動的環境下におけるコネクショニスト学習ツールの構築

塚本義明† 生天目章†

本論文では，オブジェクト指向に基づくコネクショニストモデルの構築法について提案をする．コネクショニストモデルの環境への適応性や自己成長性は，高次の認知処理を実現していくために不可欠な機能である．多層ネットワークは，コネクショニストモデルにおいて最も活発に研究されている分野の一つであるが，ネットワークの構造やユニット間の結合係数が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変化に容易に適応することが困難である．オブジェクト指向に基づくシミュレーションツールとしてニューロ・エージェントモデルを構築し，自己再編および自己成長機能を有するコネクショニストモデルについて提案をする．ニューロ・エージェントは，具備した学習メカニズムを用いて新しいオブジェクトの追加や内部パラメータを変更することにより，学習環境の変化に対して自律的に適応し自己成長できることを示す．

Tools for Connectionist Learning in a Dynamic Environment

Yoshiaki Tsukamoto† and Akira Namatame†

This paper illustrates how a high level specification agent model based upon the object-oriented language of parallel processing can describe a parallel architecture which consistently implements a specific neural network paradigm. This paper provides an agent-oriented environment for parallel distributed processing specification. Studies of distributed problem solving in the fields of artificial intelligence (DAI) and neural networks are rapidly putting parallel computation model into practice. Since the ability to learn is essential for any intelligent system, it should be a key element to consider in the design of DAI with thc agent model. Research on learning in an agent model may start to be concerned with developing ways the agents' knowledge and skill, so that not only can individual agents be better at their tasks, but the whole agents can also keep improving their performances as a result. Multi-layered networks are used as an example of considerations required to specify consistent parallel architecture. A neuro agent model is a connectionist network that facilitates modular knowledge composition. Each modular network uses a specialized a neuro agent. A neuro agent, endowed with his internal short-term memory and learning capability, represents a specific problem domain. Several agents are specialized to interact with the learning environment and some agents are specialized for the integration of those agents.

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[6]．多層ネットワークは，ネットワークの学習メカニズムによりネットワークの構造やユニット間の結合係数が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り，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を有していない．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[3),11)]．自己成長型の多層ネットワークを構築するための研究は盛んに行われている[4),7),8),12)]．これらの研究は，与えられた一組の学習例を学習する多層ネットワークを自己成長的に構築することを対象としており，ユニット間の結合構造を動的に構築するためのメカニズムの解明を主な研究目的としている．本研究では，与えられた学習例そのものが動的に変化するような学習問題を対象とし，ネットワーク構造そのものを環境の変化に対し自律的に適応（学習）するための機能をもつ新しい自律的なコネクショニストモデルを提案する．

人工知能の研究において，個別に独立に実行可能な

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

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),2),5)]．本研究では，コネクショニストモデルに基づく学習機能を有する自律的なソフトウェアモジュールをニューロ・エージェントと定義し，オブジェクト指向プログラミングに基づくシミュレーションツールを構築する．ニューロ・エージェントは，自律的で協調的な並行オブジェクト群（内部モデル）とコネクショニストモデルに基づく概念学習機能により構成される．ニューロ・エージェントの内部モデルは，新しくエージェントが生成された時に決定されるが，内部モデルを構成するそれぞれのオブジェクトには，そのオブジェクトを特徴づける集約的な学習パラメータが記憶される．多層ネットワークのユニット間の複雑な結合係数を内部モデルである並行オブジェクトの中にカプセル化した多層ネットワークと等価なモデルを，ニューロ・エージェントを用いることにより構築できることを示す．さらに，ニューロ・エージェントは，学習メカニズムにより新しいオブジェクトの追加や内部パラメータの変更は自律的に行うことができ，環境への適応機能を有する自己成長型の多層ネットワークを構築できることを示す．

2. 動的環境下におけるコネクショニスト学習

2.1 コネクショニスト情報処理

コネクショニストモデルは，ユニットと呼ぶ神経細胞のような働きをする単純な素子が，重みつきの結合係数により相互に接続されたネットワークによる情報処理である．コネクショニストモデルでは，短期記憶はユニット群の結合係数により表現され，長期記憶は特定のユニット間の結合関係により表現される．従来のコネクショニスト情報処理の主な研究課題は，**図1**に示すようにニューロンの内部パラメータを，対象とする問題に適応するようにいかに獲得する（これをネットワークの内部パラメータレベルでの適応能力という）かであり，このための数多くの学習アルゴリズムが提案されている[10)]．これらの研究の成果は，視覚系などの低次の認知処理には適しているが，高次の認知処理を実現するためには多くの課題が残されている．コネクショニストモデルによる高次認知処理法を実現するために，ユニットにそれぞれ特定の意味をもたせたり，またネットワークに特定の構造をもたせるなどいくつかの方法が提案されている[3),6),11)]．これらは，いずれも知識表現や推論メカニズムといった高次認知処理の実現のための必要な基本機能を単純なユニット間の結合構造（ネットワーク構造）の中に組み入れる方法を取っている．ネットワークに構造を持ち込むアプローチは，しかしながら，どのような構造を前提とするか，またいかにしてそのような構造を獲得するか等についてまだ解明されていない点が多い．さらに，学習環境が動的に変化するような問題にも対処できるように，構造レベルでのネットワークの適応能力をいかに向上させるかがコネクショニストモデルの基本的な課題である．ニューロ・エージェントモデルによるコネクショニストネットワークの実現法は，コネクショニストモデルと構造レベルでの適応能力を向上させるための方法として有効である．すなわち，新しい学習例の追加，入力情報の属性の変化および学習例の新しい分類法の追加等の学習環境の変化に対して，必ずしも再度学習を繰り返す必要はなく，新しいオブジェクトの追加や既存のオブジェクトの内部パラメータの変更により対処することが可能になる．学習環境の変化に対応するための従来のアプローチとニューロ・エージェントモデルのアプローチの比較を**図2**に示す．

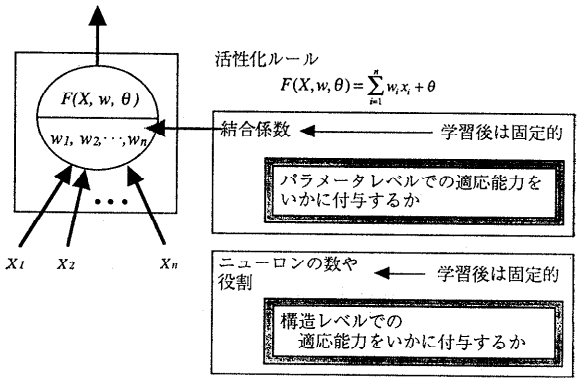


図1 ニューラルネットワーク適応モデル
Fig. 1 An adaptive model of neural networks.

2.2 コネクショニスト学習

パーセプトロンや誤差逆伝搬法に代表される学習アルゴリズムは，ネットワーク内の出力ユニットについて望ましい状態での値との誤差を利用してユニット間の結合係数を決定するため学習には時間がかかる．これは，一つの学習例に対する誤差量に対し逐次結合係数を修正していく誤差修正アルゴリズムであるため，

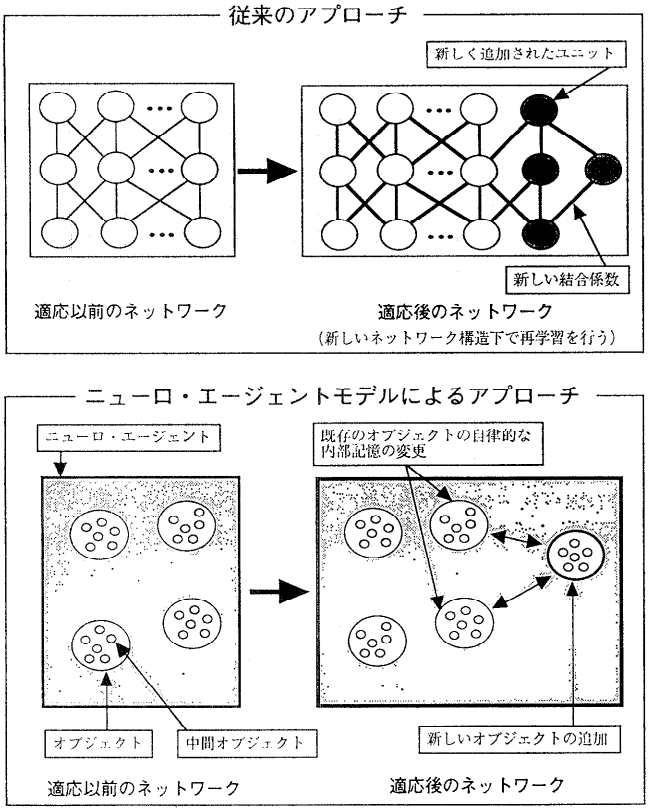


図 2 従来のニューラルネットワークのアプローチとニューロ・エージェントによるアプローチの比較
Fig. 2 A comparison of the conventional approach for neural networks with a neuro-agent model.

最終的なユニット間の結合係数を決定するためには同じ学習例を何度も提示する必要があることから学習に長時間要することになる．入力ユニットから出力ユニットまでの間に多くの隠れユニットの層がある多層ネットワークは，隠れユニットのないものよりはるかに複雑な機能を実現することができる．これは，隠れユニットの層内に学習の汎化能力に必要な内部表現が学習アルゴリズムにより獲得されるからである．しかしながら，その学習アルゴリズムは，結合係数を決定した後に新たに学習例が追加され，また入出力関係構造の変更に伴い再学習が必要な場合に，それまでに獲得した学習内容を逐次修正して新しい学習環境に対応するための適応性がない．新しい状況の下での学習例を提示し，再度学習をしてユニット間の結合係数を新しく決定する必要がある．

本節においては，文献9の学習アルゴリズムを動的な学習問題に拡張をする．文献9の学習アルゴリズムでは，学習例の集合に類似行列を定義し，その類似行列の構造から与えられた入出力関係について表現するために必要な隠れユニットの構造について決定し，またユニット間の結合係数は学習例を一度提示するだけで決定できるもので，動的な学習問題の学習アルゴリズムとしてふさわしい性質をもつ．以下，その基本定理を示す．

入力情報の集合で構成される $P\times n$ 行列を $\boldsymbol{D}=[d_{pi}]$ で表し，その入力情報に対するニューロ・エージェントの望ましい出力情報の集合を $P\times 1$ 行列 $\boldsymbol{C}=[c_p]$ および $\boldsymbol{C}$ の補完データを $\boldsymbol{I}-\boldsymbol{C}$ で表す．ここで，$\boldsymbol{I}$ は1を要素としてもつ $P\times 1$ 行列である．

定義 2.1 学習例の集合 $\boldsymbol{D}$ に対し，学習例のグループ関数を次式で定義する．

$$G^+(d_p)=[DD^T+(I-D(I-D)^T]\boldsymbol{C}$$
$$G^-(d_p)=[DD^T+(I-D)(I-D)^T]\cdot(\boldsymbol{I}-\boldsymbol{C}) \qquad (2.1)$$

定義 2.2 学習例の集合 $\boldsymbol{D}$ のグループ関数で構成される $P\times 2$ 行列

$$T(D,C)=[G^+(d_p),G^-(d_p)],\quad (1\leq p\leq P) \qquad (2.2)$$

を概念 $\boldsymbol{C}$ の下での学習例 $\boldsymbol{D}$ の類似行列と定義する．

学習例 $\boldsymbol{D}$ の類似行列 $T(D,C)$ が線形分離可能ならば，ニューロ・エージェントの内部モデルを構成するオブジェクトの活性化ルールを学習例 $\boldsymbol{D}$ を一度提示するだけで決定できる．

定理 2.1[9] 類似行列 $T(D,C)$ が線形分離可能ならば，すなわちあるパラメータ $r=(\alpha,\beta,\theta)$ が存在して

(i) 正の学習例 $d_p\in C^+$ に対し

$$\alpha G^+(d_p)-\beta G^-(d_p)+\theta>0 \qquad (2.3)$$

(ii) 負の学習例 $d_{p'}\in C^-$ に対し

$$\alpha G^+(d_{p'})-\beta G^-(d_{p'})+\theta<0$$

ならば，結合係数 w_i, $i=1,2,\cdots,n$ を，

$$w_i=\alpha\sum_{d_p\in C^+}d_{pi}-\beta\sum_{d_{p'}\in C^-}d_{p'i} \qquad (2.4)$$

で定義し，そのような結合係数をもつ活性化関数を

$$F(x)=\sum_{i=0}^{n}w_ix_i+\theta \qquad (2.5)$$

で定義すると，

(i) 正の学習例 $d_p \in C^+$ に対し $F(d_p)>0$

(ii) 負の学習例 $d_{p'} \in C^-$ に対し $F(d_{p'})<0$ (2.6)

が成立する. □

2.3 動的環境下での学習

動的環境下での学習とは，学習例の集会が逐次変化するような学習問題をいう．すなわち，初期学習例を与え学習した後に，環境が変化に伴い学習例の集合が変化した場合でもいかにして一貫性のある学習を継続するか，また以前に獲得した学習結果（知識）を再利用することにより新しい学習環境における学習を効率的に行うかが主な研究課題である．学習例の集合 $\boldsymbol{D}$ およびそれに対する望ましい出力 C の集合を学習環境と定義し，それを $[D|C]$ で記述する．学習環境 $[D|C]$ に新しく学習例が追加された場合や入力属性ベクトルが新しく追加された場合には，新しい学習環境の下で(2.2)式で定義した類似行列を求めそれに応じて(2.3)式を満たすように学習パラメータ α，β，θ を修正する．また新しい概念の追加に対してはそれに対応した内部パラメータとして(2.5)式の活性化関数や学習パラメータをもつ出力ユニットを生成する．この場合は，既存の出力ユニットの学習パラメータは不変である．このようにして，新しい学習環境に対しても容易に対応でき，また学習環境が変化する以前に獲得した類似行列や学習パラメータの再利用が可能である．すなわち，定義2.1で与えた類似行列 $T(D, C)$ は，学習環境の変化に対し次のような不変性の性質を有している．

ケース1 学習例の集合 $\boldsymbol{D}$ に新しく学習例 $(x^{\mathrm{new}}, c^{\mathrm{new}})$, $(c^{\mathrm{new}}=\{0,1\})$ が追加された場合

(1) x^{new} が正の学習例の時 $(c^{\mathrm{new}}=1)$

a. $x_t \in C^+$ に対して

$$F(x_t)+\alpha(x^{\mathrm{new}}, x_t)>0$$

b. $x_{t'} \in C^-$ に対して

$$F(x_{t'})+\alpha(x^{\mathrm{new}}, x_{t'})<0$$

c. x^{new} に対して

$$F(x^{\mathrm{new}})+\alpha(x^{\mathrm{new}}, x^{\mathrm{new}})>0 \qquad (2.7)$$

(2) x^{new} が負の学習例の時 $(c^{\mathrm{new}}=0)$

a. $x_t \in C^+$ に対して

$$F(x_t)-\beta(x^{\mathrm{new}}, x_t)>0$$

b. $x_t' \in C^-$ に対して

$$F(x_t)-\beta(x^{\mathrm{new}}, x_t')<0$$

c. x^{new} に対して

$$F(x^{\mathrm{new}})-\beta(x^{\mathrm{new}}, x^{\mathrm{new}})<0 \qquad (2.8)$$

が成り立てば，新しい学習例 $X_{\mathrm{new}}=\begin{bmatrix} X \\ x^{\mathrm{new}} \end{bmatrix}$ は概念 $C_{\mathrm{new}}=\begin{bmatrix} C \\ c^{\mathrm{new}} \end{bmatrix}$ の下で線形分離可能で，$F(x)$ は変更後の学習例の集合 $\begin{bmatrix} X \\ x^{\mathrm{new}} \end{bmatrix}$ に対しても線形分離可能になる．

ケース2 学習例の集合 $\boldsymbol{D}$ に対して新しく入力属性 $z^T=(z_1, z_2, \cdots, z_p)$ $(1\times P$ 行列$)$ が追加された場合，新しい学習例 $X_{\mathrm{new}}=(D, z)$ が次の条件を満たせば X_{new} も概念 C の下で線形分離可能で，$F(x)$ もその識別関数になる．

(1) $x_t \in C^+$ および $z_t=1$ となる x_t に対して

$$F(x_t)+(\alpha z^+-\beta z^-)>0$$

(2) $x_{t'} \in C^-$ および $z_{t'}=1$ となる $x_{t'}$ に対して

$$F(x_{t'})+(\alpha z^+-\beta z^-)<0 \qquad (2.9)$$

ここで，

$$z^+=\sum_{p=1}^{P} z_p, \quad z^-=P-z^+$$

ケース3 概念 C の下で線形分離可能な学習例の集合 $\boldsymbol{D}$ は，次の条件を満たせば新しい概念 $C_{\mathrm{new}}=[c_{\mathrm{new}}]$ の下でも線形分離可能である．

(1) $x_t \in C^+ \wedge C^+_{\mathrm{new}}$ となる学習例 x_t に対して

$$F(x_t)+(\alpha-\beta)\Delta G>0$$

(2) $x_t \in C^- \wedge C^-_{\mathrm{new}}$ となる学習例 x_t に対して

$$F(x_t)-(\alpha-\beta)\Delta G<0$$

(3) $x_t \in C^+ \wedge C^+_{\mathrm{new}}$ となる学習例 x_t に対して

$$F(x_t)-(\alpha-\beta)\Delta G>0$$

(4) $x_t \in C^+ \wedge C^-_{\mathrm{new}}$ となる学習例 x_t に対して

$$F(x_t)+(\alpha-\beta)\Delta G<0 \qquad (2.10)$$

ここで，$\Delta G=\sum_{s=1}^{T} S(x_t, x_s)(c-c_t)$.

新しい学習例が追加され(2.8)(2.9)(2.10)が成立しない場合は，以下のアルゴリズムを適用する．

<動的環境下における学習アルゴリズム>

Step 1 類似行列の修正

$$T(X_{\mathrm{new}}, C_{\mathrm{new}}) = X_{\mathrm{new}} X^T_{\mathrm{new}}[C_{\mathrm{new}}, \boldsymbol{I}-C_{\mathrm{new}}] \qquad (2.11)$$

Step 2 学習パラメータ $r_{\mathrm{new}}=(\alpha_{\mathrm{new}}, \beta_{\mathrm{new}}, \theta_{\mathrm{new}})$ の決定

$T(X_{\mathrm{new}}, C_{\mathrm{new}})$ に対して式(2.3)が成立するような r_{new} を求める．

Step 3 重み結合係数の決定

新しい学習パラメータに対して重み結合係数

$$w_i=\alpha_{\mathrm{new}} \sum_{x_t \in C^+} x_{ti}^{\mathrm{new}} - \beta_{\mathrm{new}} \sum_{x_t \in C^-} x_{ti}^{\mathrm{new}} \qquad (2.12)$$

を求める．

3. オブジェクト指向シミュレーションツールの開発：ニューロ・エージェントモデル

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．本章では，エージェントの自律行動様式をニューロ・エージェントモデルを用いて定式化する．

3.1 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の基本は，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り，エージェントの組織構造や制御メカニズムについてどのエージェントも把握していないことである．また，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・エージェントモデルの基本構成を**図3**に示す．

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2.2節で示したコネクショニスト

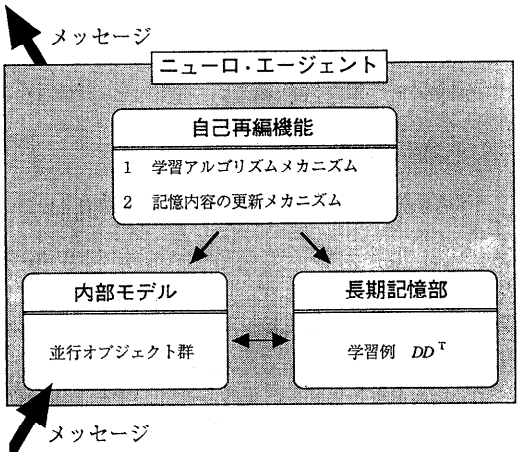


図3　ニューロ・エージェントモデルの構成
Fig. 3　Components of a neuro agent model.

学習により求める．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

$$F_j(x)=\sum_{i=1}^{n} w_{ij}x_i+\theta_j \qquad (3.1)$$

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 q_j を越えた場合には活性化する．

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数（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 $d_p=(d_{p1}, d_{p2}, \cdots, d_{pn})$ およびそれに対応して起動すべき内容を出力ベクトル $c=(c_{p1}, c_{p2}, \cdots, c_{pk})$ で表す．ニューロ・エージェントの初期の学習例の集合は，$\langle(d_p, c_p)\rangle$，$p-1, 2, \cdots, P$ 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 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 $C_{\bar{j}}$ 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，次のようなグループ化関数を定義する．

$$G_j^+(d_p)=\sum_{d\in C_j^+} S(d,d_p)$$
$$G_j^-(d_p)=\sum_{d'\in C_j^-} S(d',d_p) \qquad (3.2)$$

ここで $S(d,d_s)$ は，

$$S(d,d_p)=\sum_{i=1}^{n} d_i d_{pi}+\sum_{i=1}^{n}(1-d_i)(1-d_{pi}) \qquad (3.3)$$

で定義される学習例 d と d_p 間の類似測度である．C_j^+ は学習例の集合 $D=\{d_1,d_2,\cdots,d_n\}$ の中の正の学習例の集合を表す．すなわち，$G_j^+(d_p)$ は正の学習例全体との類似測度の総和を表し，正の学習例の特徴が集約化される．一方，$G_j^-(d_p)$ 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 d 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 $\{G_j^+(d),G_j^-(d)\}$ に対して定義される活性化ルール

$$G_j(d)=\alpha_j G_j^+(d)-\beta_j G_j^-(d)+\theta_j \qquad (3.4)$$

により判定する．

3.2 非線形識別問題の学習

本節では，(2.1)式で定義される類似行列 $T(D,C)$ に対し(2.2)式を満たすパラメータが存在しない場合について論じる．この場合は，学習例の集合 $\boldsymbol{D}$ が概念 C の下で線形分離不可能であり，したがっていくつかの中間オブジェクトが必要になる．オブジェクトの種類として(i)中間オブジェクトおよび(ii)出力オブジェクトを構築する．中間オブジェクトは，環境から入力情報または新しい学習例を，また出力オブジェクトは，中間オブジェクトからそれぞれのオブジェクトの活性化状態をメッセージ形式として受理する．中間オブジェクトは，出力オブジェクトに対する入力情報が線形分離不可能な場合，それを線形分離可能となるよう非線形変換を行うために導入され，中間オブジェクトは特定の出力オブジェクトに対し役割を果たすよう構造的に構築される．定理2.1より j 番目の出力オブジェクトの望ましい出力情報 C_j およびその補完情報 $1-C_j$ に対して求まる類似行列 $T(D,C_j)$ が線形分離可能ならば，j 番目の出力オブジェクトは入力情報を直接メッセージとして受理し，決められた望ましい出力情報をとるような活性化ルールを決定することができる．しかしながら，類似行列 $T(D,C_j)$ が線形分離不可能な場合にはいくつかの中間オブジェクトを用いて，入力情報をいったん変換することが必要である．中間オブジェクトを生成する方法として次のような手順で求める．

類似行列 $T(D,C_j)$ を線形分離不可能にしている学習例の集合を求める．すなわち，概念，C_j の下でのグループ化関数 $G^+(d_t)$，$G^-(d_t)$ の中で式(2.3)を満足しない学習例（一般性を失うことなくこれを正の学習例と仮定できる）d_s を抽出し，これを正の学習例から負の学習例へ分類変更するような $\bar{C}_j=C_j-H_s$ および新しい中間概念 H_s を生成する．ここで，H_s は s 番目の要素を1他を0とする列ベクトルである．グループ化関数は，

$$\bar{G}_j^+(x_t)=G_j^+(x_t)-(x_t,x_s)$$
$$\bar{G}_j^-(x_t)=G_j^-(x_t)+(x_t,x_s) \qquad (3.5)$$

で与えられ，概念 C_j の下で線形分離不可能となる学習例の集合 $\boldsymbol{D}$ は，新しい中間概念 $\bar{C}_j=C_j-H_s$ の下では線形分離可能な状態に近づく．以上のような手順で類似行列 $T(D,C_j)$ を線形分離不可能にしているいくつかの正の学習例 $Y=\{d_1,d_2,\cdots,d_{k-1}\}$ を抽出し，それらを局所表現した k 個の中間概念 H_s，$s=1,2,\cdots,k$ を生成する．すなわち，H_s は Y の j 番目の学習例 $d_s\in Y$ だけを唯一の正の学習例と識別し，他の学習例は負の学習例と分類する．すなわち，H_s は j 番目の要素が1で他は0となるような単位列ベクトルである．また，中間概念 H_k を次式で与える．

$$H_k=C-\bigcup_{s=1}^{k-1} H_s \qquad (3.6)$$

すなわち，中間概念 H_k は，線形分離不可能な類似行列 $T(D,C_j)$ の中からいくつかの正の学習例を抽出し，それを線形分離可能にするように選ばれた中間概念であり，したがって類似行列 $T(D,H_k)$ は線形分離可能になる．各中間概念 H_s，$s=1,2,\cdots,k-1$ はある一つの学習例を正の学習例および残りのすべての学習例を負の学習例と識別するもの（局所表現）であり，明らかに線形分離可能である．以上のような中間概念を定義することにより学習例 $\boldsymbol{D}$ はそれぞれの中間概念 H_s，$s=1,2,\cdots,k$ の下で線形分離可能である．また，概念 C_j は中間概念 $H=\{H_1,H_2,\cdots,H_{k-1}\}$ の選言形式で表現でき，したがって C_j はそれぞれの中間概念のとる値によって定義される中間概念の学習例 H に対し線形分離可能である．

このように生成された k 個の中間概念に対応して構築される中間オブジェクトは，入力情報を受け取り，受理した後の中間オブジェクトの活性化状態はそれぞれの出力オブジェクトに送られる．

4. 応用例：自己成長型多層ネットワークの構築

ニューロ・エージェントモデルを用いて，自己成長型多層ネットワークを構築した．本シミュレータは，オブジェクト指向言語 Smalltalk/V を用いて Macintosh 上で実現した．以下では，まず，Smalltalk/V 上でのインプリメント方法を簡単に示す．そして，**図4**に示すシミュレーション環境について説明する．

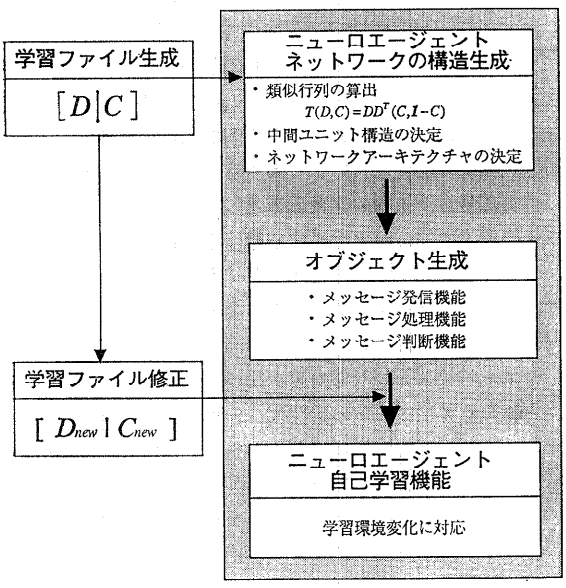


図 4 ニューロ・エージェントモデルの情報処理プロセス シミュレーションの流れ
Fig. 4 An information processing of a neuro-agent model.

4.1 Smalltalk/V 上でのクラス

以下に，作成したクラスの階層を示す．

Object
　NeuroAgent
　KnowledgeObject
　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．

● KnowledgeObject 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．

● NeuroExampleBuilder クラス

学習例の編集，ファイル管理およびユーザインタ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学習例を与え，シミュレーションを行う．

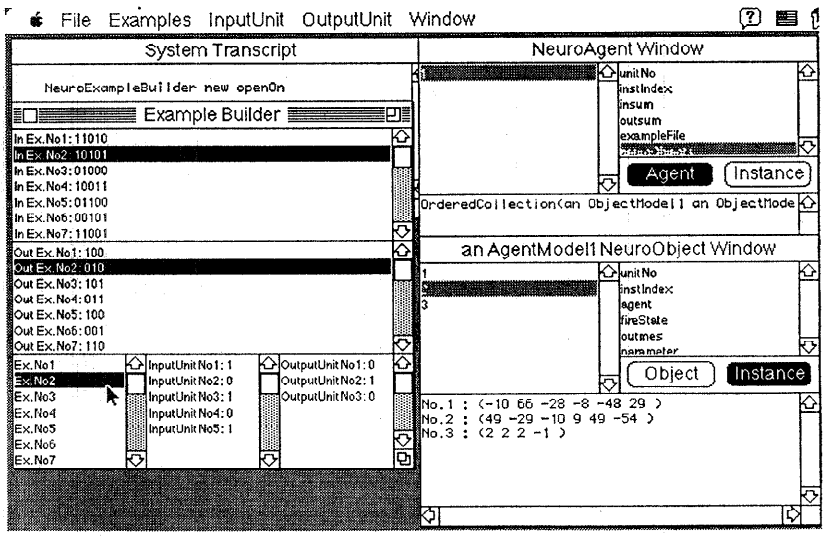


図 5 ニューロ・エージェントのシミュレーション実行例
Fig. 5 An example of simulation with a neuro agent model.

4.2 シミュレーション

図4に示した順序でシミュレーションを行う．以下簡単に説明を行う．最初に学習例を作成する．**図5**は，NeuroExampleBuilder に学習例を表示した状態である．このウィンドウでは，学習例ファイル操作，学習例の追加および出入力属性数を変更するユーザインタフェースウィンドウである．次にニューロ・エージェントを生成する．新しいニューロ・エージェントを生成して現在ウィンドウに表示している学習例を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，**図6**に示すように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，学習例が線形分離不可能な場合は，4.1 節で述べた中間オブジェクトを出力オブジェクトごとに生成する．学習が終了したニューロ・エージェントに入力パターンメッセージを送ると学習で獲得した活性化ルールによって内部状態を変化し System Transcript に内部状態を表示する．入出力パターンメッセージを送った場合は，学習メカニズムで獲得した活性化ルールによる内部状態がそのメッセージに反する場合は学習例として取り込み再度学習を行うが，反しない場合は学習例として取り込むだけで学習は行わない．ニューロ・エージェントは，自分自身の判断で入出力パターンのメッセージを取り込み，学習例の変化に対応していく．また，既存のニューロ・エージェントに学習例を与え再度学習させる場合は，ニューロ・エージェント固有の学習ファイルを修正しニューロ・エージェントに [update] メッセージを送ると新規に生成された場合と同様に動作する．

環境の変化に対応する例を示すための学習環境を**表1**に示す．初期の学習環境として $[D|C]$ を与え，以後表1に示したように学習環境を変化させニューロ・エージェントの内部状態の変化をシミュレーションした結果の一例が**表2**である．表2は，表1に示すそれぞれの組み合わせによる学習例を受理したニューロ・エージェントが内部モデルとして生成した中間および出力オブジェクトの活性化ルールのパラメータを示したもので多層ネットワークの結合係数に相当するものである．これらのパラメータは3章に示したニューロ・エージェントが具備する学習アルゴリズムが獲得したものである．表2において h_{31}, h_{32} は中間オブジェクトを表している．これは，初期の学習環境 $[D|C]$ において，望ましい出力ベクトル C_3 が線形分離不可能であることが判定され，そのために二つの中間オブジェクトが生成された．一方，望ましい出力ベクトル C_1, C_2 は，線形分離可能で中間オブジェクトは生成されていない．しかしながら，学習例が追加されることにより生成された新しい学習環境 $[DD_{new}|C]$ の下では C_3 は線形分離可能になり中間オブジェクトは消滅した．また新しく望ましい出力ベクトル C_4, C_5, C_6 を追加して生成される学習環境 $[D|CC_{new}]$ の下では，それぞれ生成された望ましい出力ベクトルに対応した3個の出力オブジェクトが新しく生成され，各出力オブジェクトごとに結合係数を獲得した結果が同じ表2に示されている．その他の場合の学習環境の変化に対しても表2に示したように学習例の変化に対応してニューロ・

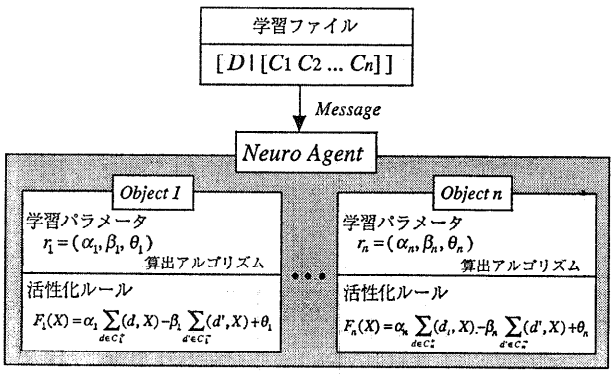


図 6 学習例からのエージェントの生成
Fig. 6 Generation of a neuro agent.

表 1 動的な環境下における学習例
Table 1 Training examples in a dynamic environment.

		D					D_{new}			C			C_{new}		
		x_1	x_2	x_3	x_4	x_5	z_1	z_2	z_3	c_1	c_2	c_3	c_4	c_5	c_6
d	d_1	1	1	0	1	0	1	0	0	1	0	0	1	0	1
	d_2	1	0	1	0	1	1	1	0	0	1	0	1	1	1
	d_3	0	1	0	0	0	0	1	0	1	0	1	0	1	0
	d_4	1	0	0	1	1	1	0	1	0	1	1	1	0	0
d_{new}	d_5	0	1	1	0	0	0	1	0	1	0	0	0	1	1
	d_6	0	0	1	0	1	1	0	1	0	0	1	1	0	1
	d_7	1	1	0	0	1	0	0	1	1	1	0	1	0	0

表 2 学習例の変化に伴いニューロ・エージェントが獲得した結合係数
Table 2 Obtained weight of neuro agent under a dynamic environment training examples.

学習例	出力オブジェクト	結合係数								
		$w1$	$w2$	$w3$	$w4$	$w5$	$w6$	$w7$	$w8$	θ
$[D \mid C]$	C_1	-12	24	-12	0	-24				12
	C_2	12	-24	12	0	24				-12
	C_3	2	2							-1
	h_{31}	-19	11	-3	-11	-11				4
	h_{32}	3	-11	-3	11	11				-18
$[D \mid C]$ ↓ $\left[\frac{D \mid C}{d_{new}}\right]$	C_1	-10	66	-28	-8	-48				29
	C_2	-49	-29	-10	9	49				-54
	C_3	2	2	2						-1
	h_{31}	-23	14	-14	-5	-23				-7
	h_{32}	13	-21	-13	29	13				-38
	h_{33}	-23	-23	23	-5	14				-51/2
$[D \mid C]$ ↓ $[DD_{new} \mid C]$	C_1	-12	24	-12	0	-24	0	0	0	12
	C_2	12	-24	12	0	24	0	0	0	-12
	C_3	-18	0	-20	0	0	-18	0	-18	27
$[DD_{new} \mid C]$	C_1	-18	38	-20	0	-38	-18	0	-20	46
	C_2	18	-38	20	0	38	18	0	20	-46
	C_3	-18	0	-20	0	0	-18	0	-18	27
$[D \mid C]$ ↓ $[D \mid C_{new}]$	C_4	19	-11	3	11	11				-4
	C_5	-10	0	10	-22	0				16
	C_6	2	2							-1
	h_{61}	3	11	-3	11	-11				-18
	h_{62}	3	-11	19	-11	11				-18

エージェントの内部状態を変化させ環境の変化に自律的に適応していく結果が得られた.

5. おわりに

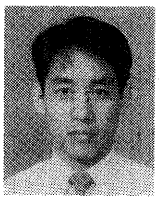
オブジェクト指向に基づくシミュレーションツールとしてニューロ・エージェントモデルを構築し，自己再編および自己成長機能を有するコネクショニストモデルについて提案をした．本モデルは自律的で協調的な並行オブジェクト群により構成され，学習メカニズムにより環境の変化に自律的に対応していくことが可能である．応用例として自己成長型多層ネットワークを構築し，環境の変化に対して自律的に自己の内部状態を変更し対応していくことを示した．ニューラルネットワークの規模の問題を解決するための方法として分散学習や複合化モデルが考えられる．これは，大規模な問題をいくつかの小さな問題に分割し，小さな問題について学習をした結果を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が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．情報統合上の加法性とは，それぞれの分散処理結果の重みづき線形総和により全体の解が得られる性質をいう．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を実現することが容易であ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 Intelligence*, Morgan Kaufmann (1988).

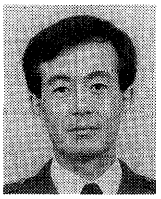
2) Marvin Minsky (安西祐一郎 (訳)) : 心の社会, 産業図書 (1990).
3) Barnden, J. A. and Pollack, J. B. (eds.): *High-Level Connectionist Models*, Ablex Publishing (1991).
4) Chen, J. R.: A Compositional Connectionist Architecture, *Proc. of the 1990 Connectionist Summer School*, pp. 188-197, Morgan Kaufmann (1990).
5) Demazeau, Y. and Müller, J. P. (eds.): *Decentralized AI*, Elsevier Science Publishers (1990).
6) Hinton, G. E. (ed.): Special Issue on Connectionnist Symbol Processing, *Artif. Intell.*, Vol. 46, No. 1-2 (1990).
7) Lee, R. C.: *Structure Level Adaptation for Neural Systems*, Kluwer Academic Press (1991).
8) Nadal, J.: Study of a Growth Algorithm for a Forward Networks, *Int. J. of Neural Systems*, Vol. 1, pp. 55-59 (1989).
9) Namatame, A. and Tsukamoto, Y.: Structural Connectionist Learning with Complementary Coding, *Int. J. of Neural Systems*, Vol. 3, pp. 19-30 (1992).
10) Reiley, D. L., Copper, L. N. and Wlbaum, O. C.: A Neural Model for Category Learning, *Biological Cybernetics*, Vol. 45, pp. 35-41 (1982).
11) Touretzky, D. S. (ed.): Special Issue on Connectionist Approaches to Natural Language Learning, *Machine Learning*, Vol. 7, No. 2-3 (1991).
12) Vaario, J. and Ohusuga, S.: Developing Neural Networks through Growth, 人工知能学会全国大会論文集, pp. 359-362 (1991).

(平成4年2月20日受付)
(平成5年2月12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年防衛大学校理工学研究科 (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助手. 日本ソフトウェア科学会, 人工知能学会各会員.



生天目 章 (正会員)

1973年防衛大学校卒業 (応用物理学専攻). 1979年スタンフォード大学大学院博士課程修了 (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, 意志決定工学などの研究に従事. 人工知能学会, 計測自動制御学会, AAAI, ACM, INNS, IEEE 各会員.