用例に基づく多義性解消における学習のための一手法

浦本 直彦
uramoto@trl.ibm.co.jp
日本アイ・ビー・エム(株) 東京基礎研究所

1 はじめに

用例に基づく自然言語処理システムにおいて，質の良い用例をどれだけ大量に獲得するかという問題は，いまだ解決されていない問題である．この問題の解決策の一つは，用例に基づいて処理を行うシステムを，用例の獲得ツールとして用い，出来るだけ人間の介入を抑えながら，必要な知識を学習していくことである．

従来の，多義性解消システムでは，次のようなプロセスを繰り返していくことで，用例を蓄積し，解析精度を向上させていく．

1. 処理を行う文集合を用意する．
2. 個々の文を解析し，多義性解消を行う．
3. 解析結果を人間が修正し，用例として加える．

用例が少ない時には，システムの解析精度はあまり高くないので，人間が解析結果を修正し用例として加える際の労力は高い．しかし，沢山の文を，解析していくにつれて，用例が増えていくのでシステムの精度が上がり，人間の介入が次第に少なくなる．しかし現実的には，このブートストラッピングの手法は次の問題点を持つ．

- どれだけ多くの用例を集めれば，安定した精度が得られるのかが不明である．比較的小さなドメイン（分野）でも，用例が十分に集まり，もはや人間の介入が不要であるといった状態には，なかなか到達しない．
- 単語や，単語の依存関係の分布は，分野が異なれば変化する．つまり，分野が変われば，それに対応した用例を集める必要がある．
- 用例を手で修正するのは，特にシステムの内部構造を知らない一般ユーザにとっては面倒である．

そこで，本論文では，用例に基づく多義性解消システムにおいて，人間の介入をできるだけ抑えながら，用例を自動的に学習する手法について述べる．まず，著者らが従来使用してきたシステムを基準として，英文の係り受けの多義性解消実験を行い，現状での評価および考察を行う．さらに，人間の介入をできるだけ抑えるための知識獲得の方法を提案し，従来のシステムに適用した結果を示す．

2 基準となるシステム

従来の用例に基づく多義性解消システムとして，著者らか開発したSENA [3] を用いる．SENAは，英文の(1)係り受けの多義性，(2)並列句のスコーピング，(3)語義の多義性，を制約と用例（およびシソーラス）を用いて解消するシステムである．

2.1 用例ベースとシソーラス

ここで，SENAで用いられている用例ベースとシソーラスについて簡単に述べる．表1にそれらの大きさを示す．用例は，単語間の2項関係の集合として記述されている．個々の用例は，下のように表現されている知識のタイプに応じて記述されている．

(E1) (("store+V" *STORE-1) ("on" TIME-SPACE-PLANE) ("disk+N" *DISK) 1)
(E2) (("store+V" ?) ("on" ?) ("disk+N" ?) 1)
(E3) (("store+V ?") (PP "in") (*+N *) 1)

(E1)は，単語間の構造と意味の両方の関係が記述された用例である．"disk" が "store" に前置詞 "on" を介して係ることを表している．*で始まるシンボルは，語義を表している．語義の多義性を解消するには，このタイプの用例が必要であるが，係り受けの多義性解消であれば，(E2)のような語義の情報がない用例でも利用可能である．更に抽象化された用例として，(E3)のようなものがある．これは，単語とヘッドの名詞によらない前置詞の間の優先度を設定するものである．(E2), (E3)のタイプの知識は，コーパスから自動的に獲得することが可能であり[1]，全用例の半分を占めている．

また，用例ベースは．対象分野毎に作られている．一般用例ベースは，一般辞書の定義文や例文から構築されており，常時使用される．分野用例ベースは，現在3種類あり，処理対象によって，使用するか否かどうかが決定される．また，個々の用例ベースに優先順位を設けることができる．現在この作業は，手動で行っているが，これを自動化しようという試みもある [2]．個々の用例の適用率を向上させるために，シソーラスを用いる．SENAでは，現在，一般の辞書から抽出した同義および上位下

[1]例えば，"Data is stored in the disk." のような文では，"in the disk" の係り先は一意に決まる．

用例ベース	
分野	大きさ
一般	30,000
計算機マニュアル	27,000
ビジネス文書	2,000
シソーラス	
同義関係	283,000 (見出し 11,006 語)
上位下位関係	6,400

表 1: 用例とシソーラスの大きさ

位関係を用いている.

2.2 多義性解消の仕組み

本章では，従来の用例に基づく多義性解消の手法を簡単に説明する．これは，入力（文）に対して，それと一致あるいは類似する用例を検索し，その用例に含まれる知識を用いて，入力に対して処理を行う手法であり，大きく次の 2 つのステップに分けることができる．

1. 用例の検索
2. シソーラスを用いた類似度計算
3. 類似した用例を用いた解の出力

ここでは，次の (S1) を入力として，SENA における前置詞句の係り先の多義性解消を例にとり，上のステップを説明する．

(S1) The system can store a new program in the repository.

S1 の前置詞句"in the repository" の係り先には，次の 2 つの解釈がある．これらの関係を単語間の head-modifier 関係で表すと次のようになる．

(解釈 1) ("store+V" (PP "in") "repository+N")
(解釈 2) ("program+N" (PP "in") "repository+N")

解釈 1 は，前置詞句が動詞"store" に係ることを示し，解釈 2 は，名詞 "program" にかかることを示す．この場合，解釈 1，解釈 2 が入力となる．多義性解消は，それぞれの解釈に対し，これともっとも類似する用例を検索することで行われる．入力と完全に一致する用例がない場合には，シソーラスを用いて単語間の類似度を計算し，類似用例を検索する．

多義性解消を行う際には，様々なレベルの知識が必要となる．SENA で用いられている知識には次のようなものがある．

(a) 用例ベースから計算される優先度

上で説明した用例とシソーラスを用いて計算される優先度である．

(b) 文法および意味制約

係り受けの非交差や，選択制約といった，解が満たさなければならない条件である．**(c) 統計的な優先度**

用例ベースを補完するために，単語の頻度情報を主に用いた優先度である．

(d) 経験則

係り先の曖昧な句に対して，最も近い句に係りやすいといった規則の集合．(a),(b),(c) の知識で多義性が解消できない時に用いる．

これらの知識は，(a),(c),(d) の順番に適用される．この際，(b) の制約が充足されていなければならない．

このように，知識の記述を分離しているのは，与えられた多義性に対して，出来るだけ適切なレベルで解消を行うためである．用例に基づく手法は，個々の単語に依存した知識を記述でき柔軟な処理を行うことができるが，従来の規則や経験則でもカバーできるものもあり，これらの場合，個々の用例で記述するのは冗長であり，用例に比べ，抽象化され適用範囲の広い規則で記述するのが効率が良い．

	Set1	Set2	Set3	Set4	Set5	合計
異なり語数	840	755	538	477	885	923
単語 (%)	85.6	84.8	90.5	86.1	82.5	85.9
依存関係 (%)	47.1	50.0	45.3	52.1	60.8	51.6

図 1: 用例のカバレッジ

2.3 実験 1: 係り受けの多義性解消

ここでは，SENA を用いて，英文の係り受けの多義性解消実験を行った結果を示す．テスト文は，計算機マニュアルからとったテスト文 500 文を 5 つに分けて用いた．用例ベースは，一般と計算機マニュアルの 2 種類を用いた．

図 1に，用例ベース (およびシソーラス) がテスト文中の (1) 単語 (1 項), (2) 依存関係 (2 項) をどの程度カバーしているかを調べた結果を示す (カバー率であり，正解率ではない).

単単語のレベルでは，テスト文に出現した単語の 85.9%が用例ベースに出現したが，2 項関係では，51.6% のカバー率であった．このことは，用例を用いた類似度計算だけでは，十分に多義性を解消きないことを示している．

また，図 2には，実験で正しく多義性が解消されたもののうち，上で挙げた (a)-(d) のタイプのどの知識によって解が一意に決定したかを調べた結果を示す．また，図 3では，適用した知識が，どの位信頼できるものであった

知識の種類	正解率 (%)
(a) 用例	43.6
(b) 制約	25.5
(c) 統計	8.9
(d) 経験則	7.2
(a)+(b)+(c)+(d)	85.2

図 2: 知識タイプ毎の正解率

知識の種類	信頼率 (%)
(a) 用例	93.0
(b) 制約	98.4
(c) 統計	68.8
(d) 経験則	68.4

図 3: 知識タイプの信頼度

かを，各々のタイプで解が一意に決まったもののうちの正解の割合を求めることで，計算した．図 2,3から，用例に関しては，正解に占める割合，および信頼性もかなり高く，用例およびシソーラスの質および量を向上させることが，システム全体の精度に大きく貢献することがわかる．次章では，これらの知識を自動的に学習する手法について述べる．

3 分野依存の単語間の関係の学習

用例に基づく多義性解消では，用例の適用率を上げるために，シソーラスが用いられる．しかし，分野に依存したシソーラスは整備されていないことが多い．また，シソーラスは単語間の上位下位関係を記述したものであるが，用例を用いた類似度計算においては，上位下位関係であることが重要なのではなく，単語がなんらかの尺度で「類似」していることが計算可能であることが重要である．本論文では，処理対象のテキストから，単語間の関係を，単語の共起関係の類似性から抽出する手法を提案する．これは，ある単語を別の単語に置き換えることができるという観点から，置換可能関係と呼ぶ．

例えば，次のような文を考える．

(S2) Insert the correct printer driver diskette in the drive.

“diskette” は，用例ベース中にも，シソーラス中にも出現しないので，類似度を計算することはできない[2]．

しかし，“diskette” は，この文が含まれるテキストには，複数回出現する．テキスト中の文脈は，その分野を

[2]この文は，パソコンのマニュアルから採った．2 章で言及した計算機マニュアル分野の用例ベースは大型機の用語集やマニュアルからとったものであり，“diskette” が全く出現していない．

(D1) (("remove+V" (THEME) "diskette+N") 1.0)
(D2) (("insert+V" (THEME) "diskette+N") 2.0)
(D3) (("install+V" ("from" SOURCE) "diskette+N") 0.75)
(D4) (("diskette+N" (SUCH-THAT) "instruct+V") 2.04)
(D5) (("diskette+N" ("from" SOURCE) "drive+N") 0.25)
(D6) (("diskette+N" ("in" GOAL) "drive+N") 1.0)
(D7) (("create+V" (THEME) "diskette+N") 8.0)
(D8) (("book+N" ("from" SOURCE) "diskette+N") 0.24)

図 4: "diskette" を含む依存関係

置換可能語	優先度	置換可能語	優先度
"file"	40.0	“data”	27.0
"view"	40.0	“library”	24.4
"job"	35.0	“definition”	24.0
"id"	32.0	“service”	24.0
"table"	28.0	“user”	24

図 5: “diskette” と置換可能な語の候補 (上位 10 語)

反映した用例そのものであり [1]，文脈中のある単語の依存関係と，用例ベース中の依存関係を比較することで，テキスト内の単語と，用例ベースの単語間の関係を推定することができる．

図 4 に，この単語を含む依存関係を示す．末尾の数字は，SENA によって付与された優先度である．これは，2.2 節で説明した多義性解消に用いられる知識の (b)-(d) までを使って計算された値である．

これらの依存関係に含まれる単語は，用例ベースにも出現することが期待できる．例えば，(D1) の依存関係に関連して，次のような依存関係が用例ベースから検索できる．

(D11) (("insert+V" *INSERT) (THEME) ("CD-ROM+N" *CD-ROM) 2)
(D12) (("insert+V" *INSERT) (THEME) ("data+N" *DATA) 1)
(D13) (("insert+V" *INSERT) (THEME) ("disk+N" *DISK) 1)

ここで，“CD-ROM”，“data”，“disk” は，“diskette” の置換可能語の候補となる．各単語に対して，置換可能性に関する優先度が，対応するテキスト内の依存関係と用例ベース内の依存関係の優先度の積をとることで計算される．D2-5 の依存関係に対しても同様の処理を行い，置換候補の単語の優先度を計算した結果を図 5に示す．

図 5をみてわかるのは，優先度の高いものが必ずしも，“diskette” と関連が深いように思えないことである．こ

置換可能語	優先度	置換可能語	優先度
"book"	6.75	"row"	4.0
"drive"	6.0	"CD-ROM"	3.5
"information"	5.0	"adapter"	3.5
"server"	5.0	"character"	3.0
"disk"	4.25	"data"	3.0

図 6: 典型的な用例を用いた "diskette" と置換可能な語の候補 (上位 10 語)

の原因は，用例の「典型性」を無視したところにある．例えば，(D7) から，用例ベース中で"create" の目的格にくる単語が置換可能語の候補となるが，"create" は，用例ベース中で多数出現し高い優先度を持っているので，(D7) からの候補が優先される結果になってしまう．ところが，(D7) は，"diskette" に関する典型的な例ではなく，その結果，正しい候補が選ばれないことになる．

これを防ぐために，典型的でない用例を除去することが考えられる．除去の基準としては，その用例の用例ベース中での頻度を用いる．本論文では，100 回以上用例ベースに出現する単語を含む用例は，計算の対象にしないという基準を設けた．その結果，(D7) は，"diskette" の依存関係から除去される．残りの依存関係，(D1-6), (D8) を用いて，置換可能語を求めた結果を図 6に示す．

4 用例の獲得

本節では，単語間の依存関係を獲得する手法について述べる．次の例文を考える．

(S3) The system can store a new program in the repository.

この文に対して，多義性を解消し，結果を依存関係として抽出した結果を下に示す

(E1) (("store+V" *STORE-1) (THEME) ("program+N" *PROGRAM-1) 1M)

(E2) (("store+V" *STORE-1) ("in" GOAL) ("repository" *REPOSITORY) 1.2M)

(E3) (("program+N" *PROGRAM-1) ("in" TIME-SPACE-PLANE) ("repository" *REPOSITORY) 0.4M)

(E4) (("system+N" *SYSTEM) (AGENT) ("store" *STORE-1) 1M)

末尾の数字は，計算された優先度である．従来は，これを人間がチェックし，正解のみを用例として登録する (その場合，(E3) は登録されない) が，人手の介入を抑えるため，優先度を付与したままで，全てを用例ベースに登録する．ただし，これが自動的に付与された用例であるためのフラグ (末尾の M) をたてておく．システムは，正しい用例と，自動獲得された用例が競合する時には，前者を優先することで，用例ベース内で矛盾が生じるのを防ぐ．フラグの立った用例は，後から，何らかの形で，正しいと判断された時に，フラグが除去される．

これらの用例は，あくまでもシステムが推定したものであり，最終的には，人間が正解かどうかチェックしなければならない．最小限の修正で，効果をあげるためには，次のような条件を満たす用例を優先して修正する必要がある．

1. 出現回数の多い用例
2. 用例ベースやシソーラス中にない単語を含む用例
3. 競合する用例の優先度に差が少ないもの

4.1 実験 2

前章で説明した知識の自動学習を行い，獲得した用例を追加して，係り受けの多義性解消実験を行った．テスト文は第 2 章で用いたものと同じものを用いた．テスト文から，抽出した知識は，置換可能関係 (それぞれの単語に対し，優先度上位 8 位までの単語) と実験 1 で作られたテスト文の優先度付きの依存関係である．この実験では学習した知識に対する人手の介入はない．結果を下に示す．

	正解率 (%)
実験 1 (自動学習前)	85.9
実験 2 (自動学習後)	90.3

5 おわりに

多義性解消に必要な知識の自動学習法について述べた．今後の課題として，どのように人間の介入をコントロールしていくかということがある．用例ベース内で疎な空間を判別したり，人間が修正しやすい環境を開発する必要がある．また，複数の分野依存の用例ベースをどのように構築し，また管理するのかということも大きな問題であろう．

参考文献

[1] T. Nasukawa. "Discourse Constraint in Computer Manuals". In *Proceedings of TMI-93*, pages 183–194, 1993.

[2] K. Takeda. "Portable Knowledge Sources for Machine Translation". In *Proceedings of COLING-94*, pages 85–89, 1994.

[3] N. Uramoto. "Example-Based Word-Sense Disambiguation". *IEICE Transactions on Information and Systems*, E77-D(2), 1994.