ニューラルネットワークの分散学習

塚本義明† 生天目章†

ニューラルネットワークの規模の問題を解決するための方法として分散学習が考えられる．これは，大規模な問題がすでにいくつかの小さな問題に分割されており，その小さな問題について学習した結果を統合して大規模な問題を解決する試みである．また，この際に全体を掌握する管理者的存在はないものとする．本論文では，多層ネットワークの学習アルゴリズムである瞬時学習法および記憶に基づく学習を統合した構造化学習法に基づく分散学習法を提案する．大規模な問題が，すでにいくつかの小さな問題に分割されており，それらを並列処理するには並列処理された結果を統合するためのメカニズムが必要である．分散アルゴリズムが加法性を有すれば，分散処理された情報を統合するためのメカニズムを容易に実現できる．本論文で提案する分散学習法は，一つの学習問題がいくつかに分割された状態で，それぞれ分割された学習問題に構造化学習法を並列的に適用し，それぞれの学習結果を統合することにより全体の問題を学習させる方法である．並列処理により獲得した分散ネットワークモジュールの線形結合として全体の問題のニューラルネットワークが求まることを示す．オブジェクト指向プログラミングにより分散学習シミュレーションツールを開発し，その適用例について示す．

Distributed Learning for a Neural Network Model

Yoshiaki Tsukamoto† and Akira Namatame†

The concept of modularization and coupling neural network modules is a promising way of a building large-scale neural network model and improving the learning performance of these networks. On the other hand, the modularization scheme would be little use if there does not exist such an appropriate learning procedure to train high-level modules separately and to integrate those functionally pre-specified modules efficiently. This paper describes a way of realizing such a learning procedure that supports the modularization and coupling connectionist network modules. We present a distributed learning procedure for neural networks composed of many separate modular networks, each of which is trained to handle a subset of the complete set of training examples. A central idea of distributed learning is the conceptual decomposition of a complex problem into many separate subproblems represented by neural network modules separately. The whole network is constructed from several network modules. This paper describes a proper way of coupling decomposed neural networks modules and shows that, with the proper decomposition, the whole network is composed as the weighted summation of the decomposed network modules. We also develop an object-oriented distributed learning simulation tool.

1. はじめに

多層ネットワークモデルは，任意の入出力関係について学習・表現することができ，コネクショニストモデルの中で最も活発に研究されている分野の一つである[16),17)]．しかしながら，従来の学習アルゴリズムは，入力の特徴数や学習例が数多い場合や大規模な問題には非効率的であり[8)]，大規模な問題に対して有効な学習アルゴリズムを確立することが大きな研究課題とされている．ニューラルネットワークの規模の問題を解決するための方法として分散学習が考えられる．これは，大規模な問題がいくつかの小さな問題に必然的に分割されており，それらの小さな問題について学習をした結果を統合して大規模な問題を解決する試みである[10),18)]．大規模な問題がいくつかの小さな問題に物理的，地理的または意図的にすでに分散されているのを並列処理するには，各問題ごとに並列処理された情報を有効に統合するためのメカニズムが必要である．この際に，大規模な問題全体を掌握する管理者的存在はないものとする．本論文では，多層ネットワークの学習アルゴリズムである瞬時学習法と記憶に基づく学習法を統合した構造化学習に基づく分散学習法について提案する．分散学習法において学習例の集合の分割

† 防衛大学校情報工学教室
Department of Computer Science, National Defense Academy

のしかたに応じて二つの形態に分類し，それぞれの形態について分散学習の基本定理を示す．構造化学習法をベースにした分散学習法を用いることにより，情報統合上の加法性を有した学習例をそれぞれ個々のネットワークにおいて並列的に独立して獲得した学習結果を線形総和することにより全体の解が求まることを示す．

人工知能の研究において，個別に独立に実行可能な計算メカニズムを備えた処理要素をエージェントと呼び，エージェントの協調により知識処理システムを実現するための研究が盛んに行われている[1),4),5),12),17)]．自律的な学習機能を有するニューラルネットワークモジュールをニューロ・エージェントと定義し，ニューロ・エージェントモデルに基づく分散学習のシミュレーション環境を構築する．本シミュレーションツールは，オブジェクト指向プログラミングに基づき開発されている．分散された学習問題を一つのニューロ・エージェントに処理させ，全体の学習問題は複数のニューロ・エージェントの統合により処理を行う．

2. 構造化学習法

2.1 瞬時学習アルゴリズム

n 次元の属性ベクトル $x_t=(x_{t1}, x_{t2}, \cdots, x_{tn})$, $t=1, 2, \cdots, T$ の集合 $\boldsymbol{X}$ で学習例の集合を表す．すなわち，$\boldsymbol{X}$ は $n\times T$ 行列

$$\boldsymbol{X}=\begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_T \end{Bmatrix} \tag{2.1}$$

である．また $x_{ti}, i=1, 2, \cdots, n$ は，2値のブール変数を表す．学習例の集合 $\boldsymbol{X}$ 上に定義される概念 $\boldsymbol{C}$ とは学習例の数 T に対応した要素 $c_t\in\{0, 1\}, t=1, 2, \cdots, T$ をもつ列ベクトルである．概念 $\boldsymbol{C}$ により分類された学習例 $\boldsymbol{X}$ の正および負の学習例をそれぞれ $\boldsymbol{C}^+$, $\boldsymbol{C}^-$ で表し，学習例の集合 $\boldsymbol{X}$ に対して次のような学習例のグループ類似度を表す関数を定義する．

$$\begin{aligned} G^+(x_t)&=\sum_{x_s\in C^+} S(x_t, x_s) \\ G^-(x_t)&=\sum_{x_{s'}\in C^-} S(x_t, x_{s'}) \end{aligned} \tag{2.2}$$

ここで $S(x_t, x_s)$ は，

$$S(x_t, x_s)=\sum_{i=1}^{n} x_{ti}x_{si}+\sum_{i=1}^{n}(1-x_{ti})(1-x_{si}) \tag{2.3}$$

で定義される学習例 x_t と x_s の類似測度である．ここで，学習例 $\boldsymbol{X}$ の類似行列を以下のように定義する．

定義 2.1 式(2.3)で定義した学習例のグループ類似度を表す関数 $G^+(x_t)$, $G^-(x_t)$ を学習例の集合 $\boldsymbol{X}$ のすべてについて求めた $T\times 2$ 行列

$$T(X, C)=[\{G^+(x_t), G^-(x_t)\} : t=1, 2, \cdots, T] \tag{2.4}$$

を概念 C の下で学習例 $\boldsymbol{X}$ の類似行列と定義する．□

補題 2.1[13),20)] 式(2.4)の類似行列は次式で与えられる．

$$T(X, C)=\{XX^T+(I-X)(I^T-X^T)\}[C, \mathbf{1}-C] \tag{2.5}$$

ここで，$\mathbf{1}$ は要素として1をもつ列ベクトルを表し，I は，要素として1をもつ $n\times T$ 行列である．□

類似行列 $T(X, C)$ は次のような性質をもつ．

定理 2.1[13),20)] 類似行列 $T(X, C)$ が線形分離可能ならば，すなわち，学習例の集合 $\boldsymbol{X}$ のそれぞれの要素 $x_t, t=1, 2, \cdots, T$ に対し

[i] $x_t\in C^+$ ならば

$$\alpha G^+(x_t)-\beta G-(x_t)+\theta>0 \tag{2.6}$$

[ii] $x_{t'}\in C^-$ ならば

$$\alpha G^+(x_{t'})-\beta G^-(x_{t'})+\theta\leq 0 \tag{2.7}$$

となるパラメータ α, β, θ が存在するならば学習例の集合 $\boldsymbol{X}$ は概念 $\boldsymbol{C}$ の下で線形分離可能である．また，各属性 x_i に対する結合係数 w_i を

$$w_i=\alpha\sum_{x_s\in C^+} x_{si}-\beta\sum_{x_{s'}\in C^-} x_{s'i} \tag{2.8}$$

とする次式で定義される関数

$$F(x)=\sum_{i=1}^{n} w_ix_i+\theta \tag{2.9}$$

は，学習例 $\boldsymbol{X}$ 上の概念 $\boldsymbol{C}$ の線形識別関数になる．すなわち，

(i) $x_t\in C^+$ ならば $F(x_t)>0$ (2.10)

(ii) $x_{t'}\in C^-$ ならば $F(x_{t'})\leq 0$ □

生物には，ある環境下で "One-Shot Learning"[3),14)] といわれている能力がある．特異なパターンの単一の表現を一度見たあとで，他の幾千もの中からそれを選び出すことができる．ここで学習例が線形分離可能な場合は One-Shot Learning の性質をもつのが瞬時学習法である．すなわち，入力ユニットを x_i, $i=1, 2, \cdots, n$ 出力ユニットを $o_j, j=1, 2, \cdots, k$ とし x_i と o_j 間の結合係数を w_{ij} とする．結合係数の修正学習戦略とは，t 番目の学習例 x_t に対し，i 番目の入力ユニットの値が x_{ti}，j 番目の出力ユニットの望ましい値が o_{tj} のとき，w_{ij} を次のように修正する方法である．

[i] $\Delta w_{ij}=\alpha_j$ if $x_{ti}=1$ and $o_{tj}=1$

[ii] $\Delta w_{ij}=-\beta_j$ if $x_{ti}=1$ and $o_{tj}=0$

[iii] $\Delta w_{ij}=0$ if $x_{ti}=0$ (2.11)

α_j, β_j, $j=1,2,\cdots,k$ は，定数である．すべての学習例を一度与えることにより，式(2.10)の修正ルールによって学習例の集合をすべて提示した後の結合係数を次のように決定できる．

$$w_{ij}=\alpha_j\sum_{t=1}^{T}x_{ti}o_{tj}-\beta_j\sum_{t=1}^{T}x_{ti}(1-o_{tj})$$
$$=\alpha_j\sum_{x_t\in C_j^+}x_{ti}-\beta_j\sum_{x_{t'}\in C_j^-}x_{t'i} \quad (2.12)$$

ここで，C_j^+ および C_j^- は j 番目の出力ユニット O_j の値によって決定される概念 C_j が定義する学習例 $\boldsymbol{X}$ の正および負の学習例を表す．

2.2 記憶に基づく学習アルゴリズム

本節では，ある特定の学習例に対してのみ発火するニューラルネットワークモデルを提案する．すなわち，学習例すべての入力パターンを記憶する記憶に基づく学習モデルである．

定理 2.4 各属性 x_i の結合係数 w_i としきい値 θ を

$$w_i=2x_{ti}-1$$
$$\theta=1-\|x_t\| \quad (2.13)$$

とする．ここで，$\|x\|$ は x のノルムを表す．次式で定義される関数

$$F(x)=\sum_{i=1}^{n}w_ix_i+\theta \quad (2.14)$$

は，学習例の集合のある特定の学習例 x_t だけに反応する識別関数になる．すなわち，

(i) $x=x_t$ ならば $F(x)>0$

(ii) $x\neq x_t$ ならば $F(x)\leq 0$ (2.15)□

証明 式(2.14)は，式(2.13)より

$$F(x)=\sum_{i=1}^{n}w_ix_i+\theta$$
$$=\sum_{i=1}^{n}(2x_{ti}-1)x_{ti}+1-\|x_t\|$$

となる．ここで，

(i) $x=x_t$ ならば $\sum_{i=1}^{n}(2x_{ti}-1)x_{ti}=\|x_t\|$ より

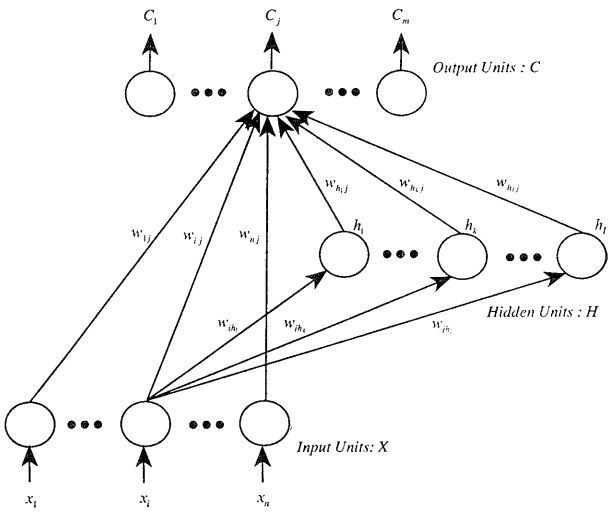
$$F(x)=1>0$$

(ii) $x\neq x_t$ ならば $\sum_{i=1}^{n}(2x_{ti}-1)x_{ti}+1\leq\|x_t\|$ より

$$F(x)\leq 0$$ □

瞬時学習アルゴリズムは，線形分離可能な問題に適用できる重み獲得学習法である．線形分離不可能な問題に対しては，瞬時学習法に記憶に基づく学習法を融合させた学習法を提案する．類似行列 $T(X,C_j)$ が線形分離不可能な場合，それを線形分離不可能にしている正の学習例 $Y=\{x_1,x_2,\cdots,x_{k-1}\}$ を抽出し，それらの学習例を記憶に基づく学習法により学習させる．これらの学習例を記憶するための中間ユニットは，中間概念を表現する．瞬時学習法と記憶に基づく学習法を融合した学習法を構造化学習法と定義する．構造化学習法は**図1**に示すように事前構造化過程と学習過程の二つの過程から構成される．事前構造化過程では，学習例 X の類似行列 $T(X,C_j)$, $j=1,2,\cdots,k$ からネットワークの構造すなわち中間ユニットの数やその活性値を決定する．学習過程では，入力層と出力層の結合係数を2.1節で示した瞬時学習アルゴリズムで求める．また，入力層と中間層間の結合係数は2.2節で示した記憶に基づく学習アルゴリズムで求める．以下にその手順を示す．

step 1.1 式(2.7)を負の学習例が満足するよう



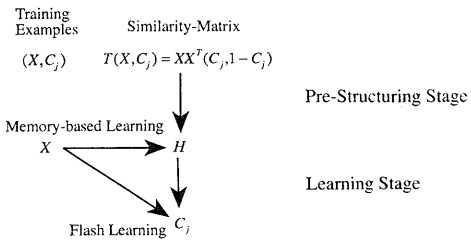


図1 構造化学習法の手順
Fig. 1 Procedures for structured connectionist learning.

にパラメータ α, β, θ を求め[20]，式(2.8)より結合係数 w_{ij} を求める．

step 1.2 *step 1.1* で求めたパラメータ α, β, θ で式(2.6)を満足しない正の学習例を抽出し，線形分離不可能にしている学習例の集合 $H=\{h_1, h_2, \cdots, h_m\}$ を求める．

step 1.3 求めた H を記憶に基づく学習アルゴリズムで学習し結合係数 w_{ih_k}* を求める．

step 1.4 中間ユニットを出力ユニットに以下の結合係数 w_{h_kj}** で結合する．

$$w_{h_kj}=\sum_{i=1}^{n}|w_{ij}|+|\theta_j| \qquad \square$$

誤差逆伝播法[15),22)]に代表される非構造的アプローチ[6),7),9)]では，ネットワークの構造について事前に規定するための方法がなく[21)]，特に中間層の構成法については試行錯誤により決定される．以上のような構造化学習法の特徴は，線形分離不可能としている学習例を事前に識別することにより，事前にネットワークの構造を決定できる点にある．

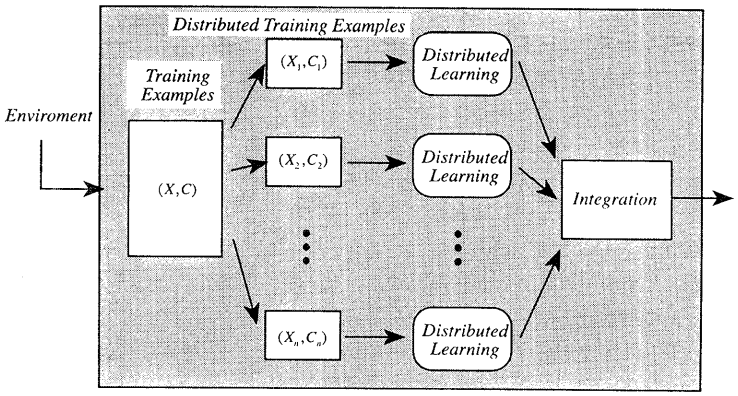


図 2 分散学習の概念図
Fig. 2 A concept of distributed learning.

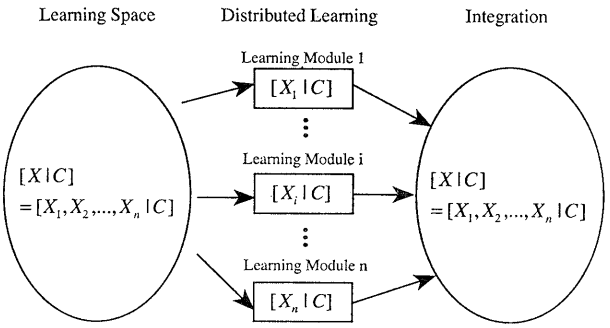


図 3 異質な学習空間の分散学習
Fig. 3 Distributed learning of heterogeneous learning examples.

3. 分散環境下での学習

分散学習は，学習空間 $L=\langle X, C\rangle$（ここで，X は入力属性空間，C は出力概念空間）の入力部分空間と出力部分空間の対からなる部分学習空間を学習し，その学習結果を統合する問題である．**図2**に分散学習の概念図を示す．ここで，大規模な問題をいくつかの小さな問題に分割する方法としては，

（i） 入力空間の分散（Type 1）

（ii） 学習例の分割（Type 2）

を考えることができる．Type 1 は，**図3**に示すように学習空間 $L=\langle X, C\rangle$ を部分空間 $L_i=\langle X_i|C\rangle$，$i=1, 2, \cdots, k$，$X_i \subset X$，$\bigcup_{i=1}^{k} X_i=X$ に分割する方法で，属性空間の分割による分散学習法である．部分学習空間 L_i を学習モジュール i で分散学習し，それぞれの学習モジュールの学習結果を統合モジュールにより統合する．これは，学習例を定義する複数の属性を分割する方法であり，各学習例に対して最終出力は一意に決定することができない．それぞれの学習モジュールは，異なった入力属性空間によって定義されることから，Type 1 による分散学習を異質の学習空間で構成される分散学習法と呼ぶことにする．Type 2 は，**図4**に示すように学習空間 $L=\langle X|C\rangle$ を部分空間 $L_i=\langle X_i|C_i\rangle$ $i=1, 2, \cdots, k$，$\bigcup_{i=1}^{k} L_i=L$ に分割する場合である．これは，学習例の集合を分割する方法であり，各学習例に対して最終出力は一意に決定することができる．それぞれの学習モジュールは，同じ入力属性空間によって定義されることから，Type 2 の分散学習を同質の学習空間における分散学習法と呼ぶことにする．

これらの分割された問題を処理するには，分散処理された情報が有効に統合するためのメカニズムが必要である．分散学習に関する従来の研究では，Type 2

---

* 入力ユニット i から中間ユニット h_k への結合係数を表す．

** 中間ユニット h_k から出力ユニット C_j への結合係数を表す．

の問題を対象にし，学習アルゴリズムは誤差逆伝播法に基づいている．このような分散学習法は，分割された問題間の相互依存度が高く，その相互依存性を考慮した新たなネットワークを統合モジュールとして構築する必要がある．したがって，分散処理の利点があまりない．分散学習アルゴリズムが，分散処理された情報を統合する上での加法性を有すれば統合のためのメカニズムが容易になる．情報統合上の加法性とは，それぞれの分散処理結果を重みつき線形総和により全体の解が得られる性質をいう．次節において瞬時学習法に基づく分散学習の基本定理を示し，情報の統合は情報の加法性の性質を有していることを示す．

3.1 異質な学習空間の分散学習法

以下では，一般性を失うことなく入力変数を二つに分割した問題を対象とする．

定理 3.1 学習空間 $\langle X|C\rangle$ の部分空間 $\langle X_i|C\rangle$, $i=1,2$ の類似行列 $T(X_1,C)$, $T(X_2,C)$ が共通のパラメータ α, β, θ の下で線形分離可能であり，学習空間の分散により学習例が矛盾を含まない時（同じ学習例に対し異なる出力を出さない時），統合された学習空間 $\langle X|C\rangle=\langle X_1,X_2|C\rangle$ はパラメータ α, β, 2θ の下で線形分離可能である．そのとき結合係数は，それぞれの分散学習で得られる結合係数によって得られる．

$$w_i=\begin{cases} w_i^1 \ for\ x_i\in X_1 \\ w_i^2 \ for\ x_i\in X_2 \end{cases} \tag{3.1}$$

$$w_i{}^j=\alpha\sum_{x_t\in C_j{}^+}x_{ti}-\beta\sum_{x_{t'}\in C_j{}^-}x_{t'i} \tag{3.2}$$

□

証明 学習空間 $\langle X|C\rangle$ の類似行列 $T(X,C)$ は以下で与えられる．

$$\begin{aligned} T(X,C)&=\{XX^T+(I-X)(I^T-X^T)\}[C,\mathbf{1}-C] \\ &=\{X_1X_1^T+X_2X_2^T+(I-X_1)(I^T-X_1^T) \\ &\quad+(I-X_2)(I^T-X_2^T)\}[C,\mathbf{1}-C] \\ &=\{X_1X_1^T+(I-X_1)(I^T-X_1^T)\}[C,\mathbf{1}-C] \\ &\quad+\{X_2X_2^T+(I-X_2)(I^T-X_2^T)\}[C,\mathbf{1}-C] \\ &=T(X_1,C)+T(X_2,C) \end{aligned}$$

ここで類似行列 $T(X_1,C)$, $T(X_2,C)$ が共通のパラメータ α, β, θ の下で線形分離可能ならば，それぞれの要素 $x_t=[x_{t1},x_{t2}]$, $t=1,2,\cdots,n$ に対し $x_t\in C^+$ の時

$$\alpha\sum_{x_s^1\in C^+}(x_t^1,x_s^1)-\beta\sum_{x_{s'}^1\in C^-}(x_t^1,x_{s'}^1)+\theta>0$$

$$\alpha\sum_{x_s^2\in C^+}(x_t^2,x_s^2)-\beta\sum_{x_{s'}^2\in C^-}(x_t^2,x_{s'}^2)+\theta>0$$

となるので，

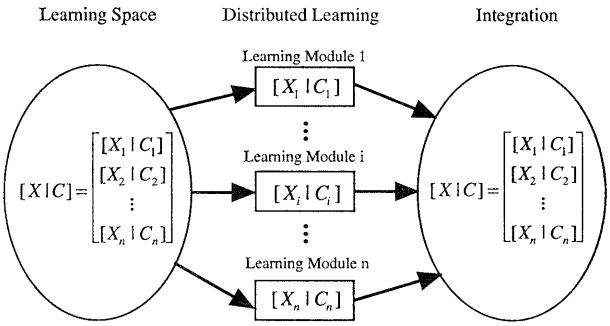


図 4 異質な学習空間の分散学習

Fig. 4 Distributed learning of homogeneous learning examples.

$$\begin{aligned} &\sum_{i=1}w_ix_{ti}+2\theta \\ &=\alpha\sum_{x_s\in C^+}(x_t,x_s)-\beta\sum_{x_{s'}\in C^-}(x_t,x_{s'})+2\theta \\ &=\alpha\sum_{x_s^1\in C^+}(x_t^1,x_s^1)-\beta\sum_{x_{s'}^1\in C^-}(x_t^1,x_{s'}^1)+\theta \\ &\quad+\alpha\sum_{x_s^2\in C^+}(x_t^2,x_s^2)-\beta\sum_{x_{s'}^2\in C^-}(x_t^2,x_{s'}^2)+\theta \\ &\quad>0 \end{aligned} \tag{3.3}$$

が得られる．また $x_t\in C^-$ の時

$$\alpha\sum_{x_s^1\in C^+}(x_t^1,x_s^1)-\beta\sum_{x_{s'}^1\in C^-}(x_t^1,x_{s'}^1)+\theta<0$$

$$\alpha\sum_{x_s^2\in C^+}(x_t^2,x_s^2)-\beta\sum_{x_{s'}^2\in C^-}(x_t^2,x_{s'}^2)+\theta<0$$

となるので，

$$\begin{aligned} &\sum_{i=1}w_ix_{ti}+2\theta \\ &=\alpha\sum_{x_s\in C^+}(x_t,x_s)-\beta\sum_{x_{s'}\in C^-}(x_t,x_{s'})+2\theta \\ &=\alpha\sum_{x_s^1\in C^+}(x_t^1,x_s^1)-\beta\sum_{x_{s'}^1\in C^-}(x_t^1,x_{s'}^1)+\theta \\ &\quad+\alpha\sum_{x_s^2\in C^+}(x_t^2,x_s^2)-\beta\sum_{x_{s'}^2\in C^-}(x_t^2,x_{s'}^2)+\theta \\ &\quad<0 \end{aligned} \tag{3.4}$$

が得られる．よって学習空間 $\langle X_1,X_2|C\rangle$ の線形識別関数は，

$$F(x)=\alpha G^+(x)-\beta G^-(x)+2\theta \tag{3.5}$$

となる．ここで，$G^+(x)=\sum_{x_p\in C^+}(x,x_p)$, $G^-(x)=\sum_{x_{p'}\in C^-}(x,x_{p'})$ である．よって結合係数は，以下で与えられる．

$$w_i=\begin{cases} w_i^1 \ for\ x_i\in X_1 \\ w_i^2 \ for\ x_i\in X_2 \end{cases}$$

$$w_i^j=\alpha\sum_{x_t\in C_j{}^+}x_{ti}-\beta\sum_{x_{t'}\in C_j{}^-}x_{t'i}$$ □

線形分離不可能な問題を含む場合は中間層を有することになり，この分散学習の結果を統合するために

は，それぞれのネットワークモジュールの中間層を統合する必要がある．

図5に示すように，統合されたネットワークの結合係数はそれぞれの分散学習で得られる結合係数によって得られる．また中間ユニットは，記憶に基づく学習アルゴリズムで学習しているので同じユニットは統合できる．異質な学習空間の特質として，各学習例に対して最終出力を一意に決定することができない．したがって，矛盾した学習例が含まれる場合が生じる．同じ入力パターンに対して異なる出力値をもつといったすなわち同一の入力ベクトルに対して異なる出力を出すような複数の学習例が同じ学習例の集合の中に存在する可能性がある．以下，矛盾した学習例をもつ場合と持たない場合の分散ネットワークモジュールの統合方法について示す．

ケース 1: 矛盾を含まない場合

step 2.1 分散された学習例が線形分離可能な共通のパラメータ α, β, θ を求める．存在しない場合は，負の学習例が式(2.6)を満足するようにパラメータ α, β, θ を求める．この際，線形分離不可能にする正の学習例を中間ユニットとする．

step 2.2 *step 2.1* で求めたパラメータ α, β, θ で分散された各ネットワークごと学習し結合係数を獲得する．

step 2.3 統合されたネットワークの結合係数は，定理3.1よりそれぞれの分散学習で得られる重みによって得られる．統合された学習による出力は各ユニットの出力となる．中間ユニットを有する場合，出力ユニットに論理和結合する． □

ケース 2: 矛盾を含む場合

学習の分割により学習例に矛盾を含む場合，そのような学習空間をそのままでは学習できない．そのためには矛盾した学習例を適切に処理する必要がある．矛盾が存在する場合は，それらの学習例をすべて同じクラスに属するように中間概念を形成する．すなわち，異なる出力をもつ学習例の集合を同じクラスとする中間ユニットを形成し，出力層において学習例に対応する正の学習例を負の学習例に変換する．これにより分散された学習例の集合は，矛盾を含まない学習例として扱える．以下に線形分離可能な場合についてその手順を示すが（**図6**を参照），線形分離不可能で矛盾を含む場合 *step 2* の手順と合わせて適用することにより対処することができる．

step 3.1 学習例 $\langle X_1|C\rangle, \langle X_2|C\rangle$ において矛盾している正の学習例を中間概念とする中間ユニットの集合 H_1, H_2 をそれぞれ生成する．

step 3.2 *step 3.1* により学習例は矛盾を含まない学習として扱えるので，*step 2* の手順により学習統合する．

step 3.3 中間ユニットの集合 H_1, H_2 を統合するには，それぞれの中間ユニットの集合に対して他方のネットワークの同じ学習例を入力として拡張し記憶に基づく学習アルゴリズムで学習して出力ユニットと論理和結合する．瞬時学習法で学習したネットワークで判別できる場合は，その中間ユニットを削除できる． □

3.2 同質な学習空間の分散学習法

定理 3.2 学習例 $\langle X|C\rangle$ の類似行列 $T(X, C)$ がパラメータ α, β, 2θ の下で線形分離可能と仮定する．学習空間 $\langle X|C\rangle$ の部分空間 $\langle X_i|C_i\rangle$, $i=1, 2$ の類似行列 $T(X_1, C_1)$, $T(X_2, C_2)$ が共通のパラメータ α, β, θ で線形分離可能であるならば，学習空間 $\langle X|C\rangle$ の結合係数 $w_i, i=1, 2, \cdots, n$ は，部分空間の結合係数 $w_i^j, j=1, 2$ の和によって得られる．

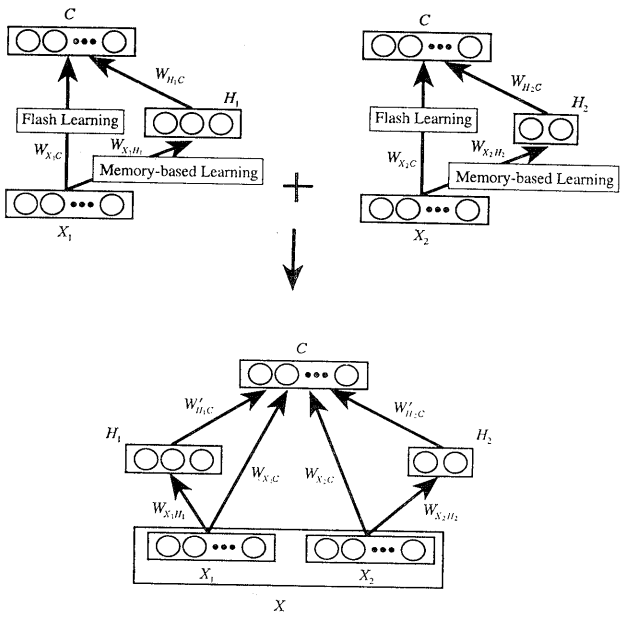


図 5 矛盾を含まない場合のネットワークの統合
Fig. 5 Integration of networks modules without contradicting training examples.

$$w_i = w_i^1 + w_i^2$$
$$w_i^j = \alpha \sum_{x_t \in C_j^+} x_{ti} - \beta \sum_{x_{t'} \in C_j^-} x_{t'i},\ j = 1, 2 \quad (3.9)$$

□

証明 学習空間 $\langle X|C \rangle$ の線形識別関数は,

$$F(x) = \alpha G^+(x) - \beta G^-(x) + 2\theta \quad (3.10)$$

で与えられる．ここで，$G^+(x) = \sum_{x_p \in C^+} (x, x_p)$, $G^-(x) = \sum_{x_{p'} \in C^-} (x, x_{p'})$ である．よって結合係数は，以下で与えられる．

$$\begin{aligned} w_i &= \alpha \sum_{x_t \in C_j^+} x_{ti} - \beta \sum_{x_{t'} \in C_j^-} x_{t'i} \\ &= \alpha \sum_{x_t \in C_1^+} x_{ti} - \beta \sum_{x_{t'} \in C_1^-} x_{t',i} \\ &\quad + \alpha \sum_{x_t \in C_2^+} x_{ti} - \beta \sum_{x_{t'} \in C_2^-} x_{t',i} \\ &= w_i^1 + w_i^2 \end{aligned}$$

□

以下では，同質な学習空間での分散学習において，線形分離不可能な問題に対する学習手順を示す．

step 4.1 線形分離不可能な正の学習例を負の学習例に変換した学習例の集合を線形分離可能とするパラメータ α, β, θ を求める．

step 4.2 *step 4.1* で求めたパラメータ α, β, θ で分散された各ネットワークごと学習し結合係数を獲得する．

step 4.3 統合されたネットワークの結合係数は，定理 3.2 よりそれぞれの分散学習で得られた重みによる和によって得られる．

step 4.4 線形分離不可能にしているそれぞれの学習モジュールを記憶した中間ユニットを求め，それらを出力ユニットに論理和結合する． □

4. ニューロ・エージェントモデル

分散学習を行うニューラルネットワークの実装モデルとしてニューロ・エージェントモデルを提案する．個別的に，また独立的に実行可能な計算メカニズムを備えた処理要素をエージェントと呼び，そのような複数のエージェントの協調により知識処理システムを実現するための研究が盛んに行われている[1),5),6),13)]．エージェントは，自分に関係することは原則として自分で処理する自律的な存在である．

4.1 ニューロ・エージェントモデルの構成と機能

エージェントモデルを確立するうえで重要なのはエージェントの自律的行動様式とエージェント間の相互作用のあり方をいかにモデル化するにある．すべてのエージェントは独立してそれぞれの目的をもって存在し，他のエージェントや外部環境からのメッセージを受理することにより活性化し，メッセージによって指定されたタスクを処理する．またどのようなタスクを行うかについては，エージェント内部の記憶の状態やタスクルールによって決定される．すなわち，ニューロ・エージェントモデルは，外部環境からのメッセージを解釈し，その内容を判断するための内部記憶(知識)を持ち，また状況の変化に対し自らの内部記憶の構造を変更するための学習メカニズムをもつ自律的なソフトウェアモジュールである．ニューロ・エージェントの内部記憶，すなわちどのような状態においてメッセージを受理し，どのような手続きでそれを処理するかなどのメタ知識は，ニューロ・エージェントが新しく生成された時に決定されるが，それ以降

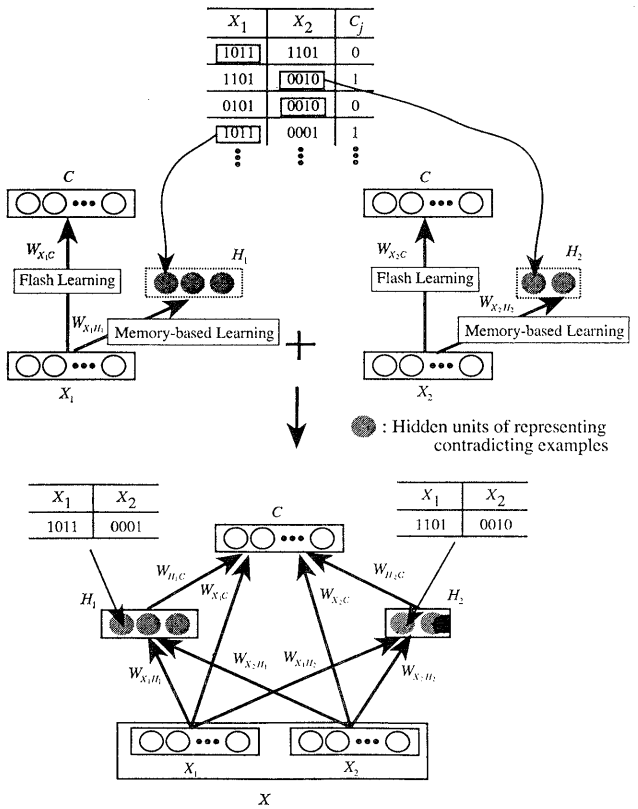


図 6 矛盾を含む場合のネットワークの統合
Fig. 6 Integration of networks modules with contradicting training examples.

はニューロ・エージェントに具備された学習メカニズムにより自己修正が行われる．ニューロ・エージェントは，メッセージを受理し解釈するための内部モデルと内部モデルの行動を規定するための学習例を記憶するための長期記憶部および長期記憶部や内部モデルの内容について自己再編するためのメカニズムをもつ自己再編機能より構成される．ニューロ・エージェントの内部モデルを構成するそれぞれのオブジェクトは，活性化ルールをもつ．活性化ルールは線形識別関数で定義し，その結合係数はオブジェクトが活性化するパターン（これを正の学習例という）と活性化しないパターン（これを負の学習例という）を用いて2章で示した構造化学習法により求める．すなわち，j 番目のオブジェクトは，メッセージに対して重みづけをするための結合係数 $w_j=(w_{1j}, w_{2j}, \cdots, w_{nj})$ と，外部からのメッセージを受理し，活性化するための判断を行うためのルールを線形識別関数として表現された活性化ルール

$$F_j(x)=\sum_{i=1}^{n} w_{ij}x_i+\theta_j \qquad (4.1)$$

を有する．それぞれのオブジェクトは，外部からのメッセージ $x=(x_1, x_2, \cdots, x_n)$ を受理し，正および負の学習例を用いて獲得したそれぞれの重み付け係数によりメッセージを解釈し，その重み付けされた値がしきい値 θ_j を越えた場合 $(F_j(x)>0)$ には活性化する．

長期記憶部は，外部からのメッセージに対して起動すべきオブジェクトとの対応関係（学習例の集合）を有する．自己再編機能とは，学習例の更新メカニズム，外部メッセージに対する重み付け係数の修正メカニズム（学習アルゴリズム）により構成される．学習例の特徴ベクトルを $x_p=(x_{p1}, x_{p2}, \cdots, x_{pm})$ およびそれに対応して起動すべき内容を出力ベクトル $\boldsymbol{c}=(c_{p1}, c_{p2}, \cdots, c_{pk})$ で表す．ニューロ・エージェントの初期の学習例の集合は，$\langle(\boldsymbol{x}_p, \boldsymbol{c}_p)\rangle$，$p=1, 2, \cdots, P$ として構成される．ここで，$c_{pj}, 1\leq j\leq k$ は $\{0, 1\}$ のブール値をとり，特に $c_{pj}=1, 1\leq p\leq P$ となる学習例の集合 C_j^+ を j 番目の出力オブジェクト（出力ベクトル $\boldsymbol{c}_p$ の j 番目の属性に対応するオブジェクト）の正の学習例，$c_{pj}=0$，$1\leq p\leq P$ となる学習例の集合 C_j^- を j 番目の出力オブジェクトの負の学習例と呼ぶことにする．ここで，それぞれ正および負の学習例の集合に対し定義されるグループ化関数 $G_j^+(x_p)$ は正の学習例全体との類似測度の総和を表わし，正の学習例の特徴が集約化される．一方，$G_j^-(x_p)$ は負の学習例全体との類似測度の総和で，負の学習例の特徴について集約して表現している．あるメッセージ x を受理した場合，それにより過去の学習例の特徴を集約したグループ関数 $\{G_j^+(x), G_j^-(x)\}$ に対して定義される活性化ルール

$$G_j(x)=\alpha_j G_j^+(x)-\beta_j G_j^-(x)+\theta_j \qquad (4.2)$$

により，$G_j(x)>0$ ならば発火する．

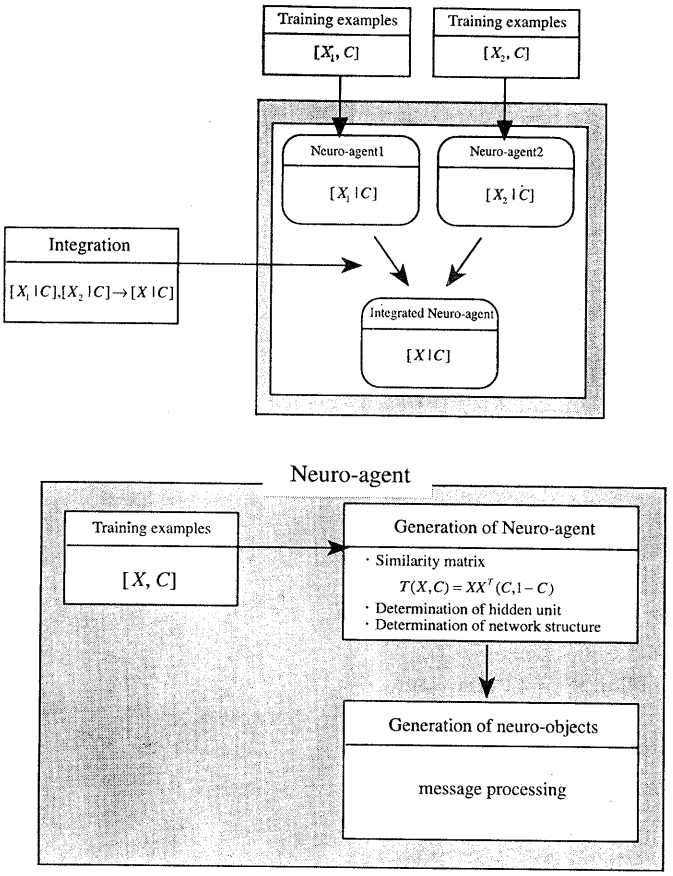


図 7 分散環境におけるニューロエージェントモデルの情報処理プロセスシミュレーション

Fig. 7 An information processing of a neuro-agent model under distributed learning environment.

4.2 シミュレーション環境

分散学習機能をもつニューロ・エージェントシミュレータをオブジェクト指向言語 Smalltalk-80 を用いて Macintosh 上で実現した．以下では，まず，Smalltalk-80 上でのインプリメント方法を簡単に示す．そして，**図7**に示すシミュレーション環境について説明する．

4.2.1 Smalltalk-80 上でのクラス

以下に，作成したクラスの階層を示す．

Object
- **NeuroAgentModel**
  - **NeuroAgent**
  - **NeuroObject**
- **ApplicationModel**
  - **NeuroAgentBrowser**
  - **NeuroObjectInspector**
  - **NeuroExampleBuilder**

• **NeuroAgent** クラス

このクラスでは，3章で述べたニューロ・エージェントの機能を提供している．出力オブジェクト群，学習例ファイルなどの情報を保持している．

• **NeuroObject** クラス

このクラスでは，3.1 節で述べたニューロ・エージェントの内部モデルであるオブジェクトを実現するクラスである．中間オブジェクト群，類似行列，活性化ルールなどの情報を保持している．ニューロ・エージェントからのメッセージにより自律的に活性化ルールを獲得するインスタンスメソッドを持つ．

• **NeuroAgentBrowser** クラス

ユーザインターフェース用のウィンドウを提供するクラスである．このウィンドウを通じてニューロ・エージェントを生成して，シミュレーションを行う．

• **NeuroExampleBuilder** クラス

学習例の編集，学習ファイル管理を提供するクラスである．このウィンドウを通じてニューロ・エージェントに作成した学習例をメッセージとして送る．

• **NeuroObjectInspector** クラス

ニューロ・エージェント内のニューロ・オブジェクトの学習結果を表示するウィンドウを提供するクラスである．学習結果としてニューロ・オブジェクトとその中間ニューロ・オブジェクトの結合係数と閾値を確認できる．

表 1 分散環境下における学習例
Table 1 Training examples in a distributed training environment.

	X										C		
	X_1					X_2							
i	x_{i1}	x_{i2}	x_{i3}	x_{i4}	x_{i5}	x_{i6}	x_{i7}	x_{i8}	x_{i9}	x_{i10}	c_1	c_2	c_3
x_1	1	0	1	0	1	0	1	0	1	0	1	1	0
x_2	1	0	1	1	1	1	0	0	0	0	0	0	1
x_3	0	1	0	1	1	1	0	1	0	1	1	1	0
x_4	1	0	1	0	0	0	1	1	1	1	1	1	0
x_5	1	1	1	1	1	1	0	0	1	0	1	0	1
x_6	1	0	1	0	0	0	1	1	1	0	0	0	1
x_7	0	1	0	1	1	1	0	0	1	1	1	1	0
x_8	0	0	1	1	1	0	1	1	1	0	0	1	0

表 2 $[X_1, C]$ を学習したニューロ・エージェントが獲得した結合係数
Table 2 Obtained weight of neuro-agent with learning example $[X_1, C]$.

オブジェクト		結合係数									
		w_{1j}	w_{2j}	w_{3j}	w_{4j}	w_{5j}	w_{h1j}	w_{h4j}	w_{h6j}	θ
c_j	c_1	0	9	-2	5	8	14			-11
	c_2	-18	10	-6	8	20	39	39		-14
	c_3	8	0	6	2	0			15	-14
h_j	h_1	1	-1	1	-1	1				-2.5
	h_4	1	-1	1	-1	-1				-1.5
	h_6	1	-1	1	-1	-1				-1.5

表 3 $[X_2, C]$ を学習したニューロ・エージェントが獲得した結合係数
Table 3 Obtained weight of neuro-agent with learning example $[X_2, C]$.

オブジェクト		結合係数										
		w_{1j}	w_{2j}	w_{3j}	w_{4j}	w_{5j}	w_{h1j}	w_{h3j}	w_{h6j}	w_{h7j}	w_{h8j}	θ
c_j	c_1	23	-3	-3	15	14	30				8	-23
	c_2	-10	5	5	0	15	21	21		21	21	-10
	c_3	12	-2	-2	4	-12			21			-4
h_j	h_1	-1	1	-1	1	-1						-1.5
	h_3	1	-1	1	-1	1						-2.5
	h_6	-1	1	1	1	-1						-2.5
	h_7	1	-1	-1	1	1						-2.5
	h_8	-1	1	1	1	-1						-2.5

4.2.2 分散学習シミュレーション

図7に示した順序で分散学習シミュレーションを行う．以下簡単に説明する．最初に NeuroAgentBrowser で二つニューロ・エージェントを生成し，そのニュー

表 4 統合したニューロ・エージェントが獲得した結合係数
Table 4 Obtained weight of integtaed neuro-agent.

オブジェクト		結合係数																	
		w_{1j}	w_{2j}	w_{3j}	w_{4j}	w_{5j}	w_{6j}	w_{7j}	w_{8j}	w_{9j}	w_{10j}	w_{h1j}	w_{h2j}	w_{h3j}	w_{h4j}	w_{h5j}	w_{h6j}	w_{h7j}	w_{h8j}	θ
c_j	c_1	0	9	-2	5	8	7	-3	-3	3	4	31								-22
	c_2	2	10	9	13	20	6	17	17	20	21	109		109	109			109	109	-108
	c_3	8	0	6	2	0	4	-2	-2	0	-6		39			39	39			-28
h_j	h_1	1	-1	1	-1	1	-1	1	-1	1	-1									-4.5
	h_2	1	-1	1	1	1	1	-1	-1	-1	-1									-4.5
	h_3	-1	1	-1	1	1	1	-1	1	-1	1									-5.5
	h_4	1	-1	1	-1	-1	-1	1	1	1	1									-5.5
	h_5	1	1	1	1	1	1	-1	-1	1	-1									-6.5
	h_6	1	-1	1	-1	-1	-1	1	1	1	-1									-4.5
	h_7	-1	1	-1	1	1	1	-1	-1	1	1									-5.5
	h_8	-1	-1	1	1	1	-1	1	1	1	-1									-5.5

ロ・エージェント各々の学習例を作成しメッセージとして送る．学習例を受け取ったニューロ・エージェントは，学習例に応じて出力オブジェクトを自律的に生成する．各出力オブジェクトは，学習例から学習パラメータを算出し，活性化ルールを獲得する．また，線形分離不可能な場合は，記憶に基づく学習アルゴリズムを用いて中間オブジェクトを生成する．学習したニューロ・エージェントを統合するには，Neuro-AgentBrowser 上で統合したいニューロ・エージェントに他方のニューロ・エージェントのコピーをメッセージとして送る．メッセージを受け取ったニューロ・エージェントは，統合ニューロ・エージェントを生成するとともに自身のコピーを生成する．ここで，分散学習アルゴリズムに基づき新たに生成された三者間でメッセージを交換し，統合ニューロ・エージェントの学習が完了する．

分散学習法は，大規模な問題に対処するために提案されたものであるが，以下では小さな学習問題に適用した例を示しながら分散学習法の妥当性について示す．分散学習の例を示すための学習環境を**表 1** に示す．二つのニューロ・エージェントを生成し，学習環境としてそれぞれに $\langle X_1, C\rangle$, $\langle X_2, C\rangle$ を与えてその学習結果を統合する．**表 2** に学習例 $\langle X_1, C\rangle$ を学習したニューロ・エージェントが獲得した結合係数を示す．出力オブジェクト c_1 において 1 番目の学習例が線形分離不可能となり中間オブジェクト h_1, h_4, h_6（添字は学習例呈示番号）が生成されていることがわかる．

表 5 $[X, C]$ を学習したニューロ・エージェントが獲得した結合係数
Table 5 Obtained weight of neuro-agent with the integrated learning examples $[X, C]$.

オブジェクト		結合係数												
		w_{1j}	w_{2j}	w_{3j}	w_{4j}	w_{5j}	w_{6j}	w_{7j}	w_{8j}	w_{9j}	w_{10j}	w_{h1j}	w_{h6j}	θ
c_j	c_1	-16	39	-30	11	24	25	-29	-29	-3	12	122		-14
	c_2	-5	5	0	5	10	0	10	10	10	15	31		-25
	c_3	29	-5	18	1	-10	12	-16	-16	-10	-33		102	-11
h_j	h_1	1	-1	1	-1	1	-1	1	-1	1	-1			-4.5
	h_6	1	-1	1	-1	-1	-1	1	1	1	-1			-4.5

表 3 には，学習例 $\langle X_2, C\rangle$ を学習したニューロ・エージェントが獲得した結合係数を示す．この二つのニューロ・エージェントを統合したニューロ・エージェントが獲得した結合係数を**表 4** に示す．それぞれ分散された学習結果を統合する際に共通な学習パラメータ α, β, θ を獲得する必要があり表 2 および表 3 に示した結合係数とは相違している．また，表 1 の全体の学習例 $\langle X, C\rangle$ を学習例として一度に示して学習した結果を**表 5** に示す．この例では，異質な学習空間における中間オブジェクトより多くの中間オブジェクトを生成している．

線形分離不可能な問題に対しては，一括して学習した場合と比し，多くの中間ユニットを生成することになり非効率的である．これは分散学習の宿命的な欠点である．しかしながら，もともと分散しているものを統合したり，分散した環境で学習例が動的に変化している場合において最初から学習するよりその学習結果を利用する方が有効である．また，従来の学習アルゴ

リズムは，ネットワークの構造やユニット間の結合係数を一度決定されてしまうと，新しい学習例の追加や入出力ユニットの変更を伴う学習環境の動的な変化に容易に適応することが困難である[11),19)]．ネットワークの構造を自己再編し自己成長するための機能，すなわちコネクショニストモデルの環境への適応能力や自己成長機能は高次の認知処理を実現していくために不可欠である[2),19)]．自律的な分散学習機能をもつニューロ・エージェントモデルは，新しい学習例の追加，入力情報の新しい属性の追加および学習例の新しい分離法の追加といった学習環境の変化に対しても容易に対処することが可能である．

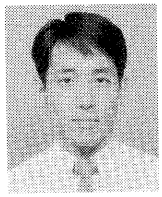
5. おわりに

ニューラルネットワークの規模の問題を解決するための方法として分散学習や複合化モデルが考えられる．これは，大規模な問題をいくつかの小さな問題に分割したり，または物理的，地理的または意図的にすでに分散されている，これらの小さな問題について学習をした結果を統合して大規模な問題を解決する試みである．大規模な問題をいくつかの小さな問題に分割し並列処理するには，並列処理された情報を有効に統合するためのメカニズムが必要である．特に，情報を統合する上で分散処理された結果が加法性を有すれば統合のためのメカニズムが容易になることを示した．また，分散されている環境で全体を知る管理者的存在がいないシステムを考えた場合に，各分散された環境ごとに学習し，その学習結果を利用して全体を学習したのと同様の結果が得られることを示した．自己学習機能をもつニューロ・エージェントモデルは情報の加法性の性質を有しており，それらのモデルの複合化により大規模で高次の認知処理システムの構築が可能になると考えるが具体的な実現の例は今後の課題である．

参考文献

1) Alan, H. B. and Gasser, L.: *Readings in Distributed Artificial Intelligenceed*, Morgan Kaufmann (1988).
2) Barnden, J. A. and Pollack, J. B.: *A High-Level Connectionist Models*, Ablex Publishing (1991).
3) Brown, R. and Fulik, J.: Flashbulb Memories, *Cognition*, Vol. 5, pp. 73-99 (1977).
4) Deen, S. M.: *Cooperating Knowledge Based Systems 1990*, Springer (1990).
5) Demazeau, Y. and Müler, J. P.: *Decentralized AI*, Elsevier Science Publishers (1990).
6) Fanty, A.: Structured Connectionist Network Learning, TR-252, University of Rochester (1988).
7) Feldmann, J. A.: Structured Connectionist Network Learning, TR-121, University of Rochester (1986).
8) Hrycej, T.: *Modular Learning in Neural Netrworks*, Wiley-Interscience (1992).
9) 石川真澄：忘却を用いたコネクショニストモデルの構造学習アルゴリズム，人工知能学会論文誌，Vol. 5, No. 3, pp. 596-603 (1990).
10) Jacobs, R. A., Jordan, M. I. et al.: Adaptive Mixtures of Local Experts, *Neural Computation*, Vol. 3, pp. 79-87 (1991).
11) Lee, T.: *Structure Level Adaptation for Artificial Neural Networks*, Kluwer Academic Publishers (1991).
12) Minsky, M.: *The Society of Minded*, Simons & Schster (1985).
13) Namatame, A. and Tsukamoto, Y.: Sructural Connectionist Learning with Complementary Coding, *International Journal of Neural Systems*, Vol. 3, No. 1, pp. 19-30 (1992).
14) Reiley, D. L., Cooper, L. N. et al.: A Neural Model for Category Learning, *Biological Cybernetics*, Vol. 45, pp. 35-41 (1982).
15) Rumelhart, D., Hinton, G. E. et al.: Learning Representation by Backpropagation, *Nature*, Vol. 323, pp. 533-536 (1986).
16) Shastri, L.: *Semantic Networks: An Evidential Formalization and Its Connectionist Realizationed*, Morgann Kaufmann (1988).
17) Sian, S. E.: The Role of Cooperation in Multi-agent Learning Systems, *Cooperative Knowledge-Based Systems*, pp. 67-84, Springer-Verlag (1990).
18) Sikora, R.: Learning Control Strategies for Chemical Processes, *IEEE EXPERT*, Vol. 7, No. 3, pp. 35-43 (1992).
19) Touretzy, D. S.: Special Issue on Connectionist Approaches to Natural Language Learning, *Machine Learning*, Vol. 7, No. 2-3 (1991).
20) 塚本義明，生天目章：多層ネットワークの瞬時学習法，情報処理学会論文誌，Vol. 34, No. 9, pp. 1882-1891 (1993).
21) Volper, D. L. and Hampson, S. E.: Connectionistic Models of Boolean Category Representation, *Biological Cybernetics*, Vol. 54, pp. 393-406 (1986).
22) Werbos, P. J.: Generalization of Backpropagation with Application to a Recurrent Gas Market Model, *Neural Network*, Vol. 1, pp. 339-356 (1989).

(平成6年3月10日受付)
(平成6年9月6日採録)



塚本 義明（正会員）

1963 年生．1986 年防衛大学校応用物理学科卒業．1992 年同理工学研究科（オペレーションズリサーチ専攻）修了．現在，防衛大学校情報工学教室研究員．日本ソフトウェア科学会，人工知能学会，日本神経回路学会各会員．

生天目 章（正会員）

1950 年 10 月 4 日生．1973 年防衛大学校卒業(応用物理学専攻)．1977 年および 1979 年スタンフォード大学大学院修士および博士課程修了(Ph. D)．同年航空幕僚監部勤務．1987-1988 年ジョージメイソン大学客員助教授．現在，防衛大学校情報工学教室助教授．人工知能，ニューラルネットワーク，意思決定工学等の研究に従事．人工知能学会，ソフトウェア科学会，神経回路学会，AAAI, ACM, IEEE 学会各会員．