慶應義塾大学学術情報リポジトリ
Keio Associated Repository of Academic resouces

Title	ILPにおける数量データの扱い : 雷の襲来予測への機械学習の応用
Sub Title	
Author	木村, 聡宏(Kimura, Akihiro)
Publisher	慶應義塾大学湘南藤沢学会
Publication year	1995
Jtitle	研究会優秀論文
JaLC DOI	
Abstract	本書は、AMEDASの気象データを使用して、機械学習による落雷予測を行うものである。ILPは事例間の関係を背景知識によって与える事で複雑な問題を学習できる為、落雷現象の学習にILPを使用することのメリットは大きい。属性値学習と比べILPの欠点として知られている「数量データの扱い」には特に重点を置き、クラスター分析とC4.5による数量データの質的データへの変換を、データの前処理として行っている。
Notes	古川康一研究会1994年秋学期
Genre	Technical Report
URL	https://koara.lib.keio.ac.jp/xoonips/modules/xoonips/detail.php?koara_id=0302-0000-0513

慶應義塾大学学術情報リポジトリ(KOARA)に掲載されているコンテンツの著作権は、それぞれの著作者、学会または出版社/発行者に帰属し、その権利は著作権法によって保護されています。引用にあたっては、著作権法を遵守してご利用ください。

The copyrights of content available on the KeiO Associated Repository of Academic resources (KOARA) belong to the respective authors, academic societies, or publishers/issuers, and these rights are protected by the Japanese Copyright Act. When quoting the content, please follow the Japanese copyright act.

SFC-SWP 94-A-016

研究会優秀論文

ILPにおける数量データの扱い

～雷の襲来予測への機械学習の応用～

木村聡宏
環境情報学部４年
古川康一研究会
１９９４年秋学期

慶應義塾大学　湘南藤沢学会
Keio University Shounan Fujisawa Academic Society

ILP における数量データの扱い
～雷の襲来予測への機械学習の応用～

慶應義塾大学
環境情報学部
知識情報コース 4 年
木村聡宏

目 次

1 はじめに – 本研究の目的 **3**

2 ローデータ **3**

3 使用した機械学習システム **4**
3.1 C4.5 4
3.2 GOLEM 4
3.3 PROGOL 4

4 実験手続き **4**
4.1 手続きの大枠 4
4.1.1 手続き 0 4
4.1.2 手続き 1 4
4.1.3 手続き 2 5
4.2 手続き 2： 数量データの質的データへの変換手続き 5
4.2.1 C4.5 によるデータ変換 5
4.2.2 クラスター分析によるデータ変換 5

5 ILP 使用データ **6**
5.1 正事例, 負事例 6
5.2 背景知識 6
5.2.1 背景知識の例 6
5.3 風向きについて 8

6 実験結果 **9**
6.1 生成されたルール 9
6.1.1 C4.5 9
6.1.2 GOLEM 11
6.1.3 PROGOL 13
6.2 生成されたルールの専門家による解釈・評価 13

7 考察 **13**
7.0.1 指標1：トレーニングデータの学習率 13
7.0.2 指標2：テストデータのカバー率 15
7.1 GOLEM,PROGOLにおける正事例データの順番とルールの生成 17
7.1.1 事例とルールの強弱 18
7.1.2 データの優先順決定手続き 18
7.2 比較: C4.5 vs クラスター分析 19
7.3 比較: C4.5 vs GOLEM, PROGOL 19
7.4 比較: GOLEM vs PROGOL 20

8 まとめ–今後の課題 **20**

9 謝辞 **21**

abstract

ILP(Inductive Logic Programming) は、機械学習と論理プログラミングを結び付ける新しい研究分野であり、背景知識を利用することが可能であり、また、一階述語論理ベースの表現力の大きい仮説言語を持っているため、従来の属性値学習では扱うことが困難であった、複雑な構造を持つ概念や問題の学習への応用が期待されている。

本研究では、AMEDAS の気象データを使用して、機械学習による落雷事例の学習実験を行なっている。実験においては、属性値学習と比べ ILP の欠点として知られている「数量データの扱い」に特に焦点を置き、クラスター分析と C4.5 による、数量データの質的データへの変換を、データの前処理として行なっている。

使用した機械学習システムは、属性値学習システムが C4.5、ILP システムが GOLEM , PROGOL である。

1 はじめに – 本研究の目的

1. ILP における数量データの扱い—数量データの質的データへの変換
2. ILP における数量データの適切な粒度の研究.
3. C4.5, PROGOL, GOLEM の表現能力, 計算能力の比較.
4. ILP への正事例データの与え方 (データの順番) の研究.
5. 気象データを使用した機械学習による雷の襲来予測

2 ローデータ

気象庁「アメダスデータ」を使用。データは下表のような属性とクラスから構成されている。

データの例

属性										クラス
降雨		日照		気温		風向 d/風速 v				半径 20km
t1	t2	t1	t2	t1	t2	t1	t2	t1	t2	落雷の有無
1.00	1.00	0.00	0.00	162.00	161.00	15.00	5.00	14.00	3.00	noflash
1.00	4.00	0.00	0.00	46.00	46.00		4.00		2.00	flash
4.00	9.00	0.00	0.00	46.00	40.00		2.00		4.00	flash
9.00	1.00	0.00	0.00	40.00	44.00		4.00		2.00	flash

t1 は 1 時間前の観測値
t2 は現在の観測値

各地点でのアメダスデータは 1 時間毎に収集されており、降水量 [mm]、日照時間 [0.1 時間]、気温 [0.1 度]、風向 [16 方位]、風速 [m/s] が記録される。今回の実験では、簡単のため、秋田市 1ヶ所におけるアメダスデータを利用し、秋田市を中心とする半径 20km 内の落雷の有無を予測することを考えた。

データ数は 302 であるが、この内、約 90%の 270 をトレーニングデータとして使用し、残り約 10%の 32 はテストデータとして使用した。正事例と不事例ー落雷したときの事例と落雷しなかったときの事例の比率ーは 1:1 である。

3 使用した機械学習システム

3.1 C4.5

C4.5 は、機械学習の代表的手法である属性値学習のシステムであり、Quinlan によって開発された。情報論的ヒューリスティクスにより、数量データを扱うことができる。

C4.5 は、属性値とクラスからなる、1 枚の表形式で与えられたデータから、シャノンの情報理論に基づき、情報量を最も減らす属性を選択していくことによって、決定木の生成を行う。

3.2 GOLEM

RLGG(相対最小汎化) と呼ばれる演算に基づいて、概念の学習を行う ILP システムであり、与えられた背景知識の元で、負事例をカバーしない範囲で正事例を一般化する。Muggleton と Feng によって開発された。

3.3 PROGOL

正事例, 負事例, 背景知識が与えられたとき、abduction の原理に基づき、最特殊節 (most specific clause) を求め、A*アルゴリズムによって最特化節の仮説空間 (仮説の body 部のリテラルの組合わせ) を探索した結果、評価関数の値がもっとも高い一情報量 $I(T|E)$ が最小となる一仮説を求める。Muggleton と Feng?? によって開発された概念学習システムである。

背景知識の記述に変数を使うことができるため、ルール形式の背景知識を与えることができる。

4 実験手続き

4.1 手続きの大枠

4.1.1 手続き 0

step1. ローデータから、正事例, 負事例の作成
↓
step2. 手続き 1, 手続き 2 へ

4.1.2 手続き 1

step1. ローデータから、背景知識の作成
↓
step2. GOLEM,PROGOL による実験

4.1.3 手続き 2

step1. 数量データ (ローデータ) の質的データ (2 次データ) への変換
↓
step2. step1. で変換された質的データ (2 次データ) から、背景知識の作成
↓
step3. GOLEM,PROGOL による実験

4.2 手続き 2： 数量データの質的データへの変換手続き

2 節のローデータは属性が数量データであるため、数量データを扱うことのできない GOLEM や PROGOL では直接使用することができない。このため、GOLEM や PROGOL で数量データを使用するためには、質的データに変換する必要がある。

数量データを質的データに変換する方法としては、統計的手法であるクラスター分析が代表的であるが、本研究では、それ以外に C4.5 による方法も試みた。C4.5 は情報論的ヒューリスティクスの使用によって数量データの扱いが可能になっている。この情報論的ヒューリスティクスは、FOIL における仮説の選択にも使用されている。

4.2.1 C4.5 によるデータ変換

step1) C4.5 のアウトプットである決定木を見て、属性毎にノードにおける分割ポイントを調べ、その分割ポイントによって各属性を幾つかのカテゴリーに分割する。
step2) 属性毎に、数量データを step1. で分割したカテゴリーに当てはめ、数量データを離散化する。

(例)

step1) 例えば、属性 rain-past の決定木における分割ポイントが、1,2,4 であれば、rain-past-under-1, rain-past-between-1-2,rain-past-between- 2-4, rain-past-over-4 という 4 つのカテゴリーを作る。
step2) 属性 rain-past の場合、3.00 という数量データは rain-past-between-2-4 という質的データに変換され、0.00 は rain-past-under-1 に変換される。

4.2.2 クラスター分析によるデータ変換

step 1) 1 属性毎に、S でクラスター分析を行い、データをクラスターに分ける。クラスター生成における距離の計算には最短距離法を使用した。

ここで注意することは、クラスタリングに使用したデータにテストデータも含めたということである。テストデータがクラスタリングによって質的データに変換されていないと、ILP が生成したルールを評価できないからである。

5 ILP 使用データ

手続き 0, 手続き 1, 手続き 2 にしたがって作成した、GOLEM,PROGOL 用のデータを以下に例としてあげる。落雷したときの事例を正事例、落雷しなかったときの事例を負事例とし、背景知識を正事例を説明する補助概念として使用する。

5.1 正事例, 負事例

手続き 0 にしたがって作成した正事例, 負事例

正事例の例

```
kekka(136,flash).
kekka(137,flash).
kekka(138,flash).
kekka(139,flash).
kekka(140,flash).
```

負事例の例

```
:-kekka(1,flash).
:-kekka(2,flash).
:-kekka(3,flash).
:-kekka(4,flash).
:-kekka(5,flash).
```

5.2 背景知識

今回の実験では、大きく分けて 3 種類の背景知識を使用した。

1 種類は、手続き 1 にしたがって、ローデータから作成したもの (B.K.1) である。これは、ローデータの数量がそのまま使われている。

残りの 2 種類は、手続き 2 にしたがって、質的データから作成したものであり、数量データの質的データへの変換に、C4.5 を使用したもの (B.K.2) と、クラスター分析を使用したもの (B.K.3) に分けられる。

5.2.1 背景知識の例

B.K.1

```
rain_past(1,0).
rain_past(2,0).
rain_pres(1,0).
rain_pres(2,0).
sun_past(1,0).
sun_past(2,10).
sun_pres(1,0).
sun_pres(2,5).
temp_past(1,182).
temp_past(2,231).
```

```
temp_pres(1,181).
temp_pres(2,230).
wind_dir_past(1,5).
wind_dir_past(2,6).
wind_spe_past(1,2).
wind_spe_past(2,3).
wind_dir_pres(1,4).
wind_dir_pres(2,6).
```

B.K.2

```
rain_past(8,between_1_2).
rain_past(60,between_2_4).
rain_pres(1,under_0).
rain_pres(2,under_0).
sun_past(95,between_0_1).
sun_past(96,between_0_1).
sun_pres(1,under_5).
sun_pres(2,under_5).
temp_past(25,between_100_104).
temp_past(66,between_100_104).
temp_pres(15,between_65_83).
temp_pres(16,between_65_83).
wind_dir_past(12,between_11_13).
wind_dir_past(14,between_11_13).
wind_dir_past(269,under_11).
wind_dir_past(270,under_11).
wind_spe_past(1,between_1_2).
wind_spe_past(3,between_1_2).
wind_dir_pres(6,over_13).
wind_dir_pres(7,over_13).
```

背景知識 rain-past(8,between-1-2) は、事例 8(負事例) について、1 時間前の雨量が 1〜2mm の間であったことを記述している。

B.K.3

```
rain_past(1,group1).
rain_past(2,group1).
rain_pres(8,group3).
rain_pres(44,group3).
sun_past(95,group2).
sun_past(96,group2).
sun_pres(1,group1).
sun_pres(4,group1).
temp_past(32,group8).
```

```
temp_past(77,group8).
temp_pres(26,group5).
temp_pres(31,group5).
wind_spe_past(8,group1).
wind_spe_past(15,group1).
wind_dir_past(1,ese).
wind_dir_past(2,se).
wind_dir_pres(1,e).
wind_dir_pres(2,se).
```

5.3 風向きについて

B.K.3 を使用した実験においては、3 種類の風向きデータ (4,8,16 方位) について、それぞれ別に実験を行った。

風向きデータは、ローデータでは 16 方位で値が与えられているが、16 方位のデータを、背景知識によって、N,E,W,S の 4 方位と N,NE,E,SE,S,SW,W,NW の 8 方位にまとめ、それらを 16 方位のデータと並行に考慮すようにした実験を、B.K.1 と B.K.2 を使用した実験で試みている。

以下、過去の風向きについて、16 方位のデータを 8 方位,4 方位にまとめる背景知識である。現在の風向きについても同様である。

背景知識の例

8 方位にまとめる背景知識.

```
wind_dir_past_oct(X,n):-wind_dir_past(X,n);wind_dir_past(X,nne).
wind_dir_past_oct(X,ne):-wind_dir_past(X,ne);wind_dir_past(X,ene).
wind_dir_past_oct(X,e):-wind_dir_past(X,e);wind_dir_past(X,ese).
wind_dir_past_oct(X,se):-wind_dir_past(X,se);wind_dir_past(X,sse).
wind_dir_past_oct(X,s):-wind_dir_past(X,s);wind_dir_past(X,ssw).
wind_dir_past_oct(X,sw):-wind_dir_past(X,sw);wind_dir_past(X,wsw).
wind_dir_past_oct(X,w):-wind_dir_past(X,w);wind_dir_past(X,wnw).
wind_dir_past_oct(X,nw):-wind_dir_past(X,nw);wind_dir_past(X,nnw).
```

4 方位にまとめる背景知識.

```
wind_dir_past_qua(X,n):-wind_dir_past(X,nnw); wind_dir_past(X,n);
        wind_dir_past(X,nne); wind_dir_past(X,ne).
wind_dir_past_qua(X,e):-wind_dir_past(X,ene); wind_dir_past(X,e);
        wind_dir_past(X,ese); wind_dir_past(X,se).
wind_dir_past_qua(X,s):-wind_dir_past(X,sse); wind_dir_past(X,s);
        wind_dir_past(X,ssw); wind_dir_past(X,sw).
wind_dir_past_qua(X,w):-wind_dir_past(X,wsw); wind_dir_past(X,w);
        wind_dir_past(X,wnw); wind_dir_past(X,nw).
```

6 実験結果

6.1 生成されたルール

機械学習システムに、気象データを学習させた結果、生成されたルールを以下にあげる。

6.1.1 C4.5

C4.5 は、今回の実験では数量データの前処理として使用した意味合いが強いが、学習結果は決定木の形で出力されている。以下、決定木の情報をルール形式に変換したものと、トレーニングデータの学習率とテストデータのカバー率である。

```
Rule 12:
     Sun_Past <= 0
     Wind_Spe_Past <= 5
     Wind_Dir_Pres <= 13
     Wind_Spe_Pres > 5
->  class flash  [91.2%]

Rule 34:
     Rain_Past > 1
     Wind_Dir_Past <= 13
     Wind_Dir_Pres > 4
->  class flash  [87.0%]

Rule 15:
     Rain_Pres <= 1
     Sun_Past <= 5
     Wind_Dir_Past <= 12
     Wind_Spe_Past > 5
     Wind_Dir_Pres <= 12
->  class flash  [85.2%]

Rule 33:
     Rain_Past > 5
->  class flash  [84.1%]

Rule 21:
     Rain_Pres > 0
     Rain_Pres <= 1
     Wind_Spe_Past > 2
     Wind_Dir_Pres <= 13
->  class flash  [82.5%]

Rule 29:
```

```
     Rain_Pres > 4
->  class flash  [77.7%]

Rule 23:
     Sun_Past > 0
     Sun_Pres <= 1
     Wind_Dir_Pres <= 15
->  class flash  [75.7%]

Rule 30:
     Sun_Past > 5
->  class noflash  [86.0%]

Rule 16:
     Rain_Past <= 1
     Rain_Pres <= 0
     Wind_Dir_Past > 12
     Wind_Dir_Pres > 5
->  class noflash  [79.8%]

Rule 36:
     Wind_Dir_Past > 13
->  class noflash  [76.6%]

Rule 11:
     Rain_Past <= 1
     Rain_Pres <= 1
     Sun_Past <= 1
     Wind_Spe_Past <= 5
     Wind_Spe_Pres <= 5
->  class noflash  [68.1%]

Rule 32:
     Rain_Past <= 5
     Wind_Dir_Pres <= 4
->  class noflash  [67.1%]

Default class: flash
```

```
Evaluation on training data (270 items):
         Before Pruning           After Pruning
        ----------------   --------------------------
        Size      Errors   Size      Errors   Estimate
          63   20( 7.4%)     41   27(10.0%)    (20.8%)   <<

Evaluation on test data (32 items):
         Before Pruning           After Pruning
        ----------------   --------------------------
        Size      Errors   Size      Errors   Estimate
          63   14(43.8%)     41   14(43.8%)    (20.8%)   <<

          (a)  (b)      <-classified as
         ----  ----
            7    9      (a): class flash
            5   11      (b): class noflash
```

6.1.2 GOLEM

学習クラス=flash, B.K.1, noise=1 のときのルール

```
kekka(A,flash) :- rain_pres(A,3).
kekka(A,flash) :- sun_past(A,3).
kekka(A,flash) :- sun_pres(A,1).
kekka(A,flash) :- sun_pres(A,4).
kekka(A,flash) :- temp_past(A,86).
kekka(A,flash) :- temp_past(A,92).
kekka(A,flash) :- temp_pres(A,95).
kekka(A,flash) :- temp_pres(A,119).
kekka(A,flash) :- temp_pres(A,121).
kekka(A,flash) :- temp_pres(A,127).
kekka(A,flash) :- wind_spe_past(A,8).
kekka(A,flash) :- wind_spe_past(A,10).
kekka(A,flash) :- rain_pres(A,1), wind_dir_pres(A,4).
kekka(A,flash) :- rain_pres(A,2), wind_dir_past(A,6).
kekka(A,flash) :- sun_past(A,2), sun_pres(A,0).
kekka(A,flash) :- sun_pres(A,0), wind_dir_past(A,7).
kekka(A,flash) :- sun_pres(A,0), wind_dir_past(A,10).
kekka(A,flash) :- sun_pres(A,0), wind_spe_past(A,11).
kekka(A,flash) :- wind_dir_past(A,6), wind_dir_pres(A,6).
kekka(A,flash) :- wind_dir_past(A,12), wind_spe_past(A,12).
kekka(A,flash) :- rain_past(A,1), wind_dir_pres(A,3), wind_spe_pres(A,3).
```

学習クラス=flash, B.K.3(8 方位), noise=1 のときのルール

```
kekka(A,flash) :- rain_pres(A,group4).
```

```
kekka(A,flash) :- wind_spe_past(A,group9).
kekka(A,flash) :- rain_past(A,group2), sun_pres(A,group4).
kekka(A,flash) :- rain_past(A,group3), wind_spe_past(A,group8).
kekka(A,flash) :- rain_past(A,group5), rain_pres(A,group2).
kekka(A,flash) :- rain_pres(A,group2), wind_dir_past(A,se).
kekka(A,flash) :- sun_past(A,group1), temp_past(A,group6).
kekka(A,flash) :- sun_past(A,group1), wind_dir_pres(A,sw).
kekka(A,flash) :- sun_past(A,group2), wind_dir_past(A,w).
kekka(A,flash) :- sun_past(A,group2), wind_dir_past(A,w).
kekka(A,flash) :- wind_dir_past(A,se), wind_spe_past(A,group4).
kekka(A,flash) :- wind_dir_past(A,se), wind_dir_pres(A,s).
kekka(A,flash) :- wind_dir_past(A,se), wind_spe_pres(A,group8).
kekka(A,flash) :- wind_dir_past(A,sw), wind_spe_pres(A,group10).
kekka(A,flash) :- rain_past(A,group1), sun_past(A,group4), temp_past(A,group3).
kekka(A,flash) :- rain_past(A,group2), sun_past(A,group1), wind_dir_pres(A,w).
kekka(A,flash) :- rain_past(A,group2), wind_spe_past(A,group8),
        wind_spe_pres(A,group1).
kekka(A,flash) :- sun_past(A,group1), temp_past(A,group3), wind_spe_past(A,group6).
kekka(A,flash) :- temp_past(A,group1), wind_spe_past(A,group2),
        wind_dir_pres(A,e).
kekka(A,flash) :- wind_dir_past(A,e), wind_spe_past(A,group8),
        wind_spe_pres(A,group6).
kekka(A,flash) :- rain_past(A,group1), rain_pres(A,B), temp_past(A,B),
        temp_pres(A,group2).
kekka(A,flash) :- rain_past(A,group1), sun_past(A,group1), wind_dir_past(A,nw),
        wind_dir_pres(A,w).
kekka(A,flash) :- rain_past(A,group4), sun_pres(A,B), temp_past(A,group3),
        temp_pres(A,B).
kekka(A,flash) :- rain_past(A,B), temp_past(A,group3), wind_spe_past(A,B),
        wind_spe_pres(A,group4).
kekka(A,flash) :- sun_past(A,group1), temp_past(A,group3), wind_dir_past(A,e),
        wind_spe_past(A,group5).
kekka(A,flash) :- temp_past(A,group3), wind_dir_past(A,e), wind_spe_past(A,group1),
        wind_spe_pres(A,group6).
kekka(A,flash) :- rain_past(A,group1), sun_pres(A,group1), temp_pres(A,group2),
        wind_spe_past(A,group8), wind_dir_pres(A,se).
kekka(A,flash) :- rain_past(A,B), rain_pres(A,group2), temp_past(A,group3),
        wind_dir_past(A,C), wind_dir_pres(A,C).
kekka(A,flash) :- sun_past(A,group1), wind_dir_past(A,B), wind_spe_past(A,group1),
        wind_dir_pres(A,B), wind_spe_pres(A,group7).
kekka(A,flash) :- rain_past(A,B), rain_pres(A,C), sun_past(A,D),
        sun_pres(A,D), wind_dir_past(A,E), wind_spe_past(A,C),
        wind_dir_pres(A,E), wind_spe_pres(A,B).
```

6.1.3 PROGOL

学習クラス=flash, B.K.2, noise=1 のときのルール

```
kekka(A,flash) :- rain_past(A,over_4).
kekka(A,flash) :- rain_past(A,between_1_2), wind_dir_past(A,under_11).
kekka(A,flash) :- rain_pres(A,over_0), temp_past(A,between_81_100).
kekka(A,flash) :- sun_past(A,between_0_1), wind_spe_past(A,over_7).
kekka(A,flash) :- temp_past(A,between_81_100), temp_pres(A,over_99).
kekka(A,flash) :- temp_pres(A,over_99), wind_spe_pres(A,over_9).
kekka(A,flash) :- rain_past(A,between_2_4), wind_dir_past(A,under_11),
        wind_dir_pres(A,between_4_12).
kekka(A,flash) :- rain_pres(A,under_0), wind_dir_past(A,under_11),
        wind_spe_past(A,over_7).
kekka(A,flash) :- sun_past(A,under_0), temp_past(A,over_158),
        wind_dir_pres(A,between_4_12).
kekka(A,flash) :- temp_past(A,between_127_145), wind_dir_pres(A,
        between_4_12), wind_spe_past(A,between_2_7).
kekka(A,flash) :- rain_pres(A,over_0), temp_past(A,between_39_81),
        wind_dir_past(A,under_11), wind_spe_past(A,between_2_7).
kekka(A,flash) :- sun_pres(A,under_5), wind_dir_pres(A,between_4_12),
        wind_spe_past(A,between_2_7), wind_spe_pres(A,between_6_9).
kekka(A,flash) :- sun_pres(A,under_5), temp_past(A,between_104_127),
        temp_pres(A,over_99), wind_dir_pres(A,between_4_12),
        wind_spe_pres(A,under_6).
```

6.2 生成されたルールの専門家による解釈・評価

7 考察

実験における言わばパラメータである (学習クラス, ノイズ, 数量データの粒度, 背景知識, ILP システム (GOLEM or PROGOL)) が結果にどのように反映しているのか、トレーニングデータの学習率 (指標 1) と、テストデータのカバー率 (指標 2) に基づき考察を行なう。

7.0.1 指標 1：トレーニングデータの学習率

学習率は、

$$\frac{\text{(トレーニングデータ数)—(一般化されなかったトレーニングデータ数)}}{\text{トレーニングデータ数 (270)}}$$

で示す。ルールに一般化されないデータの数が少ないほど学習率は高い。

ノイズ数は、ルールがカバーすることが許される負事例の数である。

クラス	ノイズ数	一般化されなかったトレーニングデータ数 (学習率 (%))					
		1 次データ (B.K.1)		2 次データ (B.K.2)		(B.K.3)	
GOLEM							
flash	0	20(85.2)		18(86.7)		15(88.9)	12(91.1) 11(91.9)
	1	7(94.8)		5(96.3)		10(92.6)	4(97.0) 7(94.8)
	2	9(93.3)		4(97.0)		6(95.6)	5(96.3) 6(95.6)
	3	9(93.3)		5(96.3)		7(95)	5(96.3) 5(96.3)
	4	9(93.3)		5(96.3)		5(96.3)	8(94.1) 5(96.3)
	5	9(93.3)		4(97.0)		7(94.8)	8(94.1) 6(95.6)
noflash	0	22(83.7)		19(85.9)		14(90.0)	13(90.4) 14(90.0)
	1	7(94.8)		8(94.1)		5(96.3)	3(97.8) 8(94.1)
	2	9(93.3)		7(94.8)		8(94.1)	6(95.6) 10(92.6)
	3	11(91.9)		6(95.6)		7(94.8)	7(94.8) 8(94.1)
	4	9(93.3)		7(94.8)		8(94.1)	6(95.6) 8(94.1)
	5	10(92.6)		7(94.8)		5(96.3)	8(94.1) 7(94.8)
total	0	42(84.4)		37(86.3)		29(89.3)	25(90.7) 25(90.7)
	1	14(94.8)		12(95.6)		15(94.4)	7(97.4) 15(94.4)
	2	18(93.3)		11(91.9)		14(94.8)	11(91.9) 16(94.1)
	3	20(92.6)		11(91.9)		14(94.8)	12(95.6) 13(95.2)
	4	18(93.3)		12(95.6)		13(95.2)	14(94.8) 13(95.2)
	5	19(93.0)		11(91.9)		12(95.6)	16(94.1) 13(95.2)
PROGOL							
flash	0	46(66.0)	40(70.4)	40(70.4)	24(82.2)	50(63.0)	46(66.0) 45(66.6)
	1	32(76.3)	27(80.0)	28(79.3)	22(83.7)	35(74.1)	32(76.3) 35(74.1)
	2	31(77.0)	28(79.3)	28(79.3)	23(83.0)	38(71.9)	27(80.0) 35(74.1)
	3	23(83.0)	21(84.4)	20(85.2)	13(90.4)	28(79.3)	29(78.5) 33(75.6)
	4	23(83.0)	24(82.2)	20(85.2)	15(88.9)	22(83.7)	21(84.4) 25(81.5)
	5	18(93.3)	22(83.7)	15(88.9)	12(91.1)	25(81.5)	17(87.4) 19(85.9)
noflash	0	38(71.9)	34(74.8)	42(68.9)	26(80.7)	31(77.0)	39(71.1) 39(71.1)
	1	26(80.7)	29(78.5)	29(78.5)	22(83.7)	26(80.7)	29(78.5) 24(82.2)
	2	25(81.5)	26(80.7)	27(90.0)	21(84.4)	21(84.4)	21(84.4) 19(85.9)
	3	26(80.7)	26(80.7)	24(82.2)	17(87.4)	26(80.7)	25(81.5) 17(87.4)
	4	22(83.7)	27(80.0)	24(82.2)	14(89.6)	15(88.9)	26(80.7) 18(93.3)
	5	24(82.2)	20(85.2)	23(83.0)	15(88.9)	17(87.4)	21(84.4) 22(83.7)
total	0	84(68.9)	74(72.6)	82(69.6)	50(81.5)	81(70.0)	85(68.5) 84(68.9)
	1	58(78.5)	56(79.3)	57(78.9)	44(83.7)	61(77.4)	61(77.4) 59(78.1)
	2	56(79.3)	54(80.0)	55(79.6)	44(83.7)	59(78.1)	48(82.2) 54(80.0)
	3	49(81.9)	47(82.6)	44(83.7)	30(88.9)	54(80.0)	54(80.0) 50(81.5)
	4	45(83.3)	51(81.1)	44(83.7)	29(88.9)	37(86.3)	47(82.6) 43(84.1)
	5	42(84.4)	42(84.4)	38(85.9)	27(90.0)	42(84.4)	38(85.9) 41(84.8)

注1： 1次データ (B.K.1), 2次データ (B.K.2) の右側は、16方位の風向きデータを使用し、16方位だけでなく4方位,8方位についても考慮されるように背景知識を記述したときの結果である。

注2: 2次データ (B.K.3) は、、風向きについて、左から順に、4,8,16の方位を値として与えたデータの結果である。

GOLEM,PROGOLともに、ノイズ0とノイズ1の結果の差が大きい。このことから、1つもノイズを許さないという条件が厳しすぎることが分かる。また、ノイズが1以上の場合、GOLEMでは、ノイズが増加しても学習率はそれほど変化しないのに対し、PROGOLでは、ノイズが増加するほど学習率が高くなる傾向がある。

GOLEMの学習率は、B.K.1よりもB.K.2、B.K.2よりもB.K.3の方が高く、クラスタリングによって前処理したデータの学習率も高いことが分かる。このことは、クラスタリングによって前処理したデータは、1次データやC4.5によって前処理した2次データよりも、ルール生成を促進させる、つまり、データの粒度が学習とよりマッチしていると判断することができる。

PROGOLの学習率は、クラスがflashのとき、ノイズが増加するほど高くなり、また、1次データより2次データの方が高く、さらに、1次データ,2次データとも、風向きデータに関して背景知識を与えた方が学習率は高まるという傾向がある。

実際に、1次データ使用でノイズ1のとき、カバーされなかった正事例数は、46であったのに対し、2次データ使用で、風向きに背景知識を与え、ノイズ5のときは、カバーされなかった正事例数は12であり、後者は、前者より34も多くの正事例をカバーするという結果が得られている。

学習率に関して、B.K.3を見ると、GOLEMではB.K.1. とB.K.2よりも少しいい結果が出ているのに対して、PROGOLではやや劣る結果が出ている。

ところで、ノイズを増加させ、ルールの生成条件を緩めることは、正事例の学習率を高める一方で、反対に、負事例のカバーを許すことよって、ルールの精度 (テストデータによるカバー率) を落としまう危険性が考えられる。そこで、ルールの精度について、以下、指標2にまとめた。(テストデータによって、生成されたルールの精度を評価している)

7.0.2 指標2： テストデータのカバー率

テストデータのカバー率は、生成されたルールを評価するものである。

ILPが生成したルールにカバーされるテストデータの数 (割合) を以下にまとめる。

カバー率は、

$$\frac{\text{カバーされたテストデータ数}}{\text{テストデータ数 (32)}}$$

で示す。

c	n	正しくカバーされたテスト事例の数 (カバー率 (%))					
		1 次データ (B.K.1)		2 次データ (B.K.2)		(B.K.3)	
GOLEM							
n	0	8.5(53.1)		9(56.2)	10.7(66.9)	10.5(65.6)	10(62.5)
o	1	9(56.2)		11(68.8)	8.3(51.9)	11.7(73.1)	10.7(66.9)
f	2	10.3(64.4)		9.5(59.4)	11.3(70.6)	11.3(70.6)	9.5(59.4)
l	3	11.3(70.6)		10.7(66.9)	12.8(80.0)	12(75.0)	8.7(54.4)
a	4	14.2(88.8)		10(62.5)	10.9(68.1)	11.5(71.9)	12.2(76.3)
	5	12(75.0)		10.5(65.6)	13.3(83.1)	11.5(71.9)	13.2(82.5)
	0	4(25)		5.5(34.4)	4.7(29.4)	6.3(39.4)	6.2(38.8)
f	1	8.8(55)		7.5(46.9)	4.8(30.0)	6.7(41.9)	6(37.5)
l	2	8.7(54.4)		9.3(58.1)	5(31.3)	6.8(42.5)	6(37.5)
a	3	8.7(54.4)		10.3(64.4)	5.2(32.5)	6.5(40.6)	7.7(48.1)
	4	8.7(54.4)		9.3(58.1)	6.3(39.4)	5.5(34.4)	8.2(51.3)
	5	8.4(52.5)		9.3(58.1)	5.8(36.3)	5.8(36.3)	8.2(51.3)
t	0	12.5(39)		14.5(45.3)	15.1(47.2)	16.8(52.5)	16.2(50.6)
o	1	17.7(55.3)		18.5(57.8)	13.1(40.9)	18.4(57.5)	16.7(52.2)
t	2	19(59.4)		18.8(58.8)	16.3(50.9)	18.1(56.6)	15.5(48.4)
a	3	20(62.5)		21(65.6)	18.0(56.3)	18.5(57.8)	16.4(51.3)
l	4	22.9(71.6)		19.3(60.3)	17.2(53.8)	17.0(53.1)	20.4(63.8)
	5	20.4(63.8)		19.8(61.9)	19.1(59.7)	17.3(54.1)	21.4(66.9)
PROGOL							
n	0	7.5(46.9)	7(43.8)	7(43.8)	5(31.3)	10.1(63.1)	11(68.8)	9(56.3)
o	1	6.5(40.6)	8(50.0)	8.5(53.1)	7.5(46.9)	9(56.3)	9(56.3)	8.5(53.1)
f	2	7(43.8)	7(43.8)	8.6(53.8)	6.5(40.6)	9.3(58.1)	12.5(78.1)	10.5(65.6)
l	3	5.5(34.4)	7(43.8)	7.5(46.9)	7(43.8)	8.5(53.1)	10(62.5)	10.5(65.6)
a	4	11(68.8)	8.5(53.1)	8.5(53.1)	6(37.5)	11.3(70.6)	10(62.5)	11.7(73.1)
	5	8.5(53.1)	8.5(53.1)	8.8(55.0)	7.5(46.9)	8.5(53.1)	10(62.5)	11.7(73.1)
	0	4.5(28.1)	4.5(28.1)	6(37.5)	4(25.0)	3(18.8)	4.5(28.1)	4(25.0)
f	1	2(12.5)	4(25.0)	6.5(40.6)	6(37.5)	5(31.3)	5(31.3)	5(31.3)
l	2	1(6.3)	4.3(26.9)	6.7(41.9)	6.5(40.6)	5(31.3)	5(31.3)	6.5(40.6)
a	3	3(18.8)	4.3(26.9)	8.5(53.1)	7.5(46.9)	5(31.3)	5(31.3)	5.5(34.4)
	4	7(43.8)	6(37.5)	8(50.0)	8(50.0)	6.5(40.6)	5.5(34.4)	8(50.0)
	5	7(43.8)	4(25.0)	8.5(55.0)	8.7(54.4)	5(31.3)	7.5(46.9)	8.2(51.3)
t	0	12(37.5)	11.5(35.9)	13(40.6)	9(28.1)	13.1(40.9)	15.5(48.4)	13(40.6)
o	1	8.5(26.6)	12(37.5)	15(46.9)	13.5(42.2)	14(43.8)	14(43.8)	13.5(42.2)
t	2	8(25.0)	10.3(32.2)	15.3(47.8)	13(40.6)	14.3(44.7)	17.5(54.7)	17(53.1)
a	3	8.5(26.6)	10.3(32.2)	16(50.0)	14.5(45.3)	13.5(42.2)	15(46.9)	16(50.0)
l	4	18(56.3)	14.5(45.3)	16.5(51.6)	14(43.8)	17.8(55.6)	15.5(48.4)	19.7(61.6)
	5	15.5(48.4)	12.5(39.1)	17.3(54.0)	16.2(50.6)	13.5(42.2)	17.5(54.7)	19.9(62.2)

注 1: B.K.1,B.K.2 の右側は、指標 1 と同じ扱いである。
注 2: B.K.3 についても、指標 1 と同じ扱いである。

指標 1 と 2 を比較すると、ノイズを増やして生成されるルールの条件を緩めても、テストデータのカバー率は低下せず、逆にカバー率は上がることが分かる。このことから、この気象データの場合には、ある程度ノイズを許さないと、条件が厳しすぎるため、十分に一般化されていない特殊なルールが生成されてしまうことが分かる。

GOLEM で 1 次データを使用し、ノイズが 4 のときに、71.6%という今回の実験の中ではもっとも高いカバー率が得られている。データがシンプルな気象データのみであることを考慮すれば、70%を越えるカバー率はかなり高いといってよい。

GOLEM と PROGOL を比較すると、全体的に GOLEM の方がいい結果を出している。また、PROGOL の場合ノイズを増やしてもカバー率が上がらないのは、PROGOL は生成されるルールの数自体が少なく、ルールが十分に一般化されていないからであると思われる。

noflash と flash を比較すると、GOLEM,PROGOL ともに、flash の方が noflash よりもカバー率が低く、クラスが flash の事例を予測する方が困難であることが分かる。これは常識的にも当然の結果である。

PROGOL において、B.K1 と B.K.2 において、風向きについて背景知識を使用していないデータと使用したデータの結果を比較すると、背景知識を使用していないデータの方が結果がよく、背景知識を使用することによって、かえって結果が悪くなってしまっていることが分かる。これは興味深い結果である。学習率は、背景知識を使用することによって、結果がよくなっているのに、その結果が、テストデータのカバー率に反映されていないのである。おそらく、背景知識を使用して学習したことによって、より特殊なルールが生成されたと考えるのが妥当であろう。

B.K.3 について、GOLEM では、クラスが noflash のとき、70%を越える高いカバー率もいくつか得られよい結果が出ているが、クラスが flash のときは、カバー率は最高で 50%と結果は途端に悪くなってしまう。学習率については、クラスが flash のとき、むしろ、いい結果が出ているため、このことから、クラスタリングによって前処理したデータは、クラスが flash の学習に対して適合性が悪いことが分かる。

同じく B.K.3 について、PROGOL では、クラスが noflash のとき、B.K.3 は、B.K.1 と B.K.2 と比べ、比較的いい結果が出ているが、クラスが flash のときは、必ずしもいい結果は出ていない。ただし、flash の学習率に関しては、逆に、いい結果が出ている。このことは上に書いたように、GOLEM と同じである。学習率の高さは、必ずしもカバー率とは結びついていないことが、このことから分かる。

7.1 GOLEM,PROGOL における正事例データの順番とルールの生成

GOLEM,PROGOL は、ルールを生成する都度、そのルールにカバーされるすべての正事例を除外し、新しいルールの生成は、すでに生成されたどのルールによってもカバーされていない残りの正事例から行なうという手続きをとる。

このため GOLEM,PROGOL では、正事例を与える順番によって、生成されるルールの集合が変わることがある。

例えば、正事例の順番によっては、多くの正事例をカバーする強い (一般的な) ルールが始めに生成されることによって、弱い (特殊な) ルールが生成されなくなってしまうことがある。

このように GOLEM,PROGOL は正事例の順番によって、生成されるルール集合が変わる可能性があるわけだが、ここで、弱いデータを上に置くことにより、強いルールが生成される前に、弱いルールを生成させるというアプローチが考えられる。目的はトレーニングデータの学習効率の向上である。トレーニングデータの学習効率を上げるためには、カバーされていなかった弱いデータをカバーする、弱いルールを

生成する必要がある。このアプローチによる弱いルールの生成は、テストデータのカバー率の向上にもつながることが期待される。

一方、先に弱いルールを生成させることの問題点は、それらのルールが、一般化すべきではないような、一般性の低い特殊なルールである可能性が高いことである。実際に、PROGOL のルール生成過程を見ると、後に残された事例からほど、ルールが生成されにくく、そのことは、それらの事例が一般性の低い特殊な事例であることを示している。したがって、そうした特殊な事例から、特殊なルールを生成することに十分な意味があるのかは、特殊なルールを生成することによるトレーニング事例の学習率の向上とも比較して、検討すべき課題である。

7.1.1 事例とルールの強弱

データ – カバーされるルールが少ないほど弱い. 多いほど強い.
ルール – カバーする事例が少ないほど弱い。多いほど強い.

7.1.2 データの優先順決定手続き

idea:

事例の強さを、その事例をカバーする" ルールの強さの総和" で計る。ルールの強さは、カバーする事例数で計る。

step0. 一般化する正事例を毎回変えて、正事例個回数 PROGOL を走らせる。その結果、135 のルールが得られる。
↓
step1. 135 のルールの内、同一のルールを排除して、異なるルールだけを残す。
↓
step2. 別のルールに包摂されるルールを排除する。その結果得られたルールを step3. に渡す。

↓ ↓ ↓

step3. 縦軸をルール番号、横軸を事例番号として、ルールと事例のカバー関係の表を作る (表 1)。
↓
step4. 表 1 の縦軸の事例 1 つに対して、その事例をカバーするすべてのルールについて、それらのルールがそれぞれいくつの事例をカバーしているか、その総和を計算する

総和の大きさが事例の強さである。

↓
step5. step4. の手続きをすべての事例に対して行なう。
step6. step4.5. から得られた値が小さい順に事例を並べる。

	1	2	3	4	5	6	7	8	9	…	(data-no)
01	○	○	○								
02				○	○	○					
03	○	○					○	○	○		
04	○						○	○	○		
05											
06					○						
07				○							
⋮											
⋮											
(rule-no)											

表 1

7.2 比較: C4.5 vs クラスター分析

C4.5 は、与えられたデータが数量データの場合、属性を数値で分割するので、利用者は、属性の分割ポイントに基づいて、数量データを質的データに変換すればよく、データの変換過程を理解しやすい。

一方、クラスター分析は、距離が近いデータやクラスターを順にまとめていくという手続きをとるので、結果として得られるクラスターがどのような特徴を持っているのかは明白ではなく、それは利用者が自ら調べ判断しなければならない。また、クラスタリングによるグループ分けは、クラスとは関係なく行われるため、クラスをもっともよく分けるように分割ポイントを生成する C4.5 とは異なる。

数量データの場合、クラスタリングを行うことによって、データから距離情報が失われてしまうことも、クラスタリングの不利点である。

クラスター分析の短所は、利用者にとって探索的な手法であり、クラスター生成のとき、距離の計算の仕方 (ex. 最短距離法, 郡平均法 etc..) をどれにするのがベストなのか、どの高さでグループ分けしたらいいのかといったことを、利用者が経験的に判断しなければならないということである。この点、C4.5 は、データさえ与えれば、後はシステムが自動的に計算してくれるので、利用者にとっては C4.5 の方が使いやすいだろう。

C4.5 の問題点は、ノードにおいてもっともよくクラスを分ける値を分割ポイントとして決定木を生成していくので、木全体で見ると、必ずしも最適な木が生成されるとは限らないということである。ノードごとにロカールマキシマムを選択していくため、木全体で見ると、グローバルマキシマムにならない場合が起こりうるのである。

テストデータの扱いも C4.5 とクラスター分析では異なる。4.2.2 にも記述したように、クラスター分析の場合には、テストデータも含めてクラスタリングをする必要があるが、C4.5 の場合には、テストデータを含める必要はない (含めてもよい)。

7.3 比較: C4.5 vs GOLEM, PROGOL

本研究で使用したアメダスの気象データは、データ構造が複雑でなく、対象間に関係性を示していないので、C4.5 で扱うほうが望ましいデータである。しかし、C4.5 はデータの周期性/不連続性を扱うことができないため、今回の実験でも、周期性/不連続性を持つ風向きデータについては扱うことができなかった。ILP では、背景知識を与えることによって、データの周期を扱うことができる。ただし、今回のデータは、事例が独立であり、風向きの周期を背景知識で与えても特に意味がないため、背景知識は与えなかった。

計算時間については、C4.5 の方が明らかに GOLEM,PROGOL よりも速い。計算量の多さに起因する計算の非効率は、GOLEM,PROGOL などILP の大きな短所となっている。ILP の計算量の多さは、仮説言語が表現力の大きい一階述語論理ベースのものであることや、背景知識の導入などによっている。C4.5 は、仮説言語が表現力の劣る命題論理ベースのものであり、背景知識も扱うことができないので、それが制約となっている分、計算はILP よりもかなり効率的である。

7.4 比較: GOLEM vs PROGOL

GOLEM はPROGOL と比べ、生成されるルール節の数が多い。およそ、PROGOL の 1.2 倍から 1.7 倍である。このことは、GOLEM と PROGOL のルールの生成方法の違いに起因している。簡単に言うと、GOLEM よりも PROGOL の方がルール生成の条件が厳しいため、GOLEM よりも生成されるルール数が少ないのである。

GOLEM は、背景知識に変数を許さないため、背景知識はすべてグランドで記述しなければならない。一方、PROGOL は、背景知識の記述に変数を使うことができるため、背景知識をルールの形式で記述することができる。今回の実験でも、風向きの方位をまとめるルールを背景知識に与えている。

計算量については、どちらも非常に多く、実際の使用においては、計算効率の悪さは問題となる。ただし、PROGOL の方が GOLOEM よりはだいぶ効率的であり、実際の使用上も許容範囲内に収まっているといってよい。ただし、構造の複雑なデータにおいて、述語の引数間の関係を背景知識に与えたりすると、計算量はとたんに爆発してしまう。

8 まとめ–今後の課題

落雷現象は、気象現象であるとともに電気現象である。本研究は、気象データによる落雷の予測を目的としているため、アメダスの気象データのみを使用し、電気データを使用しなかった。しかし、電気データも付加することによって、落雷予測の精度はさらに上がることが予想される。

また、アメダスの気象データには、湿度や気圧など落雷現象に関係がある思われるいくつかのデータがなく、このことも予測精度に影響していると思われる。必要十分なデータを用意することも、予測精度を上げるうえで重要である。

また、AMEDAS の気象データでは、落雷の事例をそれぞれ独立なものとして扱っているため、落雷間の関係性を示すようなルールを学習することはできなかった。しかし、実際の落雷事例には、例えば時系列で関係しているものもあるため、そのような関係の学習をすることは、より精度の高い落雷予測を行なう上で重要になる。

今回の実験で使用した気象データでは、落雷時刻が分からないため、落雷の時系列での関係性を学習することはできなかったが、落雷時刻の分かるデータを使用すれば、そうした学習は可能になるため、今後の研究で取り上げたい。

また、そうした関係性の学習にILP は非常に有用である。それは、ILP が、事例間の関係を背景知識によって与えることによって、複雑な問題を学習することができるからである。特に、落雷は非常に複雑な現象であるため、落雷現象の学習にILP を使用することのメリットは大きいと思われる。

今後の課題としては、他に、落雷現象の地域や季節による比較がある。落雷に関して一般に知られていることとして、日本海沿岸では冬に落雷が多いことや、都市部では周辺の郊外部に比べ落雷が多いといった事実があり、それらの事実を説明するような学習も今後の課題として残されている。

今回は、ローデータの前処理にクラスター分析を使用したが、落雷の予測自体には統計的手法やニューラルネットワークは用いなかった。統計的手法やニューラルネットワークと機械学習との予測の比較をす

ることにも意味があるため、重回帰分析や判別分析などの統計的手法やニューラルネットワークによる、落雷現象の解析も、今後の重要課題として残る。

9 謝辞

三菱総合研究所の村野氏には、気象データの提供や、生成されたルールの解釈をしていただきました。ここに感謝致します。

政策・メディア研究科の八木氏は、実験を進めていくなか、多くの問題点や疑問点に答えて下さいました。ここに感謝致します。

最後に、3年の春学期より現在まで、4学期に渡り、常に御指導そして激励をして下さった古川教授に感謝致します。

参考文献

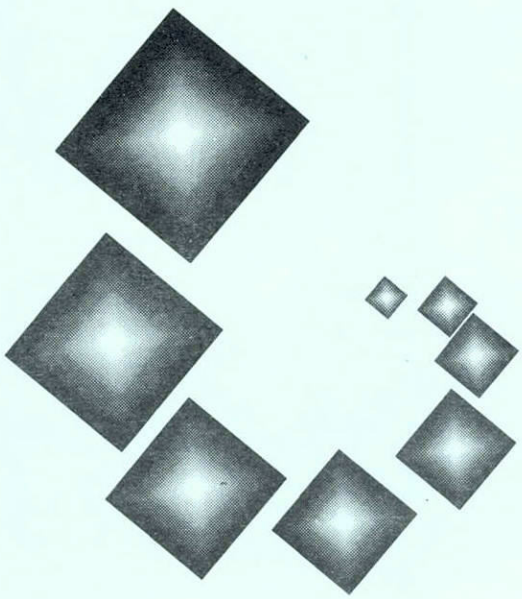
[1] Ivan Bratoko and Saso Dzoroski: Engineering Apprications of ILP,

[2] 古川康一: 論理と応用 (3), 人工知能学会誌, Vol7, No6(1992), pp955-963.

[3] 川村正: 帰納論理プログラミングー論理プログラミングの帰納的一般化を中心に, コンピュータソフトウェア, Vol10, No5(1993), pp3-15.

[4] 八木直樹 古川康一: C4.5 と GOLEM の表現能力の比較について, 人工知能学会全国大会 (第 8 回) 論文集, pp129-132(1994).

[5] 饗庭貢: 雷の科学, コロナ社, 1990.



＊　本論文は研究会で優秀と認められ　出版されたものです　＊