

論理和表現を含む複数の概念記述を同時に学習する
複合多重集約アルゴリズム

2C-7

田代 勤、 薦田 憲久
(株)日立製作所 システム開発研究所

1. はじめに

対象状態と結論の対からなる実例を与えれば、より多くの対象状態に適用できるルールを自動的に導く機械学習によるルール獲得手法が、ルール型制御システム、エキスパートシステム等の開発のボトルネックを解消する一手段として注目されている。本手法は、結論を概念名、対象状態をその具体的表現、学習後の概念記述をルールの条件として対応させた実例からの概念学習の問題として扱われる。

従来この問題は、単一概念に対し、その正(負)の実例(その概念である(でない)ことを表わす具体的表現)から、正の実例を全て満足し、負の実例を一つも満足しない論理積表現の概念記述を導く場合が主に扱われてきた。しかし、広範な現実問題への適用を考える時、(1) 複数の概念の同時学習、(2) 概念毎に独立した概念記述の学習、(3) 概念の論理和表現の学習、(4) 逐次的な学習を行えることが要求されるが、全てを満足できる方式はない。本稿では、論理和表現を含む独立した複数の概念記述を並列かつ逐次的に学習する新しいアルゴリズムを提案する。

2. 学習問題例

理解の助けのため、飛行体に関する4種類の概念の学習問題(図1)の例を説明しておく。概念及び実例は、パワー属性とウイング属性の組合せから成る概念記述言語で記述され、それぞれ図に示す属性値を取る。属性値は半順序関係 \ll を持ち、これにより概念記述の一般化関係が定義される。与えられた実例と矛盾しない、より一般的記述(上位の属性値を含む記述)を求めることが問題となる。

概念集合: {宇宙船(SS), 飛行機(AP), グライダー(GD), その他(UFO)}
 概念記述言語: $[x(\text{パワー属性}), y(\text{ウイング属性})]$
 $x \in \{\text{レシプロエンジン}(re), \text{ジェットエンジン}(je), \text{ロケットエンジン}(ro), \text{エンジンなし}(ne), \text{空気エンジン}(ae), \text{任意パワー}(*)\}$
 $y \in \{\text{主翼のみ}(mo), \text{主翼と補助翼}(ms), \text{補助翼のみ}(so), \text{翼なし}(nw), \text{主翼あり}(hm), \text{主翼なし}(nm), \text{任意翼}(*)\}$
 一般化関係: $[x_1, y_1], [x_2, y_2]$ が下記の半順序(上程大)において $x_1 \ll x_2$ かつ $y_1 \ll y_2$ なる時、後者は前者より一般的。
 \ll ; パワー属性 ウイング属性

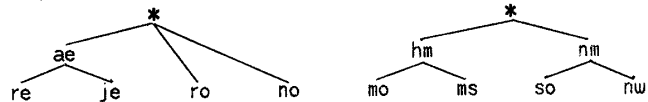


図1 飛行体に関する概念の学習問題

3. 複合多重集約アルゴリズム

3.1 バージョン空間法

上記学習問題は、概念記述言語の張る空間内を学習条件を満たす記述を見い出すべく探索する問題といえる。これを扱う枠組みに、単一概念の逐次的学習に使用され、全空間を効率的に探索できるバージョン空間法¹⁾がある。これは、与えられた正の実例を全て満足し負の実例を一つも満足しない最も一般的な論理積表現の集合を上限、最も特殊な同表現の集合を下限として保持し(上限と下限の間をバージョン空間と呼ぶ)、新たな正、負の実例が与えられる度に以下の操作1、2により上、下限をその差が縮まるように変更してゆき、両集合が一致した際の要素を学習結果として得る方法である。

【操作1】下限の最小一般化: 新たな正の実例を満足し、下限の各要素より一般的な概念記述を見出し、そのうち最も特殊な記述を新しい下限とする。また、新たな正の実例を満足しない上限要素を取り去る。

【操作2】上限の最小特殊化: 新たな負の実例を満足せず、上限の各要素より特殊な概念記述を見出し、そのうち最も一般的な記述を新しい上限とする。また、新たな負の実例を満足する下限要素を取り去る。

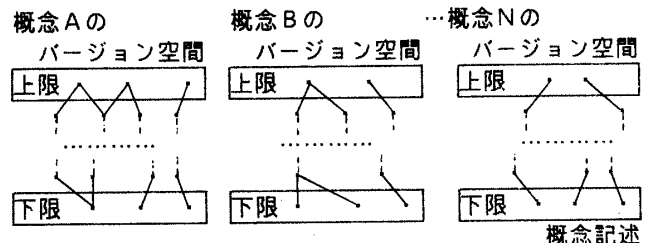
本稿の複合多重集約アルゴリズムもこの枠組みを採る。

3.2 アルゴリズムのアイデア

提案アルゴリズムは更に、複数概念を同時に扱うため、学習する概念毎にバージョン空間を持つ(図2)。

Combined Multiple Convergence Algorithm for Disjunctive Multiple Concept Learning

Tsutomu TASHIRO, Norihisa KOMODA, HITACHI Ltd.



a. 独立した上限要素の例

b. 飛行機のサブ空間の例

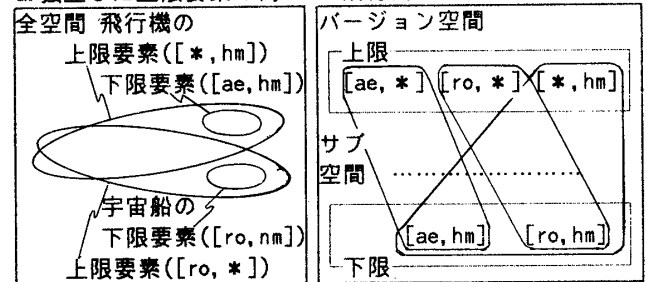


図2 複合多重集約アルゴリズムの枠組み

また、概念間の独立性の保証のため、従来に加え、上限の要素は他のバージョン空間の下限のどの要素ともオーバーラップしないという条件を持つ。例えば図1の例では、飛行機概念の上限要素 $[*,hm]$ は、宇宙船概念の下限要素 $[ro,nm]$ が満足する実例 $[ro,so]$ 、 $[ro,nw]$ を満足しないように管理される。これにより、上限と下限には含まれる空間は、最終的に共通部分を持たない概念記述に収束し、独立した概念記述を得ることができる。

更に、論理和表現を扱えるよう、一つでも正の実例を満足する概念記述は候補として上、下限に残しておく。このため、各バージョン空間は上限の要素毎に幾つかの下限の要素(全てでなくてよい)を包含するサブ空間を形成する。図1の例の学習途中の飛行機概念のバージョン空間では、実例 $[re,mo]$ 、 $[re,ms]$ 、 $[je,mo]$ 、 $[je,ms]$ を満足する下限要素 $[ae,hm]$ 及び上限要素 $[ae,*]$ が残され、 $[ro,mo]$ 、 $[ro,ms]$ を満足する下限要素 $[ro,hm]$ 上限要素 $[ro,*]$ が残されている。更に、以上の実例全てを満足する上限要素 $[*,hm]$ が残されており、それぞれのサブ空間を形成している。複合多重集約アルゴリズムは、これらのサブ空間をせばめて行くことにより最終的に複数の概念記述によって全ての正の実例をカバーする概念の論理和表現を得る。

以上の枠組みで従来同様の操作にて学習を進める際、各サブ空間がせばまるにつれ、これからはみ出る下限要素がでてくる。また、概念間の独立性を保証する条件のため、上限の要素が特殊になり過ぎて新たな正の実例をその上限が受け入れられなくなる場合がある。同様に負の実例では、その実例以外の概念の上限において必要がないのにその実例を排除してしまう場合がある。これらに対して複合多重集約アルゴリズムは、可能性のある全ての概念記述を最後まで追及するという立場から、できる限り問題を回避するように下記操作3、4を用いてバックトラックを発生させながら学習を進める。

【操作3】下限の最大分配：サブ空間からはみでる下限要素に対しこれより特殊な概念記述を求め、そのサブ空間の上限要素より特殊な記述のうち最も一般的な概念記述をサブ空間の新たな下限要素として登録する。
 【操作4】上限の最大一般化：各上限要素に対しこれより一般的な概念記述を求め、これまで与えられた負の実例を満足しない記述のうち最も一般的な概念記述を新たな上限要素として登録する。

3.3 アルゴリズムの処理フローと性質

複合多重集約アルゴリズムの処理フローを図3に示しておく。下線部分の前節で述べたバックトラック処理である。複合多重集約アルゴリズムは、下限の要素を捨てることなく最大分配により最後まで保持することにより得られた概念記述が全ての正の実例を満足することを保障している。また、上限の要素は、全ての負の実例及び他の概念の下限を排除するように最小特殊化される。上限の最大一般化時も、同様である。このことにより、得られた概念記述の無矛盾性及び独立性を保障している。更に、バックトラックを発生させ学習を進めるため、最後までバージョン空間の上限あるいは下限の要素が無くなるといった不都合が発生せず、必ずアルゴリズムが停止し条件を満足する解を得ることができる。

4. 適用事例

図1の問題に対し正の実例を順に入力した場合の結果を図4に示す。途中経過は省略してあるが、実例 viii、ix の入力時に、実例を受け入れることのできる上限の要素が存在しなくなり、図3の処理5、6の操作によりバックトラックを発生させ学習を続けている。通常バージョン空間法は、正の実例のみでは容易には収束しない。負の実例による上限の特殊化と下限の一般化とが合い待ってこそ効率的な学習が可能となる。しかし、複合多重集約アルゴリズムでは、概念間の独立性を保証する条件から正の実例を得た場合でも、他の概念の拡張バージョン空間の上限が特殊化される。従って、正の実例のみでも充分効率的な学習が可能である。

5. おわりに

提案したアルゴリズムは、機械学習手法の実際問題への応用可能性を拓けたものといえる。今後、本アルゴリズムを基に広範な対象の現実問題を扱うシステムを構築できることが期待される。

参考文献：1) Mitchell, T.M.: Version Spaces: A Candidate Elimination Approach to Rule Learning, Proc. of IJCAI '77, pp.305-310(1977)

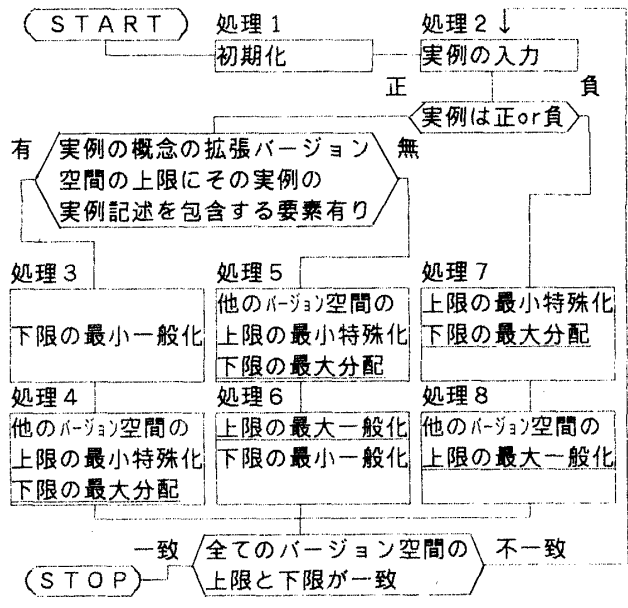


図3 複合多重集約アルゴリズムの処理フロー

入力実例： i+[[re,ms],AP], ii+[[re,nw],UFO], iii+[[je,mo],AP], iv+[[ro,ms],AP], v+[[ro,nw],SS], vi+[[ro,so],SS], vii+[[je,so],UFO], viii+[[no,ms],GD], ix+[[no,mo],GD], x+[[no,nw],UFO], xi+[[no,so],UFO]

学習結果：
 ・宇宙船のバージョン空間 ; 上下限 {[ro, nm]}
 ・飛行機のバージョン空間 ; 上下限 {[ae, hm], [ro, hm]}
 ・グライダーのバージョン空間 ; 上下限 {[no, hm]}
 ・その他のバージョン空間 ; 上下限 {[ae, nm], [no, nm]}

図4 飛行体に関する概念の学習例