TM-1189

会話文理解のための知識の文脈構造解析に基づく抽出

住山　一男、小野　顕司、知野　哲朗、
浮山　輝彦（東芝）

July, 1992

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03)3456-3191 ~5
Telex ICOT J32964

Institute for New Generation Computer Technology

会話文理解のための知識の文脈構造解析に基づく抽出 *

Discourse structure analysis based knowledge extraction from text for conversation understanding *

住田 一男, 小野 顕司, 知野 哲朗, 浮田 輝彦

Kazuo Sumita, Kenji Ono, Tetsuro Chino, Teruhiko Ukita

(株) 東芝

Toshiba Corp.

概要

文脈構造に基づいて, 会話文理解のための知識を抽出する方法と, その方法に基づいて PSI-II(ICOT が開発した逐次型推論マシン) 上に試作した知識抽出システムについて述べる. 本稿での知識の抽出の目的は, 会話文においてシステムが取り扱える表現を拡張することにある. ここでは, 会話の話題に関連するテキストに現れる様々な表現の間の相対関係を, 命題の間の関係としてとらえ, この命題間の関係を知識としてテキストから抽出することについて論じる.

本稿で述べる方法は, テキストの文脈構造を解析し, その構造情報に基づいて, 省略情報の補完処理を行い, 命題間の関係を抽出することを特長としている. また, 抽出処理で用いている文脈構造の解析は, 修辞的な表現などの言語情報だけに基づいて解析処理を行う方法である.

例題として, 人間の似顔絵作成を自然言語で指示するシステム (顔の部品の物理的な形状などを指示し, 操作する) を選び, そこで用いる知識を顔と人の性格についての記述のあるテキストから抽出する. そして, 抽出した知識 (命題間の関係) を似顔絵作成システムに適用することにより, 主観的な表現を含む入力文を取り扱うことができるように拡張できることを示す.

*本研究は ICOT ((財) 新世代コンピュータ技術開発機構) からの再委託により, 第五世代コンピュータプロジェクトの一環として行ったものである.

1 はじめに

自然言語による対話システムでは, 利用者が入力する様々な表現を処理できる能力が重要である. 通常, 何らかのタスクに対して自然言語対話システムを構築しようとした場合, 与えられたタスク固有の処理を行うアプリケーション処理部と, 利用者の自然言語入力をそのアプリケーション処理へのコマンドに変換する自然言語処理部とに機能を分けて実現する. そして, 普通, このようなアプリケーション処理部が受け付けるコマンドは, 限定された語彙で設計されることになる.

例えば, アプリケーション処理が, データベースへの問い合わせであると仮定する. この場合, アプリケーションで必要となるコマンドは, 項目の指定のための属性や, それをそれらの属性を結合する AND, OR や NOT などの論理的な結合子などの語彙から構成される. アプリケーション処理と利用者との間のインタフェースとなる自然言語処理部では, 利用者の様々な表現を, このような限定された語彙に置き換えるための知識を何らかの形で用意しておかなければならない.

この種の知識を会話の履歴から獲得する事例ベースの方法が, 提案されている [Shim 91]. しかしこの方法では, ある程度の量の会話のコーパスをあらかじめ準備しなければならない. ところが, 処理しなければならないタスクについての会話が, あらかじめ準備されている場合は少なく, 現実味が少ない. つまり, このように会話の履歴から知識を獲得する能力は, システムが処理できる表現を動的に拡大していく枠組みとしては重要な能力であると

考えられるが,初期能力の向上させることはできない.そこで,初期の表現処理能力をあげるために,すでに存在するテキストなど書かれた文章から,表現処理能力を向上させることが考えられる.

本稿では,ICOT プロジェクトの一環で開発した似顔絵作成システムを例として,システムにおける表現の処理能力を拡大するため,テキストからの知識抽出について考察する.

2 会話文理解のための知識

2.1 自然言語入力の特長

通常,あるアプリケーションに対して,自然言語を入力手段とする場合の特長としては以下のような点が考えられる [Ukit 88].

- 利用者が自分の言葉で入力できる.
- 特別な訓練がいらない.
- 入力装置としてはキーボードだけでよい.

これらの特長のうち,2番目の特長と3番目の特長は,日本語入力の場合,仮名漢字変換を行わなければならないので,その使い方に慣れないといけないという問題はある.しかし,近年のワードプロセッサなどの流通にともなって,日本語入力は一般化しており,特別な訓練を必要としない入力手段となりつつある.

一方,1番目の特長は,自然言語入力にとって最も重要視される特長である.あるアプリケーションについての自然言語インタフェースを実現する上で,この特長を満足しないインタフェースは使い良いものとはいえない.つまり,利用者が自分の言葉で入力できることを実現するためには,自然言語処理部は,次のような機能を持つ必要がある.

- 省略や代名詞などの指示先を求め省略された情報を補う機能
- 利用者の表現をシステムの内部表現に対応させる機能

このような機能の実現には,何らかの知識に基づいた処理が必要となってくる.例えば,前者の省略の補完機能を考えた場合,次のような知識が必要である.

- 概念の間の関係 (上位 - 下位, 部分 - 全体)

  ex. 「そのボタン を押したらどうなるのか」
  (ビデオの操作法案内に関する入力)

  “そのボタン”が例えばビデオのボタンであることを解析するためには,ビデオがどんな部品から構成されているのかというような概念間の関係についての知識が必要である.

- 命題の間の関係 (因果関係)

  ex. 「ビデオの再生ボタンを押したけれど,動かない.」

  “動かない”では,動かないものは何であるかが明示されていない.この省略を解決するには,“ビデオの再生ボタンを押すと,ビデオが動作する”というような命題間の関係についての知識が必要である [Sumi 91].

また,後者の表現間の対応をとる場合についても,やはり同様の知識が必要となってくる.このような観点から必要となる知識には,次のようなものがある.

- 概念の間の関係 (同義語)

  ex. 「電源スイッチ を押した」

  アプリケーション処理の側が‘電源ボタン’という用語を内部表現として用いていた場合,‘電源スイッチ’を‘電源ボタン’に置き換えるための概念間の関係についての知識が必要である.

- 命題の間の関係

  ex. 「ビデオの電源を ON にした」

  アプリケーション処理の側が「ビデオの電源ボタンを押す」という命題に対して処理が可能である場合,「ビデオの電源を ON にした」という命題を上記命題に対応させるための知識が必要である.

通常,自然言語インタフェースを構築する場合,アプリケーションごとに,あらかじめ自然言語処理部に上記のような知識を格納することになる.この知識の準備が,自然言語インタフェースを構築する際に大きな作業項目となっている.

本稿で考察する似顔絵作成では,例1のような入力がなされる.

Example 1

(1) 口はもう少し大きい.

(2) 優しそうだ.

文(1)では, 口の大きさを指定している.ここで, 似顔絵作成をコマンドレベルで処理するアプリケーション処理部は, 顔の部品の大きさや形などの具体的な形状の指定に対して処理できるものと仮定する.この場合, アプリケーション処理部は単にもう少し大きい口にすればよく,自然言語インタフェース部は, 単に入力された自然言語をアプリケーション処理部に対するコマンドに変換する処理を行えばよい.

ところが, 文(2)の場合, アプリケーション処理部は, 単純な変換処理ではこの要求を満たすことができない.この場合, 自然言語インタフェース部は,「優しそうだ」という主観的な表現を,アプリケーション処理部が取り扱える具体的な形状に変換する必要がある.すなわち,「優しい」と例えば「目が垂れている」というような命題間の関係を, 知識として用意しておかなければならない.

上の例からも容易にわかるように, 命題間の関係は処理するタスクに強く依存しており, タスクごとに, 自然言語インタフェース部にこの種の知識を用意する必要がある.しかし, あらゆるタスクについてあらかじめ準備することは, 現実的には困難であり,自然言語インタフェースを構築する上で一つの問題となっている.本稿では, このような命題間の関係をテキストから抽出する.

2.2 命題間の関係の抽出

命題間の関係は, 一文の中で述べられる場合と, 複数の文にわたって述べられる場合がある.次の例は, 人間の性格と顔の特徴についての記述である.

Example 2

(1) 目が垂れていると人柄がよい.

(2) 目が垂れている人は人柄がよい.

上記例では, 両方の文は, 同じ関係を述べている.したがって, 命題間の関係の抽出のためには, 一文の中でのいくつかの表現パターンを考慮し,抽出処理を行う必要がある.

また, 上記の例とは異なり, 二文以上にわたって解析しなければ正しい関係が得られない場合がある.特に, 省略された情報がある場合, それを正しく補わなければ正しい関係が抽出できない.

Example 3

(1) 目の垂れている人は人柄がよく, 人にだまされやすく, 損をしがちです.

(2) 例えば, 上の絵の人はこのタイプです.

(3) さらに, 目が大きいと優しく, 愛嬌のある性格といえます.

(4) 下の絵の人はこのタイプです.

上記の例の場合, 文(3)は,「目の垂れている人は, さらに目が大きいと優しい性格といえます」ということを意味している.文(3)では,「目が垂れている人」という話題は明示されておらず, 文脈からその情報を補完する必要がある.

通常, このような話題情報の補完を行う場合, 直前の文の話題情報を用いて補うことが多い.ところが, 今の場合, このような単純な方法では,「上の絵の人」という語句が補われることになり正しい補完ができない.

文(2)は文(1)の例示となっており, 全体の話しの流れに対して補足的な情報しか持っていない.したがって, このような位置付けにある文から, 補完する情報を得るべきではなく,今の場合, 文(1)から情報を得るべきである.このように, 文の間の相対関係を考慮することにより, 補完すべき情報を得る文の範囲を限定することができる.

文間の相対関係を捉えるために, 本稿では, 文脈構造解析処理に基づき, 命題間の関係を抽出する.

3 命題間関係の抽出

3.1 文脈構造

本稿では, 文脈構造に基づいて, 命題間の関係抽出を行う.筆者らは, 複数の文からなるテキストを解析するために, 文脈構造の解析のための処理系を開発している ([Ono 89], [Sumi 92]).

文脈構造は, テキスト内の文や文のまとまりの間の相対関係を表示するものである.テキストに対する文脈構造解析は, 文に対する構文解析に対応し,複数の文からなるテキストを処理する上で基本的な処理である.これまでいくつかの研究がなされてきている (例えば [Cohe 87] など) が,計算機上で実テキストを対象として動作するものはない.以下, 本稿で用いる文脈構造解析について概説する.

テキストにおいては, 様々な修辞表現が議論の展開を明確にするために用いられる.これらの中で,

表 1: 接続関係の一例

RELATION		EXAMPLES and EXPLANATION
順接	<SR>	だから(thus, therefore), よって(then)
並列	<PA>	同時に(at the same time), さらに(in addition)
対比	<CT>	一方(however), 反面(on the contrary)
例示	<EG>	例えば(for example), ... 等である(and so on)
展開	<EX>	これは(this is)
:		:

<EG> : 例示
<PA> : 並列
<EX> : 展開

((1 <EG> 2) <PA> (3 <EX> 4))

図 1: 文脈構造の例 (例 3 に対応)

文の間の関係を直接表現する, 接続詞や接続的な副詞(「しかし」, 「だから」など)ならびに, 文末の表現(「…だからだ.」, 「…などである.」など)など文間の関係を明示する接続表現は, 特に重要な働きを持つ.

筆者らは, これらの接続表現を 18 種の接続関係に分類し, この接続関係に基づいて文脈構造を表現している. 一例を, 表 1に示す.

図 1は, 例 3 に対する文脈構造を示している. ここで, 関係 <EX> は, 明示的な接続表現が存在しない場合に設定する関係である. 図示した構造は, 文(2) は文(1) の例示であること, 文(3) は前方の 2 文に対して並列関係にあること, を表現している.

3.2 文脈構造解析

文脈構造解析は, 次の 5 つステップからなる.

(a) 接続関係解析
テキストの各文に対し, 形態素解析, 構文解析を行った後, 接続表現の抽出を行う. その後, 抽出した接続表現から接続関係を接続関係テーブルに従って求め, 接続関係系列を作成する.

接続関係系列は, ある文が前方の文に対してどのような関係を持っているかを表示するものであり, 文を同定するための文番号と, 接続関係からなる系列である. 例えば,

(1 <EG> 2 <PA> 3 <SR> 4)

というような接続関係が得られた場合, 文(2) は, 文(1) に対する例示であること, 文(3) は, 文(1) と文(2) か, 文(2) のいずれかと並列関係にあること, 文(4) は, 文(1) から文(3) か, 文(2) と文(3), あるいは文(3) のいずれかと順接関係にあることを示している.

(b) セグメンテーション
この処理では, 入力のテキストを参照し, 大局的な構造を明示する修辞表現を抽出し, 接続関係系列に反映させ, 制約付き接続関係系列を生成する.

処理は, セグメンテーション規則と呼ぶ if-then 規則に基づいている. 例えば, 次のような表現がテキスト中に現れたと仮定する.

"…. ^{M} 確かに, …. …. …. ^{N} しかし, …."

この場合, 文 M から N-1 は内容的なまとまりが感じられる. そして, その文のまとまりに対して, 文 N が逆接という関係にあることがわかる. セグメンテーション規則は, このような文間の修辞表現の依存関係を, 文脈構造に反映させるために記述する規則である.

図 2にセグメンテーション規則の一例を示す. セグメンテーション規則の if 部には, 表層的な表現パターンを記述し, また, then 部には, その表現パターンに対応する制約を記述する. 制約は, 図に示すように '{' や '}', '@' というような記号を用いて

If-part :

Sentence No.: N M

Input string: "…。確かに…。…。 しかし…。"

then-part : [… {N <EX> {N+1 … M-1}} @<NG> M …]

図 2: セグメンテーション規則の例

表現している．'{' と '}' は，それらの記号で囲まれた系列が一つまとまりであることを表示している．また，'@' は，その記号の現れる位置が文のまとまりの境界であってはならないことを表示している．

(c) 構造生成

この処理では，セグメンテーション処理で得た制約付きの接続関係系列に基づいて，可能な文脈構造候補をすべて生成する．

(d) 構造絞り込み

構造生成で生成した文脈構造を入力として，構造選好規則に基づき構造候補の絞り込みを行う．構造選好規則は，文脈構造上で隣合う接続関係と，それに対応するより自然な構造についての規則である．例えば，次のような文の系列を考える．

"P。例えば、 Q。したがって、 R。"

ここで，P， Q， R は文または，文のまとまり表現している．接続関係系列に変換すると，次のようになる．

(P <EG> Q <SR> R)

このような系列に対して，可能な文脈構造候補は，次の 2 候補が考えられる．

((P <EG> Q) <SR> R)

(P <EG> (Q <SR> R))

第 1 の候補は，R の内容が P と Q の両方の内容からの帰結となっていることを，それに対して，第 2 の候補は，R の内容が Q の内容だけの帰結となっており，かつそれらが P の例示となっていることを示している．

明らかに，この場合，自然な議論の展開としては，第 1 の候補の方が第 2 の候補に比べ自然な表現である．すなわち，ある文について，その前後の接続関係がそれぞれ <EG> と <SR> である場合，その文の直後で，文のまとまりを閉じる構造がより自然であるということになる．

構造選好規則は，このような隣接する接続関係と，構造との関係を規則として表現したものである．筆者らは，接続関係のすべての組み合わせに対して，いずれの構造が適当であるかを判定し規則として整備した．

具体的には，接続関係のすべての組み合わせに対して，次の 4 つのカテゴリに分類した．ただし，ここで 'r1' と 'r2' は，それぞれ接続関係であり，'P' は，文または文のまとまりを意味している．

POP-type : "r1 P) r2" を許可
("r1 (P r2" を棄却)

PUSH-type : "r1 (P r2" を許可

NEUTRAL-type : POP と PUSH の両方の構造を許可

NON-type : 非構造 "r1 Q r2" を許可

POP-type は，このタイプに属する接続関係の組み合わせ (r1, r2) に対して，議論の展開が閉じられる構造のほうが自然であることを意味している．したがって，これに対応して，"r1 (P r2" という部分構造を含む文脈構造を棄却することになる．

逆に，PUSH-type は，このタイプに属する接続関係の組み合わせの場合，議論の展開が開始する構造のほうが自然であることを意味している．したがって，これに対応して，"r1 P) r2" という部分構造を含む文脈構造を棄却することになる．

(e) 話題による選好

構造の絞り込み処理で残った構造候補のうち，話題とその照応先との関係から構造を選択する．選択の観点としては，話題とその照応先とが，構造上より近くになる候補を優先する．ここで，話題とは「…は」や「…については」などの表現で指定される名詞句をいう．

以上の処理により，テキストの文脈構造を解析する．

3.3 話題の補完

この処理では，話題が明示されていない文に対して文脈構造を参照し補完を行う．接続関係ごとに左右のノードの重要性を判定する．

説明を簡単にするため，以下の記号を導入する．

> LA(P)：文脈構造中のあるノードPに対して，そのノードを右側の子孫に持つノードのうちで最もPに近いノードをLA(P)と表記する．

話題の補完処理は，次のステップで行う．

Step.1 各文ごとに明示された話題を取り出す．
(ここで明示された話題とは，“…は”，“…には”，“…については”などの表現で指定された名詞句である．)

Step.2 各部分木の話題を重要ノードの側の話題とする．
(ここで，重要ノードとは，全体の論旨の流れに対して中心的な内容を述べている側のノードである．例えば，ある2つの文が例示関係にある場合，右の文は左の文の例示であり，右の文が全体の論旨の流れに対して中心的な内容を述べていることになる．したがって，この場合，重要ノードは左側の文ということになる．
なお，このステップは，ターミナルノードから初めて，ルートノードまで順次処理する．)

Step.3 ある文Pで話題が省略されている場合，LA(P)の左側の子ノードの話題を，Pの話題とする．

この処理のために，接続関係ごとに左右のノードの重要性についての情報を用意している．

前節で述べた文脈構造解析により，例3から図1に示した文脈構造が得られる．その構造に従って図3に示すように話題の補完処理が行われる．

この例の場合，まずはじめに，**Step.1** で文(1)，(2)，(4)の話題が，それぞれ“目の垂れている人”，“上の絵の人”，“下の絵の人”ということが得られる。次に **Step.2** で，ノードaの話題が“目の垂れている人が”得られ．最後に，**Step.3** で文(3)の話題をノードaから補完する．

3.4 命題間の関係の抽出

前節の処理で，話題の補完処理を終えた各文を入力として，命題間の関係抽出を行う．この処理は，単純なテンプレートマッチングで行う．あらかじめ準備したテンプレートに照合した場合，その照合した情報を，命題間の関係として抽出する．

抽出すべき関係は，似顔絵作成システムにおいて，利用者の入力文を，システムのコマンドに対応させるために用いる．本稿で述べる似顔絵作成システムのアプリケーション処理部では，各顔部品についての物理的な属性についての指定を受け付ける．そこで，テンプレートは，次のように顔部品の物理的属性に関しての表現を検出するパターンを表現している．準備したテンプレートの例を以下に示す．

```
pattern([(theme,
            (人,[(n_comp,
                (P,[(v_comp,A)]))]))],
         [physical(A),part(P)],
         [P,[(v_comp,A)]]).

pattern([(if,(A,[(comp,P)]))],
        [physical(A),part(P)],
        [P,[(v_comp,A)]]).
```

前者は，例えば“目の垂れている人は”というような表現に照合するテンプレートである。また後者は，“目が大きいと”というような表現に照合する．

上記の例からわかるように，テンプレートは，3引数からなり，第1引数がマッチングパターン，第2引数が制約条件，第3引数が抽出パターンを表現している．上記のテンプレートにおいて，第2引数に設定されている制約条件のうち，‘physical(A)’は 変数 A に照合する語が物理的な属性であることを意味している．また，‘part(P)’は変数 P に照合する語が顔部品のいずれかであることを意味している．

例3に関しては，次のような関係が知識として抽出される．

```
if-then([(目,[(v_comp, 垂れる)])],
        [(よい,[(comp, 人柄)])]).

if-then([(目,[(v_comp, 垂れる)]),
         (目,[(v_comp, 大きい)])],
        [優しい]).
```

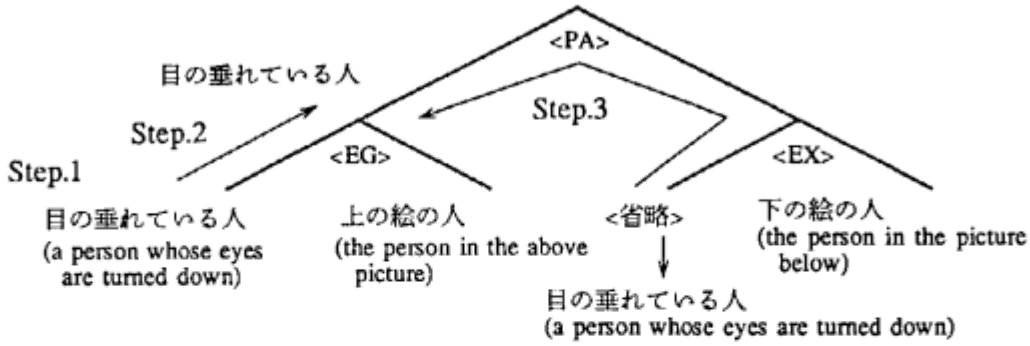

図 3: 話題の補完例

4 自然言語インタフェースへの適用

似顔絵作成システムは, 利用者の自然言語を似顔絵作成のコマンドに変換・解釈する自然言語処理部と, 変換されたコマンドに従って実際の処理を行うアプリケーション処理部からなる. 図 4にシステムの全体の構成を示す.

アプリケーション処理部は, 次のようなコマンドを受け付け, そのコマンドの内容に従って, 顔部品の操作を行う.

```
[(topic, 目),(attribute, 大きさ),
 (大きさ, 大きい),(operation, 代える),
 (degree, ちょっと),
 (comp,[右目 #001, 左目 #001])].
```

上記コマンドは, 「目の大きさをちょっと大きくしろ」という入力文に対応するコマンドである. ここでは, このコマンドを操作コマンドと呼ぶ. 操作コマンドで, 操作対象である顔部品 (目, 眉, 鼻, 口, 輪郭を 'topic' に設定) について, 属性 (大きさ, 長さ, 垂れ具合などを 'attribute' に設定) を指定することにより, 表示されている顔部品の変更を指示することができる. また, 'operation' には '代える' や '移動' などの操作の種類が, 'degree' には属性の度合が, 'comp' には比較されるインスタンスが設定される.

このシステムでは, 各顔部品ごとに 30 個づつ画像データをあらかじめ準備し, 画像データベースに登録している. また, あらかじめ物理的な属性 (大きさ, 長さ, 垂れ具合など) ごとに, 顔部品の順序づけ情報を次のような形式で設定している.

```
order(目, 大きさ, [22,25,13, 2, …]).
order(目, 丸さ,   [1, 24,11,23, …]).
                …
```

この情報は, 一般の日本語辞書から顔部品の物理的な属性に関する形容となる表現を人手で取り出し, 各々の属性について順序づけ情報を作成した. アプリケーション処理部は, この属性についての順序づけ情報を参照することにより, 操作コマンドに対応する画像データをデータベースから検索し, 表示されている画像と入れ替えることになる.

自然言語処理部は, 利用者の入力を操作コマンドに変換する. この処理は, 大きく分けて解釈部と, 再解釈部の 2 要素からなる.

解釈部では, 形態素解析, 構文解析を行い, 格パターンに変換する. そして, 必須格が省略されている文については, 利用者が入力した文の履歴を参照し, 省略部分の補完を行う.

次に, 解釈部は, 格パターンを操作コマンドに変換する. この処理は, 格パターンの部分構造を if 部とし, 部分的な操作コマンドを then 部とするルールに基づいた処理である. ルールの一例を以下に示す.

```
st([{r(theme, は),[{h(目),[]}]}]) →
      search(目,P),
      put([(topic, 目),(comp,P)]).
st([{h(する),[p(肯定)]},
    {r(adju,nil),[{h(A),[]}]}]) →
      nominalize_a(A,V),
      put([(attribute,V),(V,A),
           (operation, 代える)]).
```

第 1 の例は, "目は" という表現が入力文に含まれており, その時点の目の表示が, '右目 #001' と '左目 #001' である場合, 操作コマンドに '[(topic, 目),(comp,[右目 #001,左目 #001])]' を付け加えることを意味している. また第 2 の例は, "大きくしろ" という表現が入力文に含まれている場合, 操作コマンドに '[(attribute, 大きさ),(大

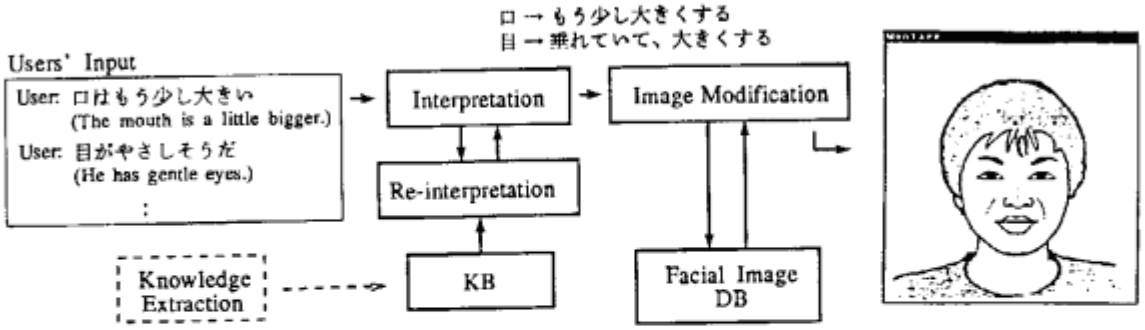

図 4: 自然言語による似顔絵作成システムの概要

きさ, 大きい),(operation, 代える)]' を付け加えることを意味している.

if 部と照合する部分が存在する場合, then 部の指示に従って, 対応する操作コマンドを出力することにより, 格パターンから操作コマンドを作成する.

上記の処理で操作コマンドの作成に失敗した場合, 再解釈部は, 3 で抽出した命題間の関係に基づき, 入力の格パターンを物理的な属性を含む格パターンに変換する. そして, この変換に成功した場合, これを入力として解釈部を再度起動し, 操作コマンドへの再変換を試みる.

5 まとめ

会話文理解において取り扱える表現を拡張するため, テキストから命題間の関係を知識として抽出する方法について述べた. 本方法は, 命題間の関係を抽出するために, テキストの文脈構造を解析し, その構造に基づいて省略された情報を補うことを特長としている. 本稿では, 抽出した知識を, 自然言語により似顔絵作成を行う会話システムで扱える表現を拡張するために用いた.

本稿では, 文脈構造の情報を, 省略情報の補完のために用いたが, 命題間の関係が, 複数の文にわたって表現される場合がある. このような場合, それぞれの関係がどの文の範囲にわたる関係であるかを明確に認識する必要がある. これに対しても, 文脈構造の情報は有効な情報を与えると考えられる.

今後, 他のアプリケーションに適用するなどの実験を通じ, 知識抽出方法の改良を行っていきたい.

参考文献

[Cohe 87] Cohen, R.: "Analyzing the Structure of Argumentative Discourse", *Computational Linguistics*, Vol.13, pp.11–24, 1987.

[Ono 89] Ono, K., et al.: "An Analysis of Rhetorical Structure", *IPS Japan Technical Report NL* 70-2, pp.1–8, 1989.

[Sumi 91] Sumita, K., et al.: "Disambiguation in Natural Language Interpretation Based on Amount of Information", *IEICE Trans.*, Vol.E 74, No.6, pp.1735–1746, 1991.

[Sumi 92] Sumita, K., et al.: "A Discourse Structure Analyzer for Japanese Text", *Proc. Int. Conf. Fifth Generation Computer Systems 1992 (FGCS'92)*, pp.1133–1140, 1992.

[Shim 91] Shimazu, H., et al.: "Acquiring Knowledge for Natural language Interpretation Based-on Corpus Analysis", *IPS Japan Technical Report AI* 75-10, pp.87–96, 1991, (in Japanese).

[Ukit 88] Ukita, T., et al.: "An operation guidance system with natural language input", *IEICE Technical Report OS* 88-18, pp.13–18, 1988, (in Japanese).