例文からの学習による生成規則の自動修正

2J－8

中澤 聡　東京大学
濱田 喬　学術情報センター

1 はじめに

近年、例からの学習が盛んに研究されており、それにともない、例からの形式言語の学習もいくつかの成果が挙げられている。

中でも、実用上重要な言語である文脈自由言語 (CFL) の学習においては、文脈自由文法 (CFG) を多項式時間で学習するアルゴリズム [1] が既に提案されている。

このアルゴリズムでは、内部ノードはラベル付けされていない導出木が、入力される例文に対してそれぞれ与えられる。

また、MAT (Minimally Adequate Teacher) と呼ばれる教師の存在を仮定しており、学習システムはこの教師に、任意の記号列に対して、それが目標の言語 L に属しているか尋ねるという所属性質問と、任意の文法 G に対して、$L = L(G)$ かどうか判定し、推測結果が正しくない場合には教師が、反例を返すという等価性判定質問をすることが許されている。

一方、人間のユーザがコンピューターと対話的に学習を進めるシステムを考えた場合、与える例文全てに正確な導出木を要求するのは大きな負担である。また、等価性判定質問は決定不能問題であり、人間にとってもある文法と別の文法が等価であるかどうか判定するのは大変な問題である。さらに、この等価性判定質問では、無数の等価の文法のうち、どれかに到達すれば学習が終了したことになるので、学習システムが終了してもユーザに都合の良い構造を持つ文法を得られる保証がない。

そこで本研究では、望ましい文法構造の骨組みとして、元となる大まかな文法を学習システムに与え、さらに、ユーザが「典型例」と見なせる例文を入力することにより、例文を受理できるよう文法の生成規則に肉付け、修正を行なってゆき、効率良く CFG を学習するというシステムを提案する。

2 学習システムの構成

図1が提案している学習システムの大まかな構成である。

ユーザは最初に骨組みとなる元文法を、そして以後は、元文法に追加、あるいは削除したい形式の文を正負の具体例として入力していく。負の例が必要となるのは、既存の生成規則を削除する場合である。

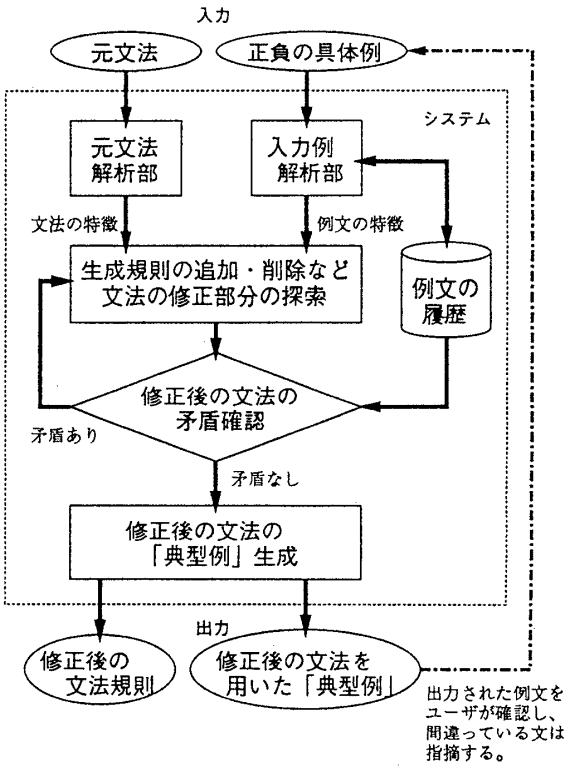

図 1: 学習システムの構成

学習システムは、まず与えられた元文法を解析し、おおまかな特徴を取り出す。ここで取り出された特徴が、今後例文が入力された際の学習の基準となる。

また、与えられた入力例は全て例文の履歴として記録され、それらの相関関係が調べられる。

学習システムはこれらの情報から、与えられた例文のおおまかな導出木を推測し、それと現在の文法構造の差から、追加・削除すべき生成規則の探索を行なう。

修正部分が求まると、システムは修正の結果がいままで入力された例文に矛盾していないか確認を行なう。もしここで矛盾が発見された場合は、再び探索の動作に戻り、次の候補を調べる。

さらに学習結果の確認のため、修正部分の特徴を表すような例文を生成し、ユーザに提示する。ユーザはこれを調べて、間違っている文は反例として再びシステムに入力する。これを繰り返すことにより、正しい修正部分

Automatic Modification of Production Rules by Learning from Example Sentences
Satoshi Nakazawa[1], Takashi Hamada[2]
[1]University of Tokyo
[2]National Center for Science Information Systems

に辿り着ける。

以上のようにして、元文法から文法規則を修正していく。修正後の文法を次の元文法とすることにより、最初の大まかな骨組みから、複雑な CFG を例文から学習させることができる。

3 元文法の利用

元文法から取り出す構造情報として、まず考えられるのが、階層構造である。人間にとって理解し易い CFG は特に記号間に何らかの上下関係があることが多い。しかし、そのままの生成規則の形ではこれを判断するのが難しい。

そこで、各非終端記号に対して、それを左辺に持つ生成規則の右辺に現れる記号の集合を取る。ついで、その集合に含まれる非終端記号 1 つ 1 つに対して再び、その記号を左辺に持つ生成規則の右辺に現れる記号の和集合をとる。ただし再帰を回避するため、それまでに 1 回以上現れている記号はそれ以上辿ることはしない。こうして、ある非終端記号 N から r 回生成規則を適用したとき右辺に現れる記号の集合の表 $Descendant(N,r)$ を求める。

また、同様にして、ある一つの記号 $\alpha \in (N \cup T)$ を含む何らかの記号列から r 回生成規則を用いて還元したとき得られる可能性のある記号の集合の表 $Ancestor(\alpha,r)$ を求める。

これらの表に現れる可能性のある記号は、生成規則の数を n としたとき、$r = n$ までに必ず現れるので r は最大で n まで調べれば良い。

このようにして、実際の導出木を調べるよりは不正確だが、次のような元文法のおおまかな構造を掴むことができる。

- 文法の導出木の特徴
  開始記号から辿れる r の数が生成規則の数 n に比べて少ないほど、その文法の導出木は深さが浅く、幅が広いものになり、逆に多いほど、導出木は深く、幅狭いものになる傾向がある。
- 再帰構造の目安
  再帰構造があるところでは、必ず同じ記号が 2 回以上表の異なる r のところで現れる。ただし、逆は必ずしも成り立たない。
- 記号のグループ分け
  記号を導出木で用いられる段数によって、おおまかに r 個のグループに分けて扱える。

次に生成規則全体に対して、生成規則の右辺における記号間の前後・組関係、あるいは、ある記号を右辺に含む生成規則の左辺の種類、などといった統計情報を調べる。

これらにより、ある部分記号列は導出木の途中でかならずその非終端記号に還元されるといった重要な非終端記号を見つけることができ、元文法が受理する文の導出木を大きく縦にグループ分けするのに使用できる。

4 「典型例」の利用

典型例として人が提示する例文と、ランダムな例文との大きな違いの 1 つに提示される順番が考えられる。ランダムな例文では、ある例と次の例が同じ修正箇所の特徴を表しているとは限らないが、人間が与えた場合には普通同じ修正箇所を示している。

そこで具体的には、入力された例文を 1 つ 1 つ取り上げて、その修正すべき生成規則の候補を絞っていくのではなく、近くの例文と比較して、それらとの不変項を見つけ出し、また、遠くの例文との不変項は逆に考慮しないようにする。

不変項として考えられる特徴としては、

- 記号間の順序関係。あるいは (A) のような括弧関係
- 記号 A があるなら B もある、C があるなら D もあるといった因果関係。あるいは排他関係

などが挙げられる。

5 修正部分の探索

基本的な方針は、元文法およびユーザが入力する「典型例」から取り出した特徴を用いて、入力された例文の導出木の大まかな構成を推測し、それに従って修正すべき生成規則の大域的な探索を行なうということである。

ついで、局所的な探索として、絞り込んだ生成規則の候補を 1 つずつランダムに取り上げ、それを用いた文例を生成して、入力された「典型例」との相似度を調べてゆく。

6 おわりに

以上、元文法と入力例に含まれる特徴を利用した CFG の学習システムの方針について述べた。

ただ、本資料で提案した手法は CFG 一般に有効ではなく、元文法は何らかの構造を持っていなければならない。

今後は、2 節で述べた *Descendant* などを調べる方法が、どのような CFG にどの程度有効なのか定式化していく予定である。

参考文献

[1] Y. Sakakibara. "Learning Context-Free Grammars from Structural Data in Polynomial Time", *Theoretical Computer Science*, Vol.76, No.2-3, pp.223-242, Nov. 1990.