくつかの定義をする。便宜のため、教師信号の i 番目の要素について考える。

$$\begin{aligned} &z_i(t+1) = F_{ui}(\boldsymbol{z}(t)) \\ &F_{ui} : \boldsymbol{B}^N \to \boldsymbol{B},\ i = 1 \sim N,\ t = 1 \sim T. \end{aligned} \tag{3.6}$$

ただし、$\boldsymbol{F}_u \equiv (F_{u1}, \cdots, F_{uN})$ となる。F_{ui} は N 次元空間での超立方体を用いることで表現できる: $\boldsymbol{z}(t) \in \boldsymbol{B}^N$ は 2^N の頂点の一つに対応し、$z_i(t+1) = F_{ui}(\boldsymbol{z}(t))$ はその頂点のバイナリ値に対応する。例えば、+1 ≡ ●と −1 ≡ ○となる。$\boldsymbol{F}_u$ は図 3.3 に示すように N 次元の超立方体に対応する。

F_{ui} について、もし $z_i(t+1) = +1$ ならば、要素 $\boldsymbol{z}(t) \in \boldsymbol{B}^N$ は真頂点といい、もし $z_i(t+1) = -1$ ならば、偽頂点という。i 番目の教師信号について、真頂点と偽頂点の集合はそれぞれ $\boldsymbol{U}_i = \{\boldsymbol{u}(1), \cdots, \boldsymbol{u}(T_u)\}$, $\boldsymbol{V}_i = \{\boldsymbol{v}(1), \cdots, \boldsymbol{v}(T_v)\}$ で示され、$T_u + T_v = T$ を満たす。

N 次元 2 値空間 $\boldsymbol{B}^N$ では、$\boldsymbol{U}$ でも $\boldsymbol{V}$ でもない要素は “don’t care” という。式 (3.1) の j 番目の中間層ニューロンは N 次元空間での j 番目の SHP に対応する。

$$\text{The } j\text{-th SHP: } \sum_{i=1}^{N} w_{ji}x_i(t) - (\sum_{i=1}^{N} |w_{ji}| - \beta_j) = 0 \tag{3.7}$$

SHP の例を図 3.3 に示し、これに対応する DBNN のネットワークを図 3.4 に示す。式 (3.6) での i 番目の座標の真頂点すべてが完全に分離されるなら、i 番目の中間層部分集合は完全であるという。我々の学習アルゴリズムは $i = 1 \sim N$ についてすべての中間層部分集合を完全にするように中間層ニューロンのパラメータ (w_{ji}, β_j) を見つけようとする。もし、すべての中間層部分集合が完全になるなら、その出力層ニューロンのパラメータは中間層部分集合の論理和演算を実現するように自動的に決定される。学習アルゴリズムは真頂点 $\boldsymbol{U}_i$, $i = 1 \sim T_u$ を用いてパラメータ (w_{ji}, β_j) を決定する。以下に学習アルゴリズムを示し、フローチャートを図 3.5 に示す。

Step 1:（初期化）j をある中間層ニューロンの番号とし、$j = 1$ とする。

Step 2:(SHP 構成) GA サブルーチンを適用し、j 番目の SHP を決定する。評価関数として、j 番目の SHP により分離される真頂点数を利用する。この際に、$F_{u1}, \cdots, F_{uN}$ 全てのネットワークに SHP を適用し分離される真頂点数をカウントする。この $F_{u1}, \cdots, F_{uN}$ での分離数を合計したものを評価関数とする。

Step 3:(SHP 共有) j 番目の SHP を決定する際に真頂点がカウントされた F_{ui} へ SHP を適用する。これを共有するという。さらに、分離された真頂点を “don’t care” とする。

Step 4: $j = j + 1$ とし、Step2 へ戻る。全ての真頂点が分離されるまで繰り返す。

GA サブルーチン

適応度は分離された真頂点数に対応する。M_g 個の染色体 $\{\boldsymbol{C}_1, \cdots, \boldsymbol{C}_{M_g}\}$ は j 番目の SHP の荷重 w_{ji} の候補である。

$$\boldsymbol{C}_k = (w^k_{j1}, \cdots, w^k_{jN}), k = 1 \sim M_g \tag{3.8}$$

初期染色体は教師信号の真頂点を基にランダムに作成する。それぞれの $\boldsymbol{C}_k$ について、最も良い適応度になるようにパラメータ $\beta_j \geq 1$ を選択する。M_g 個の染色体はエリート戦略とランキング選択を G_{max} 回繰り返すことで次の世代に選択される。2 点交叉は確率 P_c。突然変異は確率 P_m で一つの遺伝子 w^l_{ji} を選択し、その値はそれぞれ 1/3 の確率で $-1, 0, +1$ のいずれかになる。

3.4　数値実験

　学習アルゴリズムの効果と学習後の DBNN の特性について検証するためにいくつかの数値実験を行う。ここで重要なことは、学習アルゴリズムにより決定される DBNN のいくつかのパラメータである。学習により得られるネットワークとパラメータについて示し、その DBNN のダイナミクスを確認するために Gmap を適用する。

3.4.1　2 スパイラルに対応した教師信号

　まずは、図 3.6 に示すような 2 スパイラル問題 [3] の一部を取り出したものを教師信号として用いる。□を “-1”、■を “+1” として教師信号として変換したものを表 3.2 に示す。この教師信号は 8 次元で 8 周期の周期的な信号パターンを表わすものである。パラメータを以下のように設定し、学習を行う。

$$(M_g, P_c, P_m, G_{max}) = (30, 0.8, 0.1, 30) \tag{3.9}$$

学習後の DBNN を図 3.7 に示し、この DBNN のパラメータを表 3.3 に示す。この DBNN は、中間層ニューロンを 8 個用いることで構成される。このネットワークの特性を確認するために Gmap を適用する。その結果を図 3.8 に示す。図では、赤い周期軌道は教師信号の周期解を示している。よって、図を見ると教師信号の周期解を銘記することに成功していることがわかる。また、この図には定常状態にある軌道のみを表わしている。つまり、教師信号以外の初期値 248 個から出発した軌道は全て教師信号の周期解に収束するということが確認できる。学習では教師信号の周期解以外の情報は一切使われていないので、自動的に安定化されているといえる。これは、学習の汎化能力の一つである。この自動安定化は、教師信号が回路の制御信号などに対応する場合、回路のロバスト化に貢献できると考えることができる。したがって、2 つ目の教師信号からは回路のスイッチ信号に対応した教師信号を学習していく。

3.4.2　インバータのスイッチ信号に対応した教師信号

　次は、実際の回路として使われているインバータのスイッチ信号を用いる。このインバータは、DC/AC 変換機である [17]。図 3.9 に回路モデルを示す。図に示すように、スイッチング素子を 6 個配置し、それぞれのスイッチング素子のオン/オフのタイミングを制御することで直流を交流へ変換する。図 3.10 に示すスイッチングパターンを教師信号として DBNN で学習させる。このスイッチングパターンを用いてスイッチ制御を行い、出力される電圧波形を図 3.11 に示す。ここで、スイッチングパターンを教師

信号として変換したものを表 3.4 に示し、その教師信号について説明する。図 3.10 との対応を確認すると、この教師信号で用いられている “-1” がスイッチオフ状態、“+1” がスイッチオン状態に対応している。この教師信号は 6 次元で 6 周期の周期的な信号パターンを表わすものである。パラメータを以下のように設定し、学習を行う。

$$(M_g, P_c, P_m, G_{max}) = (20, 0.8, 0.1, 20) \tag{3.10}$$

学習後の DBNN を図 3.12 に示し、この DBNN のパラメータを表 3.5 に示す。この DBNN は、中間層ニューロンを 6 個用いることで構成される。このネットワークの特性を確認するために Gmap を適用する。その結果を図 3.13 に示す。図より、赤い周期軌道が表れている。したがって、この実験でも教師信号の周期解を銘記することに成功していることがわかる。また、先ほどと同様に考えれば、教師信号以外の初期値 58 個から出発した軌道は全て教師信号の周期解に収束するということが確認できる。よって、この場合も教師信号の周期解以外の情報を使うことなく自動安定化される。

今回、実際用いられている回路のスイッチ信号に対応した教師信号を学習させ、その汎化能力も確認した。しかし、これだけでは特定のただ一つの回路に対して有効であるとしかいえない。最後の数値実験として、さらにスイッチの数を増やし、AC/AC 変換の直接変換を可能にしたマトリックスコンバータのスイッチ信号に対応した教師信号を学習させ、DBNN の有効性を検証する。

3.4.3 マトリックスコンバータのスイッチ信号に対応した教師信号

最後に、マトリックスコンバータ [18] [19] のスイッチ信号を用いて数値実験を行う。このマトリックスコンバータとは交流電力を直接、周波数や振幅が異なる交流電力に変換することができる AC/AC 変換器である。このマトリックスコンバータは既存のインバータに比べて、外形寸法が小さく、変換効率が高く、また、高周波電流の発生量が少ない。さらに、メンテナンス・フリーが実現できるといったメリットを備えている [20]。このマトリックスコンバータは 9 個のスイッチング素子で構成される。これ

らの素子のオン/オフを最適なタイミングで切り替えて、所望の周波数と振幅の交流出力を得ることができる。このスイッチング素子の制御方法として AC 直接方式がある。この制御方法に用いるスイッチングパターンを教師信号として、DBNN の数値実験を行った。

まず、図 3.14 にマトリックスコンバータの回路モデルを示す。入力が 3 相交流に、出力が 3 相交流に対応した回路である。スイッチング素子を格子状に 9 個配置し、それぞれのスイッチング素子のオン/オフのタイミングを制御することで、交流電力を振幅や周波数が異なる交流電力に変換する。次に、図 3.15 で回路へ入力される電圧波形を示す。この 3 相交流波形は V_r が R 相入力波形、V_s が S 相入力波形、V_t が T 相入力波形である。これらの波形について図 3.16 に示すスイッチングパターンを用いて AC 直接方式で回路制御を行う。

ここで、DBNN で学習する教師信号について説明する。図 3.16 に示したスイッチ信号は V 相におけるスイッチングパターンであり、出力波形の振幅の最大値を 78[V] に変換する制御を行うものである。これを教師信号として表記したものを表 3.6 に示す。次に、表 3.6 に示す教師信号を説明する。この教師信号の下位 3 ビットはスイッチングパターンを示す。“-1” がスイッチオフ状態であり、“+1” がスイッチオン状態である。その左隣の 3 ビットがその時のスイッチングパターンの継続時間を示す。残りの 2 ビットが関数関係を成り立たせるための矛盾回避に用いている。この教師信号は V 相におけるスイッチングパターンを示していることを説明した。次に残りの 2 相のスイッチ制御について説明する。まず、U 相におけるスイッチ信号は V 相のスイッチングパターンを示すビットを左へ 1 ビットシフトさせ、位相を変えることでスイッチ信号を生成できる。次に、W 相におけるスイッチ信号は V 相のスイッチングパターンを示すビットを右へ 1 ビットシフトさせ、位相を変えることでスイッチ信号を生成できる。各相のスイッチングパターンを図 3.17 に示す。したがって、1 相分の教師信号を学習させ DBNN を生成し、その特性について考察する。今回は V 相についての教師信号を学習させる。

パラメータを以下のように設定し、学習を行う。

$$(M_g, P_c, P_m, G_{max}) = (30, 0.8, 0.1, 30) \tag{3.11}$$

学習後のDBNNを図3.18に示し、このDBNNのパラメータを表3.7に示す。このDBNNは、中間層ニューロンを20個用いることで構成される。このネットワークの特性を確認するためにGmapを適用する。その結果を図3.19に示す。この図にある周期軌道は教師信号の周期解を示している。図より、前2つのGmapと同様に、赤い周期軌道が表れている。したがって、教師信号の周期解を銘記することに成功していることがわかる。さらに、教師信号以外の初期値237個から出発した軌道は全て教師信号の周期解に収束するということが確認できる。よって、この場合も教師信号の周期解以外の情報を一切使うことなく自動安定化される。

この教師信号やそれにより得られるDBNNを用いてスイッチ制御を行い、出力される3相の電圧波形を図3.20から図3.22に示す。図に示す各相における出力電圧波形の振幅の最大値は74[V]となっている。誤差は生じるが、入力電圧波形の振幅を変換した出力電圧波形が生成できる。この結果から、インバータやマトリックスコンバータなどパワーエレクトロニクスの回路に適用できることを確認できた。したがって、自動安定化できるこのDBNNのシステムを用いることで、ロバスト性をもたせた回路にすることができる。

3.5　むすび

本章では、2スパイラル問題の一部をBPOとした教師信号やインバータやマトリックスコンバータのスイッチ制御信号に対応するBPOを教師信号としたDBNNについて考察した。その教師信号の学習方法や学習後に得られるDBNNのダイナミクスを確認し、その特性について検討した。典型的な結果として、その教師信号BPOを銘記することができ、教師信号以外の初期値から出発する信号は全て教師信号の周期解に収束することが確認できた。また、学習では教師信号の周期解以外の情報は一切使われていないことから、自動安定化されるシステムであることを示した。これにより、DBNN

の汎化能力を確認することができた。

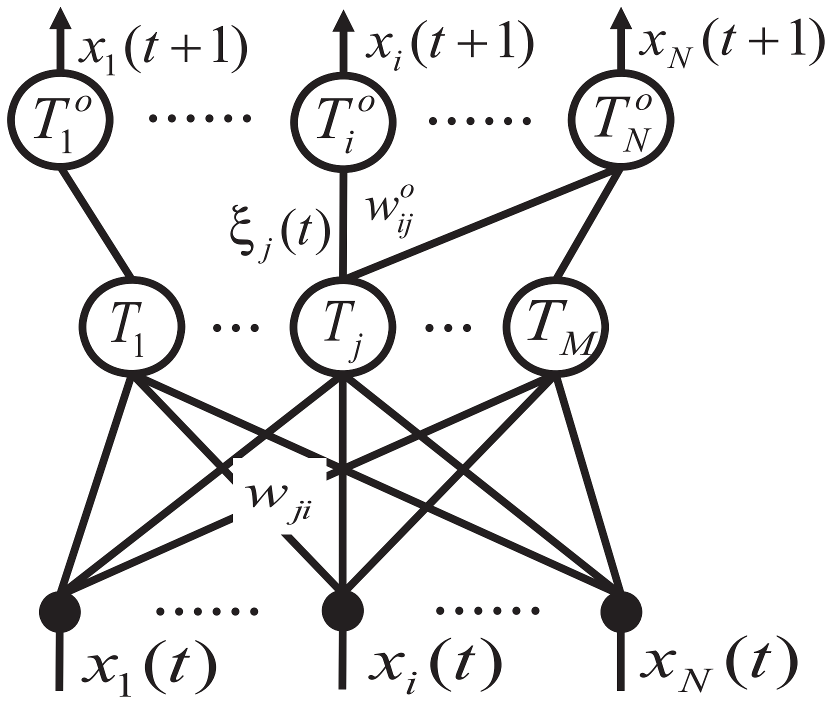

図 3.1: DBNN の構造図

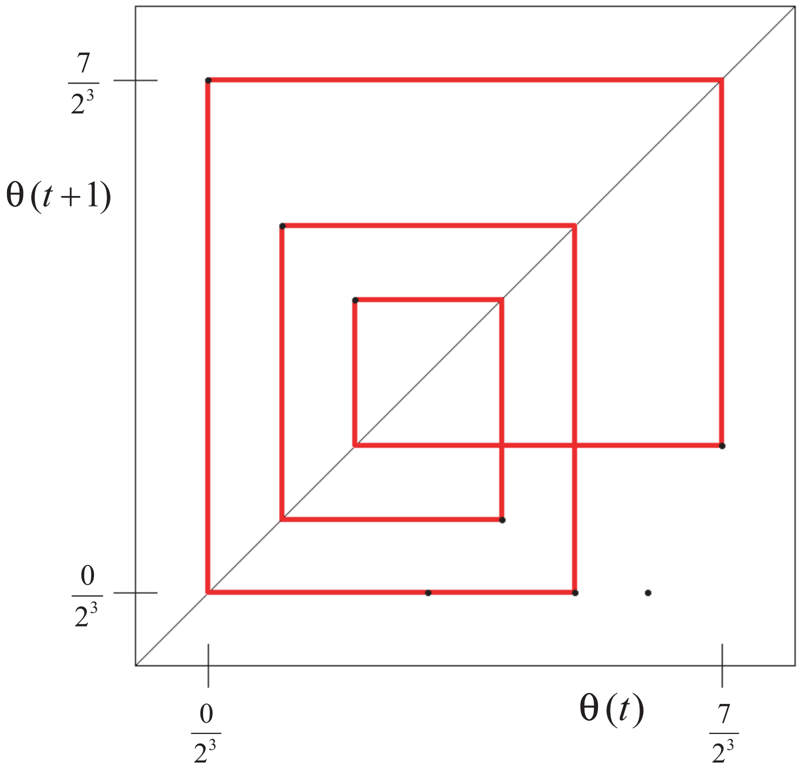

図 3.2: DBNN により得られる Gmap
赤い軌道は教師信号の周期解を示す。

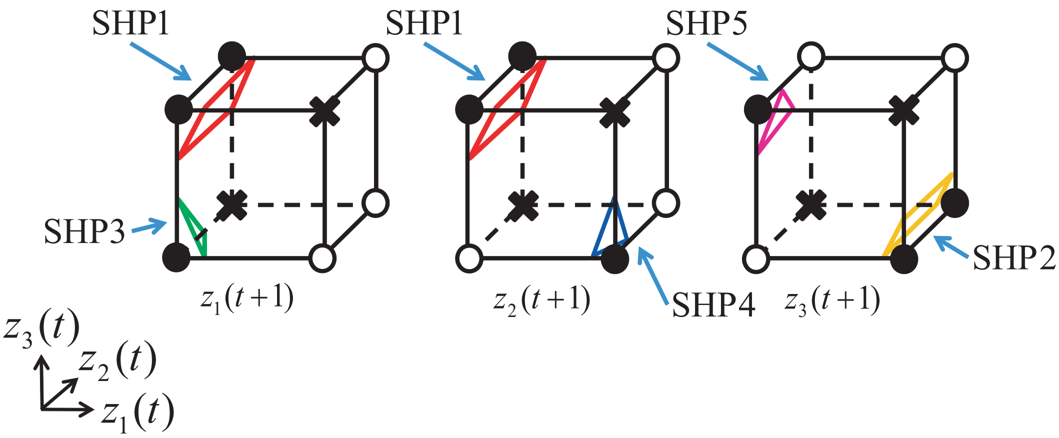

図 3.3: SHP による真頂点 (●) の分離図
図の●、○、×はそれぞれ教師信号の出力 “+1”、“-1”、“don’t care” を意味する。

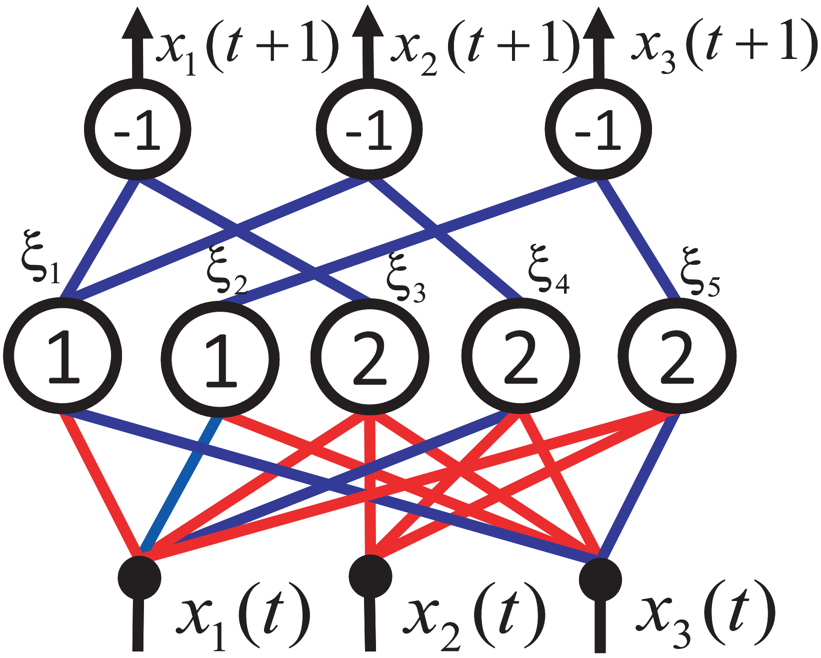

図 3.4: 図 3.3 に対応した 3 次元の DBNN
青い線を $w = +1$、赤い線を $w = -1$、$w = 0$ は接続なしとする。

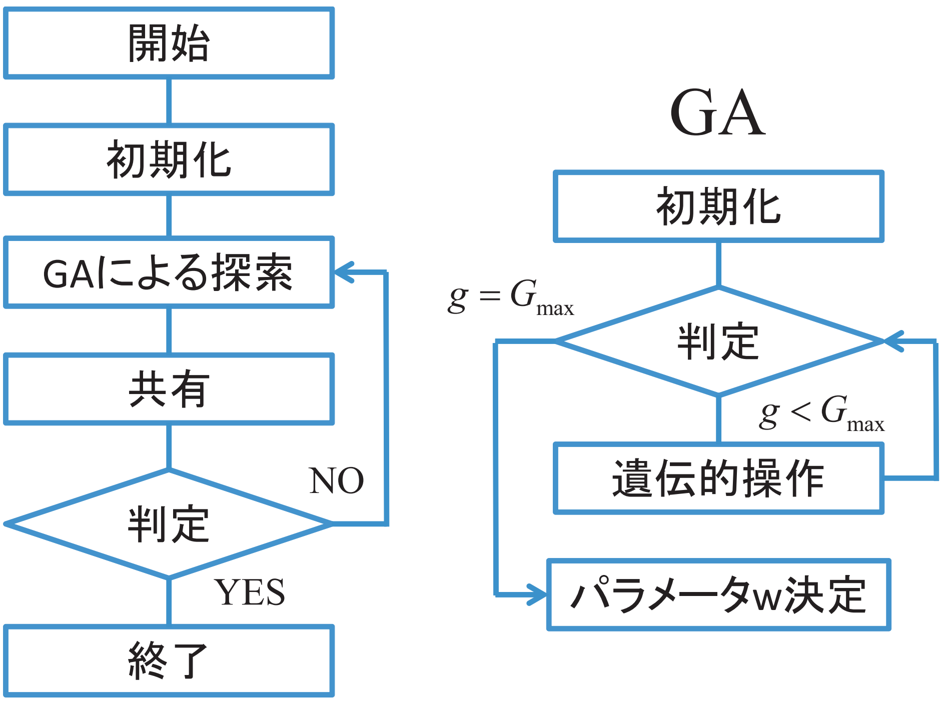

図 3.5: フローチャート (DBNN)

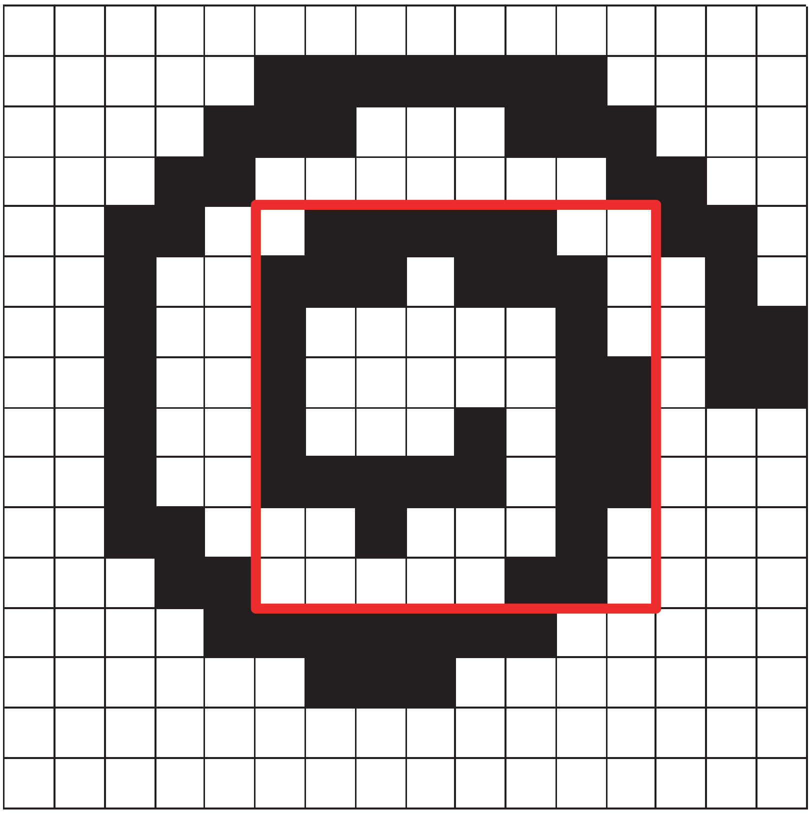

図 3.6: 2 スパイラル問題
赤い枠線内部を教師信号として変換し、学習に用いる。

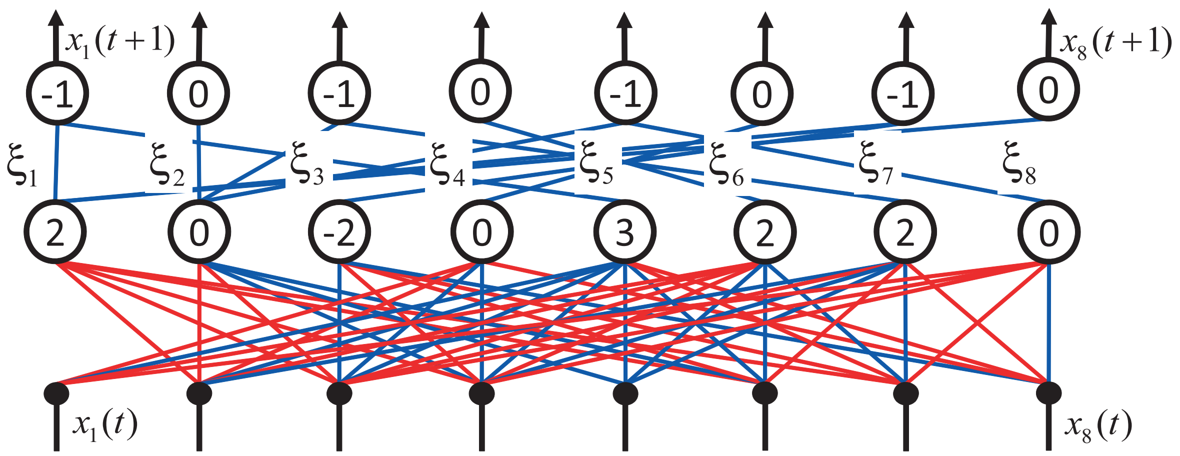

図 3.7: 表 3.2 に示す教師信号を学習後の DBNN

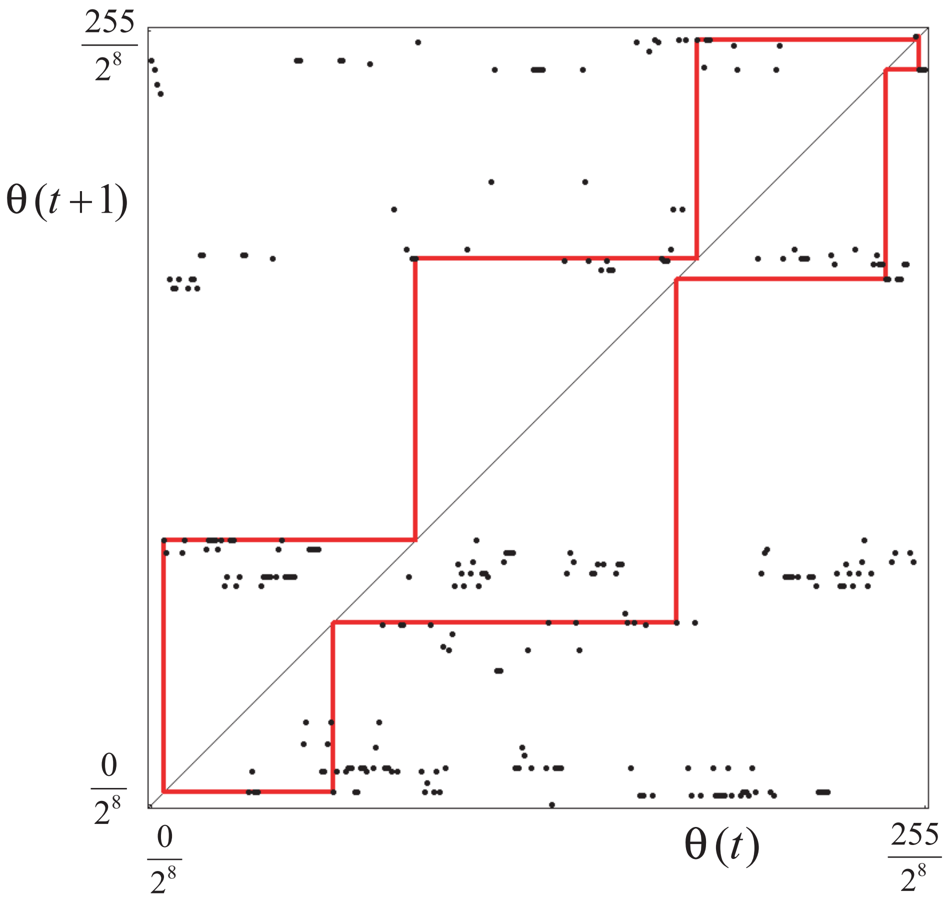

図 3.8: 図 3.7 の DBNN により得られる Gmap

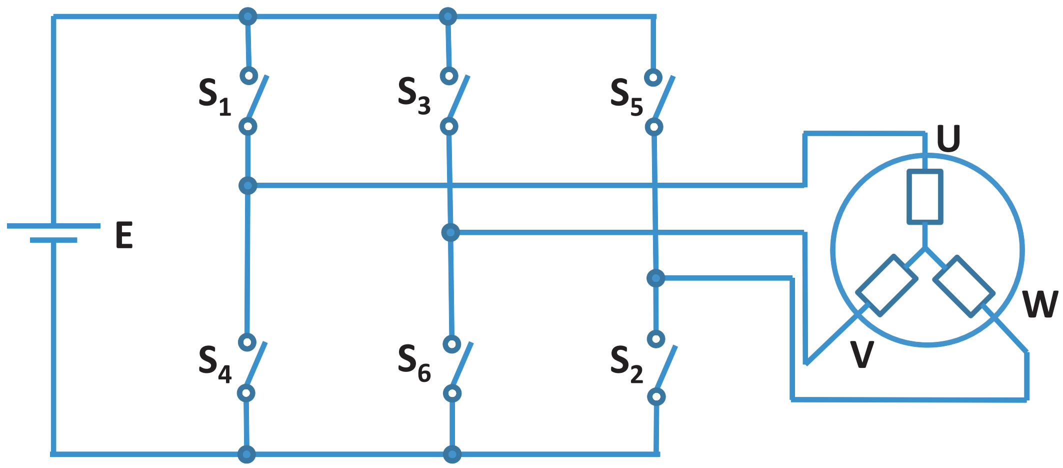

図 3.9: インバータの回路構成

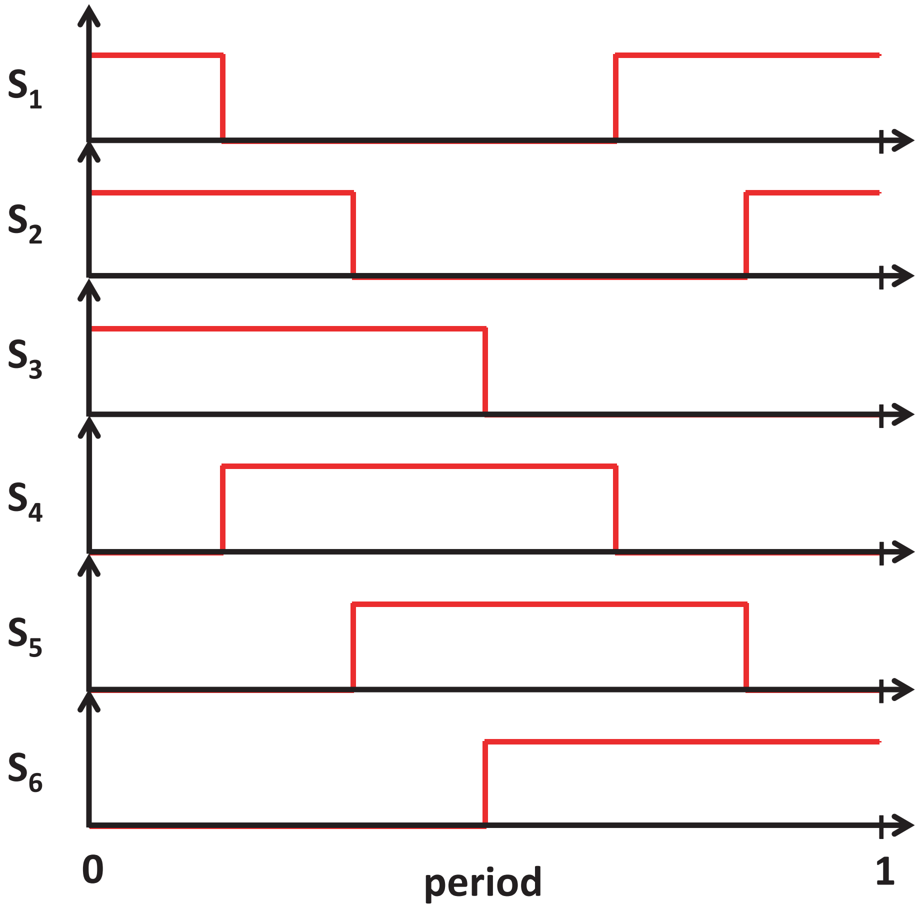

図 3.10: インバータのスイッチングパターン

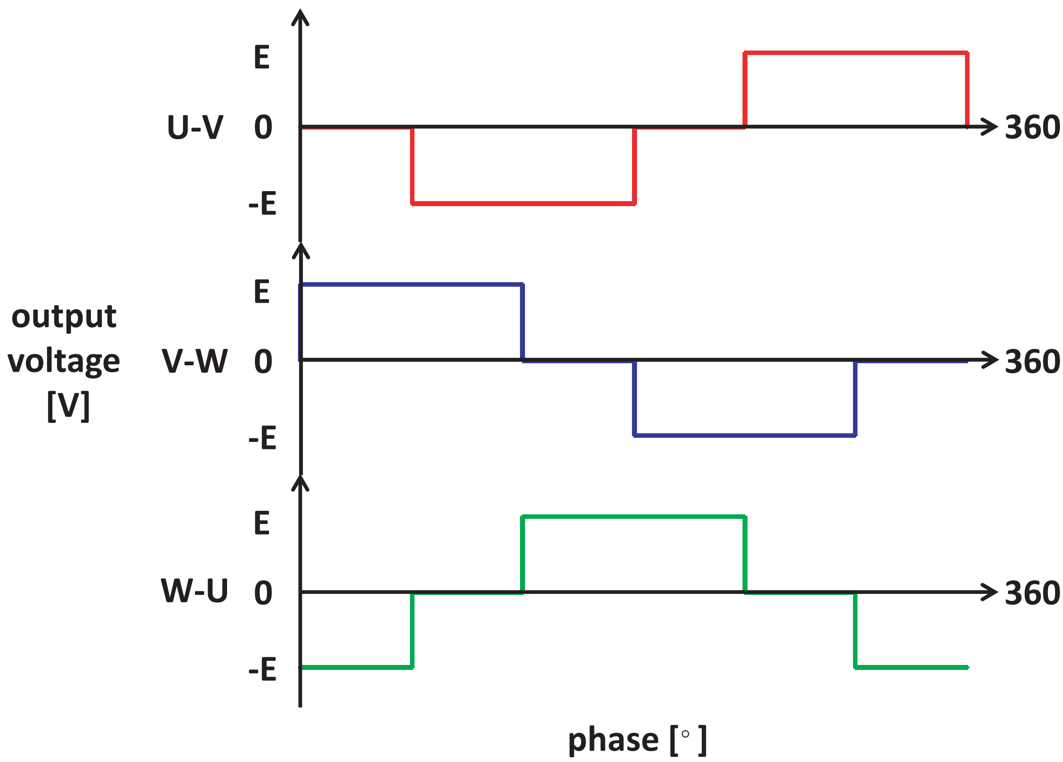

図 3.11: インバータの出力波形

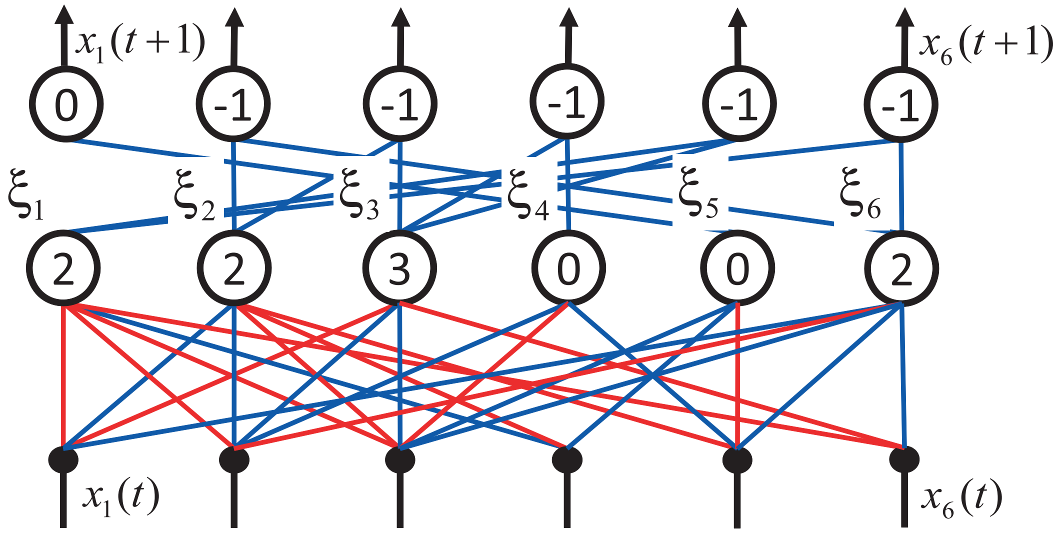

図 3.12: 表 3.4 に示す教師信号を学習後の DBNN

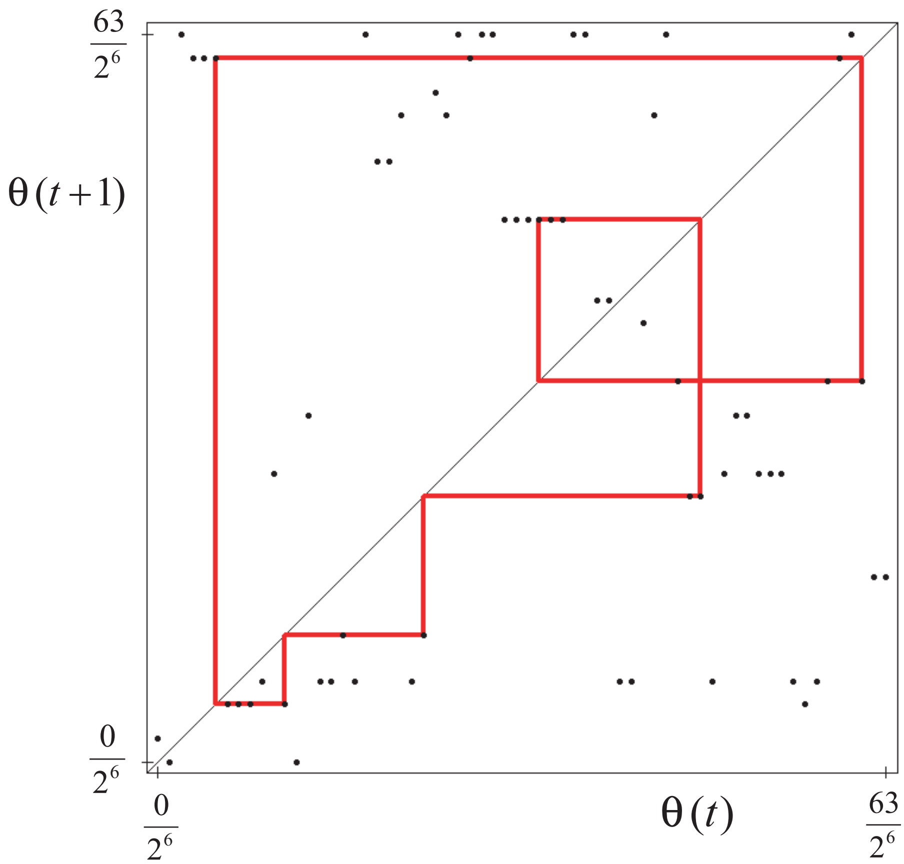

図 3.13: 図 3.12 の DBNN により得られる Gmap

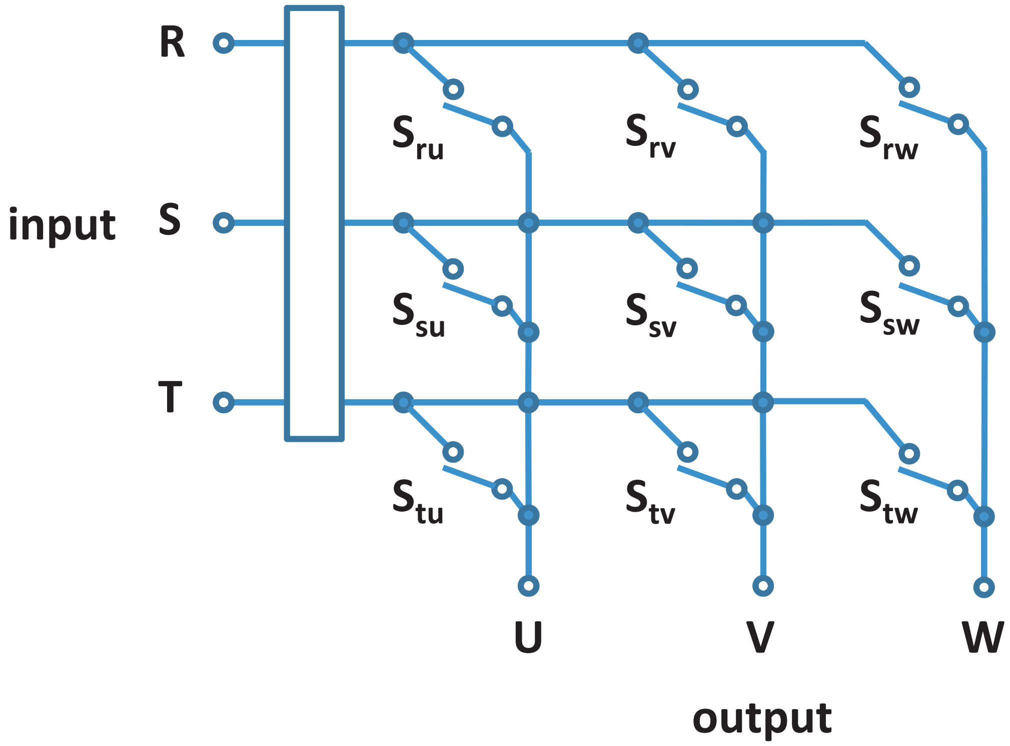

図 3.14: マトリックスコンバータの回路構成

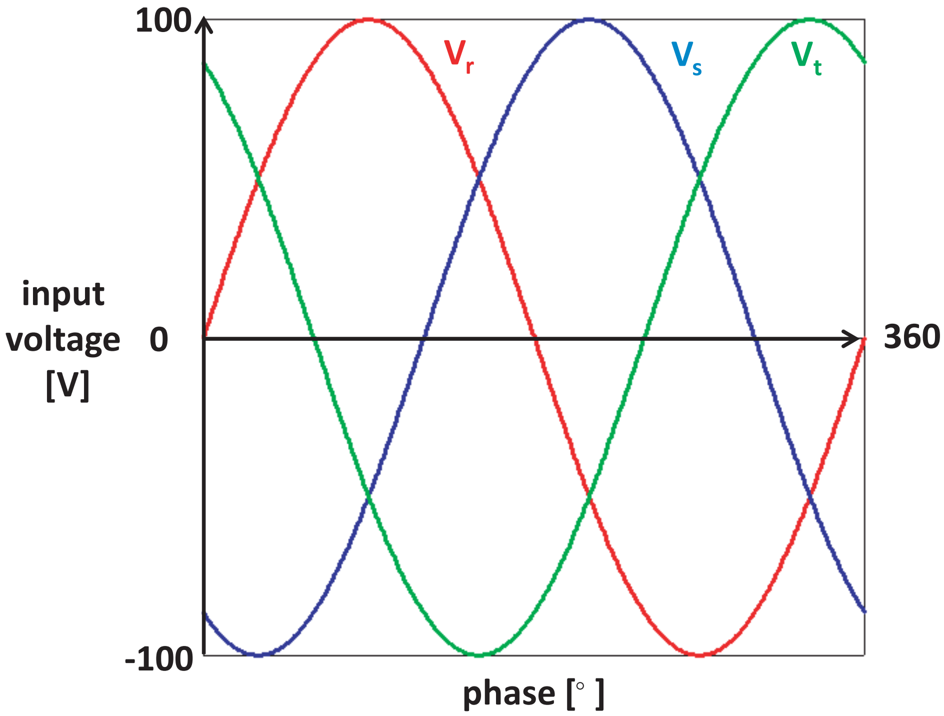

図 3.15: マトリックスコンバータの入力波形

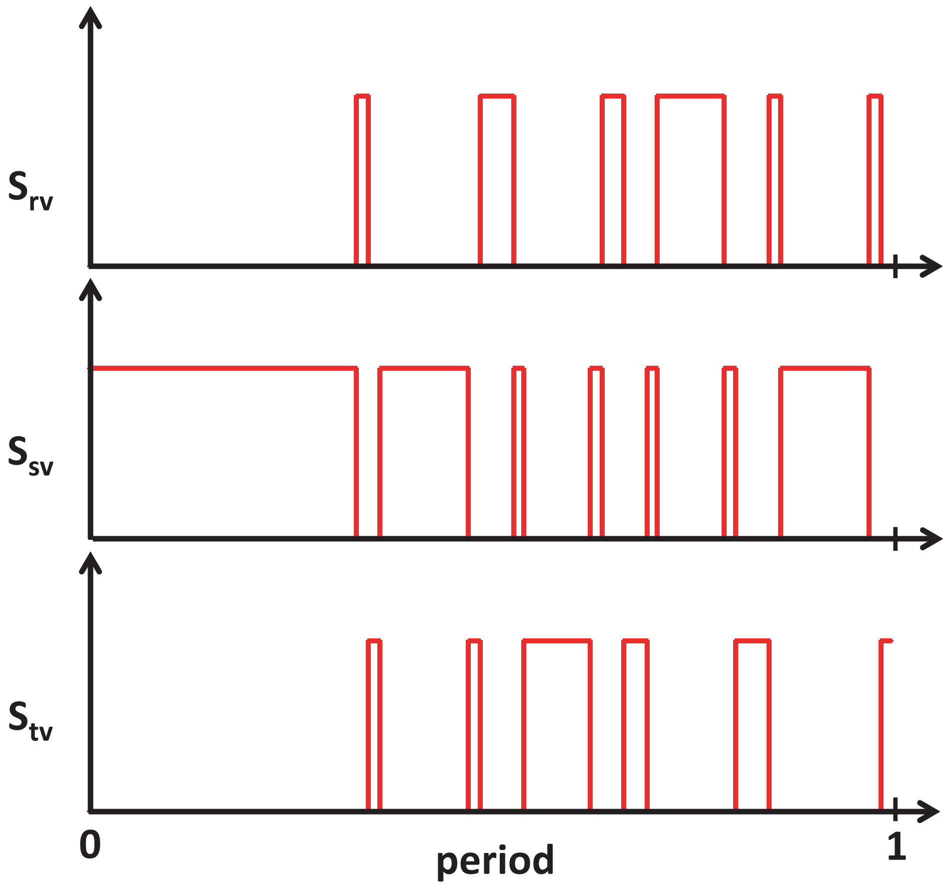

図 3.16: V 層におけるマトリックスコンバータのスイッチングパターン

U-phase

z(10)	0 0 1 0 1 1 0 0
z(11)	0 0 0 0 0 0 0 1
z(12)	0 0 0 0 0 0 1 0
z(13)	0 0 1 0 0 1 0 0
z(14)	0 1 0 0 0 0 1 0
z(15)	0 0 0 1 0 0 0 1
z(16)	0 0 0 0 0 1 0 0
z(17)	0 0 0 1 1 0 1 0
z(18)	0 1 0 0 0 1 0 0
z(19)	0 0 0 0 1 0 0 1
z(1)	0 0 0 0 1 0 1 0
z(2)	1 0 0 0 0 1 0 0
z(3)	0 0 0 1 1 0 0 1
z(4)	1 1 0 0 0 1 0 0
z(5)	0 0 0 1 1 0 0 1
z(6)	0 1 0 0 0 0 0 1
z(7)	0 1 1 0 0 1 0 0
z(8)	1 0 0 0 0 0 0 1
z(9)	1 0 0 0 0 0 1 0

shift to left

V-phase

z(1)	0 0 1 0 1 0 1 0
z(2)	0 0 0 0 0 1 0 0
z(3)	0 0 0 0 0 0 0 1
z(4)	0 0 1 0 0 0 1 0
z(5)	0 1 0 0 0 0 0 1
z(6)	0 0 0 1 0 1 0 0
z(7)	0 0 0 0 0 0 1 0
z(8)	0 0 0 1 1 0 0 1
z(9)	0 1 0 0 0 0 1 0
z(10)	0 0 0 0 1 1 0 0
z(11)	0 0 0 0 1 0 0 1
z(12)	1 0 0 0 0 0 1 0
z(13)	0 0 0 1 1 1 0 0
z(14)	1 1 0 0 0 0 1 0
z(15)	0 0 0 1 1 1 0 0
z(16)	0 1 0 0 0 1 0 0
z(17)	0 1 1 0 0 0 1 0
z(18)	1 0 0 0 0 1 0 0
z(19)	1 0 0 0 0 0 0 1

shift to right

W-phase

z(19)	0 0 1 0 1 0 0 1
z(1)	0 0 0 0 0 0 1 0
z(2)	0 0 0 0 0 1 0 0
z(3)	0 0 1 0 0 0 0 1
z(4)	0 1 0 0 0 1 0 0
z(5)	0 0 0 1 0 0 1 0
z(6)	0 0 0 0 0 0 0 1
z(7)	0 0 0 1 1 1 0 0
z(8)	0 1 0 0 0 0 0 1
z(9)	0 0 0 0 1 0 1 0
z(10)	0 0 0 0 1 1 0 0
z(11)	1 0 0 0 0 0 0 1
z(12)	0 0 0 1 1 0 1 0
z(13)	1 1 0 0 0 0 0 1
z(14)	0 0 0 1 1 0 1 0
z(15)	0 1 0 0 0 0 1 0
z(16)	0 1 1 0 0 0 0 1
z(17)	1 0 0 0 0 0 1 0
z(18)	1 0 0 0 0 1 0 0

図 3.17: 各相スイッチングパターン (教師信号)

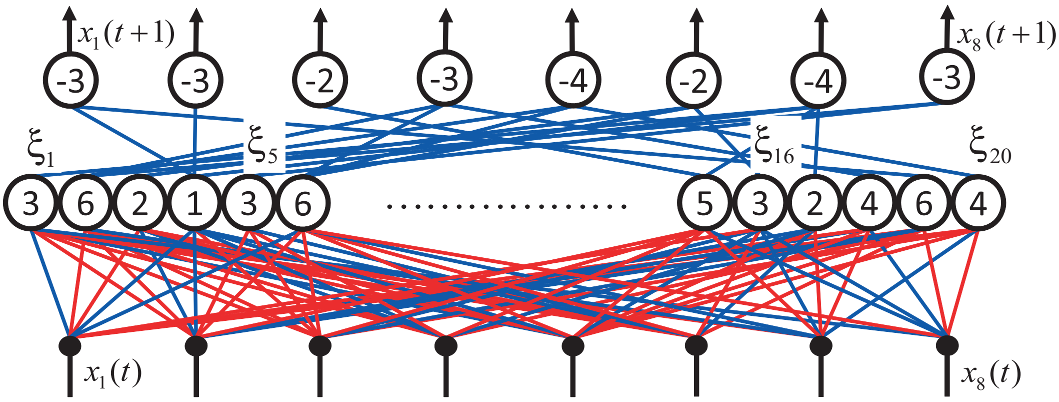

図 3.18: 表 3.6 に示す教師信号を学習後の DBNN

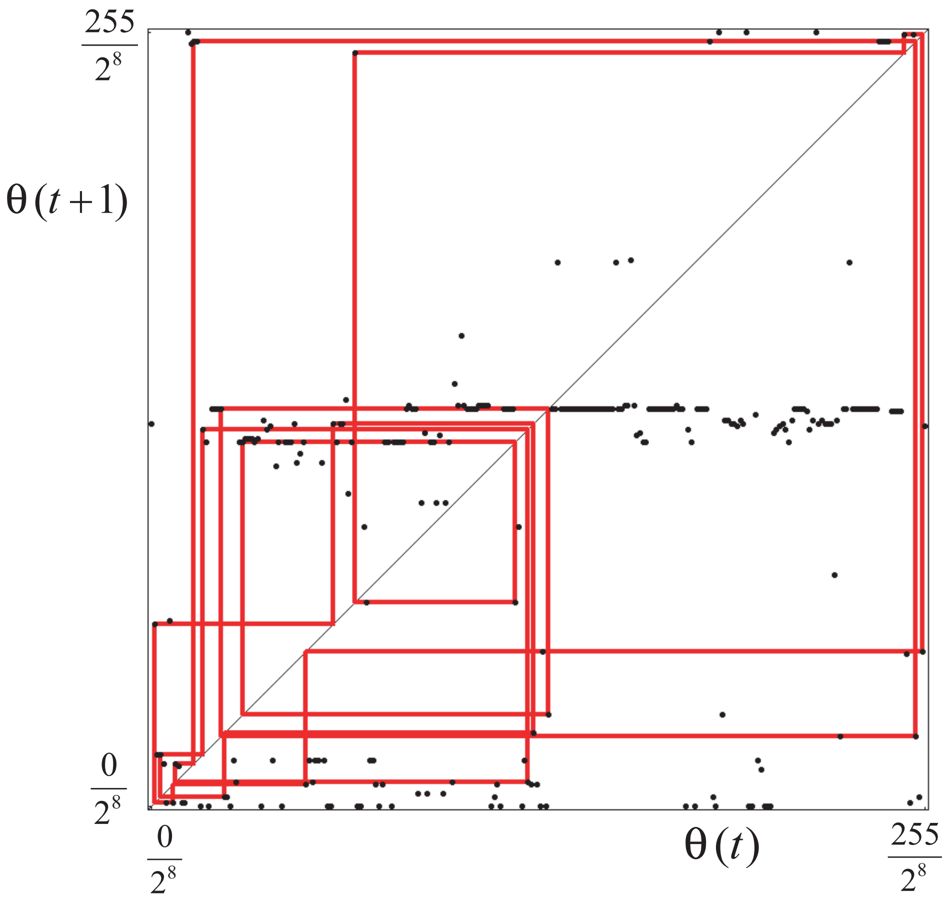

図 3.19: 図 3.18 の DBNN により得られる Gmap

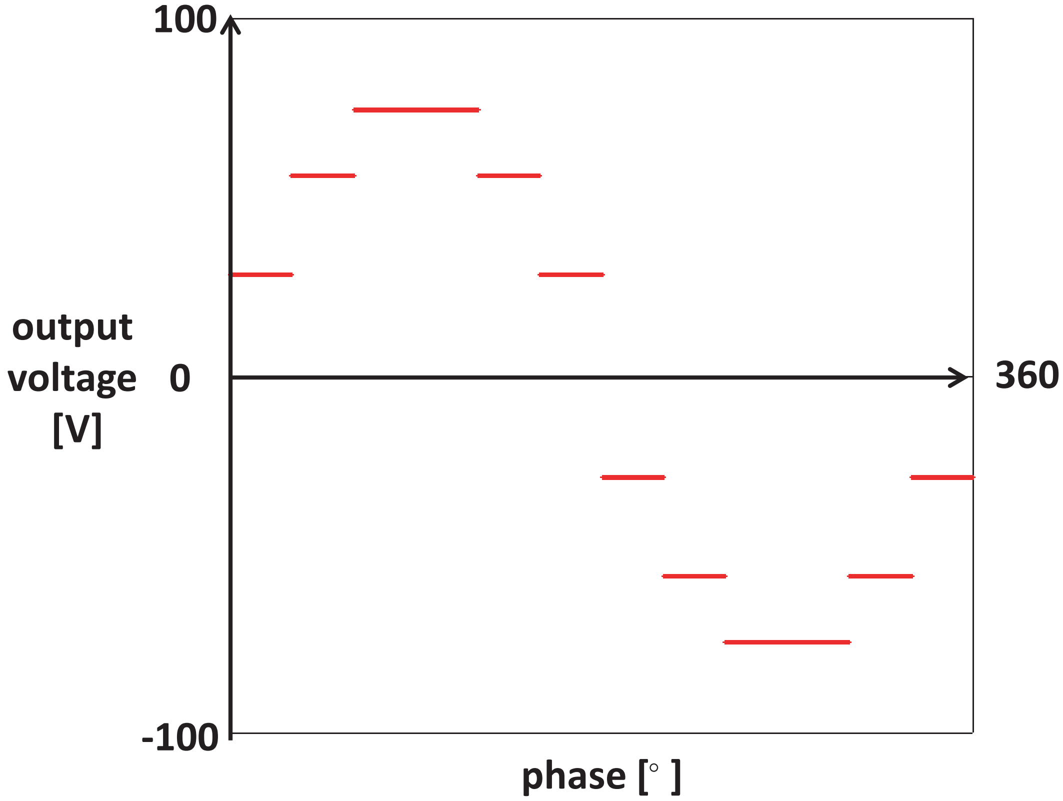

図 3.20: マトリックスコンバータの U 相出力波形

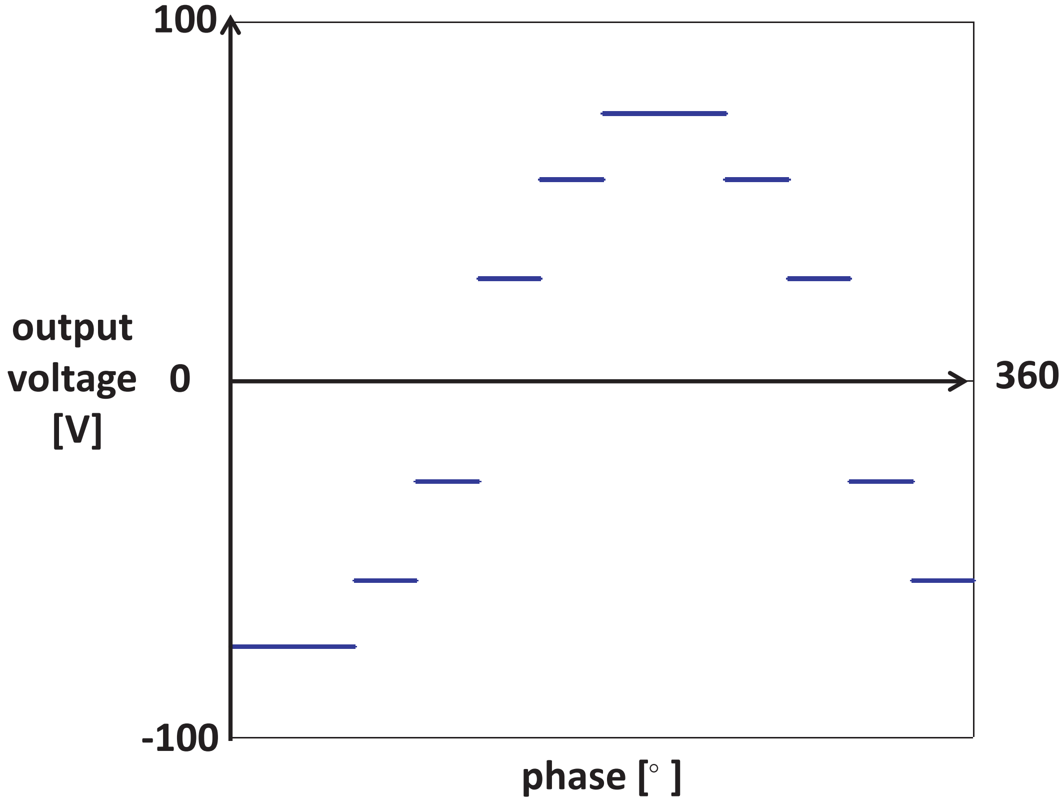

図 3.21: マトリックスコンバータの V 相出力波形

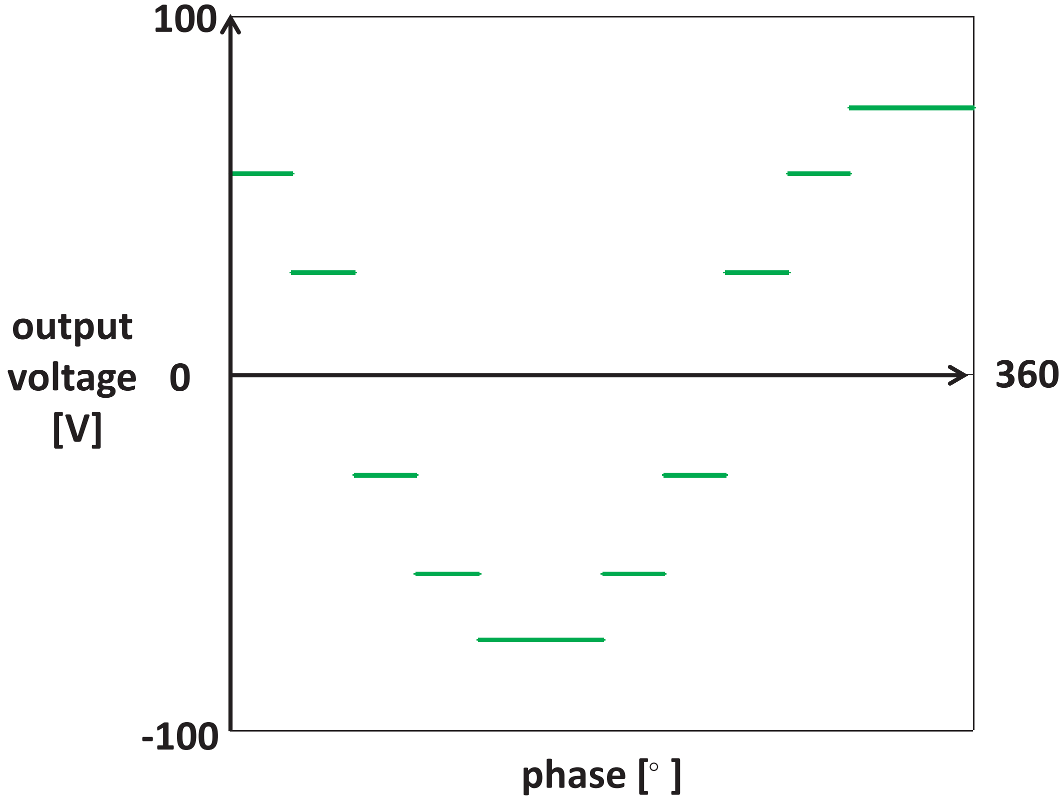

図 3.22: マトリックスコンバータの W 相出力波形

表 3.1: 3 次元教師信号

$\boldsymbol{z}(1)$	$(-1, -1, -1)$
$\boldsymbol{z}(2)$	$(+1, -1, -1)$
$\boldsymbol{z}(3)$	$(-1, +1, +1)$
$\boldsymbol{z}(4)$	$(+1, +1, -1)$
$\boldsymbol{z}(5)$	$(-1, -1, +1)$
$\boldsymbol{z}(6)$	$(+1, +1, +1)$
$\boldsymbol{z}(7) = \boldsymbol{z}(1)$	$(-1, -1, -1)$

表 3.2: 2 スパイラルの一部を用いた教師信号

$\boldsymbol{z}(1)$	$(-1, +1, +1, +1, +1, +1, -1, -1)$
$\boldsymbol{z}(2)$	$(+1, +1, +1, -1, +1, +1, +1, -1)$
$\boldsymbol{z}(3)$	$(+1, -1, -1, -1, -1, -1, +1, -1)$
$\boldsymbol{z}(4)$	$(+1, -1, -1, -1, -1, -1, +1, +1)$
$\boldsymbol{z}(5)$	$(+1, -1, -1, -1, +1, -1, +1, +1)$
$\boldsymbol{z}(6)$	$(+1, +1, +1, +1, +1, -1, +1, +1)$
$\boldsymbol{z}(7)$	$(-1, -1, +1, -1, -1, -1, +1, -1)$
$\boldsymbol{z}(8)$	$(-1, -1, -1, -1, -1, +1, +1, -1)$
$\boldsymbol{z}(9) = \boldsymbol{z}(1)$	$(-1, +1, +1, +1, +1, +1, -1, -1)$

表 3.3: 表 3.2 の教師信号について学習後のパラメータ

j	w_{j1}	w_{j2}	w_{j3}	w_{j4}	w_{j5}	w_{j6}	w_{j7}	w_{j8}	β_j
1	0	-1	-1	-1	0	-1	-1	0	3
2	0	-1	-1	+1	+1	+1	0	0	5
3	0	0	+1	-1	0	-1	-1	+1	7
4	-1	-1	+1	+1	0	0	0	-1	5
5	+1	+1	+1	+1	+1	+1	-1	-1	5
6	-1	-1	-1	-1	+1	+1	+1	0	5
7	0	+1	-1	+1	+1	-1	+1	-1	5
8	-1	0	-1	-1	0	0	-1	+1	5

表 3.4: インバータのスイッチ信号に対応した教師信号

$\mathbf{z}(1)$	$(+1, +1, +1, -1, -1, -1)$
$\mathbf{z}(2)$	$(-1, +1, +1, +1, -1, -1)$
$\mathbf{z}(3)$	$(-1, -1, +1, +1, +1, -1)$
$\mathbf{z}(4)$	$(-1, -1, -1, +1, +1, +1)$
$\mathbf{z}(5)$	$(+1, -1, -1, -1, +1, +1)$
$\mathbf{z}(6)$	$(+1, +1, -1, -1, -1, +1)$
$\mathbf{z}(7) = \mathbf{z}(1)$	$(+1, +1, +1, -1, -1, -1)$

表 3.5: 表 3.4 の教師信号について学習後のパラメータ

j	w_{j1}	w_{j2}	w_{j3}	w_{j4}	w_{j5}	w_{j6}	β_j
1	-1	-1	-1	+1	0	-1	3
2	+1	+1	-1	-1	-1	0	3
3	-1	+1	+1	0	0	-1	1
4	0	+1	-1	0	+1	0	3
5	0	0	+1	+1	-1	0	3
6	+1	-1	+1	0	+1	+1	3

表 3.6: マトリックスコンバータのスイッチ信号に対応した教師信号

$\boldsymbol{z}(1)$	$(-1, -1, +1, -1, +1, -1, +1, -1)$
$\boldsymbol{z}(2)$	$(-1, -1, -1, -1, -1, +1, -1, -1)$
$\boldsymbol{z}(3)$	$(-1, -1, -1, -1, -1, -1, -1, +1)$
$\boldsymbol{z}(4)$	$(-1, -1, +1, -1, -1, -1, +1, -1)$
$\boldsymbol{z}(5)$	$(-1, +1, -1, -1, -1, -1, -1, +1)$
$\boldsymbol{z}(6)$	$(-1, -1, -1, +1, -1, +1, -1, -1)$
$\boldsymbol{z}(7)$	$(-1, -1, -1, -1, -1, -1, +1, -1)$
$\boldsymbol{z}(8)$	$(-1, -1, -1, +1, +1, -1, -1, +1)$
$\boldsymbol{z}(9)$	$(-1, +1, -1, -1, -1, -1, +1, -1)$
$\boldsymbol{z}(10)$	$(-1, -1, -1, -1, +1, +1, -1, -1)$
$\boldsymbol{z}(11)$	$(-1, -1, -1, -1, +1, -1, -1, +1)$
$\boldsymbol{z}(12)$	$(+1, -1, -1, -1, -1, -1, +1, -1)$
$\boldsymbol{z}(13)$	$(-1, -1, -1, +1, +1, +1, -1, -1)$
$\boldsymbol{z}(14)$	$(+1, +1, -1, -1, -1, -1, +1, -1)$
$\boldsymbol{z}(15)$	$(-1, -1, -1, +1, -1, -1, -1, +1)$
$\boldsymbol{z}(16)$	$(-1, +1, -1, -1, -1, +1, -1, -1)$
$\boldsymbol{z}(17)$	$(-1, +1, +1, -1, -1, -1, +1, -1)$
$\boldsymbol{z}(18)$	$(+1, -1, -1, -1, -1, +1, -1, -1)$
$\boldsymbol{z}(19)$	$(+1, -1, -1, -1, -1, -1, -1, +1)$
$\boldsymbol{z}(20) = \boldsymbol{z}(1)$	$(-1, -1, +1, -1, +1, -1, +1, -1)$

表 3.7: 表 3.6 の教師信号について学習後のパラメータ

j	w_{j1}	w_{j2}	w_{j3}	w_{j4}	w_{j5}	w_{j6}	w_{j7}	w_{j8}	β_j
1	+1	-1	-1	-1	+1	-1	-1	+1	5
2	-1	-1	-1	-1	-1	-1	+1	0	1
3	-1	+1	0	+1	-1	-1	0	0	3
4	+1	+1	-1	+1	+1	+1	0	0	5
5	0	-1	-1	-1	0	+1	0	0	1
6	+1	-1	-1	0	-1	-1	+1	-1	1
7	0	0	+1	+1	0	-1	0	+1	3
8	+1	0	-1	0	-1	0	-1	+1	1
9	+1	0	0	0	-1	+1	0	-1	1
10	-1	+1	0	-1	-1	+1	-1	0	1
11	-1	-1	+1	-1	+1	+1	-1	0	3
12	0	-1	+1	0	-1	-1	-1	-1	3
13	0	-1	-1	-1	+1	-1	0	0	1
14	+1	+1	0	-1	-1	-1	+1	-1	1
15	0	-1	-1	-1	-1	0	-1	+1	1
16	-1	0	+1	0	+1	-1	+1	+1	3
17	0	+1	0	+1	-1	+1	-1	0	3
18	-1	+1	0	0	-1	0	-1	+1	1
19	-1	+1	+1	-1	-1	-1	0	-1	1
20	-1	0	-1	0	-1	0	+1	-1	1

第4章　むすび

本論文では、ニューラルネットワークの入出力を2値としたBNNやDBNNについて研究を行った。学習アルゴリズムにはそれぞれが抱える問題に対処するため、ヒューリスティックなアルゴリズムであるGAに基づいたアルゴリズムを提案した。その学習アルゴリズムの有効性を検証するために、それぞれ数値実験を行うことによってその問題である中間層ニューロン数や計算コストの増加を抑制できることを確認した。また、数値実験では代表的である手法との比較を行った。さらに、実際に用いられている回路への適用も考察した。

第2章では、BNNと論理合成法を対応させ、それぞれの特性について確認した。BNNのパラメータであるβの値により、得られるBNNが異なることを示した。パラメータ$\beta = 1$であるとき、DCFと等価となる。また、$\beta \geq 1$であるとき、その得られるBNNはQMCによる論理合成法を行ったDCFより簡単な構造を実現できることを明らかにした。このBNNは比較的高次元のブール関数にも対応し、近似することができる。また、ETRの導入により曖昧さを伴った学習を行うことができた。

第3章では、いくつかのBPOをもつ教師信号をDBNN学習アルゴリズムで学習し、考察を行った。この章では、主に実際に用いられている回路のスイッチ信号に対応する教師信号を用いて学習を行い、Gmapを用いてDBNNのダイナミクスを確認した。その結果、DBNNを用いて教師信号BPOを銘記することができた。また、その教師信号以外の初期値から出発する信号は全て教師信号の周期解に収束することが確認できた。さらに、学習では教師信号の周期解以外の情報は一切用いられていない。したがって、自動安定化できることを示した。この実験により、DBNNの汎化能力を確認できた。このDBNNのシステムを実際の回路に用いてスイッチ制御を行えば、回路のロバスト化に貢献できる。

今後の課題として、BNN や DBNN の学習時に用いるパラメータの改善や学習過程の解析、そのダイナミクスや Gmap の解析などが挙げられる。また、ハード化も考えていく。

参考文献

[1] D. L. Gray and A. N. Michel, “ A training algorithm for binary feed forward neural networks,” IEEE Trans. Neural Networks, 3, 2, pp. 176–194, 1992.

[2] J. H. Kim and S. K. Park, “ The geometrical learning of binary neural networks,” IEEE Trans. Neural Networks, 6, 1, pp. 237–247, 1995.

[3] A. Yamamoto and T. Saito, “ A flexible learning algorithm for binary neural networks,” IEICE Trans. Fundamentals, E81-A, 9, pp. 1925-1930, 1998.

[4] F. Chen, G. Chen, Q. He, G. He, and X. Xu, “ Universal perceptron and DNA-like learning algorithm for binary neural networks: non-LSBF implementation,” IEEE Trans. Neural Networks, 20, 8, pp. 1293-1301, 2009

[5] F. Chen, G. Chen, Q. He, G. He, and X. Xu, “ Universal perceptron and DNA-like learning algorithm for binary neural networks: LSBF and PBF implementations,” IEEE Trans. Neural Networks, 20, 10, pp. 1645-1658, 2009

[6] M. Muselli and D. Liberati, “ Training Digital Circuits with Hamming Clustering,” IEEE Trans. Circuits Syst. I, 47, 4, pp. 513-527, 2000

[7] S. Kabeya and T. Saito, “ A GA-based flexible learning algorithm with error tolerance for digital binary neural networks,” in Proc. IEEE-INNS Joint Conf. Neural Netw., pp. 1476-1480, 2009.

[8] R. Ito, Y. Nakayama and T. Saito, “ Learning of dynamic BNN toward storing-and-stabilizing periodic patterns,” in Proc. ICONIP, pp. 606-611, 2011.

[9] C. Zhang, J. Yang and W. Wu, “ Binary higher order neural networks for realizing boolean functions,” IEEE Trans. Neural Networks, 22, 5, pp. 701-713, 2011

[10] R. Ito, Y. Nakayama and T. Saito, “ Analysis and Learning of Periodic Orbits in Dynamic Binary Neural Networks,” Proc. IEEE-INNS/IJCNN, pp. 502-508, 2012.

[11] R. Ito and T. Saito, “ Dynamic binary neural networks and evolutionary learning,” in Proc. IEEE-INNS Joint Conf. Neural Netw., pp. 1683-1687, 2010.

[12] K-J. Kim1 and S-B. Cho, “ Evaluation of Distance Measures for Speciated Evolutionary Neural Networks in Pattern Classification Problems,” (C. S. Leung, M. Lee, and J. H. Chan (Eds.): ICONIP2009), Lecture Notes in Computational Sience, 5864, pp. 630 637, 2009.

[13] A. Rakitianskaia and A. P. Engelbrecht, “ Training Neural Networks with PSO in Dynamic Environments,” in Proc. IEEE Congress Evol. Comput., pp. 667-673, 2009.

[14] N. L. Mineu, T. B. Ludermir and L. M. Almeida, “ Topology Optimization for Artificial Neural Networks using Differential Evolution,” in Proc. IEEE-INNS Joint Conf. Neural Netw., pp. 4062-4068, 2010

[15] A. P. Engelbrecht, “ Fundamentals of computational swarm intelligence,” Willey, 2005.

[16] Y. Nakayama, R. Ito and T. Saito, “ A simple class of binary neural networks and logical synthesis,” IEICE Trans. Fundamentals, E94-A, 9, pp. 1586-1589, 2011.

[17] “ はじめてのインバータ,” https://www.mitsubishielectric.co.jp/fa/ssl/wap/eln/index.do?SAStruts.method=startStudy

[18] P. D. Ziogas, “ Analysis and Design of Forced Commutated Cycloconverter Structures with Improved Transfer Characteristics,” IEEE Trans. PESC, pp. 271-280, 1986

[19] J. Oyama, T. Higuchi, E. Yamada, T. Koga and T. Lipo, “ New control strategy for matrix converter,” in Conf. Rec. IEEE Trans. PESC, pp. 360-367, 1989

[20] “ マトリックス・コンバータの基礎から応用まで,” http://techon.nikkeibp.co.jp/article/LECTURE/20100929/185966/

研究業績

（論文）

Y. Nakayama, R. Ito and T. Saito, “ A Simple Class of Binary Neural Networks and Logical Synthesis,” IEICE Trans. Fundamentals, E94-A, 9, pp. 1586-1589, 2011

（国際会議）

R. Ito, Y. Nakayama and T. Saito, “ Analysis and Learning of Periodic Orbits in Dynamic Binary Neural Networks,” Proc. IEEE-INNS/IJCNN, pp 502-508, 2012

R. Ito, Y. Nakayama and T. Saito, “ Learning of Dynamic BNN toward Storing-and-Stabilizing Periodic Patterns,” (ICONIP 2011), Lecture Notes in Computational Science, 7063, pp. 606-611, Springer-Verlag

Y. Nakayama, R. Ito and T. Saito, “ Evolutionary Learning of Binary Neural Networks and Its Application to Logical Synthesis,” Proc. International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC), pp. 1036-1039

（国内発表）

中山雄太, 斎藤利通, “ 動的バイナリーニューラルネットワークの応用：マトリックスコンバータのスイッチ信号の学習,” 電子情報通信学会技術研究報告 (NC 研究会), vol. 112, No. 345, pp. 13-18, 豊橋, 2012 年 12 月

上月良太, 中山雄太, 斎藤利通, “ 動的バイナリーニューラルネットの学習と応用,” 電子情報通信学会技術研究報告 (NC 研究会), vol. 112, No. 227, pp. 85-89, 福岡, 2012 年 10 月

中山雄太, 斎藤利通, “ 遺伝的アルゴリズムに基づく動的バイナリーネットの学習,” 第 11 回情報科学技術フォーラム (FIT 2012), pp. 349-350, 法政大学, 東京, 2012 年 9 月

中山雄太, 伊藤良, 斎藤利通, “ 動的バイナリーニューラルネットの学習過程の解析,” 電

子情報通信学会技術研究報告 (NC 研究会), vol. 111, No. 483, pp. 159-164, 玉川大学, 東京, 2012 年 3 月

斎藤利通, 伊藤良, 中山雄太, “ 動的バイナリーニューラルネットとその学習について,” 電子情報通信学会総合大会, AK-1-1, 岡山, 2012 年 3 月

Y. Nakayama, R. Ito, and T. Saito, “ Sequential Learning of Binary Neural Networks based on Genetic Algorihm,” Proc. of IEEE Workshop on Nonlinear Circuit Networks, pp. 48-50, 2011

岡本祐輔, 中山雄太, 江野澤瑶子, 斎藤利通, “ ART-Map に基づく論理合成,” 電子情報通信学会技術研究報告 (NLP 研究会), vol. 111, No. 243, NLP 2011-89, pp. 169-173, 静岡, 2011 年 10 月

中山雄太, 伊藤良, 斎藤利通, “ 論理合成のためのバイナリーニューラルネットの学習について,” 電子情報通信学会技術研究報告 (NLP 研究会), vol. 111, No. 62, NLP 2011-11, pp. 49-53, 小豆島, 香川, 2011 年 5 月

中山雄太, 伊藤良, 斎藤利通, “ バイナリーニューラルネットと論理合成,” 電子情報通信学会総合大会, A-2-8, 東京都市大学, 東京, 2011 年 3 月

中山雄太, 伊藤良, 斎藤利通, “ バイナリーニューラルネットによる論理合成へのアプローチ,” 電子情報通信学会技術研究報告 (NLP 研究会), vol. 110, No. 299, NLP 2010-108, pp. 43-47, 東北大学, 宮城, 2010 年 11 月

謝辞

本論文は著者が法政大学大学院工学研究科電気工学専攻に在学中の 2 年間、同大学理工学部電気電子工学科斎藤利通教授の指導下で行ったものである。研究活動を遂行するにあたり，同教授から終始懇切に御指導，御鞭撻下を賜りました。心から深謝致します。

最後に法政大学理工学部電気電子工学科斎藤利通研究室の皆様にはいろいろな有益な御討論・ご助言を戴きました。ここに深謝致します。