ニューラルネットからの並列ルール抽出

新居 学*　荻野 康介*　坂部 智一*

220 Y0/prext

ライブラリー名： prext

概要： 並列ルール抽出プログラムは，パターン識別問題に対して学習済みのニューラルネットから言語的識別ルールを抽出するプログラムである．

概要形式：コンプリートプログラム

作成年月日：2002年11月

使用言語：C

作成者：新居 学，荻野 康介，坂部 智一

計算方法：ファジィ演算，区間演算，ニューラルネット

計算対象：パターン識別問題に対して学習済みのニューラルネット

1　はじめに

多層階層型ニューラルネットは関数近似やパターン識別等の分野で広く利用されている．ニューラルネットは未知データに対する汎化能力に優れる等の長所を持つが，出力値に対する説明能力の欠如という問題点がある．この問題に対して，種々のネットワーク構造や学習アルゴリズムが提案されている [1–6]．しかし，これらの手法は既に一般的な構造のニューラルネットが利用されている場合には適用することができず，学習用データを用いてシステムを再構築しなければならない．これに対して，一般的な学習後のニューラルネットからルール抽出を行う手法が提案されている [7]．この手法では，学習済みのニューラルネットに何ら変更を加えることなく入出力関係を説明するルールを抽出することができる．しかし，ルールを抽出するためにファジィIf-Thenルールの条件部ファジィ集合の組合せを全て入力する必要があり，高次元問題に対して学習済みのニューラルネットへの適用は困難である．この問題に対処するため並列ルール抽出手法を提案している [9]．並列ルール抽出プログラムは，ルール抽出手法のアルゴリズムを変更することなく並列化したものである．

並列ルール抽出プログラム（Parallelized Rule Extraction：PREXT）は，学習後のニューラルネットの入出力関係を説明するファジィIf-Thenルールを生成するコンプリートプログラムである．本プログラムは，一般的な多層階層型ニューラルネットの入出力関係を以下のようなファジィIf-Thenルールで表現することができる．

*にい まなぶ，おぎの こうすけ，さかべ ともかず（姫路工業大学 大学院 電気系工学専攻）

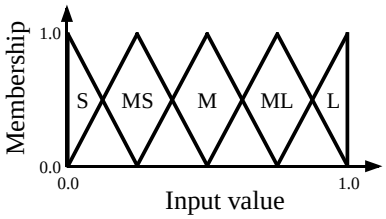


図 1: 言語値のメンバーシップ関数.

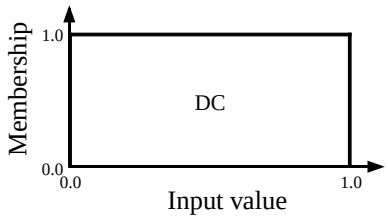


図 2: “ *Don't Care* ”のメンバーシップ関数.

Rule R: If x_1 is *small* and x_2 is *large*

then Class 1 with $CF = 0.9$

たとえば，このルールは，データの属性x_1が*small*という値を持ち，属性x_2が*large*という値を持てばそのデータはクラス１に識別され，そのときの確信度が0.9であることを示している．

本プログラムにより，ブラックボックス型のシステムの入出力関係が言語的に表現可能となるため，たとえばその分野のエキスパートによる検証等が可能となる．

2 計算方法

2.1 ファジィIf-Thenルール

学習後のニューラルネットの入出力関係を言語的に表現するために，本プログラムではファジィIf-Thenルール形式を使用する．

ニューラルネットが対象とする問題をn次元cクラスパターン識別問題とする．このとき，パターン空間はn次元超立方体$[0,1]^n$に規格化されていると仮定する．

Rule R: If x_1 is A_1 and $\cdots$ and x_n is A_n

then Class C with CF (1)

ここで，A_i, $i = 1,2,\cdots,n$は言語値を表し，図1のメンバーシップ関数により表現されるファジィ数である．本プログラムでは，図1の他に図2で表される“ *Don't Care* ”も使用する．“ *Don't Care* ”は図から明らかなように，入力xがいかなる値をとっても常に出力が１となる．

本プログラムでは，ルールの条件部に使用する言語値の種類を図1と2に限定しているが，他の言語値を使用することももちろん可能である．

2.2 ニューラルネットからのルール抽出

n次元cクラスパターン識別問題に対して学習済みのニューラルネットからルール抽出することを考える．

まず，実数データ$\mathbf{x} = (x_1,\cdots,x_n)$の識別を行う場合を考える．実数データ$\mathbf{x}$は学習後のニューラルネットへ入力され，対応する出力ベクトルが計算される．このとき，ニューラルネットの入出力関係は次のように表される．

$$\text{Input units: } o_i = x_i,\ i = 1,2,\cdots,n \tag{2}$$

$$\text{Hidden units: } o_j = f(net_j),\ j = 1,2,\cdots,n_{\mathrm{H}} \tag{3}$$

$$net_j = \sum_{i=1}^{n} w_{ji} \cdot o_i + \theta_j \tag{4}$$

$$\text{Output units: } o_k = f(net_k),\ k = 1,2,\cdots,c \tag{5}$$

$$net_k = \sum_{j=1}^{n_{\mathrm{H}}} w_{kj} \cdot o_j + \theta_k \tag{6}$$

ここで，o_i, o_j, o_kは実数出力，w_{ji}, w_{kj}は実数結合強度，θ_j, θ_kは実数バイアスを表している．

式（2）–（6）に示すように出力ベクトル$\mathbf{o} = (o_1,\cdots,o_c)$が計算される．実数データ$\mathbf{x}$はこの実数出力ベクトル$\mathbf{o}$を用いて以下の式によりクラス$h$に決定される．

$$o_h > o_k \text{ for } \quad k = 1,2,\cdots,c \quad (k \neq h) \tag{7}$$

一方，ファジィIf-Thenルールを抽出する場合には，条件部ファジィ集合の任意の組合せにより構成されるファジィ入力ベクトルを入力する必要がある．ニューラルネット内部のファジィ演算はα–レベル集合を用いた区間演算により近似的に行われる．

ファジィベクトル$\mathbf{A} = (A_1,\cdots,A_n)$がニューラルネットへ入力されるとファジィ出力ベクトル$\mathbf{O} = (O_1,\cdots,O_c)$が計算される．式（7）をファジィ数の

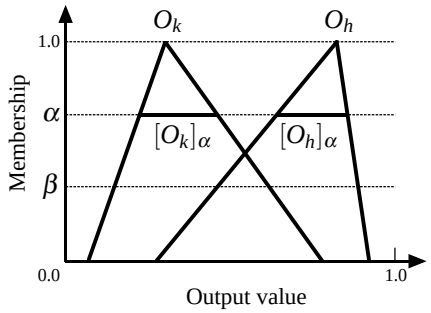


図 3: 大小関係$[O_h]_\alpha > [O_k]_\alpha$.

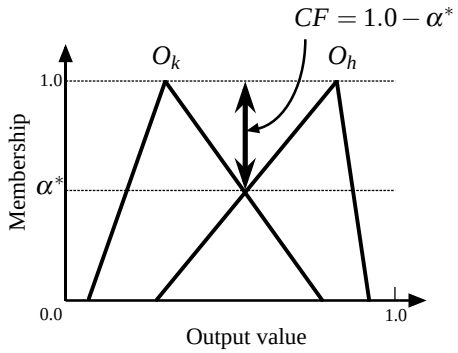


図 4: 確信度CFの決定.

場合に拡張すると以下の式（8）のようになる.

$$O_h > O_k \text{ for } \quad k = 1, 2, \cdots, c \quad (k \neq h) \qquad (8)$$

ここで，ファジィ数O_hとO_kの大小関係をレベル集合の大小関係を用いて式（9）のように定義する.

$$O_h > O_k \quad \Longleftrightarrow \quad [O_h]_\alpha^L > [O_k]_\alpha^U \qquad (9)$$

ここで，$[\cdot]_\alpha^L$ と $[\cdot]_\alpha^U$ は α–レベル集合の下限値と上限値である.

この式より$[O_h]_\alpha^L$と$[O_k]_\alpha^U$の２つのα-レベル集合が重ならないときのみ大小関係が成り立つ．図3の例では，α–レベルのとき大小関係が成り立ち，β–レベルでは大小関係が成り立たない.

次に，ルールの確信度の決定を行う．確信度CFは，式（9）の大小関係が成り立つレベルの最小値α^* に基づき式（10）のように定義される．従って，図4のように確信度が決定される.

$$CF = 1 - \alpha^* \qquad (10)$$

ルール抽出アルゴリズムの詳細は文献 [7]を参照されたい.

2.3 ルール抽出手順

ルールを抽出する基準となるレベルαの値は事前にユーザが設定する．できるだけ多くのファジィIf-Thenルールを抽出したい場合は，αを１に近い大きな値に設定し，確信度の高いファジィIf-Thenルールのみ抽出したい場合は，αを０に近い小さな値に設定すればよい.

Step 1: ファジィIf-Thenルールの条件部ファジィ集合に対応したファジィ数入力ベクトル**A**を生成する.

Step 2: Step１で生成したファジィ数入力ベクトル**A**のα–レベル集合をニューラルネットに入力し，ファジィ数出力ベクトル**O**のα–レベル集合を計算する．この計算は区間演算により近似的に行われる.

Step 3: 出力されたファジィ数出力ベクトル**O**のα–レベル集合を調べることで，ファジィ数入力ベクトル**A**のα–レベル集合の識別可能性を調べる．その結果，識別不可能な場合はファジィIf-Thenルールの生成を行わなずにStep１へ戻り，次のファジィ数入力ベクトルを生成する．識別可能な場合は，その識別結果をファジィIf-Thenルールの結論部クラスとしStep４へ進む.

Step 4: 様々なレベルの値を用いてファジィ数入力ベクトル**A**のレベル集合の識別可能性を調べ，式（10）により確信度CFを決定する.

2.4 ファジィIf-Thenルールの包含関係

抽出されたファジィIf-Thenルールから“ *Don't Care* ”を考慮することにより，包含される冗長なルールを削除可能である．ここで，ファジィIf-Thenルールの条件部において“ *Don't Care* ”以外の言語値を持つ属性の数をルールの“長さ”と定義する.

図5のような２次元３クラスパターン識別問題を用いてファジィIf-Thenルールの包含関係について考える．図1の５個の言語値と図2の“ *Don't Care* ”の６個を用いてルール抽出を行った結果を図6と7に示す.

図6より，長さ１のルール抽出を試みた場合は次の２つのファジィIf-Thenルールが抽出できている

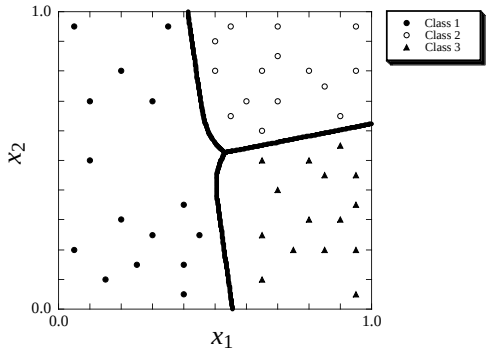


図 5: ３クラス識別問題.

x_2 \ x_1	S	MS	M	ML	L
DC	Class 1 (1.00)	Class 1 (0.65)			

図 6: 長さ１のファジィIf-Thenルール.

x_2 \ x_1	S	MS	M	ML	L
L	Class 1 (1.00)	Class 1 (0.64)	Class 2 (0.32)	Class 2 (1.00)	Class 2 (1.00)
ML	Class 1 (1.00)	Class 1 (0.68)	Class 2 (0.19)	Class 2 (0.59)	Class 2 (0.52)
M	Class 1 (1.00)	Class 1 (0.81)		Class 3 (0.23)	Class 3 (0.40)
MS	Class 1 (1.00)	Class 1 (0.94)	Class 1 (0.07)	Class 3 (0.77)	Class 3 (1.00)
S	Class 1 (1.00)	Class 1 (1.00)	Class 1 (0.20)	Class 3 (0.77)	Class 3 (1.00)

図 7: 長さ２のファジィIf-Thenルール.

ことがわかる.

$$\text{If } x_1 \text{ is } S \text{ then Class 1 with } CF_1 = 1.00 \quad (11)$$

$$\text{If } x_1 \text{ is } MS \text{ then Class 1 with } CF_2 = 0.65 \quad (12)$$

また図7より，長さ２のルール抽出を試みた場合は24個のファジィIf-Thenルールが抽出できていることがわかる．式（13）にその一例を示す.

$$\text{If } x_1 \text{ is } MS \text{ and } x_2 \text{ is } L \text{ then Class 1 with } CF_3 = 0.64 \quad (13)$$

式（12）の条件部は式（13）の条件部を包含しているため，式（13）のルールは式（12）に含まれていることがわかる．このように冗長なルールを

x_2 \ x_1	S	MS	M	ML	L
L	Class 1 (1.00)	Class 1 (0.65)	Class 2 (0.32)	Class 2 (1.00)	Class 2 (1.00)
ML			Class 2 (0.19)	Class 2 (0.59)	Class 2 (0.52)
M				Class 3 (0.23)	Class 3 (0.40)
MS			Class 1 (0.07)	Class 3 (0.77)	Class 3 (1.00)
S			Class 1 (0.20)	Class 3 (0.77)	Class 3 (1.00)

図 8: 冗長なルールを削除した後の長さ１から２のファジィIf-Thenルール.

削除することにより，抽出された26個のルールから図8のように16個までルールを削減できる.

以上の例で明らかなように，長さlのルールに包含されるルールの長さは$l+1$以上であり，同じ長さのルール間では包含関係が成り立たない．並列ルール抽出プログラムでは，この性質を利用して長さごとにルール抽出を行うことにより，包含ルールの削除を確実に行うことができる.

2.5 ルール抽出手法の並列化

ルール抽出手法は，学習済みのニューラルネットにファジィIf-Thenルールの条件部ファジィ集合の組合せを入力し，出力されるファジィ出力値に基づいてクラスと確信度を決定する．このとき，条件部ファジィ集合の組合せは，各属性に対して与えられている言語値の全ての組合せである．n次元問題に対して図2の“ *Don't Care* ”を含めたK個の言語値が各入力変数に与えられている場合，K^n個のファジィ入力ベクトルを構成することになる．そのため，多数の属性を伴う高次元問題ではニューラルネットへ入力するファジィIf-Thenルールの条件部ファジィ集合の組合せの数が指数関数的に増大する.

ルール抽出手法において，それぞれのファジィ数入力ベクトルに対してルール抽出可能性および確信度の計算は独立に実行できる．したがってこの性質を利用することで並列化を行う．すなわち，ファジィIf-Thenルールの条件部ファジィ集合の組合せをそれぞれノードに振り分けることで計算に要する時間を短縮することができる.

また，2.4節で述べたように，長さlのルールが抽出できた場合には長さが$l+1$以上のルールのいく

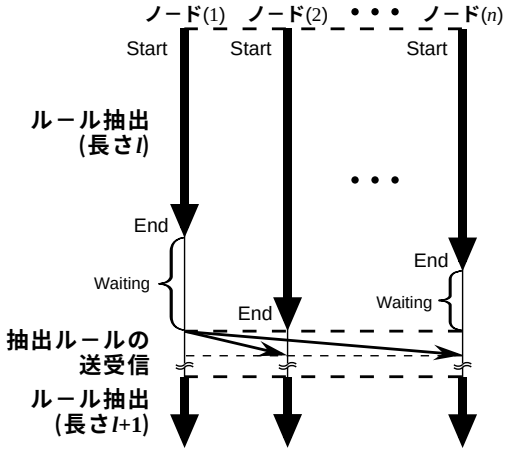


図 9: 並列ルール抽出手法の並列処理の流れ（長さ l）.

表 1: 入力データの一覧.

引数	内　容	型
1	# of input units	unsigned int
2	# of hidden units	unsigned int
3	# of output units	unsigned int
4	Threshold CF_{th}	double
5	Rule length (start)	unsigned int
6	Rule length (end)	unsigned int
7	Weights & biases file name	char[]

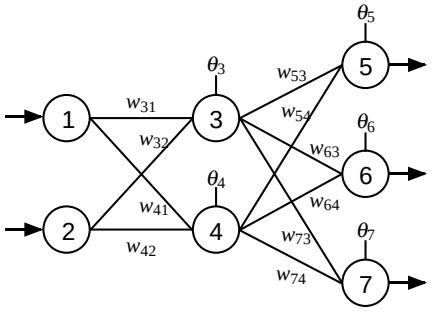
つかは長さlのルールに包含されている可能性がある．したがって，長さが短いルールから抽出していき，抽出したルールに包含される条件部ファジィ集合の組合せに対してはルール抽出を行わないように並列化を行う．

並列ルール抽出手法の並列処理の流れを図9に示す．各ノードはそれぞれルール抽出の計算を開始し，計算終了したノードは全ノードが計算を終了するまで待機する．そして，全ノードが計算を終了すると，ノード（1）から順番に抽出したルール数とそのルールを示す情報を他のノード全てへ送信する．

3　使用法と使用例

3.1　入力データ

本プログラムでは，表1に示す入力データ（引数）を必要とする．これらの入力データにより，しきい値CF_{th}より大きな確信度を持つルールが抽出される．ここで，学習後のニューラルネットの結合荷重データ（引数番号７），は，図10に示すニューラルネットの結合荷重を図11に示すようなフォーマットで表したものである． 実際の結合荷重データは図12のようになる．



w_{ji}: ユニットi, j間の結合強度
θ_i ：ユニットiのバイアス値

図 10: ニューラルネットの結合荷重.

w_{31}, w_{32}, θ_3
w_{41}, w_{42}, θ_4
w_{53}, w_{54}, θ_5
w_{63}, w_{64}, θ_6
w_{73}, w_{74}, θ_7

図 11: 結合荷重データのフォーマット.

```
-23.150140, -3.305322, 12.892372
5.311805, -26.429060, 11.025826
13.122677, 0.622349, -6.867582
-12.922465, -13.998045, 6.909474
-13.198914, 13.534509, -6.562125
```

図 12: 結合荷重データ例.

3.2　出力データ

本プログラムでは，抽出したファジィIf-Thenルールをファイルに出力する．各プロセッサは，それぞれ抽出したルールを“`rule-[Proc ID]-[File No.].data`”という名前のファイルに出力する．各ルールファイルに対する最大出力ルール数は，10,000,000に設定しており，それを越えた場合は，`File No.`の値をインクリメントしたファイルに抽出ルールを出力する．

ルールファイルでは，図1と2で示した言語値は表2のように対応した値に置き換えられる．

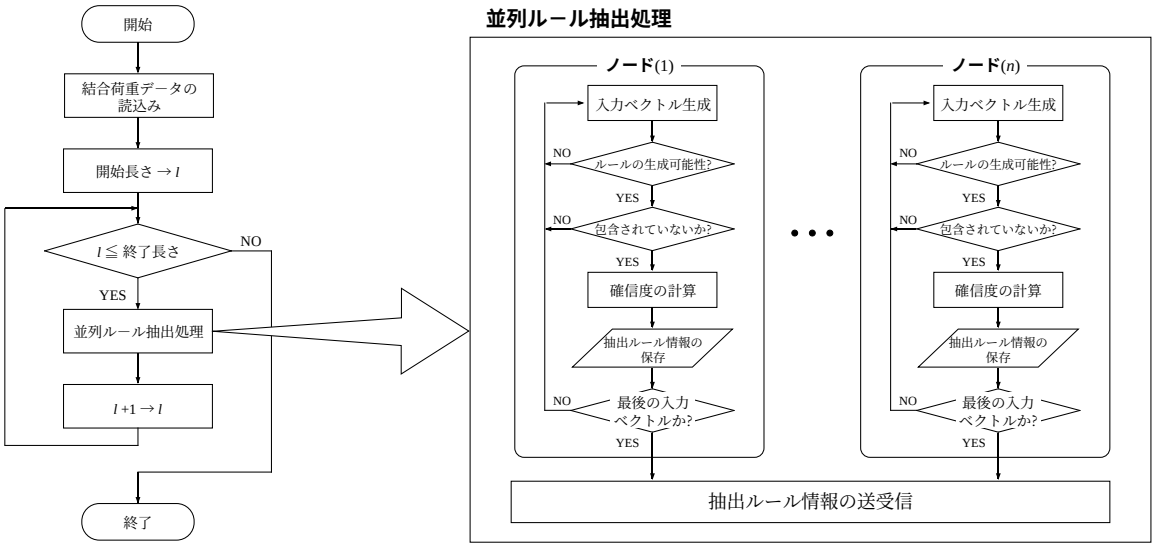


図 13: 並列ルール抽出プログラムの流れ図.

表 2: 言語値の対応表.

言語値	DC	S	MS	M	ML	L
値	0	1	2	3	4	5

たとえば，次のようなファジィIf-Thenルールが抽出された場合を考える．

If x_1 is *ML* and x_2 is *DC* and x_3 is *S*
then Class 0 with 0.75

If x_1 is *M* and x_2 is *MS* and x_3 is *L*
then Class 1 with 0.20

ルールファイルは以下のように出力される．

```
If 4, 0, 1, then 0, 0.75
If 3, 2, 5, then 1, 0.20
```

3.3 流れ図

本プログラムの流れを図13に示す．

3.4 使用例

本プログラムをUCI Machine Learning Repository（ftp://ftp.ics.uci.edu/pub/machine-learning-databases）より利用可能な甲状腺機能低下症データを用いて学習を行ったニューラルネットからルールの抽出を行った．甲状腺機能低下症データは，学習用データ数3,772，テスト用データ数3,428の21次元３クラスパターン識別問題である．学習後のニューラルネットには，学習率0.5，慣性項係数0.01，中間層ユニット数５，学習回数5,000回というパラメータ設定で学習を行ったものを使用した．

本プログラムのコンパイルは以下のように行った．

```
VPP% mpicc prext.c -o prext -Kvp
```

しきい値αの値を0.01として長さ１から７までルール抽出を行った．以下に，このとき使用したNQSスクリプトを示す．

```
#
# A Parallelized Rule Extraction Method
#
# @$-q h
# @$-lT 15:00:00
# @$-lM 7gb
# @$-lP 32
# @$-eo
#
set -x
VPP_MBX_SIZE=134217728; export VPP_MBX_SIZE
#
cd $QSUB_WORKDIR
#
./prext 21 10 3 0.01 1 7 ann-tyroid_21-10-3.wt
#
```

その結果，表3に示すようなルール抽出結果を得ることができた．また，このとき抽出された長さ４のルールを以下に示す．

```
If 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 4, 1, 0, 0, then 2, 0.05
If 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 1, 0, 0, then 2, 0.06
If 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 1, 0, 0, then 2, 0.10
If 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 0, 1, 0, then 2, 0.02
If 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 1, 4, 2, 0, 0, then 2, 0.02
If 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 4, 2, 0, 0, then 2, 0.05
If 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 2, 0, 0, then 2, 0.04
If 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 2, 0, 0, then 2, 0.09
If 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 3, 0, 0, then 2, 0.01
If 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 3, 0, 0, then 2, 0.05
If 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 3, 0, 4, then 2, 0.02
If 1, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 5, 0, 0, then 2, 0.02
If 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 4, 0, 0, then 2, 0.04
If 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 1, 4, 5, 0, 0, then 2, 0.03
If 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 4, 5, 0, 0, then 2, 0.11
If 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 5, 0, 0, then 2, 0.02
If 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 5, 0, 0, then 2, 0.13
If 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 0, 0, 4, then 2, 0.03
If 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 0, 0, 5, then 2, 0.03
If 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 0, 0, 5, then 2, 0.05
If 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 5, 0, 5, then 2, 0.07
If 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 5, 0, 5, then 2, 0.14
If 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 2, 0, 5, then 2, 0.05
If 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 5, 3, 0, 5, then 2, 0.05
```

表 3: ルール抽出結果.

ルールの長さ	抽出ルール数	計算時間[sec.]
1	0	0.01
2	0	0.08
3	0	0.51
4	24	2.59
5	1,594	13.76
6	60,165	118.26
7	1,435,044	6030.10
合計	1,496,827	6165.53

謝辞

本プログラムの開発にあたり，大型計算機を利用させていただきました京都大学学術情報メディアセンターの方々に深く感謝いたします.

参考文献

[1] Y. Hayashi: A neural expert system with automated extraction of fuzzy if-then rules and its application to medical diagnosis, In R.P. Lippmann, J.E. Moody, D.S. Touretzky (eds.), Advances in Neural Information Processing Systems 3, pp. 578–584, Morgan Kaufmann, San Mateo(1991).

[2] S. Sestito and T. Dillon: Knowledge acquisition of conjunctive rules using multilayerd neural networks, International Journal of Intelligent Systems, 8, pp. 799–805(1993).

[3] G. Towell and J.W. Shavlic: Extracting refined rules from knowledge-based neural networks, Machine Learning, 13, pp. 71–101(1993).

[4] L. Fu: Rule generation from neural networks, IEEE Trans. on Systems,Man, and Cybernetics, 24, pp. 1114–1124(1994).

[5] R. Andrews, J. Diederich, and A. B. Tickele: Survey and critique of techniques for extracting rules from trained artificial neural networks, Knowledge-Based Systems, 8, pp. 373–389(1995).

[6] C. Matthews and I. Jagielska: Fuzzy rule extraction from a trained multilayerd neural network, Proc. of IEEE International Conference on Neural Networks, pp. 744–748(1995).

[7] H. Ishibuchi and M. Nii: Generating fuzzy if-then rules from trained neural networks: Linguistic analysis of neural network, Proc. of IEEE

International Conference on Neural Networks, pp. 1133–1138(1996).

[8] H. Ishibuchi, M. Nii, and K. Tanaka: Linguistic rule extraction from neural networks for high-dimensional classification problems, Proc. of Complex Systems, pp. 210–218(1998).

[9] M. Nii, K. Ogino, T. Sakabe, H. Sakagami and Y. Takahashi: A parallelized rule extraction method for high-dimensional pattern classification problems, SICE Annual Conference 2002 in Osaka, sice02-0737(2002).

センターからの補足

本稿のプログラムprextは、ライブラリ登録されており、VPP上でご利用いただけます。したがって、VPPで利用される場合には、本稿3.4節に使用例として記述されているコンパイル作業は必要ありません。また、同節に記述されているNQSスクリプト内には”./prext...”とありますが、標準PATHに設定されていますので、先頭の”./”は書かないでください。VPPで使うNQSスクリプトの例は次のようになります。

```
#
# A Parallelized Rule Extraction Method
#
# @$-q h
# @$-lT 15:00:00
# @$-lM 7gb
# @$-lP 32
# @$-eo
#
set -x
VPP_MBX_SIZE=134217728; export VPP_MBX_SIZE
#
cd $QSUB_WORKDIR
#
prext 21 10 3 0.01 1 7 ann-tyroid_21-10-3.wt
#
```