知識工学と人工知能 66 － 2
（１９８９．９．２２）

Prologとニューラルネットワークの結合システム Neuro-Prolog

今中　武　　上原邦昭　　豊田順一
大阪大学産業科学研究所

本稿では、Prologインタプリタ C-Prologとニューラルネットワークシミュレータ SunNetの結合システム Neuro-Prologについて述べる。Neuro-Prologは知的システムの開発環境を向上させるために、論理的情報と感性のような論理表現できない情報を1つのシステム内で同時に利用できるようにしたシステムである。Neuro-Prolog内では論理的情報を扱うために Prologが用いられ、感性情報など非論理的情報を扱うためにニューラルネットワークが用いられる。両者は互いに情報を交換しながら処理を行うために、Prologを用いた論理的情報処理に感性情報を取り込んだり、ニューラルネットワークを用いた感性情報処理に Prolog表現された知識を導入するなどの処理が可能となる。

また、本稿では Neuro-Prologの有効性を確認するために開発した実験システムについても述べる。実験システムは、「暖かい」など色の持つイメージを表した自然語からイメージに合う色出力を行うものである。実験システムでは、イメージを表す自然語と色出力に用いる RGB値の対応付け部分をニューラルネットワークを用いて実現しており、一度出力した色に対して「もう少し暖かい色」などイメージの量を変化させることによって色の修正ができるようになっている。

An Integration of Prolog and Neural Networks to Deal with Sensibility in Logic Programs

Takeshi IMANAKA, Kuniaki UEHARA and Jun'ichi TOYODA
The Institute of Scientific and Industrial Research
Osaka University, 8-1 Mihogaoka, Ibaraki 567, Japan

We have developed Neuro-Prolog which integrates a Prolog interpreter C-Prolog and a neural network simulator SunNet. By use of Neuro-Prolog, users can handle both definite information like logical rules and mutable information like sensibility concurrently in a single system. In Neuro-Prolog, the definite information is handled in Prolog and mutable information is handled in neural networks. The two systems Prolog and a neural network communicate with each other by execution of additional built-in predicates which drive neural networks from Prolog programs in Neuro-Prolog. In contrast with this, in conventional systems, only either one of these two different types of information can be handled at one time.

Furthermore, we have also developed an example system by use of Neuro-Prolog to make sure that advantages of Neuro-Prolog are still effective in real world problems. The example system computes RGB values from image words and outputs colors on CRT for the computed RGB values. For example, "very warm, a little delight and very lovely color" could be output by this system. In this system, a neural network which is set up based on statistics is used to compute RGB values from image words. Prolog rules which represent common sense like "If a certain part of picture means the sky, the part can not be painted in a green color" is applied to decide colors to paint. Furthermore, the example system could output reasonable colors even for very flexible inputs like "much more remarkable and less lovely color" containing mutable features.

1 はじめに

近年の人工知能分野ではシステムの問題解決能力だけでなく、利用者にとっての使いやすさが重要視され、より人間に親しみやすいインタフェースの開発が多数行われている。このようにインタフェースを人間にとって親しみやすいものにする場合、「美しい」などの形容詞で表される人間特有の感性に関する情報(以降では感性情報と呼ぶ)を扱う必要がある。しかしながら、感性情報には①どの程度「美しい」かなどを表す感性の量(以降では感性量と呼ぶ)が存在し、「より美しい」というように感性量は比較可能な連続値となる、②感性量が連続であるために、感性情報を含んだ処理は有限個のパターンで表せない、③感性量を入出力に持つ関数を、あらかじめ固定的に決定することは困難であるなどの特徴があり、従来の論理的情報処理の枠組みで感性情報を処理することは困難であった。さらに、感性情報の処理を現実の問題に適用するには論理的情報も同時に必要となる。たとえば、「暖かい」などのように色のイメージを表す自然語(以降ではイメージ語と呼ぶ)でポスターに塗る色が指定され、実際に塗るべき色を決定する場合を考える。この場合、暖かさの程度などの感性量の扱いに加え、配色などの決定には「重要な部分ならば他の部分に比べて派手に塗るべきである」といったような論理的知識の適用が必要である。したがって、感性情報処理には、感性情報が持つ上記の3つの特徴を上手く扱い、かつ論理的情報と同時に利用できるような枠組みが必要となる。

このような枠組みとして、我々はニューラルネットワークとPrologの結合システムNeuro-Prologを開発した。Neuro-Prologは、Prologインタプリタ C-PrologとニューラルネットワークシミュレータSunNetを結合したシステムで、現在ワークステーションSun-3/260上で稼動中である。Neuro-Prologでは、ニューラルネットワークを利用して感性情報などの非論理的情報を扱い、Prologを用いて論理的情報を扱う。ニューラルネットワークには、①0から1までの連続値を入出力値として扱うことができる、②いくつかの入出力パターンを設定すれば、未設定の入力パターンに対しても設定済みの入出力パターンに基づき妥当な出力ができる、③入出力パターンの設定のみを行えば、入出力関係をあらかじめ明確な形式で定義しなくてもよいなどの利点がある。したがって、感性量を0から1の実数値で表せば、感性情報の持つ上記の特徴を上手く扱うことができ、ニューラルネットワークを用いて感性情報処理を行うことが可能である。また、Neuro-Prologの構成は、Prologインタプリタ上にニューラルネットワークを操作するための組み込み述語を新たに実装したものであり、推論途中で組み込み述語を実行すれば、Prologの推論結果をニューラルネットワークの処理に、ニューラルネットワークの処理結果をPrologの推論に利用することができる。

さらに、我々はNeuro-Prologの有効性を確認するために、CRT上でポスターに色を塗る実験システムを開発した。この実験システムは、「非常に暖かい」などのように、イメージ語「暖かい」と、どの程度「暖かい」のかを表すイメージの量(以降ではイメージ量と呼ぶ)を示す「非常に」などの言葉を入力とし、これらの入力とProlog表現された常識知識を用いてポスターの各部分に塗るべき色を決定する。たとえば、ポスターの最重要部分に対して「非常に鮮やかな色」といったイメージ語が入力された場合、Prolog表現された知識を用いて重要部分は目立たせる必要があると判断し、「非常に鮮やかな色で、目立つ色」を塗るべき色のイメージとして結論づける。塗るべき色のイメージが得られると、ニューラルネットワークを用いて、これらのイメージに合う色のRGB出力値を求め、画面上に色を出力する。本実験システムでは、イメージ量を実数値で扱っており、出力色に対して利用者が「さらに目立つ色」などの修正を行った場合、「目立つ」といったイメージのイメージ量を増加させて再計算し、修正された色を出力することができる。また、本実験システム内に実装しているニューラルネットワークは、[川添1987]、[小林1988]で示されている統計データを用いて初期設定しており、イメージ語入力に対する出力色は統計データに基づいたものとなっている。

2. Prologとニューラルネットワーク

Prologの推論機能は規則と事実を用いた後向き推論に基づいているために、推論結果に対する説明機能を容易に実現することができる。たとえば、以下のPrologプログラムに対して、tomの祖母は誰かといった問い合わせを意味するゴール"grandmother(tom,Z)"を実行すれば、結論"Z = mary"が得られる。

```
grandmother(X,Z):- mother(X,Y), mother(Y,Z).
mother(tom,jane).
mother(jane,mary).
```

この結論に対し、起動されたルールをもとに、以下のように推論過程について簡単な説明を行うことができる。

tom の mother は jane です。
jane の mother は mary です。
mother の mother は grandmother です。
したがって、tom の grandmother は mary です。

一方、Prologには、感性量などのように連続値で、かつ入出力関数が固定的に定まらない情報を上手く扱う能力がないという欠点がある。たとえば、「暖かい」イメージ持っている色のRGB値の計算や、「より暖かい」色といった、イメージ量の増加に伴うRGB値の変化を計算するなどの処理を行うことは困難である。さらに、これらの計算には「非常に暖かく、かつ男性的な」イメージを持つ色といったようにイメージやそれぞれのイメージに対するイメージ量が複雑に関わり合うなどの特徴があり、新たに確信度などの手法を導入した場合でも容易ではない。

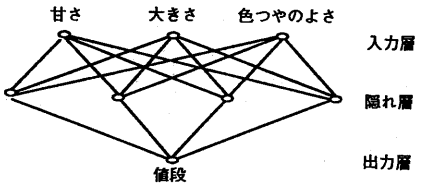

図1　果物の値段を決定するニューラルネットワーク

以上のようなPrologの持つ特徴に対し、ニューラルネットワークには連続値をとる入力を扱う機能、いくつかの入出力のパターンを設定すれば未設定の入力パターンに対しても妥当な値を出力する汎化機能などがある[Rumelhart 1986]。このようなニューラルネットワークの特徴を説明するために、果物の値段を作柄に応じて決定する例を用いる。まず、図1に示すような入力層、隠れ層、出力層の3層で構成されているニューラルネットワークを考える。入力層の各ユニットには「甘さ」、「大きさ」、「色つやのよさ」が割り当てられる。このネットワークに対して、以下の4つの値段決定例を設定する。

甘さ	大きさ	色つやのよさ	値段
0.9	0.9	0.9	0.9
0.1	0.1	0.1	0.1
0.5	0.3	0.9	0.4
0.9	0.6	0.1	0.8

これらの設定は、値段の決定に「甘さ」が最も重要であるという事実を暗に示している。ニューラルネットワークは、上記の4つの決定例が設定されるとバックプロパゲーションメカニズムを用いてこの事実を自動的に学習し、"(甘さ、大きさ、色つやのよさ)=(0.8, 0.2, 0.1)"といった未定義の入力パターンに対しても、「甘さ」を最重要視して妥当な値段を出力することができる。さらに、計算結果が好ましくない場合には、新たな入出力パターンを追加して、ネットワーク設定を動的に修正することもできる。このような値段の決定をあらかじめ定義した固定的な関数で行うことは、①入力される要因が3個以上に増加し、それらが複雑に関わり合った場合に関数が複雑になりすぎ、関数の定義自身が困難である、②値段決定は例年同じではなく、関数を動的に修正しなければならないなどが原因で現実的ではない。以上のように、ニューラルネット

ワークは入出力が連続値で、すべての入出力パターンを予め設定できない、入出力関数をあらかじめ固定的に決定できないなどの特徴を持つ問題に対して有効である。

しかしながら、ニューラルネットワークはPrologに比べ計算過程を説明することが困難であるといった欠点があり、個々の知識を組み上げて1つの結論を得るような論理的情報の処理には不適当である。したがって、説明可能な論理的情報はPrologで扱い、論理的に説明することが本質的に不可能な感性情報などはニューラルネットワークで扱うことが合理的である。たとえば、果物の値段を決定する例では単純に作柄に関する3つの要因のみから値段を決定するだけでなく、実際には最低限の値段、原価率、利益計算などの論理的に表現可能な情報と合わせて総合的に値段の決定を行うことが合理的である。以上のように、Prologとニューラルネットワークがそれぞれに持つ利点は同時に必要となるものであり、それぞれが持つ欠点は互いに補い合うものである。すなわち、両者を結合することによって有用なシステムを作成することが可能となる[Imanaka 1989]。

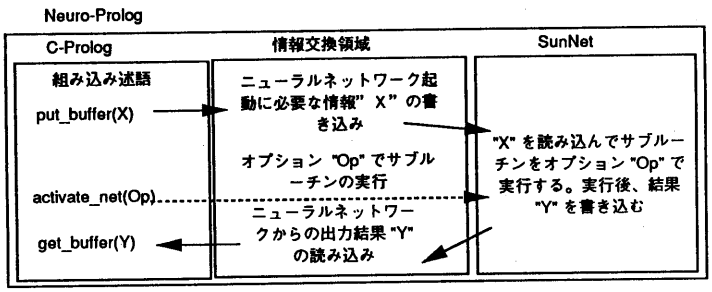

図2　Neuro-Prologのシステム構成図

3. Neuro-Prologの実現手法

3.1 Neuro-Prologの構成

Neuro-Prologのシステム構成図を図2に示す。Neuro-Prolog内では、Prologとニューラルネットワークの間での情報交換は、すべて情報交換領域を介して行われる。また、情報交換領域におけるデータの書き込みや、読み取り、ニューラルネットワークの起動などはすべてNeuro-Prolog内に新たに設定した組み込み述語を用いて行われる。現在、Neuro-Prolog中で使用可能なニューラルネットワーク操作用の組み込み述語は約10種類あり、最も重要な以下の3つの述語について説明する。

1. put__buffer(Altribute,X)
2. activate__net(Op)
3. get__buffer(Attribute,X)

"put__buffer(Attribute,X)"は、引数"X"に代入された情報を図2に示した情報交換領域に書き込むための述語である。情報"X"は各層のユニット数や入出力パターンなど用い方によって多様な属性を持つために、引数"Attribute"の値によって"X"の属性を指定する。"activate__net(Op)"は、引数"Op"に代入された値によって、ネットワークの作成、設定、計算を行うための述語である。"get__buffer(Attribute,X)"は、情報交換領域に書き込まれた計算結果などの情報を引数"X"に代入する述語である。引数"Attribute"は"put__buffer"と同様の用い方をする。各述語はPrologプログラム中に自由に埋めこむことができ、これらの述語をPrologプログラムの実行中に呼び出すことによって、ニューラルネットワークの起動や情報交換領域を介した情報の受け渡しを行うことができる。

3.2 Neuro-Prologを用いたニューラルネットワークの操作

前節で示した3種の述語"put__buffer(Attribute, X)", "activate__net(Op)", "get__buffer(Attribute, X)"を用いてニューラルネットワークの作成、初期設定を行った後に、ニューラルネットワークを用いて計算を行う操作を順に説明する。まず、ニューラルネットワークを新たに作成するために以下の述語を順に実行する。

```
put__buffer(input__layer, Number__of__input__units).
put__buffer(output__layer, Number__of__output__units).
put__buffer(hidden__layer, Number__of__hidden__units).
activate__net(cons__net).
```

述語"put__buffer(input__layer, Number__of__input__units)", "put__buffer(output__layer, Number__of__output__units)", "put__buffer(hidden__layer, Number__of__hidden__units)"は、それぞれ入力層、出力層、隠れ層に定義するユニットを第2引数で指定し、その内容を情報交換領域に書き込むためのものである。述語"activate__net(cons__net)"は、述語"activate__net"をオプション"cons__net"で実行するもので、情報交換領域に設定されたユニットを各層に定義し、新たなネットワークを作成する述語である。

次に、作成したネットワークに初期設定を行う操作を示す。初期設定はニューラルネットワークのバックプロパゲーションメカニズムを用いて行っており、いくつかの入出力パターンを用いて設定すれば、ニューラルネットワークが設定された入出力関係を計算できるように自動的に学習する。Neuro-Prologでは、以下の述語を順に実行すれば、初期設定が行える。

```
put__buffer(input, Ideal__input__pattern__list).
put__buffer(output, Ideal__output__pattern__list).
activate__net(add__pattern).
activate__net(initial__learn).
```

"put__buffer(input, Ideal__input__pattern__list)", "put__buffer(output, Ideal__output__pattern__list)"は、入出力パターンをリスト表現した引数"Ideal__input__pattern__list", "Ideal__output__pattern__list"を情報交換領域に書き込む述語である。"activate__net(add__pattern)"は情報交換領域に書き込まれた入出力パターンをニューラルネットワークのパターンファイルに登録するための述語で、述語"activate__net"をオプション"add__pattern"で実行するものである。パターンファイルとは、ネットワークに設定すべき理想的な入出力パターンをすべて保存するファイルであり、ネットワークの設定はすべてこのファイル中のパターンを用いて行われる。また、複数のパターンを設定する場合は、以上の2つの述語の実行を繰り返し、すべてのパターンをパターンファイルに蓄積する。述語"activate__net(initial__learn)"は、パターンファイルに設定された入出力パターンを用いてネットワークを設定するためのもので、述語"activate__net"をオプション"initial__learn"で実行するものである。

最後に、設定の完了したネットワークを用いて計算を行う操作を示す。計算は入力パターンを与えてネットワークを起動し、出力パターンを得るものである。Neuro-Prologでは、以下の述語を順に実行する。

```
put__buffer(input, Input__pattern__list).
activate__net(activate).
get__buffer(output, Output__pattern__list).
```

述語"put__buffer(input, Input__pattern__list)"は、入力パターンを情報交換領域に書き込むものである。述語"activate__net(activate)"は、情報交換領域に書き込まれた入力パターンからニューラルネットワークを用いて出力パターンを計算し、結果を情報交換領域に書き込むものである。この述語は、述語"activate__net"をオプション"activate"で起動するものである。述語"get__buffer"は、情報交換領域に書き込まれたニューラルネットワークの出力パターンを読み取り、変数"Output__pattern__list"に結果を代入するものである。

以上の操作を用いて2章の例をNeuro-Prologで実現するプログラムを以下に示す。

```
start :-
    put__buffer(input__layer, [sweetness, size, color]),    /*入力層のユニット数と各ユニットの名前の指定*/
    put__buffer(output__layer, [price]),                    /*出力層のユニット数と各ユニットの名前の指定*/
    put__buffer(hidden__layer, [h1, h2, h3, h4]),           /*隠れ層のユニット数と各ユニットの名前の指定*/
    activate__net(cons__net),                               /*ネットワークの作成          */
    put__buffer(input, [0.9, 0.9, 0.9]),                    /*設定用入力パターン          */
    put__buffer(output, [0.9]),                             /*設定用出力パターン          */
    activate__net(add__pattern),                            /*パターンファイルに蓄積      */
    put__buffer(input, [0.1, 0.1, 0.1]),                    /*設定用入力パターン          */
    put__buffer(output, [0.1]),                             /*設定用出力パターン          */
    activate__net(add__pattern),                            /*パターンファイルに蓄積      */
    put__buffer(input, [0.5, 0.3, 0.9]),                    /*設定用入力パターン          */
    put__buffer(output, [0.4]),                             /*設定用出力パターン          */
    activate__net(add__pattern),                            /*パターンファイルに蓄積      */
    put__buffer(input, [0.9, 0.6, 0.1]),                    /*設定用入力パターン          */
    put__buffer(output, [0.8]),                             /*設定用出力パターン          */
    activate__net(add__pattern),                            /*パターンファイルに蓄積      */
    activate__net(initial__learn).                          /*ネットワークの初期設定      */
```

述語"start"を実行すると、図1のネットワークが作成され、初期設定も行われる。初期設定が完了した時点でニューラルネットワークを用いた計算が可能になり、入力パターン(甘さ、大きさ、色つやのよさ)=(0.5, 0.6, 0.9)に対する値段を知りたければ以下のゴールを実行し、変数"X"に値段を求めることができる。

```
:-  put__buffer(input, [0.5, 0.6, 0.9]),
    activate__net(activate),
    get__buffer(output, X).
```

4. Neuro-Prologを用いた実験システム

4.1 システムの概要

我々の開発した実験システムは入力された線画を指定色で塗り、CRT画面上に出力するものである。この実験システムは、色の指定をイメージ語を用いて行える、イメージ量を変化させながら色の修正ができるなどの特徴を持つ。たとえば、「非常に派手で、少し女性的な色」と指定することができ、出力された色が満足できるものでなければ、「もっと派手に」などの入力を行って色を修正することが可能である。線画の入力はマウスを用いて行えるようになっている。また、本システムでは線画中にある各部分が何を意味しているのかも入力として受け付ける。たとえば、「部分Aは家を表わす」、「部分Bは空を表わす」などが入力される。このように各部分の意味を入力することにより、予めシステム内に蓄積した「空を表わす部分は、緑色には塗らない」等のPrologのルールを適用することができ、非常識な色使いなどをしないようになっている。さらに、Prologのルールを適用して塗るべき色を決定した場合は、塗った色に対する説明を行うことができる。たとえば、空を意味する部分の色を決定する際に上記のルールを適用したならば、2章で示したように適用ルールを自然言語風に示して塗った色に対する説明とする。

4.2 イメージ語からの色出力

本実験システムは、入力が与えられると、イメージ語間の距離を定義した図3のイメージスケールに基づいて入力イメージを基本イメージ量に変換する。基本イメージ量は11個の0から1までの実数値の並びであり、各実数値は11個の基本イメージ語「暖かい」、「明るい」、「目立つ」、「軽快な」、「親しみやすい」、「派手な」、「上品な」、「若々しい」、「男性的な」、「陽気な」、「好きな」に対応するイメージ量を表している。ただし、基本イメージ量の各実数値は、0.5を基準としており、0.5未満の数値は逆のイメージを持つことを表す。たとえば、基本イメージ量(0.1, 0.4, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.9, 0.5, 0.5)は、「暖かい」、「明るい」に対するイメージ量が0.5より小さいためにそれぞれ逆のイメージを持ち、非常に冷たく、少し暗い色で、「男性的な」に対するイメージ量が大きいために、非常に男性的な色を表す。また、11個の基本イメージ語は川添、千々岩が色の持つイメージを統計データとして集めた際に用いたイメージ語である[川添1987]。基本イメージ量の計算は、入力されたイメージ語と各基本イメージ語とのイメージ

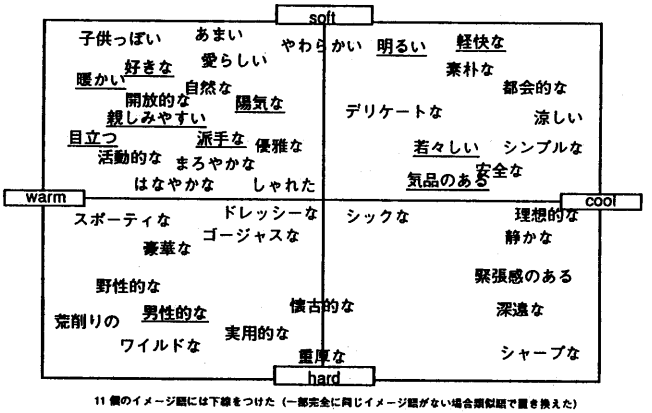

図3　イメージスケールの一部分([小林 1988]より引用)

スケール中における距離の近さを0から1までの実数値で表したものである。たとえば、入力に「開放的な」といったイメージ語が与えられると距離が近い基本イメージ語、「暖かい」、「好きな」、「親しみやすい」、「目立つ」、「軽快な」、「派手な」、「陽気な」に対するイメージ量は大きな値、距離が遠い基本イメージ語「明るい」、「若々しい」、「気品のある」、「男性的な」に対するイメージ量は小さな値になる。このような基本イメージ量の計算は、イメージスケール中の各イメージ語間の距離で初期設定した図4のニューラルネットワークを用いて行っている。図4のニューラルネットワークでは、イメージスケール中のすべてのイメージ語が入力層の各ユニットに割り当てられ、11個の基本イメージ語が出力層の各ユニットに割り当てられる。また、隠れ層のユニット数は入力層のユニット数と同数にしている。このように、ニューラルネットワークを用いて基本イメージ量の計算を行っているために、「非常に豪華で少ししゃれた」色といったように複数のイメージ語と、それぞれのイメージ量が同時に入力された場合でも、各入力イメージに対応する入力ユニットの値を「豪華な」は大きな値0.8、「しゃれた」は小さな値0.4に設定するのみで、複数のイメージ語に対する基本イメージ量を計算することができる。ただし、本システム

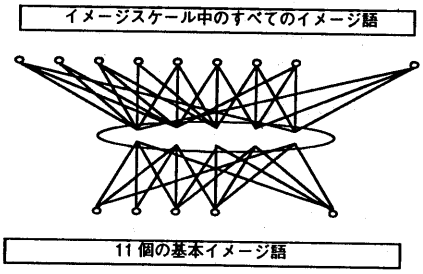

図4　基本イメージ量の計算

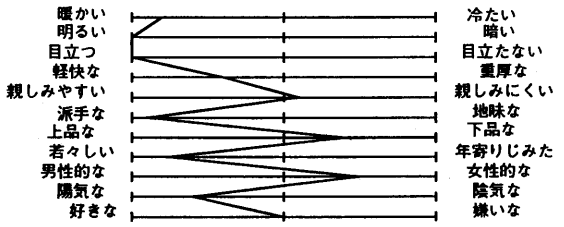

図5　色のイメージに対する統計データ([川添 1987]より引用)

では実験システムであることを考慮して、入力されたイメージ語が程度を表す言葉で修飾されている場合、「非常に」は0.8に、「少し」は0.4に、修飾されていない場合、0.5に設定するといった単純な処理でイメージ量の入力を扱っている。また、入力されなかったイメージ語のイメージ量は0に設定される。これは、基本イメージ量では、0.5が基準となっているのに対し、入力イメージ量では0が基準となっているためである。

以上の操作により、入力されたイメージ語に対する基本イメージ量が図4のニューラルネットワークで計算される。基本イメージ量が計算されると、図5の統計データに基づいて初期設定した図6のニューラルネットワークを用いてRGB値を計算する。図5の統計データは、マンセル表色系で7RP 5/13のピンク色に対するイメージの統計データであり、川添、千々岩によって合計30色に対して同様のデータが求められている。RGB値を計算するニューラルネットワークの構成は図6のように11個のイメージ語を入力層の各ユニットに、RGB出力値を出力層の各ユニットに割り当てたものである。隠れ層のユニット数は入力層と同じ11個にしている。本実験システムでは、30色すべてを用いて図6のニューラルネットワークを初期設定した。

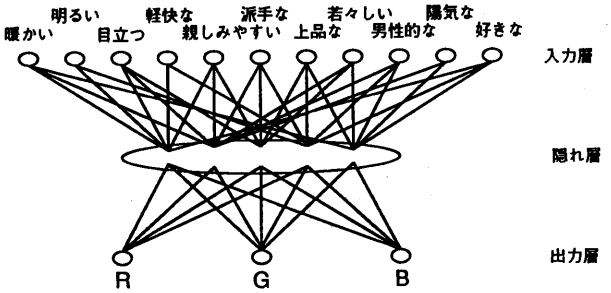

図6　基本イメージ量からの色出力

以上のように、入力されたイメージ語に対するRGB値の計算をニューラルネットワークで行っているために、一度出力した色に対して、イメージ量の変更で色の修正ができる。たとえば、「もう少し豪華に」を意味する入力が与えられると、図4のニューラルネットワークの「豪華な」に対応する入力ユニットの値をさらに大きな値に変化させて、基本イメージ量を再計算することができる。基本イメージ量が計算されると、図6のニューラルネットワークを用いて修正された色のRGB値を求め、修正した色を出力することができる。

4.3 実行例

実験システムを以下の入力に対して実行した例を示す。ただし、実際には線画はマウス入力され、他の入力はすべてPrologのファクト形式で入力される。たとえば、「部分Aは、駐車場の方向を意味する」といった入力は"means(partA, direction)"と入力される。

線画: 図7に示すようなポスター
各部分の意味: 部分Aは、駐車場の方向、部分Bは駐車場という名称
色の指定: 部分Aは、非常に自然で、少し高級な感じで、少し涼しい色
部分Bは、非常に子供っぽくて、少し幻想的で鋭い色

まず、実験システムは部分Aについて図4のニューラルネットワークの入力パターンを決定する。「非常に」が、0.8に、「少し」が0.4に変換され、「自然な」、「高級な」、「涼しい」に対応する入力ユニットの値がそれぞれ(0.8, 0.4, 0.4)に設定される。さらに、Prologには、以下のような規則があらかじめ蓄えられており、部分Aに塗るべき色をPrologを用いて求めるために、ゴール"color(partA, X)"が実行される。ゴールが実行されると、ルール(R1)、(R2)が起動されて変数"X"には"[remarkable]"が代入される。この結果、先の3つの入力ユニット以外に「派手な」に対応する入力ユニットも大きな値に設定される。

```
(R1) color(X,A) :-                    /* 部分Xに塗るべき色のイメージがCに代入される */
         bagof(C,colors(X,C),A).
(R2) colors(X,A) :-                   /* 最も重要な部分Xは、派手な(remarkable)色に */
         most_important(X),           /* 塗るべきである */
         A = remarkable.
(R3) colors(X,A) :-                   /* 空を意味する部分は、緑色や紫色には塗らない */
         means(X,the_sky),
         A = [not(green), not(purple), ...].
(R4) most_important(X) :-             /* 方向を表す部分最重要部分である */
         means(X,direction).
(R5) most_important(X) :-             /* 値段を表す部分は最重要部分である */
         means(X,price).
(R6) most_important(X) :-             /* 電話番号を表す部分は最重要部分である */
         means(X,phone_number).
```

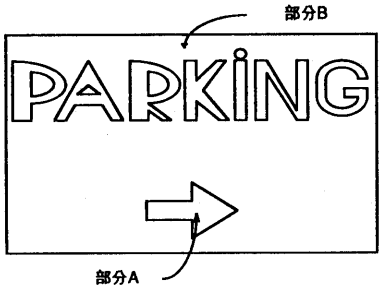

図7　入力線画

以上のようにして、図4のニューラルネットワークの入力パターンが決まり、ネットワークが起動されて、基本イメージ量が求められる。求められた基本イメージ量は、さらに図6のニューラルネットワークの入力層に移され、図6のニューラルネットワークが起動される。図6のニューラルネットワークが起動されて、RGB値が出力層に求められると、部分Aのみ色が塗られたポスターが画面上に示される。この時、出力色に対して説明を求れば、起動したPrologのルールを用いて以下のように説明が行われる。

The part A means the direction of the parking.
The direction is the most important part of this picture.
The most important part of the picture should be painted remarkable.

さらに、出力色に対して「さらに涼しい色」等の修正を行えば、図4のニューラルネットワークの「涼しい」に対する入力ユニットの値を少しずつ増加させ、出力色の修正ができる。以上のようにして、部分Aの色が満足できるものになれば、次の部分について同様の処理を行う。

5. まとめ

本稿では、Prologとニューラルネットワークの結合システムNeuro-Prologと、Neuro-Prologを用いて開発した実験システムについて述べた。Neuro-Prologを用いることによって、論理的情報と感性などの非論理的情報を1つのシステム内で同時に利用することができた。さらに、このような特徴は実験システムで示したように現実の問題に当てはめることもできる。特に、実験システムで用いている図6のニューラルネットワークにおいては、30色のみを用いた初期設定から基本イメージ量の変化によって、初期設定にはない色を統計データに基づいた方法で出力できるようになっている。現在、我々は実験システムの改良を行っており、以下のような点をさらに考慮する予定である。

1. 入力のイメージ量を変化させて出力した色と入力イメージのずれを調べる統計処理を行う
2. 30色以外の色に対する統計データを収集する
3. 配色の持つイメージを考慮する(2色、3色の組み合せから生じるイメージ)
4. 現在は、「非常に」などの程度を表す言葉は単純に数値に変換しているが、このような程度を表す言葉自体も感性情報と見なすことができ、ニューラルネットワークを用いて扱う方法を検討する
5. 色見本とCRT画面上の出力色を見比べて、統計データ[川添1987]に示されている30色のRGB値を求めているが、さらに厳密にRGB値を求める

参考文献

[Imanaka 1989] Imanaka, T., Fukuda, H., Uehara, K. and Toyoda, J.: "An Integration of Prolog and Neural Network to Deal with Ambiguity in Analogical Reasoning", IEA/AIE-89, p.1096 (1989).

[川添1987] 川添泰宏, 千々岩英彰: 色彩計画ハンドブック, 視覚デザイン研究所 (1987).

[小林1988] 小林重順(監): カラー・イメージ事典, 講談社 (1988).

[Rumelhart 1986] Rumelhart, D.E., McClelland, J.L. and PDP Research Group: "Parallel Distributed Processing", The MIT Press (1986).