題名：「学習システム」

第１章　まえがき

自然言語処理では、単語を増やし、その意味や文法情報を順次拡大していかねばならない。パターン認識でも、対象パターンを増やしていかねばならない。その数が限られているならば、人手でそれらを定義し、辞書などに登録していけばいいのであるが、実際問題、実用レベルで、数千、数万というデータを用意しなくてはならない。そして、それらはどんどん増加していくものなのである。とても、個人の力では対応できないし、しかも単純労働であることから、人間が行うのは問題である。

このように、人工知能では、計算機が自分で知識を獲得できるようにしなければならないことが沢山ある。しかし、こうした自己学習問題は、非常に難しいのが現状である。処理手続きをどう学習させるか、処理手続きを勘案した有効なデータ（知識）をどう組織だって獲得していくか、問題を考えれば考えるほどに、それが複雑に見えてくる。実際、考えなくてはならないことが沢山あり、一つ、二つ仕組みを考えただけでは克服できないものである。

本論では、できるだけ単純に知識というものを捉え、自然言語処理とパターン認識で用意すべき学習機構は基本的にどうであるべきか論じてみた。それは一言で言うと、学習システムは推論システムとカテゴリ管理システムからなると捉えるものである。だから、推論システムとカテゴリ管理システムを詳しく分析していくつもりである。

第２章　推論システム

推論システムは学習の基盤である。これの実現を考えてみたい。

２．１　推論システムの基盤

推論を次の３つを組み合わせた技術として捉えたい。

（１）集合論
（２）対応付け（コネクション）
（３）検索

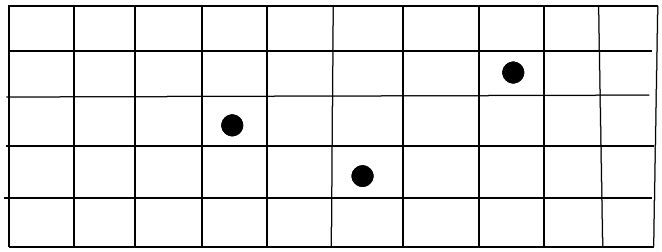
（１）の集合論は、推論の中で現れる要素をマップの要素に対応付け、集合論演算を実現するものである。

マップは１次元のマップ、２次元のマップ、３次元、Ｎ次元と考えられる。しかも有限個の要素を収納する有限マップと無限の要素を収納できる無限マップがある。マップは単独で集合演算の場として機能するほかに、更なる詳細を表すマップや大局を表すマップへの対応付け（コネクション）を持つこともある。その他、関連情報へのコネクションも沢山もつのが普通である。

集合論の演算としては、

（１）∀　（全ての）
（２）∃　〔ある～が存在して〕
（３）partof（部分）

（４）restof（残余）
（５）∩
（６）∪
（７）－（マイナス）
（８）＋（プラス）
（９）A → B（ならば）
を用意する。



● ：コネクションのある要素

図２．１　２次元マップ例

検索は、マップや、知識を表現したデータエリアであるフレームのコネクションをたどって、目的のデータを探す操作である。マッチングパターン（検索キーとなるデータ）を探し歩き、指定の情報を取り出すことを行う。プログラムでこれを実現するには、何を探すかということ（マッチングパターン）と、何処を探すかということ（マッチングパターン）とをパラメータに指定し、目的のデータオブジェクトを返す、というインターフェースを採用することになる。

集合論演算システムとコネクションシステム、検索システムは学習システムの一部として、作りつけで用意する。これらは、プリミティブプロセスとして、学習システムのベースとなるものである。

２．２　一階述語論理との関係

推論システムの代表的なものに一階述語論理がある。これを２．１で述べた、技法でどう表現できるかを示す。

一階述語論理の構文は以下のように定義されている。

（１）Sentence → Atomic Sentence
| Sentence Conective Sentence
| Quantifier Variable,・・・Sentence
| ^Sentence
|(Sentence)

（２）Atomic Sentence → Predicate(Term,・・・)|Term=Term

（３）Term → Function（Term,・・・）
　　|Constant
　　|Variable

（４）Connective →=>| ∧ ¦ ∨ ¦ <=>
（５）Quantifier →∀¦ ∃
（６）Constant → A| X1 | John|・・・
（７）variable →　a| x| s|・・・
（８）Predicate → Before| HasColor| Raining|・・・
（９）Function → Mother| LeftLegOf|・・・

推論規則例）∀x　NaganoCitizen(x)→Japanese(x)
　　NaganoCitizen(June)

　　Japanese（June）

推論規則の説明）全ての x について、NaganoCitizen ならば Japanese である。
　　June は NaganoCitizen である。
---（推論すると）
　　June は Japanese である。

この一階述語論理を解決するプログラムは、人手で、（１）から（９）の要素を表現したフレームを作り、さらに推論規則を表すフレームを作り込み実現する。実行時にパラメータに指定の値（例では x に June を入れている）を代入してこのプログラムを実行する。処理系は推論規則をくまなく検索し、最後に変数のなくなった、それ以上「=.>」の無い規則（葉という）を結果として出す。

Prolog の処理系や、一階述語論理処理プログラムを読めば、それが検索を駆使して推論問題を解決していることがよく分かる。

例をフレームで表したものを下図に示す。

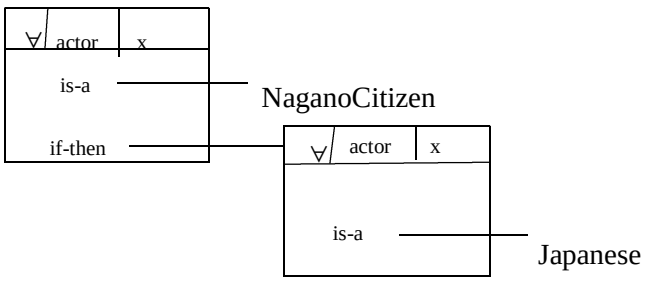


図２．２　推論フレーム例

検索が主な処理であるが、途中のデータ管理の処理として、「NaganoCitizen というマップに June が属する」というように、集合論演算が行われる。「全ての x は・・・である」なんて表記も集合論演算である。

推論にはレベルがある。概略は次のようなものである。

（1）連想レベル

データの共起関係で、次々に関連項目を提示していくもの。

「以下の条件の時」では、「以下」と「の」、「条件」、「の」、「時」の各単語が次々に連想して一つの意味を表している。

（2）構造レベル

データに構造があるときに、行われる推論である。文法とか、簡単な意味構造とかを持つデータを解析する。

「私は昨日、学校へ行った。」は主格（私は）、副詞（昨日）、目標格（学校へ）、動詞（行った）からなる文であることから、文を理解する。

（3）解析レベル

曖昧性処理など、知識を動員して、問題を解決するもの。

「空を飛ぶ船から見たカモメは・・・」の「空を飛ぶ」のは「船」なのか「カモメ」なのか。知識で解決しなければならない。

（4）思考レベル

アルゴリズムを組み立て、問題を解決していくもの。

レベルが深いほど、その推論システムを実現することは難しくなる。

第３章　カテゴリ管理システム

自然言語システムでは、単語は品詞分けされ、品詞も処理の中で句とか節とかの作業品詞がつけられたりする。また、意味としても、動物とか植物、鉱物など必要に応じて分類しなければならない場合がでてくる。

パターン認識も、インスタンスとして入力されたイメージは何かという判定をして、特定のカテゴリに分類していく。

人工知能はこのカテゴリ分けを自動学習していくべきである。手作業では限界がある。プリミティブのみを人手で作って、種として、あとは人工知能が管理していくというかたちになる。このため、カテゴリ管理を学習のキーポイントとして考察していく。

３．１　カテゴリの新規性の判断

新しいカテゴリが要求されたら、人工知能は、それを作らねばならない。新しいと判断するのはどんなきっかけからだろうか。考えられるのは、

（１）今までと違う操作が必要になった。

自然言語だと、単語の文法が今までと微妙に違う事が明確になった。それは文章のコーパスを解析していて、どのカテゴリにも入らないことから判断される。

パターン認識では、既存のカテゴリに入らないとか、イメージとは別の情報から、既存の分類では間違いであることが示された、といったことから判断される。

（２）既存のカテゴリで良好なのだが、更にサブセットとして分離できるような固まりとなっていることが判明した。特殊化として、サブカテゴリを設けるべきであろう。

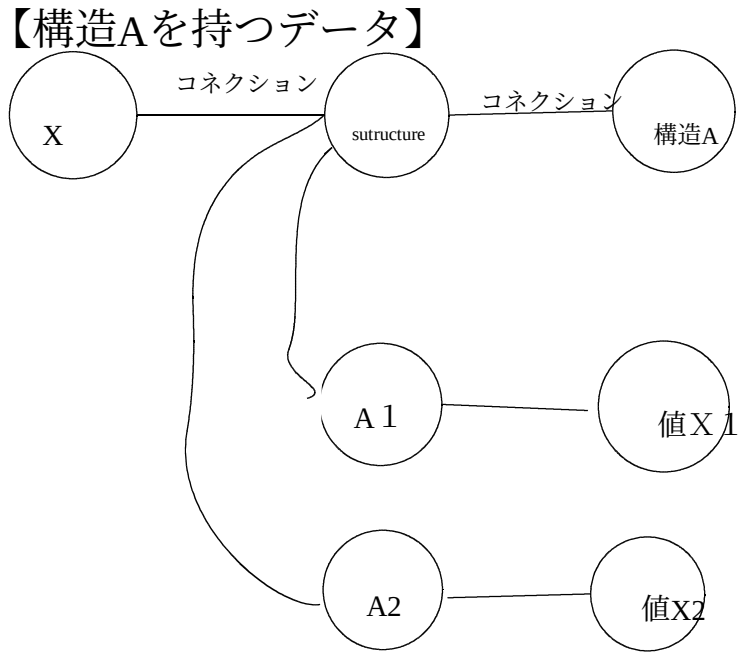
（３）幾つかのカテゴリをまとめた方が、大局的に情報処理が円滑になることが判明した。この場合は汎化として、スーパーカテゴリを設けるべきだろう。

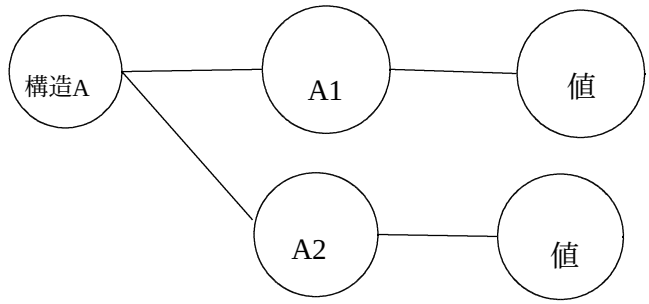
新規性の判断は、専用の辞書を設けるべきである。全文検索では時間が足りなくなる。カテゴリやカテゴリセットをキーにして、定義情報をデータにして、辞書を作っておく。コネクションもキーにすることも工夫の一つだろう。とにかく、高速に新規性の判断ができるように辞書、辞典などを設けることになる。

３．２　カテゴリ構造の管理

カテゴリ群は、サブセット、スーパーセットといった構造を持つ。また、関連カテゴリとか、反意カテゴリなどなど人工知能が生きていく上で、次々と新しいカテゴリができていき、しかも単なるコネクションでは情報が不足するものである。それはカテゴリが構造を要求するからである。左右配置とか、上下、周期性があるなどなど、きりがない程の構造情報が必要になる。その構造を踏まえて、カテゴリを管理していくのである。

では、カテゴリの構造は何で示すか。それは構造を示すフレーム情報（必要に応じてボクセルを用いる）をプリミティブでもって、カテゴリ間のコネクションにその構造へのコネクションを持たせることで実現する。こうすれば、人工知能が柔軟に構造を選択し、利用できるようになる。





【説明】

・下の図の構造 A は要素として A1、A2 をマップとか、フレーム、ボクセルで持っている。

・上の図のカテゴリ X は構造 A を持つことを、構造 A へのコネクションで示す。
構造要素 A1 に値Ｘ１を構造要素 A2 に値Ｘ２を持つ。

図３．１　カテゴリ構造の管理例

第４章　学習システム

「学習」を考える際、データを学習するのか操作（プロセス）を学習するのかを分けて対応していかねばならない。データはコネクション（連想）で対応できるが、操作はアセンブルするプリミティブ操作とそのシーケンスを厳密に定義していかなければならないからである。

確かに、データの学習もそのデータによって操作をよりよく制御していくことが目標だから、操作の学習と似ている点は多々ある。

（１）データの学習も操作の学習も、人工知能があらかじめ持っているプリミティブ操作はその学習に決定的な役割を演じる。つまり、プリミティブ操作の組み合わせが覆える範囲しか、能力は達成できないのである。プリミティブ操作は学習システムの限界を決める。

（２）データ学習は操作学習の制御部を抽出したものと同じである。操作学習は操作の実行部も学習によって賢くなろうというものである。

（３）自然言語処理での学習は記号世界のことであるから、データ学習が主になる。

（４）パターン認識はパターンというデータを学習するが、パターンを解析するプロセスを学習によって作っていく必要があるので、操作学習も大きく入ってくる。

（５）運動学習は操作学習がメインとなる。

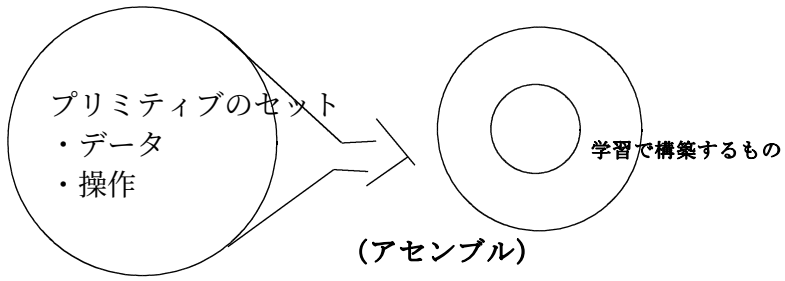


図　４．１プリミティブセットと学習（アセンブル）

４．１　自然言語処理での学習

「りんご」の学習を考えてみよう。

（１）赤いとか、すっぱいといった、属性の学習がある。

「赤い」、「すっぱい」 は視覚、味覚からの情報があって初めて人工知能が処理できるようになるデータである。ここから自然言語は教師がなけてば成り立たないものという判断が出る。しかし、視覚系や味覚系を装備したとして、そのパターン認識の結果をシンボル（カテゴリ）化して、例えば red とかの記号で「赤い」を表現するという、コネクションを作り、更に「りんご」の知識フレームの説明項目に、「"りんご" is red」を加える。

そもそも、「りんご」はパターン認識システムと連動して、りんごイメージを「りんご」

と記号化しなくてはいけない。
（２）文法規則として、品詞がある。
　これはパターン認識システムで、物体を表す情報を得るので、それをもって、名詞と判断する。ちなみに、動きなら、動詞と判断する。属性ならば、形容詞、副詞とする。
（３）植物であるとか、果物であるとかのカテゴリ構造がある。
　基本的に教師あり学習になるが、既に「ぶどう」が学習済みであれば、「ぶどうと同じもの」という情報だけで、「ぶどう」と同じコネクションを引き継ぐという方法で、学習効率を高めるようになっているべきである。

「与える」の学習を考えてみよう。
（１）意味の学習が必要である。
「与える」はパターン認識系で動作を表すの情報を得て、品詞として動詞を表すことが分かったとする。
その「与える」の前後関係を解析する必要がある。
・「与える」の前の状態：I have a ball.
・「与える」の操作：I give it to her.
・「与える」の後の状態：She has it.
意味処理で、著者は、動画システムを提案したが、それを用いて、この意味解析をしていく。give のプリミティブとして、move をとり、have-a 状態の change であるという、意味を抽出して、「与える」の知識フレームに記す。
（２）格の学習が文法上必要である。
それは、文章コーパスを解析して、direct-object に「を格」を、target に「に格」を取ることを抽出する。

意味プリミティブについては、論文「意味処理」でふれた。
自然言語文法プリミティブは次のようなものである。
（１）配置
top　：文頭
tail　：文末
before　：前方
just before　：直前
after　：後方
just after　：直後
N th　：N 番目の位置
（２）機能
modify　：修飾する
modified　：修飾される
pass throgh　：通過越える
!　：否定

（３）品詞

SS	：文、節
PP	：句
XX	：助詞
ND	：名詞
VB	：動詞
CM	：コンマ
CJ	：接続詞
AN	：形容詞
AV	：副詞
AX	：助動詞
??	：その他

データから有益な特徴を抽出する技術はデータマイニングである。既存のデータマイニング技術を応用し、さらに自然言語処理やパターン認識固有の問題に向けた技術を開発していこう、というのが本論文の目的でもある。

４．２　パターン認識での学習

人間のパターン認識を内省してみると、大枠のパターンというものは生得的に持っているようだ。目とか人の顔、人体表面に関することは教師なしでも判別できた。それと、森とか山、川、地面に空。すなわち生活圏も教師無しで判断できた。教えられて認識したのは、文字とか、人や生活圏の微妙な違いを区別することだった。

人工知能もそうするのが自然だろう。例えば、

（１）「円が２つ横に並んでいたら要注意。目である可能性大。」という情報を持つ。

（２）「閉曲線は藪（葉のかたまり）の可能性大。」という情報を持つ。

その他、動物の形とか、川、木といったものの基本パターンを持つ。学習はそれらの図形の変形、詳細化過程であると考える。無論、全く別のものになることも考えられる。動物から自動車、汽車が生まれても良い。それは別カテゴリにするが、その大枠は動物のパターン認識処理を利用する。

基本戦略はそのように大局を掴んでおこなうのであるが、そのベースとなるプリミティブは、微細にパターンの特徴を解析していくものでできていなくてはならない。あらゆる事柄に対応していくためである。

パターン認識も上記のようにカテゴリシステムを構成していくことが、学習の大きな作業となる。このカテゴリシステムは自然言語処理と同じである。

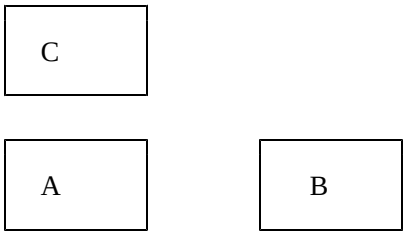
パターンのプリミティブについては、論文「パターン認識」でふれた。

４．３　操作（プロセス）の学習

操作の学習は試行錯誤をベースに行う。推論も行っていく。試行するとき、あってほしい状態を羅列し、今の状態からそこにたどり着く操作を既に持っているかを検索するのである。利用した操作のパスを記録し、それを学習データとする。目的が完遂しなかったら、プリミティブ操作を探し、それを組み合わせる処理で操作を作り出し、それで目標を完遂しようと試行する。上手くいけば、それを新しい操作に組み込む。

積み木の例で考えよう。次に示す状態 A を状態 B にしたい。

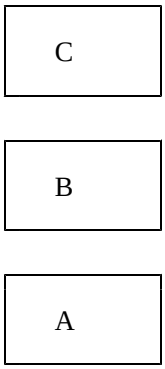
【状態 A】



----------(テーブル)

積み木 A の上に積み木 C が乗っている。積み木 B は離れてテーブルに乗っている。

【状態 B】



----------（テーブル）

積み木 A の上に積み木 B が乗り、その上に積み木 C が乗っている。

図４．２　積み木問題の例

人間にとって、直ぐに分かることは、状態 A で積み木 C を積み木 A から取り除き、一旦テーブルに置き、次に、積み木 B を積み木 A の上に置き、最後に積み木 C を積み木 B

に置くということである。
　これを、計算機に推理させるとしたらどうであろうか。更に、推論も学習によって獲得しなければならない能力だったら、どうであろうか。

　操作のプリミティブが必要である。それは、
（１）アームの左右への動き
（２）アームの上下への動き
（３）積み木の保持
（４）積み木の解放

　センサが必要である。それは、
（１）積み木の認識
（２）積み木の位置の認識
（３）空席の認識

　プリミティブな推論能力
（１）マップシステム（積み木の位置を把握する）
（２）集合論演算能力

　これだけあって、やっと問題が解けるのである。その手順は、
（１）センサによって、積み木の位置をマップシステムに記録する。
　マップは、状態 A と状態 B と作業用のものの３つ必要である。
（２）作業マップで、積み木 A の上に積み木 C があって、積み木 B を移動できないことを認識する。
（３）試行錯誤で、積み木 C をテーブルに置くと、積み木 A の上が空席になることを知る。
（４）積み木 A の上に積み木 B を置く。
（５）積み木 B の上に積み木 C を置く。

　問題が解決したら、（３）の工程を学習によって、改良する。「目標の操作を阻害されたら、それを取り除く」という知識を獲得するのである。それは試行錯誤のなかで行った行動をトレースして、最短パスを記録し、学習された操作とすることである。

　基本的に、プリミティブに無いことはできないので、プリミティブセットの設計は神経使うことである。

第５章　学習データ収集システム

学習機構の必要性が切実な問題となっているのは、自然言語処理であろう。何万、何十万という単語や相当語を用意しなくては、使いものにならない。文法や意味情報を用意するとなると、人海戦術も限界を感じる。単純労働は避けるべきであろうし。

そこで、自動学習システムを是非とも作りたい。それはどんなものであるだろうか。まずは、パターン認識系を持った、完全な自己学習システムは必要だ。一方で、教師あり学習の効率を高めるものも欲しい。シンボルの世界になれば、イメージのパターン認識は必要なくなるわけで、記号世界でのパターンを解析することに心血を注ぎたい。

イメージのパターン認識は人間と一緒に生活するロボットとして実現するのが素直だろう。

教師ありの学習システムは記号入力が効率よくできれば良いだけなので、学習エディタで、自然言語入力などがスムーズに行え、コーパスの収集、分類が手際よく行えるものが欲しい。パターン認識の記号処理部分も操作できればなおよい。

自然言語処理には、とにかく、学習システムを早急に用意する必要があるのである。

第６章　まとめ

研究段階を越え、実用的な自然言語処理システムを作っていこうとしたとき、大きな壁となるのは、大きな辞書すなわち知識データをどうつくっていくかということである。３千語ぐらいならなんとか用意できるが、それ以上は体力的にも時間的にも無理である。しかも、自然言語処理は曖昧性処理など多くの難しい問題を抱えており、システムはスクラッチアンドビルドを繰り返して成長させるしかない。プログラムはなんとかできるが、データの用意はまず無理。

学習システムは、推論とカテゴリをどう作っていくか、どう自動化するかという問題である。そこでは、プリミティブ（データもプロセスも）設計と、学習システムを管理運営するメタデータ、メタプロセスの設計が重要な解決課題となる。

学習システム自身は、データの獲得、プロセスの獲得による成長といったマンパワーの問題はない。自然言語処理やパターン認識では、データを大量に獲得し、成長していくので、マンパワー的に対応が困難である。だから、まず学習システムを作り、自然言語処理やパターン認識処理に展開していくのが王道というものだろう。

著者はそれで、学習システムの攻略が先決と判断し、パターン認識の勉強をし、自然言語処理研究に結びつけてきた。本論はその一端を紹介するものである。ここでも、超並列計算機でプログラミングできたら、もっと研究が進むのにと現状を嘆いたものである。早い時期に計算機環境が大きく前進することを望む。

以上