自然言語処理 89－8
（1992. 5. 15）

事例ベースを用いた発見的規則制御の最適化

野美山 浩† 諸橋 正幸† 細野 公男‡ 原田 隆史‡
梅田 栄廣⋆ 関根 さゆり⋆⋆

†日本アイ・ビー・エム株式会社 東京基礎研究所
‡慶應義塾大学 文学部図書館・情報学科
⋆ 日本電気
⋆⋆ 日立製作所

自然言語処理システムは、一般に非常に多くの曖昧な規則から構成されている。規則間の制御の流れを明示的に記述したシステムにおいては、多くの規則を含んだ規則系全体を常に大局的に把握する必要があり、その開発および使用環境への適合化が困難を極める。システムの精度を上げるには、個々の環境におけるシステムの適合化が避けられない以上、個々の規則を独立なものとして捉え、個々の使用環境毎の規則と一般的な規則が大局的な有効性の尺度から統合化されるような機構が必要となる。本稿では、規則間の制御を正解の集合である事例ベースから計算することによって、与えられた環境において規則の精度を向上させ、効率的な規則系を生成するための機構を提案する。

Optimization of Heuristic Rule Control by Using Case Base

Hiroshi Nomiyama†, Masayuki Morohashi†, Kimio Hosono‡, Takashi Harada‡,
Eiko Umeda⋆, Sayuri Sekine⋆⋆

†IBM Research, Tokyo Research Laboratory
5-19 Sanbancho, Chiyoda-ku,Tokyo 102 Japan

‡School of Library and Information Science, Keio University
2-15-45, Mita, Minato-ku,Tokyo 108 Japan

⋆ NEC Corporation

⋆⋆ Hitachi Ltd.

Natural language processing systems consist of sets of huge numbers of heuristics rules, which are not always true. It is difficult to develop or customize the systems, where control flows among rules are explicitly declared, bacause developers must grasp the whole flow of systems. In addion, users must costomize systems for every environments where they are used. It is nessesary to integrate general rules and environment-specific rules effectively from a global view in each environment. In this paper, we propose a mechanism to costomize systems for a given environment by using casebase.

1 はじめに

現状の自然言語処理システムは、使用環境に依存した多くの細かい発見的規則によって構成されている。実用的なシステムにおいて精度の向上に貢献しているのは、個々の細かい発見的規則であり、確実に成立する、本来の意味での、規則はほとんど存在しない。

ほとんどのシステムでは、このような多くの発見的規則間の制御の流れを明示的に記述している。そのようなシステムの開発・保守は、常に巨大な規則系を大局的に把握する必要があり、困難を極める。例えば、ある規則を加えた場合、それが結果にどのように影響するかを見極めるには、全体の制御の流れをたどって調べる必要がある。あるいは、ある特定の入力について特定の規則を適用させたい場合も同様に局所的には解決できない。また、一般に制御の流れが変わることによって規則系が与える結果は変化する。しかし、制御の流れの記述の有効性を判断するための明解な指針はなく、各々の開発者の直観によっている。

このような開発・保守の困難性に加え、運用上の問題として、多くの例外的な規則の適用によって生ずる非効率性の問題がある。ほとんど成立することのない規則を、処理の都度評価することで、全体の処理効率が低下する場合がある。このような非効率性の解消にも、明確な指標はなく、開発者自身が経験的な直感に基づいて行っている。

規則の多くは使用環境に依存したものであり、ある特定の環境に対して記述した規則が別の環境で有効であるか、あるいは、効率的であるかは全く保証されない。また、規則が決定すべきこと、および、規則が与える結果の重要度も環境によって異なる場合がある。

使用環境間の偏りが大きいのであれば、個々の使用環境における、規則の獲得・適合化は必須となり、それらを容易にする手段を提供しなくてはならない。使用環境毎に記述された規則を有効に活かすためには、システム開発者が作成した規則と環境毎の規則が使用環境別に効率的に統合されるような機構を考える必要がある。そのためには、個々の規則の有効性および効率性が、使用環境において大局的な尺度から評価されなくてはならない。さらに、規則そのものに対してだけではなく、規則間の制御も使用環境に適合するように、分野に依存した情報源から動的に決定される必要がある。

本稿では、与えられた環境の元で規則集合が与える結果の有効性を向上させ、かつ、制御の流れを効率化する手法を提案する。規則間の制御の流れを使用環境に応じて動的に決定するために、個々の発見的規則を独立した規則として捉える。規則間の制御は与えられた環境(正解の集積である事例ベース)、および、規則が与える結果に対する評価関数から計算する。規則適用の順序だけでなく、1つだけでは有効性が低い規則を、同じ結果を与える規則と組み合わせることによって、より有効性のある規則として解釈する。

また本手法では、規則作成者が定義する規則は、条件とそれによって決まる属性を定めるものであり、属性の値は記述しない。すなわち、"ここときこを調べればこれの値が決まる" というように定義し、具体的な値は記述しない。その値は事例ベース中の個々の事例から抽出される。

また、本手法では、規則作成者は規則が特定の環境に対して有効であるかどうかを気にせず記述できる。規則集合を最適化することによって、最終的には有効な規則のみが選別される。

従来のほとんどの自然言語処理システムは個々の規則とそれらの間の制御の流れを明示的に記述し、規則は独立なものとして解釈されない [1] 。このようなシステムでは規則間の制御を分野に適合化させるために動的に変えることができない。事例ベースから抽出した確率を利用するシステムも提案されているが [2] 、これは最終的な候補の選択を行なうためであり、制御の最適化は行なわない。事例を用いて自然言語処理を行なうシステムが提案されている [3] 。これは個々の事例を規則とみなし、それら規則の独立性を保証し、制御は類似度から計算する。制御を明示的記述以外のものから得るという点で同様であるが、同じ結果を与える規則を組み合わせて、より確からしいと判断するものでない。また、環境に応じて、制御の流れを変化させるには、類似度を与える体系を変えることによって可能であるが、頻度を考慮することで使用環境別に効率化を図るものではない。

2 事例を用いた発見的規則制御の最適化

2.1 規則の結果に対する評価関数

本手法においては、規則は、それが決定する属性毎に記述される。規則は、条件部とそれが与える値よりなる。規則の値は、ある使用環境においてその条件部が成立した時、属性が取る値の分布を与えるものである。この値は、事例ベースより計算されるので、規則記述時には指定する必要はない。

ここで、本手法の基本的な考え方を説明するために1つの例を考える。事例ベース全体において属性1の値が図1のような分布を持ち、規則 A の条件部が成立する場合は、図2のような分布を与えたとする。

この規則が与える分布は、全体の分布が与える空間と比較して Val2 の方へ歪んでいる。その意味でこの規則は、 Val2 の性質を表していると言える。つまり、規則記述者は Val2 を検出する目的で規則を記述しているのかもしれないが、その意味においては、40%の

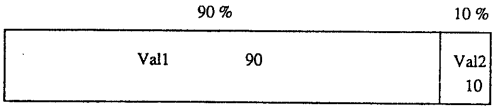

図 1: 値の分布

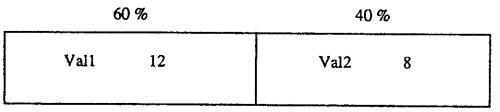

図 2: 規則 A の条件部が成立する場合の値の分布

精度しか得られていない。一方、規則 A が Val1 であるという結果を与えると考えると、60%の精度を持つ。つまり、規則系全体が与えるの精度の観点から見ると、この規則は明らかに精度を下げることになる。しかし、もし、Val2 である場合を漏らさず検知するためのシステムであるならば、規則 A は有効であるといえる。

このように、規則が持つ性質とそれが与える結果に対する精度は別にして考える必要がある。開発の過程においては規則が表す性質を考慮することは重要であるが、それを実際に適用するにはその規則はない方が良い精度を与える場合がある。

本手法では、規則が適用される環境、および、それが与える値に対する評価基準を評価関数によって定義する。規則評価関数は特定の規則が特定の環境で与える値が正解する確率を与えるものとする。一般的には、その規則が与える結果の分布の事例ベースでの最も大きな確率がその規則の評価関数として定義される。しかし、値として何を取ることが重要であるのかは個々の問題によって異なる。Val1 であることが検知できればよいといった場合は Val1 である確率あるいは Val1 でない確率の和のどちらか大きな方が評価値として採用される。

評価関数は、規則が成立した場合の分布から計算される。例えば、Val1 を検出する目的の規則系において、規則 A が環境 E 中で一致した場合の分布を

```
(Val1 = 0.3, Val2 = 0.5, Val3 = 0.2)
```

とすると、規則 A の環境 E での評価関数の値 $(EV(A,E))$ は、Val1 でない確率 (0.7) となる。

2.2 規則集合の最適化

前節で述べた規則の集合を、評価関数および事例ベースを用いて、最適化を行なう（図3参照)。まず、最初に規則が与える結果の有効性の観点から有効な規則の選別および組み合わせを行ない、次に、実行時の効率の観点から得られた組み合わせ規則集合を最適化する制御の流れを決定する。

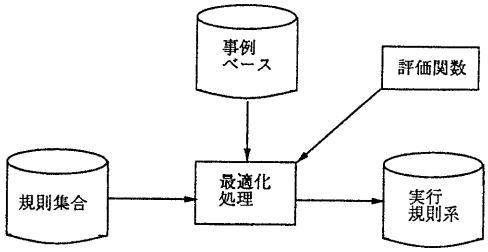

図 3: 最適化処理

2.2.1 有効な規則の選別および組み合わせ

ある規則が使用環境において有効であるかどうかは、規則が全くない場合の分布と比較することによって決定する。すなわち、恒真の規則が与える評価値（事例ベース全体の値の分布に対する評価値）よりも大きいかどうかによって判断される。ある規則の評価値が恒真の規則の評価値よりも低ければ、その規則は規則系に反映されない。

しかし、1つの規則として見ると有効とされないが、いくつかの規則を組み合わせ、条件をより厳しくすることによって有効になる場合がある。

2つの規則 A,Bを考えた場合、その組み合わせは、$A\&B, A\&\bar{B}, \bar{A}\&B, \bar{A}\&\bar{B}$の4通りあるが、規則が一致した場合と、それが一致しなかった場合をそれぞれ別の規則と考えることによって、$A\&B$の組み合わせのみを考える。

規則は、組み合わせることによって評価値が向上しなければ組み合わせを行なわない。規則 A と規則 B を組み合わせる場合、$(EV(A\&B,E) > EV(A,E))$ & $(EV(A\&B,E) > EV(B,E))$ ならば規則 $A\&B$ を組み合わせ規則として追加する。ただし、組み合わせの途中で組み合わせ規則の評価値が1になった場合は、それ以上の組合せを行なわない。

このようにして、規則集合中の事例ベース中で条件が一致した規則をすべて組み合わせ、事例ベース全体の値の分布に対する評価関数の値より大きい評価値を持つ規則を採用する。

最後に、すべての規則が一致しなかった場合の省略値を設定する。これは環境で恒真の規則に対する値である。

2.2.2 実行規則系の生成

次に、実行時の効率の観点から、得られた組み合わせ規則の集合の制御の流れを最適化する。

規則が効率的であるかどうかは、その規則が、使用環境において、より高い頻度で成立するかどうかによ

る。効率性は、組み合わせたものでない規則が成立する頻度によって判断する。この基準に従い、成立する確率が高いほど、早く規則を評価するような制御の流れとなるような実行系を生成する。

このような規則系を生成するために、まず、規則集合の規則を事例ベース中でその条件部が真になった回数が多い順に並べる。その順に組合せ規則の集合を並べ替え、並べ替えた順番によって Top-Down,Left-to-Right に、図4に示すような3進木を生成する。

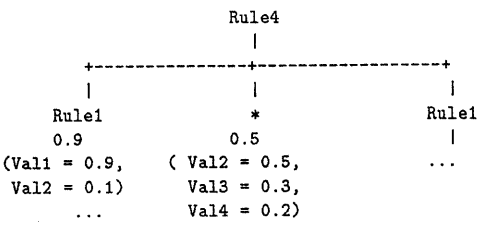

図 4: 3進木構造

ノードの中で、規則名を持つものは、その条件部を評価し、その値が真である場合は、左のリンクをたどる。たどった先のノードはその条件部が真であった場合の評価値を持つ。例えば図で、ルートノードの最も左の子ノードは *Rule4* の条件部が真である時その評価値が 0.9 でその値が (Val1 = 0.9, Val2 = 0.1) であることを示している。偽である場合は真中のリンクをたどる。その規則と独立である場合は左のリンクをたどる。

2.3 規則処理系

規則処理系は、最適化処理で得られた実行規則系を解釈する (図5 参照)。条件部解釈結果リストは、途中で評価された、組み合わせたものでない規則の解釈の結果を保持するために用いられる。評価値リストは、途中で評価された組み合わせ規則の評価結果を保持するために用いられる。

3進木においてノードに規則名がある場合は、まず条件部解釈結果リストを調べ、すでにその規則の条件部が解釈されているかどうかを調べる。もしリスト中にあった場合は、その結果を用い、ない場合は、条件部を解釈し、その結果を条件部解釈結果リストに加える。

条件部を解釈した結果が真なら左のノードをたどる。そのノードに評価値が設定されていて、その値が1ならば、その値を結果とし、処理を終了する。1未満ならば、その評価値と規則の値の対を評価値リストに加える。

条件部を解釈した結果が偽なら中央のリンクをたどる。そのノードに評価値が設定されていて、もしその値が1ならその値を結果とし、処理を終了する。もし1未満ならばその評価値と結果の対を評価値リストに加える。

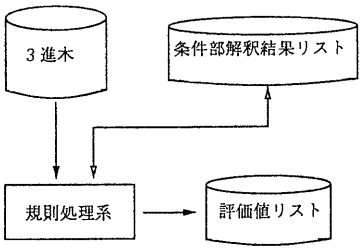

図 5: 規則処理系

このように条件の適用を、3進木中で Top-Down,Left-to-Right に走査する。途中で評価値が1の場合は、そこで処理が終了する。3進木をすべて走査し終えた後、評価値1のものがなかった場合は、最後に評価値リストを調べ、最も高い評価値をもつ値を結果とする。もし評価値リストが空であったならば、省略値を結果とする。

3 実験

3.1 実験の目的・前提

本手法を文のタイプを識別する問題に適用し、実験を行なった (規則および結果の詳細は [4] 参照)。

これは、抄録からのキーワード自動抽出において、抄録中で文献に記述してあることを述べている文 (主題文) を抽出することによって、その文のキーワードを重視し、キーワード抽出の精度を上げようという試みの一環において行なったものである。

入力として、JICST の抄録を用い、入力文の構文解析木に対して規則を適用し、それを、出典, 前提文, 主題文, 説明文, 方法文, 結果文に分類する。

最終的な目的は、抄録の中から主題文を抽出することにある。そのため評価関数として、主題文である確率と主題文でない確率のどちらか大きな値を取ることにした。

実験対象として、最適化のために用いる訓練標本 (1,060 文) と、それによって得られた実行系を評価するための評価標本 (536 文) を用いた。標本における文タイプの分布を表3に示す。

規則は全部で 129 となった。規則作成者から見て、規則がどの文のタイプの性質を捉えているかを指定した内訳を表1に示す。個々の規則は原則として各々の

表 1: 規則の内訳

規則が検知しようとする文のタイプ	規則の数
主題文	69
?	20
結果文	18
前提・結果文	8
前提文	7
方法文	2
説明文	2
出典	1
主題・結果文	1
前提・説明文	1
合計	129

文のタイプの性質を捉えたものである。しかし、どの文タイプの特徴であるかは、明らかではないが、何らかの文の特徴を捉えている性質を表している規則も存在する。そのような規則は表中?で示している。

目的は、主題文を抽出することであるため、主題文の特徴を表している規則が最も多くなっている。

3.2 実験

これらの規則に対し訓練標本を用いて、本手法によって組み合わせ規則を求めた結果、911 通りの組合せ規則が求められた。最初に記述した、組み合わせたものでない規則 (129) のうち、10 個は有効性が認められず、実行系に組み入れられなかった。

組み合わせ規則の例として、得られた組み合わせ規則のうち、訓練標本中で最も頻度の高かった規則を以下に示す。

$\bar{Rule}3\&Rule94\&Rule99 \qquad (SYUDAI = 1.0)$

この規則は、主語がなく ($\bar{Rule}3$) 、かつ、時制が過去 ($Rule94$) 、かつ、動詞の意味分類が "表現" であることを意味している。これは、主題文における典型的なパタンを表している。このようにいくつかの規則においては有効な組み合わせが得られた。

しかし、少数の例外を救うために、あまり有効でない規則の組み合わせが生じた。例えば、訓練標本中で $Rule55$ が成立する場合は、前提文 1, 主題文 135 となった。このただ一つの前提文となった例外を救うために、その例外で成立しない規則のいくつかが組み合わされた規則が生成された。

次に、これらの組合せ規則から規則実行系を生成し、それを用いて訓練標本自身および別の評価標本に適用した結果を表 4に示す。

まず、規則の適用回数についての結果を考察する。

表 2: 評価標本に対する文タイプ別の規則評価回数と抽出率

	1 文当たり規則評価回数	抽出率
出典	20.0	100.0
前提文	69.1	71.2
主題文	59.8	91.2
説明文	87.2	9.0
方法文	86.7	13.2
結果文	74.7	13.5

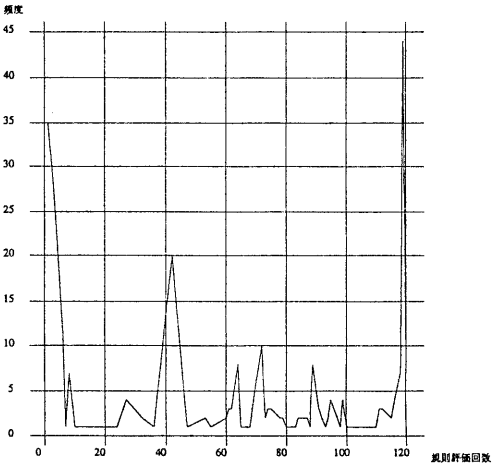

図 6: 規則の評価回数別の頻度 (主題文のみの場合)

文タイプ別の 1 文当たりの規則評価回数とその抽出率を表 2に示す。

出典[1]に次ぎ、主題文に対する規則の評価回数は少ない。また、抽出率が高いほど、規則評価回数は少なくなっている。

規則の評価回数について、主題文のみを取り出した場合について見てみる。図 6は、主題文について、規則の評価回数と、その出現回数を示したものである。規則の評価回数 119 のピークは、全ての規則が評価されたことを示す。また、規則評価回数 120 は、省略値の評価値が用いられたことを意味する。

図 7に、規則評価回数と、その精度および累積精度を示す。精度は、かなり波があるが、累積精度は緩やかに下降している。すなわち、評価回数が少ないほど精度が高い傾向があることを示している。

次に精度について考察する。主題文適合率は訓練標本に対しても、最大精度 (付録参照) よりも低い精度

[1]この文は非常に形式的であり、ほとんど一意に決定できる

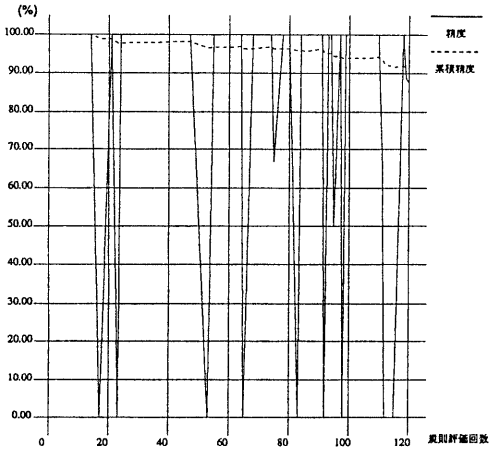

図 7: 規則の評価回数別の精度および累積精度（主題文のみの場合）

を示した。すなわち、再現性は100%ではないことを意味している。これは、評価値の低い規則が実行規則系に入れられなかったことによるものと考えられる。

評価標本に対して適用した結果は、訓練標本に対する結果よりも低いものであったが、かなり高い精度で文のタイプ分けができた。評価標本に対する適用結果の詳細を表5に示す。

ほぼ一意に決定できる出典を除き、主題文が抽出率および適合率において飛び抜けて高い値を示している。これは、主題文の性質を記述した規則が多いことにもよるが、評価関数を主題文検知用に設定したためにもよるものと考えられる。

これを検証するため、全く同じ規則と標本を用い、評価関数のみを変えて実験を行なった。ここでは、結果文を抽出することに対して評価をするものとして、結果文である確率と結果文でない確率のどちらか大きな値を評価関数の値とした。その結果を表6に示す。

ここでは、前の場合に比べ、結果文に対する適合率・抽出率はどちらも向上していることが分かる (それぞれ+4.8%,+10.8%) 。一方、主題文に対する抽出率は多少向上したもの (+3%) の、適合率は大幅に低下した (-23.4%) 。

我々は、構文解析を用いてキーワード抽出を行なう問題に対しても本手法を適用した。文タイプ識別の実験と同じ標本を用い、キーワードの抽出率82%, 適合率53%を得た[5] 。

4 議論

本手法では、明らかに成立する規則でも事例ベース中になければ、その規則は、実行規則系に反映されない。また、明らかに規則が成立する事例が事例ベース中に存在した場合でも、その規則の結果が取られない場合がある。それは、明らかに成立する規則を規則 A 、確実ではないがたまたま事例ベース中ではすべての場合に成立した規則を規則Bとする。もし、規則 A と規則 B が同時に成立する事例がなく、かつ、規則Bの成立する頻度が規則 A よりも高ければ、規則 A と規則Bが同時に成立する場合は、規則Bの結果が優先される。

その規則が成立する事例を最低1つ加えることによってその規則を実行系に反映することができるが、実験でも明らかであったように、少数の例外によって、有効でない組み合わせが生成されてしまう場合がある。これらは、誤った結果を与えたり、システムの効率を低下させる要因となる。このような問題に対する対処としては2つ考えられる。

1つは、有効性を判断する最低頻度を決め、頻度がそれ以下の場合には有効性を認めないとするものである。この方法では、事例を1つ加えることだけでは、規則系の変更が保証されなくなり、見通しが悪くなる。もう1つの方法は、事例ベースに加える事例を慎重に選別し、例外的と思われる事例は加えないことである。しかし慎重に選別することで事例が大量に集められないのであれば本手法は実現可能ではなくなる。

どちらの方法も完全ではないが、実際の運用においては、選別された事例を集めた事例ベースと選別のない事例ベースを、また、確実な規則と曖昧な規則を、区別して扱う必要があるであろう。

本手法で、実際に規則系を構築してみたが、従来の方法に比べ、規則系を構築する上で最も重荷であった制御の記述から解放されたという点で、非常に容易にシステムを構築できた。また、個々の規則が有効であるかどうかを考えることなく記述でき、かつ、システムの最適化の結果を調べることによって、従来、規則の有効性の検証をするために別の様々なツールを用いて行なっていたことが統合的にできるようになったことは開発の効率化につながった。

しかし、本手法における本質的な問題として、大きな事例ベースを作成するコストがかかってしまうという問題がある。しかし、従来のシステムの開発者は実際の事例を綿密に調べることによって直観的に同じようなことを行なっている。また、規則の制御全体を理解しなければ開発・保守はできないので、規則系全体を熟知した人しか規則系を変更することができない。一方、事例ベースを作成するのは、システムの専門家である必要はない。以上の点を総合的に考えるならば、決して高いコストであるとはいえない。

本手法では規則が真偽の2値を返すことを前提とした。そのために、単語に依存するような細かい規則については、その数が多くなってしまった。しかし、多値であることを許せば、これらは1つの規則として記

表 3: 文タイプの内訳

	出典	前提文	主題文	説明文	方法文	結果文	合計
訓練標本	9	172	602	117	44	116	1,060
割合 (%)	0.8	16.2	56.8	11.0	4.2	10.9	
評価標本	3	85	272	67	38	74	536
割合 (%)	0.6	15.9	50.7	12.5	7.1	13.1	

表 4: 適用結果

	主題文適合率 最大精度	主題文 適合率	主題文 抽出率	規則 評価回数	1 文当たり 規則評価回数
訓練標本 (1,060 文)	97.5%	97.0%	92.1%	67,205	63.4
評価標本 (539 文)	98.0%	85.8%	91.2%	36,858	68.3

表 5: 評価標本に対する適用結果

解析結果 ＼ 人による分類	出典	前提文	主題文	説明文	方法文	結果文	合計	抽出率
出典	3	0	0	0	0	0	3	100 %
前提文	0	61	14	8	0	2	85	71.2%
主題文	0	11	248	7	6	0	272	91.2%
説明文	0	52	8	6	1	0	67	9.0%
方法文	0	21	7	4	5	1	38	13.2%
結果文	0	45	12	7	0	10	74	13.5%
合計	3	190	289	32	12	13	539	
適合率	100%	32.1%	85.8%	18.8%	41.7%	76.9%		

表 6: 適用結果 2：評価関数が結果文の場合

解析結果 ＼ 人による分類	出典	前提文	主題文	説明文	方法文	結果文	合計	抽出率
出典	3	0	0	0	0	0	3	100 %
前提文	0	29	49	3	2	2	85	34.1%
主題文	0	8	259	2	1	2	272	95.2%
説明文	0	15	49	3	0	0	67	4.5%
方法文	0	12	22	0	4	0	38	10.5%
結果文	0	17	36	1	2	18	74	24.3%
合計	3	81	415	9	9	22	539	
適合率	100%	35.8%	62.4%	33.3%	44.4%	81.8%		

述できる。本手法は、多値にも拡張可能である。しかし、多値の場合、場合の数が多くなり過ぎ、どこまでを厳密に捉えれば良いかを考慮する必要がある。

5 おわりに

正解の集合である事例ベースを用いて、与えられた規則集合を最適化する手法を提案した。本手法によって、規則を、より柔軟なものとして捉えることができるため、開発を、より効率化することが可能となる。また、個々の使用環境毎に記述された規則を有効に組み入れることができる。

本質的問題として、事例ベース作成のコストの問題がある。しかし、個々の事例は規則系の使用者に最も近い規則であると考えることができる。それらから、より一般化された知識を得ることによって、自然言語処理で大きな問題である知識獲得を解消する可能性を探るべきであると考える [6] 。

謝辞

本研究の初期の段階で有益な議論をしていただいた荻野紫穂氏に感謝致します。

参考文献

[1] 中村, “文法記述ソフトウェアＧＲＡＤＥ,” 情報処理学会自然言語処理研究会,38-4,1983

[2] Fujisaki, T., “An Approach to Stachastic Parsing,” Proc. of Coling 84, pp16-19,1984.

[3] Sato, S. and Nagao, M., “Toward Memory-based Translation,” Proc. of Coling 90..

[4] 梅田, “抄録からの主題文の自動抽出,” 慶應義塾大学図書館・情報学科卒業論文,18801604,1991

[5] 関根, “抄録からのキーワード自動抽出の高度化,” 慶應義塾大学図書館・情報学科卒業論文,18805236,1991

[6] 野美山, “事例の一般化による機械翻訳,” 情報処理学会第４３回全国大会,Vol. 3, pp.191-192, 1991.

A 規則集合Ｒの環境Ｅにおける最大精度

規則集合Ｒが環境Ｅで取り得る理論的最大精度であり、規則集合の制御の方法によらない、精度の定量的上限を与える。定性的な上限を与えるものではない。

規則集合の規則の数は５であるとする。それらの規則を適用する環境Ｅは、全部で１０の事例からなるとする。環境Ｅにおける規則集合Ｒの適用状況を図７に示す。これから明らかなように規則の適用状況が全く同じであるのに、結果値が異なるものが存在する。このような例においては、ここで用いた規則集合Ｒでは区別ができない。ここで、同じ適用状況のうち最も頻度の多い結果を取るものとすると、規則の組み合わせの仕方に拘らず規則集合Ｒが環境Ｅで取りうる最大の精度が決まる。図７では同じ適用状況を取るパタンが１通り存在し(４番目と５番目)、それらを考慮すると最大精度は、$\frac{(10-1)}{10} = 90\%$ となる。

表 7: 適用結果

回数	規則 1	規則 2	規則 3	規則 4	規則 5	結果値
2	1	0	1	0	0	Val1
2	0	1	1	0	1	Val2
1	1	0	0	0	0	Val3
4	0	0	1	1	0	Val3
1	0	0	1	1	0	Val2