人　工　知　能 98－2
（1995．1．18）

概念束を用いた選言概念の逐次的学習

宮川 聡　　上原 邦昭　　前川 禎男

神戸大学 工学部 情報知能工学科

内容梗概

近年，機械学習に関する研究が盛んに行なわれており，従来では学習困難とされていた DNF 概念の学習方法なども提案されている．しかしながら，これらの多くはすべての訓練例を一度に使用する一括処理型アルゴリズムがほとんどであり，逐次的に DNF 概念を学習することは困難とされている．本研究では，事例記述中の特徴の連言を束形状に表現した概念束を用いることによって，事例の入力順序に左右されずに逐次的に知識ベースを作る方法を提案する．また，概念束を用いることにより，DNF 概念の逐次的な学習が可能となることを示し，さらに，概念束の逐次的な縮小法，および概念束からのルール抽出についても検討する．

Incremental Learning of Disjunctive Concepts Using Concept Lattice

Satoshi Miyagawa　Kuniaki Uehara　Sadao Maekawa

Department of Conputer and Systems Engineering, Kobe University

Abstract

Recently, in the field of machine learning, many researchers have concentrated on concept learning algorithms capable of learning DNF concepts. Most of these algorithms are constructed on the assumption that all training instances can be given at a time, although incremental learning algorithms are needed in various practical domains. In this paper, we propose an incremental learning algorithm that is able to learn DNF concepts incrementally by using *comcept lattice*. We also describe methods of both incremental reduction of the concept lattice and rule extraction from the concept lattice.

1 はじめに

近年，各種エキスパートシステムの開発に際して，知識獲得のボトルネックが問題となってきている．知識工学の概念が定着し，種々の実用エキスパートシステムが開発されるにつれ，有効な多くの知識を獲得するには多大な労力が必要なことが認識されつつある．このため，知識獲得の支援に対する要望が強まっている．以上の問題に対処する手段として，機械学習による知識ベースの自動的獲得手法が注目を集めている [1][2][3][4][6][7]．機械学習の考え方を導入すれば，いくつかの典型的対象状態 (特徴記述) と結論 (クラス) の対からなる事例を与えるだけで，一般的な知識ベースを自動的に生成することができるため，上記問題の解決に役立つことが期待されている．

知識ベースの自動的獲得のために，現在，さまざまな学習アルゴリズムが提案されているが，それらの多くは事例を一括して処理する方法をとっている (以後，すべての事例を一度に処理するアルゴリズムを一括処理型アルゴリズムと呼ぶ)．一括処理型アルゴリズムの中には，構成的帰納法を応用した方法 [1][2]，あるいは学習後に知識精錬フェーズを追加した方法 [3][4] などを用いて，従来，学習困難であった DNF 概念を学習可能にしたアルゴリズムもある．前者は，決定木学習[1]等で得られた知識中の特徴を連言でつないで新特徴を作り，既存の特徴に新特徴も含めて再び学習を行なうというサイクルを繰り返す方法である．後者は，多くの候補知識を生成した後で，候補知識の組合せを考えることにより，最適な知識の集合を求める方法である．

上記の 2 種の方法は，DNF 概念のルールベース学習法としてはかなり有効であるが，学習に必要な事例を予めすべて与えることができるという仮定に基づいている．しかしながら，実際問題としては，必要十分な事例を最初から与えることは困難な場合も考えられるため，逐次的な学習方式の導入が必要とされている．言い換えると，いくつかの事例を与え，その範囲で学習された知識を試用して，もし誤った結論が導かれれば，正しい結論を持つ事例をさらに訓練事例として与えるといったサイクルを繰り返すことが望ましい．すなわち，学習状況に応じて事例を増やしながら，逐次的に学習を進めることのできる方式が必要である．以後，このような学習アルゴリズムを逐次処理型アルゴリズムと呼ぶ．

従来考案されている逐次処理型アルゴリズムには，ヴァージョン空間法に代表されるような連言表現を学習するルールベース学習法や，事例ベース学習などが有名であるが，いずれも DNF 概念などの学習には不向きである．本研究では，各事例を構成する特徴および特徴の連言を束形状にした概念束によって知識を表現し，概念束を用いて分類を行なう方法を提案する．また，概念束を用いた学習法によって，DNF 概念の逐次的な学習が可能であることを示す．

[1]ID3 などのように，獲得された知識が決定木の形で表現されるような学習法を決定木学習と呼ぶ．

2 研究背景

2.1 DNF 概念

本節では，本研究で学習対象とする選言概念および DNF 概念について説明する．選言概念とは，概念記述が選言形式で表現される概念を指している．選言概念の中で，特徴間に何らかの従属関係がある概念を，特に DNF (選言標準形: Disjunctive Normal Form) 概念と呼ぶ．

DNF 概念の例として，三目並べ問題 (図 1 参照) がある．三目並べ問題は「三目並べのゲーム終了時における×の勝ち」という概念を学習する問

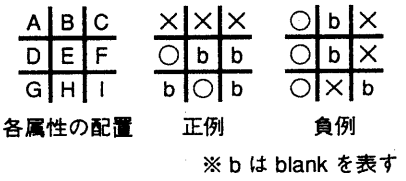

図 1: DNF 概念の例 (三目並べ問題)

題である．各事例は，ゲーム終了時のゲーム盤の状態を表しており，×が三連を達成していれば正例，それ以外は負例である．また，各属性は三目並べのゲーム盤のマス目に対応しており，左上から右に向かって順に A，B，... ，I という属性名であるとし，それぞれの属性は“×”，“○”，“blank (空白)” のいずれかの値を持つとする．三目並べ問題の場合，各特徴は正例中にも負例中にも一様に存在している．たとえば，三目並べ問題で (A=×) という特徴に注目して事例集合を分割しても，分割後のどちらの集合中にも正例と負例が同程度に含まれているため，特徴選択方式は意味をなさない．. (A=×) ∧ (B=×) ∧ (C=×) のような特徴の連言を用いることによって，はじめて正例と負例を分割することができる．

また，他の DNF 概念の例としてパリティ概念がある．パリティ概念とは，属性値がランダムに 0 または 1 の値を持ち，特定のいくつかの属性値の合計が偶数ならば正例，奇数ならば負例という概念である．三目並べと同様にパリティ概念でも，たとえば，ある属性の値が 0 であるという特徴を持つ事例は正例中にも負例中にも同程度に存在している．

従来の学習アルゴリズムでは，正例と負例をうまく弁別できるような特徴を一つづつ選択しながら，ルール記述を構成する特徴選択方式をとっているものが多い．しかしながら，DNF 概念の学習では，有効な特徴を選択することが難しい．なぜならば，DNF 概念の各特徴は各クラス中に同程度に存在するため，選択すべき特徴を特定できないためである．このように，特徴の連言を選択しなければ，正しいルールが構成されない概念を DNF 概念と呼ぶ．

2.2 一括処理型アルゴリズム

DNF 概念を学習可能なアルゴリズムは大きく二つに分けることができる．第一の方法は，構成的帰納法を用いた方法 [1][2] である．この方法は，まず決定木学習等で学習を行ない，それによって得られた知識中で使用されている特徴の連言を新特徴とし，それらの特徴も使って事例を再び記述し直し，学習を繰り返すという方法である．たとえば，先ほどの三目並べ問題では，学習過程で (A=×) ∧ (B=×) や (A=×) ∧ (B=×) ∧ (C= ×) などの新特徴が作られる．(A=×) ∧ (B=×) ∧ (C=×) という特徴は負例中に存在せず，正例中にのみ存在するため，これらの新特徴を使えば DNF 概念の学習が可能となる．

第二の方法は，多くの候補知識を生成した後で，それらの組合せを考えることにより，最適な知識の集合を求めるものである．このようなアルゴリズムとして，RF2[3] がある．RF2 は，特徴選択方式の問題点を回避するために，基本戦略として special-to-general 探索を採用している．spcial-to-general 探索は，ある事例記述 (この事例を seed と呼ぶ) と他の事例記述との共通部分をくくり出すことによって，seed の記述を徐々に一般化し，なるべく多くの正例をカバーしつつ，負例をカバーしないような連言記述を生成するものである．この spcial-to-general 探索を，すべての事例が連言のどれかにカバーされるまで，seed となる事例をを変えながら繰り返す．DNF 概念は一つのクラス内に複数のサブクラスを持つ (各サブクラスが DNF 記述中の各連言によって表現される) 概念であると考えることができる[2]．したがって，理論的には，それぞれのサブクラスごとに seed を選ぶことができれば，各サブクラスに対応した連言をすべて作ることができる．これらの連言集合中からもっとも望ましい連言の組合せを選び出し，それらを選言でつないで DNF 形式にすれば，求める概念記述が得られるはずである．RF2 では，このような考え方を IDA* 探索によって実現し，必要最小限の連言を選び出すようにしている．

また，決定木学習の一つである C4.5 からのルール抽出法 [4] では，決定木から個々の連言を取り出し，それらから不要な特徴を取り除いて連言集合とし，それらの組合せを MDL 基準 [5] を利用した探索法によって選びだし，DNF 形式のルールを生成している．

[2]先ほどの三目並べ問題は 8 つのサブクラスがある．

2.3 逐次処理型アルゴリズム

DNF 概念を学習するには，前節で述べた二種の方法が有効となるわけであるが，これらはいずれも一括処理型アルゴリズムにのみ可能な方法である．逐次処理型アルゴリズムにおいては，学習後に新特徴を構成してからの再学習，又は知識の精錬などのフェーズを導入することが不可能である．また，DNF 概念に限らず対象が選言概念の場合には，学習の初期段階ではルールの要素として用いるべき特徴を正しく選択できないために，逐次的なルールベース学習は困難であることが知られている．これは，通常の一括処理型のルールベースアルゴリズムでは，DNF 概念の学習において，正例をカバーし，負例をカバーしないような連言，すなわち，弁別的な連言の組合せによってルールセットを生成することが可能であるが，逐次処理型アルゴリズムの場合には，逐次的に弁別的な連言の組合せを選ぶことは不可能なためである．

つぎに，非ルールベースの逐次学習法について説明する．前述のように，ルールベースで逐次学習を行なうことには無理があるため，ルールを用いない学習法によって逐次的に選言概念を学習する方法も提案されている．非ルールベースの学習法としては事例ベース学習 IBL[8] が有名である．事例ベース学習とは，概念記述としてルールを生成せず，未知の例が入力されたときには，過去の事例ベース中から最も類似した事例を検索し，入力例がその事例と同じクラスに属するかどうかを判定する学習法である．この手法は，逐次的学習が容易であるため，事例検索が容易な問題領域においてはかなり有効な手段であるが，ほとんど DNF 問題を学習することが不可能であるという問題がある．事例ベース学習が DNF 概念を学習できない理由は，類似度の計算時に特徴間の従属関係を考慮することができないためである．これは，単にユークリッド距離によって事例間の類似度を求めていることによる．また，事例ベース学習に構成的帰納法を導入したもの [9] もあるが，有効な背景知識等のバイアスが必要となり，汎用性には欠けている．

逐次的な選言概念，特に DNF 概念の学習に関してまとめると，以下のようになる．

- 唯一の DNF 形式のルールを保持しながら学習を行なう方式では学習が困難である (ルールベース学習は不向き)．
- 単にユークリッド距離によって類似度を求めるのではなく，各特徴の連言に関する情報を有効に利用する方法が必要である (ルール候補となる連言を集合として保持する)．

本研究では，事例中の各特徴およびその組合せに注目するために，事例の一般化空間として概念束を用い，概念束を利用することよって逐次的に DNF 概念を学習可能な手法 [10] を提案する．

3 概念束を用いた一般化

3.1 本研究の概要

概念束とは，事例記述中の特徴および特徴の連言を，一般・特殊の半順序関係によって束の形状に表現したものである．まず，表 1 の例について考える．第 1 例から第 4 例までが Class 1，第 5 例

表 1: 入力事例

事例＼属性	A	B	C	Class
例 1	a	a	a	1
例 2	b	b	a	1
例 3	a	a	b	1
例 4	b	b	b	1
例 5	a	b	a	2
例 6	b	a	a	2
例 7	a	b	b	2
例 8	b	a	b	2

から第 8 例までが Class 2 に属し，属性 A, B, C はそれぞれ値として a と b を持っている．表 1 では，各クラス中の全例に共通した特徴がなく，どの特徴も全クラスの事例中に一様に存在している．たとえば，$A = a$ という特徴は Class 1 中と Class

2 中に同数ずつ存在するため，ID3 などの決定木学習で用いられる情報量の期待値等による特徴選択法は意味をなさない．なお，各事例はそれぞれ $Class1 = AaBa \vee AbBb$, $Class2 = AaBb \vee AbBa$ (Aa, Bb は属性 A の値が a，属性 B の値が b であることを表す) という法則にしたがっている DNF 概念である．

表 1 の事例から生成される概念束を図 2 に示す．図中のノードを結ぶ実線をエッジと呼び，最上段のノードをトップノード，最下段のノードをボトムノード，ボトムノードの一つ上のノード (事例記述そのもの) をリーフノード，それ以外のノードを中間ノードと呼ぶ．また，各ノードはそれぞれ唯一の連言を持っており，連言中の要素数 (and-結合された特徴の数) をそのノードのレベルと呼ぶ．

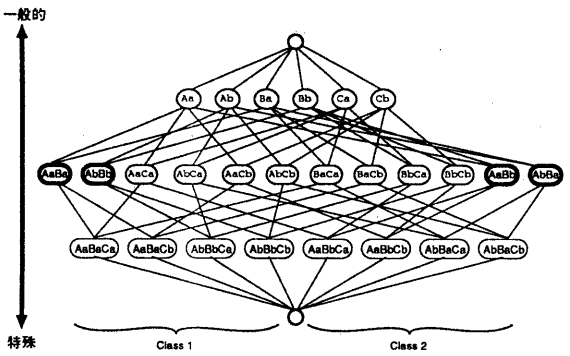

図 2: 概念束の例

概念束中には，$AaBa$ や $AbBb$ のように，事例を弁別する際の鍵となる特徴の組合せ，すなわち連言 (図中の太線部) が現れている．これらの連言を発見することができれば，未知の事例 (テスト事例) の分類時におけるクラスの予測に役立てることができる．概念束を用いて概念学習を行なうことには，つぎのような利点が考えられる．

- 概念束は逐次的に生成することが可能 [11] である．また，事例を構成する特徴のすべての組合せを概念束中に保存しているため，事例の入力順序に全く影響を受けない．
- 特徴の連言をそのまま保存するため，連言の組合せを考える必要がない．そのため，DNF 概念の学習に際して連言の組合せを誤ることもなく，生成されたルールによって分類能力を低下させることもない．

3.2 概念束の生成法

本節では，逐次的に概念束を生成する手法について説明する．新たな訓練事例が入力されると，逐次更新アルゴリズム: *Update_Lattice* によって概念束の逐次的な更新がなされる．*Update_Lattice* は，入力事例と概念束中の各ノードとの交差 (intersection) をとり，新たなノードの生成を試みるアルゴリズムである．図 3 では，新たな事例 $AaBaCa$ とノード① の交差をとり，新たなノード③ を生成し，それらを結ぶエッジを生成している状態を示している[3]．逐次更新アルゴリズム *Updata_Lattice*

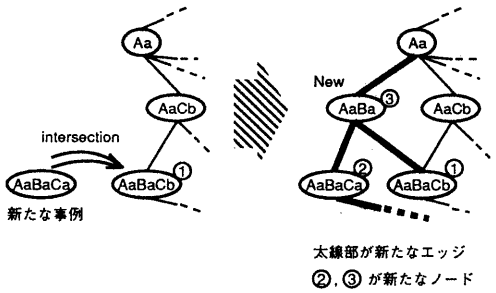

図 3: 概念束の更新

の詳細を図 4 に示す．図 4 の A は概念束，s は連言，i はノード，$\mathcal{U}$, $\mathcal{L}$ は連言の集合を表している．また，関数 *Make_New_Leaf*, *Intersection*, *Upper_Boundery_Set*, *Lower_Boundery_Set*, *Make_New_Node* はそれぞれ以下のような働きをする関数である．

各関数の説明

[3] 新たな事例が入力された時点で，リーフノード② が生成される

```
Algorithm Update_Lattice(A, s);
begin
  d_s := 事例 s の特徴記述を連言でつないだもの;
  A := Make_New_Leaf(A, d_s);
  for i ∈ A do begin
    d_i := ノード i の連言記述
    d_x := Intersection(d_s, d_i);
    if d_x が空
      or A 中に，d_x を持つノードが存在
      or A 中に，d_x より特殊で
        d_i より一般的な連言を持つノードが存在
    then 何もしない;
    else begin
      U := Upper_Boundery_Set(A, d_x);
      L := Lower_Boundery_Set(A, d_x);
      A := Make_New_Node(A, d_x, U, L)
    end
  end
end.
```

図 4: 概念束の逐次更新アルゴリズム

$Make\_New\_Leaf(A, d_s)$ ... 連言 d_s を持つ新たなリーフノードを作り，そのノードを概念束 A 中に組み込んだ新たな概念束を返す.

$Intersection(d_s, d_i)$ ... 連言 d_s と連言 d_i の共通部分を返す.

$Upper\_Boundery\_Set(A, d_x)$ ... 概念束 A 中のノードで，連言 d_x よりも一般的な連言を持つもののうち，もっとも特殊なノードの集合を返す.

$Lower\_Boundery\_Set(A, d_x)$ ... 概念束 A 中のノードで，連言 d_x よりも特殊な連言を持つもののうち，もっとも一般的なノードの集合を返す.

$Make\_New\_Node(A, d_x, \mathcal{U}, \mathcal{L})$ ... 連言 d_x を持ち，ノード集合 $\mathcal{U}$ 中のノードを親に，ノード集合 $\mathcal{L}$ 中のノードを子に持つ新ノードを作り，そのノードを概念束 A に組み込んだ新たな概念束を返す.

なお，各ノードは，その連言によってカバー[4]することのできる各クラスの事例数，そのエントロピー についての情報を持っているものとする.

エントロピーとは，そのノードにおける各クラスの混在の度合を示す関数である．ノード i のエントロピー e_i は，

$$e_i = \sum_{j=1}^{n}(-P_{i,j} log P_{i,j}) \ [bits]$$

で表される．ここで，n は総クラス数，$P_{i,j}$ はノード i における総被カバー例数に対するクラス j に属する例の比率である．未知の事例が入力されたときには，これらの情報を使って分類が行なわれる．その方法については次章で説明する.

4 概念束を用いた分類

4.1 予測ノードの探索法

本節では，未知の事例 (テスト事例) が入力されたときの分類方法，つまり，束形状に保存された知識の利用法について述べる．前述したように，逐次学習の初期段階では，ルール記述の候補となるべき連言，すなわち，正例をカバーし負例をカバーしないような連言が多く存在しているが，それらの連言の組合せを逐次的に選択することは困難であるため，ルールを生成することが難しい．一方，本手法によって生成される概念束中には，ほぼ単一のクラスの事例のみをカバーする連言を持ったノードがそのまま保存されているため，テスト事例の分類時にその事例が属すると思われるノードを予測すればよい.

以下に，テスト事例の分類方法について説明する．テスト事例 t が入力されると，つぎに示す評価指標に基づいて，t が属すべきノード (これを t の**予測ノード**と呼び p_t で表す) を概念束中から選択し，t のクラスを予測する．すなわち，p_t の連言がもっとも多くの事例をカバーしているクラスに t も属していると予測する.

評価指標とは次のようなものである.

1. 連言記述 d_i が t の事例記述の連言表現 d_t をカバーするノード i.

[4]ここでは，ノード i の連言 c_i が事例 s の記述 c_s と同じかまたはそれより一般的であるとき，「事例 s は連言 c_i にカバーされる」または「事例 s はノード i にカバーされる」と表現する.

2. エントロピー e_i が閾値 T_1 以下のノード i.

3. 1. と 2. を満たすノードのうち，もっとも多くの訓練事例をカバーしているノードが p_t である．

概念束中の全ノードを調べて p_t を求めることも可能であるが，ここではより効率的に図5に示す探索法によって求めている．この手法をサンプル

```
Algorithm Search_Predict_Node(A, t);
begin
  d_t := 事例 t の特徴記述を連言でつないだもの;
  l := 1;
  h_score := 0;
  f := off;
  while f = off do begin
    f := off;
    N_l := レベル l で,
      評価指標 1. と 2. を満足するノードの集合
    k_node := N_l 中でもっともカバーする
      事例数の多いノード; ..... (※)
    k_score := k_node のカバー事例数;
    if h_score < k_score then begin
      h_node := k_node;
      h_score := k_score;
      f := on
    end;
    l := l + 1
  end;
  return h_node
end.
```

図 5: 予測ノードの探索法

データに適用した結果を図6に示す．使用した問題は，$A \sim I$ の9つの2値の属性を持ち，各事例はつぎの法則に従う DNF 概念の分類である．

Class 1 ←
$ABC \vee DEF \vee GHI \vee ADG \vee BEH \vee CFI$

Class 2 ←
$\bar{A}\bar{B}\bar{C} \vee \bar{D}\bar{E}\bar{F} \vee \bar{G}\bar{H}\bar{I} \vee \bar{A}\bar{D}\bar{G} \vee \bar{B}\bar{E}\bar{H} \vee \bar{C}\bar{F}\bar{I}$

比較のために，図6には C4.5 の決定木での結果およびルールでの結果を載せている．C4.5 の結果では，ルールによる方法が決定木による方法を上回っており，学習後の知識の精錬が DNF 概念に対して効果的であることが分かる．一方，本手法は知識の精錬等の処理をしない逐次的学習でありながら，この問題においては C4.5 と同等以上の分類結果を得ることができている．これは，概念束中に弁別に有用な情報，すなわち，予測ノードが存在することを示しており，本章で提案した探索法によって妥当な予測ノードを選出することが可能であることが分かる．

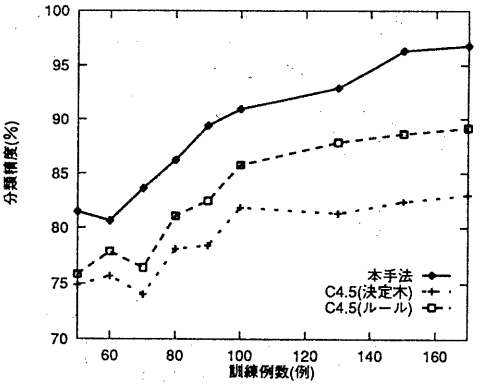

図 6: 概念束による分類結果

4.2 k-nearest-neighbor ライクな方法

IBL などでは，もっとも類似した k 個の事例で予測を行なう k-nearest-neighbor 法によって知識の利用時における誤りを削減し，分類精度を高めることができるが，本手法においても k-nearest-neighbor ライクな方法によって分類能力を高めることができる．ただし，本手法は類似度を使用していないので，アルゴリズム中の (※) のところで，カバーする事例数の多い順に k 個のノードを選ぶことにする．

この方法を用いて k の値を 3 ～ 5 とした場合に，先ほどの例に適用した結果を図7に示す．図のように k の値を増すに従って分類精度が上がっており，この手法によって分類能力を高めることが可能であることが分かる．

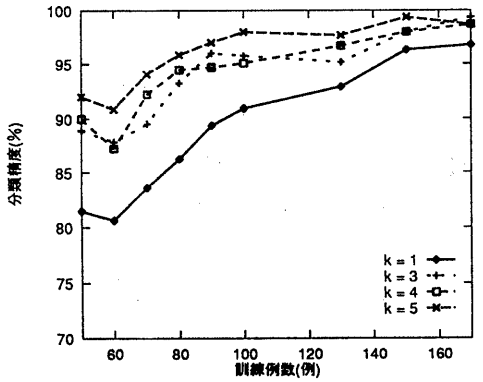

図 7: 概念束による分類結果 ($k = 1, 3 \sim 5$)

5 概念束の逐次的縮小

本手法の問題点として，事例の増加にともない，概念束中のノードは爆発的に増大するということが挙げられる．たとえば，前節の例では，100 例の学習でノード数が 5000 近くにもなってしまう．これらのノードのうちで，弁別のために必要となるのは，束中の一部のみであると考えられるため，概念束をすべて保持する必要はない．このような，束中の不要なノードを削除することにより，処理時間，メモリ使用量の軽減を図ることができると考えられる．

概念束中のノードはつぎのグループに分けて考えることができる．

- 特殊ノード … トップノードとボトムノード
- 混在ノード … エントロピーが閾値 T_2 以上の中間ノード
- エントロピーが閾値 T_2 以下の中間ノード
  - 有用ノード … 親ノードのエントロピーが閾値 T_2 以上であるノード (混在ノードに接するノード)
  - 準有用ノード … 親ノードのエントロピーが閾値 T_2 以下であるノード
- リーフノード

このうち，弁別の際に必要となるノードは有用ノードのみである．したがって，特殊ノード以外の他のノードは削除してもよいことになるが，逐次的にノードを削除する際には，つぎにあげるような問題があるため，ノードの削除は慎重に行なう必要がある．

1. エントロピーが閾値以上の混在ノードを生成するたびに，そのノードを削除することは問題である．ある時点でそのノードのエントロピーが閾値以上であったとしても，学習が進むにつれて閾値以下になっているかも知れないからである．
2. 実際に必要となる有用ノードが揃わないうちに，準有用ノードやリーフノードを削除してしまうと，概念束生成時における交差の相手となるノードが足りず，必要な有用ノードが生成されなくなる場合がある．
3. ある有用ノードのエントロピーがが閾値以上になってしまったときに，その子ノードがすでに消去されているという場合も起こり得る．

これらの問題のうち，1. については，分類時の評価指標に用いられる閾値 T_1 の値と，削除用の閾値 T_2 の値を $T_1 < T_2$ となるように少し差をつけて設定することによって対応できる．2. と 3. については現在考慮中であるが，つぎのような方法によって解決可能であると思われる．2. については，各有用ノードにノードの有用性を測る信頼度のような重みを付け，ある重み以上のノードの子にあたる準有用ノードおよびリーフノードを削除するようにするなどの方法が考えられる．また，削除する準有用ノードを有用ノードより一定レベルより下位のものにすることによって 3. の問題を回避することができる．

先ほどの例に混在ノードを削除する方法を適用した場合の結果を図 8 に示す．ノード数は 100 例の時点で 300 程であり，大幅なノードの削減がなされている．それにもかかわらず，分類精度は若干の劣化は見られるものの，ほぼ良好な結果が得られていることが分かる．

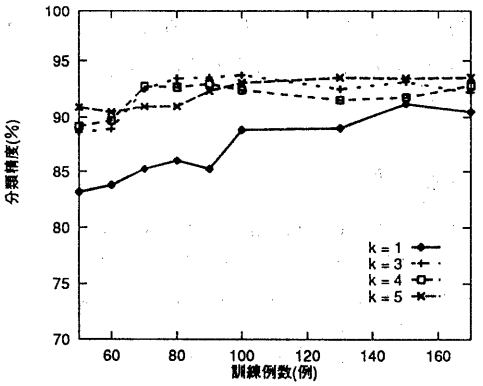

図 8: 縮小概念束による分類結果

6 MDL 基準を用いたルール化

本章では，MDL 基準を用いた概念束からのルール化について説明する．本アルゴリズムは，概念束中の知識を連言形式で保存しているため，学習後にそれらの組み合わせによってルールを得ることも可能である．事例を一括して与えることができる場合，または，ルール形式の知識が要求される場合には，概念束からルールを抽出することによって，分類時の探索コストとの削減および知識の記憶量の削減を測ることができる．

ここで用いるルール化は，基本的には C4.5 からのルール化に用いられているものと同様の考え方に基づいており，ルールの記述量と誤分類される事例の記述量の和によって導かれるコストを最小にするようなルールセットを求めるようにしている．コストの定義を以下に示す．

$$Cost = TheoryCost + log_2 \left(\begin{array}{c} r \\ fp \end{array} \right) + log_2 \left(\begin{array}{c} n-r \\ fn \end{array} \right)$$

ただし，$TheoryCost$ はルールセットを記述するために必要なビット数，r はルールセットによりカバーされる訓練事例数，fp は間違ってカバーされている負例数，n は全訓練事例数，fn は間違ってカバーされていない正例数である．

ノードの連言にクラス名のラベルを付けたものをルールとし，それらを組合せて (論理和をとって) ルールセットとするわけであるが，ルールの組合せのすべてのコストを計算することは効率的ではないので，ここではつぎの 2 段階の方法によって各クラスのルールセットの探索を行なっている．

クラス c のルールセットの選出法

1. **候補連言集合の選出** … エントロピーが閾値以下で，そのノードにカバーされる事例の大多数がクラス c に属し，かつそれより上には閾値以下のものがないノードの連言記述を候補連言とする．候補連言をコストの低い順に並べ変え，候補連言リスト $\mathcal{B}$ とする．

2. **ルールセットの選出** … *Search_Rule_Set* アルゴリズムによってルールセットを選出する．

Search_Rule_Set アルゴリズムを図 9 に示す．

```
Algorithm Search_Rule_Set(B);
begin
  O := B;
  C :=∅;
  h_score := とても大きな数;
  h_ruleset := φ;
  while 終了基準が満たされていない do begin
    a_ruleset := O の先頭を取り出す;
    if a_score < h_score then begin
      h_ruleset := a_ruleset;
      h_score := a_score
    end;
    Subset(a_score, B, O, C)
  end;
  return h_ruleset
end.
```

図 9: ルールセットの選出法

なお，アルゴリズム中の $\mathcal{O}$ はルールセットの順序付集合 (リスト)，$\mathcal{C}$ はルールセットの集合，$h_{ruleset}$ と $a_{ruleset}$ はルールセットを表している．

関数 *Subset* では，a に $\mathcal{B}$ 中の要素のうち，

- a 中にないもの．
- a に付加しても，$\mathcal{C}$ 中の要素と同じにならないもの．

を付加したルール集合を作り，それぞれのコストを計算し，これらを順序よく O に入れるという処理をしている．また，終了基準には規定回数以上 $h_{ruleset}$ が更新されないことを条件としている．

本章で提案したルール化方法を 4 章で使用した例に適用して，得られたルールでテストした結果を図 10 に示す．

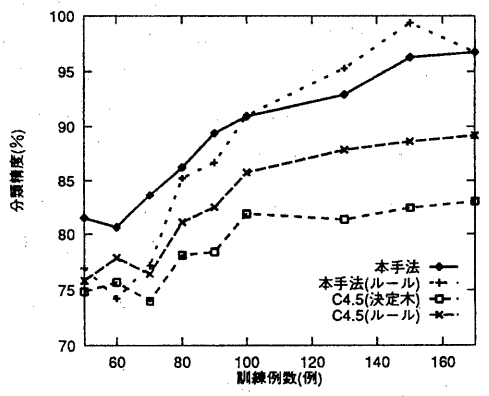

図 10: 概念束からのルールによる分類結果

図 10から，本手法では良好な結果が得られ，一括処理型のルールベースアルゴリズムとしても本手法が有効であることが分かる．また，分類精度についても，4 章で述べた手法の結果とルールによる手法の結果との差はあまりみられない．本章で述べたルール化方法は，学習後の知識の精錬によって分類能力を向上させることを目的としているのではなく，知識の可搬性を増すことのみを目的としているためである．つまり，4 章で述べた方法が，知識の精錬を行なうことなく高精度の分類をするという本研究の目的を達成していることを示すものでもある．

7 おわりに

本稿は，選言概念，特に DNF 概念を逐次的に学習可能なアルゴリズムを提案し，概念束を用いた学習法により，DNF 概念の逐次的学習の可能性を示した．また，概念束の縮小法およびルール化法を提案した．今後は，さらに多くのケースに適用してその有効性を検証すること，および概念束の縮小方法とその実現方法について検討することが必要である．

参考文献

[1] Pagallo, G.: Learning DNF by Decision Trees, *Proc. of the 11th IJCAI*, pp.639-644 (1989).

[2] Matheus, C. J. and Rendell, L. A.: Constructive Induction on Decision Trees, *Proc. of the 11th IJCAI*, pp.645-650 (1989).

[3] 斉藤 和巳, 中野 良平: 事例からのルール抽出: RF2 アルゴリズム, 情報処理学会論文誌, Vol.33, No.5, pp.628-635 (1992).

[4] Quinlan, J. R.: *C4.5: Programs for Machine Learning*, pp.45-56, Morgan Kaufmann, San Mateo, CA (1993).

[5] Quinlan, J. R. and Rivest, R. L.: Inferring Decision Trees Using the Minimum Description Length Principle, *Information and Computation 80*, pp.227-248 (1989).

[6] Murray, K. S.: Multiple Convergence: An Approach to Disjunctive Concept Aquisition, *Proc. of the 10th IJCAI*, pp.297-300 (1987).

[7] 田代 勤, 鷹田 憲久: 複数概念の選言表現の逐次的学習のための複合多重集約アルゴリズム, 情報処理学会論文誌, Vol.30, No.9, pp.1073-1083 (1989).

[8] Aha, D. W., Kibler, D. and Albert, M. K.: Instanse-Based Learning Algorithm, *Machine Learning, Vol.6*, pp.37-66 (1991).

[9] Aha, D. W.: Incremental Constructive Induction: An Instance-Based Approach, *Proc. of the 8th International Workshop on Machine Learning*, pp.117-121 (1991).

[10] 宮川 聡, 上原 邦昭, 前川 禎男: 概念束を用いた選言概念の逐次的学習, 平成 6 年電気関係学会関西支部連合大会, G9-46 (1994).

[11] Carpineto, C. and Romano, G.: GALOIS: An Order-Theoretic Approach to Conceptual Clustering, *Proc. of the 10th International Workshop on Machine Learning*, pp.33-40 (1993).