自然言語処理 112－11
（1996．3．15）

距離反比例型スコアを導入したコロケーションの自動抽出法

尾本 貴志　北 研二

徳島大学 工学部

コーパスからコロケーションを自動的に抽出するための様々な方法が提案されてきたが，その方法の多くは連続したコロケーションの抽出が目標であった．そこで，本稿では，不連続なコロケーションの自動抽出を目的とし，そのための新しい評価関数である『距離反比例型スコア』を導入する．また，距離反比例型スコアを用いて行なった抽出実験の結果を示す．さらに，より効率的に不連続なコロケーションを抽出するため，相互情報量をもとにした評価関数を提案する．

Automatically Extracting Collocations Using a Distance-inverse Score

Takashi Omoto　Kenji Kita

Faculty of Engineering, Tokushima University

Many methods have been suggested for automatically extracting collocations from a corpus. Most of them, however, do not apply to extracting discontinuous collocations. In this paper, we propose a new measure, called a distance-inverse score, to automatically extract discontinuous collocations. We also show examples of extracted collocations using this measure. For a further research, we present another measure based on the mutual information to fully automatically extract discontinuous collocations.

1　はじめに

コロケーション (Collocation) とは，テキスト中に頻出する単語の集まりあるいは連語を指す用語であり，これらは単語間の共起情報を与える重要な知識源である．コロケーションは，言語的あるいは慣用的な表現であることから様々な形態が考えられる．例えば，英語においてよく用いられる “Thank you very much.” や “I would like to ～” のような表現はそれぞれの単語が連続しているが “not only ～ but (also) ～” や “not so much ～ as ～” のように必ずしもコロケーションは連続した単語ばかりだとは限らない．

これまでに，コーパスから自動的にコロケーションを抽出する方法として，構文情報を利用した方法 [1]，相互情報量を用いた方法 [2]，仕事量基準を用いた方法 [8] をはじめとして様々な方法が提案されている [3, 4, 5, 6]．我々は，相互情報量を用いた方法と仕事量基準を用いた方法の比較実験を行ない，相互情報量を用いた方法は，複合語やコーパスのドメインに依存した複合名詞句の抽出に適しており，仕事量基準を用いた方法は，述語型の定型表現や慣用句の抽出に適していることを示した [9, 10, 11, 12]．また，仕事量基準を用いたコロケーションの抽出法において，データ構造を木構造化することにより，cpu 時間の短縮とメモリの効率化を実現することができた [13]．

しかし，従来の方法では，連続したコロケーションは抽出できても不連続なコロケーションは抽出することができないため，最近は，不連続な離散型のコロケーションの抽出に関する研究も行なわれている [7]．本稿では，不連続なコロケーションの抽出を目的とした新しい方法を提案し，従来の方法による結果との比較を行ない，その方法の有効性について考察する．

2 コロケーションについて

コロケーション (Collocation) とは, コーパステキスト中に頻繁に出てくる単語の集まりであり, その例としては "in spite of" や "for the purpose of" のような慣用句 (Idioms), "Thank you very much" や "I would like to" のような定型表現 (Frozen Expression), "high school" や "father-in-law" といった複合語 (Compound words) などがある. これらコーパスから抽出されたコロケーションの知識は, 次のことがらによって重要であると考えられる.

第一に, コロケーションの知識は, 音声認識や文字認識の分野に応用できる. 例えば, 音声認識である単語の認識ができなくても, その単語の前後にある単語を含むコロケーションを検索することによって, 認識できなかった単語の候補を絞り込むことができる.

第二に, コロケーションは第二言語学習者にとって習得するのが難しいといわれる. 例えば, "strong" と "powerful" はよく似た意味で使われるが, "strong tea" と "powerful tea" および "strong car" と "powerful car" の違いは, 日本人にとっては分かりづらい. しかし, 母国語話者は "strong tea" と "powerful car" と使い分ける. このように, 似たような意味の単語でも使われ方など微妙な区別がある [14, 15].

第三に, コロケーションは分割翻訳に適用できるということである. 人間の翻訳過程は, まず文章を適当な小片に分け, 次にそれぞれの小片を類推によって翻訳し, その小片ごとに翻訳されたものを一つの文章にする. この小片にあたるコロケーションを, コーパスを解析することにより, 他の言語に相当するコロケーションを検索し, 二つ以上の言語間での翻訳を行なうことが可能である.

このほか, コロケーションの知識は, 正しくない単語の使い方や誤った文法の発見にも適用可能であり, 生徒の英作文の添削において辞書的な活用ができると考えられる.

3 従来の方法における評価関数

コーパスからコロケーションを自動的に抽出するためには, 何らかの基準となる評価関数とその値に対する閾値を導入して, それによってコロケーションとみなすか否かを決定する. この基準の善し悪しがコロケーションを抽出する際によい結果が得られるかどうかの重要なポイントとなる. 我々はこれまでに相互情報量と仕事量基準の二つの評価関数を用いてコロケーションの抽出を行なってきたが, そのそれぞれの評価関数の特徴について考察してみる.

3.1 相互情報量

相互情報量は二つのことがらに関して, その間の共起や関連性を表す尺度である. 二つの単語 (列)x,y を同時に観測する確率 $P(x,y)$ と, x,y を独立に観測する確率 $P(x)$, $P(y)$ とを用いて, 相互情報量 $I(x;y)$ は次式で定義される [2].

$$I(x;y) \equiv \log_2 \frac{P(x,y)}{P(x)P(y)} \tag{1}$$

$$P(x) = f(x)/N \quad P(y) = f(y)/N$$

$$P(x,y) = f_w(x,y)/N$$

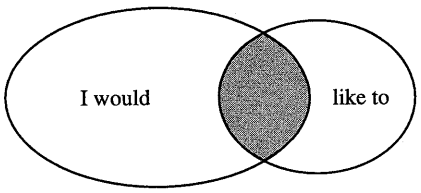
ここで, Nはコーパスに含まれる全単語数, $f(x)$ はコーパステキストにおける単語 (列)x の出現頻度を表す. よって $P(x)$ は, N個の単語からなるコーパステキストでの単語 (列)x の出現確率を表す. また $P(x,y)$ は連続した w個の単語の中で同時に単語 (列)x,yが観測される確率を表している.

例えば, "I would" の出現回数が 160 回, "like to" の出現回数が 80 回, "I would like to" と連続して出現した回数が 40 回, また "yacht" と "harbor" の出現回数が各々3 回, "yacht harbor" と連続して出現した回数が 2 回として, さらにコーパスの規模が 10 万語である場合, 二つの単語列 "I would like to" と "yacht harbor" の相互情報量は次のようになる.

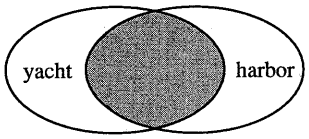
$$\begin{aligned} I(\text{I would; like to}) &= \log_2 \frac{\frac{40}{100000}}{\frac{160}{100000}\frac{80}{100000}} \\ &= 8.29 \\ I(\text{yacht; harbor}) &= \log_2 \frac{\frac{2}{100000}}{\frac{3}{100000}\frac{3}{100000}} \\ &= 14.44 \end{aligned}$$

相互情報量は, 図 1における二つの集合の和集合に対する共通集合の占める割合が大きい程, 値が大きくなる. つまり図中の濃い色の部分が全体に対して大きな割合を占めると値が大きくなる.

前の計算からわかるように "I would like to" という単語列は "yacht harbor" という単語列よりも多く



I wouldの出現回数 = 1 6 0回
like to の出現回数 = 8 0回
I would like to の出現回数 = 4 0回



yacht の出現回数 = 3回
harbor の出現回数 = 3回
yacht harbor の出現回数 = 2回

図 1: 二単語列の出現回数

出現しているにもかかわらず, 相互情報量の値では小さくなっている. 言語的にコロケーションとしてふさわしいのは "I would like to" であるが, 相互情報量の値から判断するとコロケーションとして抽出されない恐れがある.

相互情報量は二つの単語 (列) の結びつきの強度はわかっても, 関数の中ではそれぞれの単語 (列) の出現回数は比で表されることになり, その情報を失ってしまうことになる. よって出現回数が少ないものでも同時に出現する割合さえ高ければ大きなスコアを与えられるので, 滅多に出現しないあまり重要でない単語列に対しても大きな値を与える可能性がある. 単語列をコロケーションであるか判断する上で重要なことは, 単語の結びつきの強度よりもむしろその単語列の出現頻度であり, この出現頻度を考慮されていない点が相互情報量による方法の欠点であるといえる.

3.2 仕事量基準

相互情報量を用いた方法では, コロケーションとして採用するか否かの重要な判断基準である単語列の出現頻度の情報が失われる. これに対し, 頻出する単語列ほどより重要な単語列であるという概念のもとに提案されたのが仕事量基準である. 北らはコーパステキスト中にあらわれる単語列 α に対して, 次のような関数を仕事量基準として定義した [8].

$$K(\alpha) = (|\alpha| - 1) \times n(\alpha) \qquad (2)$$

なお, 式 (2) の中で, 単語列 α に対して次のような表記法を用いている.

$|\alpha|$ ・・・ 単語列 α の長さ (語数)
$n(\alpha)$ ・・・ 単語列 α のコーパス中での出現回数

この関数の意味は, コーパステキスト中に出現する長さ $|\alpha|$ の単語列αを一まとまりの単語列としてみなすことにより, コーパステキスト上をサーチする仕事量を一回の出現で $(|\alpha| - 1)$ 削減でき, よって全体で $(|\alpha| - 1) \times n(\alpha)$ の仕事量が削減できるということである. 仕事量基準は単語列の長さと出現頻度に比例するので, 頻繁に出現する単語列ほど値は大きくなる.

しかし, 単に単語列といっても, 二つの単語からなるものもあるし, 三つの単語からなるものもあり, 単純に単語列の出現回数で比較するわけにはいかない. 例えば, 単語列αの出現回数が n であるとき, αの部分単語列βの出現回数は必ず n 以上となるため, 単純に出現回数だけからは真のパターンは抽出できない.

例えば, "in spite" と "in spite of" のように一つの単語列がもう一つの単語列の部分単語列になっている場合を考える. コーパステキスト中の二つの単語列の出現頻度がそれぞれ 50 と 20 とすると, 二つの単語列に対する仕事量基準は

$$\begin{aligned} K(\text{in spite}) &= (2-1) \times 50 \\ &= 50 \\ K(\text{in spite of}) &= (3-1) \times 20 \\ &= 40 \end{aligned}$$

となり, このままであると "in spite" の仕事量基準の方が大きくなる. しかし, "in spite" が参照された

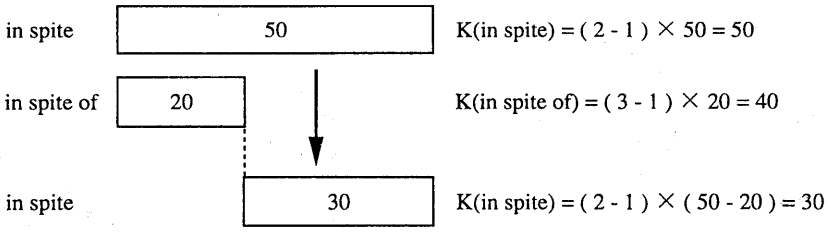


図 2: 部分単語列に対する仕事量基準

50 回の内, "in spite of" の部分単語列として参照された回数が 20 回あるので, 純粋に "in spite" が参照されたのは (50 − 20) で 30 回である. よって, "in spite" に対する仕事量基準は

$$\begin{aligned} K(\text{in spite}) &= (2-1)\times(50-20) \\ &= 30 \end{aligned}$$

として計算するのが妥当である.

このように部分単語列には再計算を施すことによって, 長い単語列に対しても均整のとれた値が算出できるようになる.

4 距離反比例型スコアについて

今回, 我々は不連続なコロケーションも抽出できるような手法を提案する. そのために, 抽出のための新たな評価関数『距離反比例型スコア』を導入する.

コーパステキスト中に出現する二つの単語列α, βの間のスコアを, 次のように定義する. なお式中のf_rは, 二つの単語列が距離 rにおいて出現した回数を表す.

$$\begin{aligned} S(\alpha,\beta) &= f_1 + \frac{f_2}{2} + \frac{f_3}{3} + \cdots + \frac{f_n}{n} \\ &= \sum_n \frac{f_n}{n} \end{aligned} \quad (3)$$

また, 二単語列の平均距離 Dを次のように表す.

$$D(\alpha,\beta) = \frac{1\times f_1 + 2\times f_2 + \cdots + n\times f_n}{\sum_n f_n} \quad (4)$$

上記の関数は二つの単語列の距離 1 から n まで (隣接する単語列の距離を 1 とする) の出現回数に対し, それぞれの出現回数に距離の逆数を掛け, それを合計したものが値となる.

なぜコロケーションを抽出するための評価関数としてこのような関数を用いるかという理由については, まず第一に, 前節で述べたように単語列の出現頻度はコロケーションであるかを判断する上での重要なファクタであり, 評価関数は出現頻度に比例もしくはべき乗の形となるのがふさわしいと考えられる. そして二番目に, 二つの単語列の間の距離が大きくなれば, その二単語列は偶然的に同時に出現する確率が高くなるので, 距離に対しては反比例となるようにすれば適当であると考えた. また, 平均距離は二つの単語列が連続しているかどうかを判断する指標として設定した.

ところが, この方法では n の値が大きくなればなるほど単語列の出現頻度の個数が多くなり計算の処理が膨大となる. 無論, 計算機の処理能力に限界があるので, n の値は適度な数の範囲でなければならない. ここでテキスト中の単語から二つの単語列を決定する方法について述べる.

例えばここに異なる 5 つの単語からなるテキストがあるとする.

a b c d e

この 5 つの単語を二つの単語列に分ける組合せは
(a,b) (a,bc) (a,bcd) (a,bcde) (a,c) (a,cd) (a,cde)
(a,d) (a,de) (a,e) (ab,c) (ab,cd) (ab,cde) (ab,d) (ab,de)

⋮

(bc,d) (bc,de) (bcd,e) (c,d) (c,de) (c,e) (cd,e) (d,e)
の 35 通りである.

一般的に n 個の単語からなるテキストを上のように二つの単語列に分ける組合せは

$$\frac{1}{24}(n-1)n(n+1)(n+2)$$

となり, 計算のオーダーが $O(n^4)$ となってしまい, かなりの計算量となってしまうのが本手法の欠点である.

実際にコーパステキストに出現した二つの単語列 (be able, to) に対する距離反比例型スコアの計算について述べる. 二つの単語列の距離 1〜10 での出現頻度は下の通りとなった.

r	1	2	3	4	5	6	7	8	9	10
f_r	67	0	2	2	1	0	1	0	1	0

よって二単語列 (be able, to) のスコア Sは

$$\begin{aligned} S(\text{be able, to}) &= 67+\frac{2}{3}+\frac{2}{4}+\frac{1}{5}+\frac{1}{7}+\frac{1}{9} \\ &= 68.6206 \end{aligned}$$

となり, また平均距離 Dは

$$D(\text{be able, to}) = \frac{1\times 67+3\times 2+4\times 2+5\times 1+7\times 1+9\times 1}{67+2+2+1+1+1} = 1.378$$

となる.

5 実験概要

我々は, この新しい評価関数である『距離反比例型スコア』を用いて, コロケーションの抽出実験を行なった. 実験には, ATR 対話コーパス [16] を用いた.

5.1 実験方法

実験の方法について簡潔に述べる.

1. コーパスから単語/文節だけを抽出しトレーニングテキストを作成する.
2. 作成したトレーニングテキストを用いて, 二つの単語列がある距離だけ離れて出現する回数を調べる.
3. それぞれの二単語列に対するスコア Sと平均距離 Dを計算する.
4. 算出されたスコアを降順に並べ, 上位をコロケーションとして抽出する.

表 1: 実験に用いたデータのサイズ

内容	言語	総文数	延べ単語数
会議	英語	21,087	267,916
	日本語	23,823	299,118
旅行	英語	16,314	228,334
	日本語	19,577	271,000

実験に用いるコーパスの規模は表 1に示すとおりである. それぞれのコーパスに含まれる延べ単語数は 20 万〜30 万語程度であるが, 前節で述べた単語列の総組み合わせ数は 1000 万〜3000 万通りになるので, 我々の計算機では莫大な計算時間が必要となる.

そこで計算時間短縮のため, 与えられたコーパスに対し次のような前処理をしてからコロケーションの抽出を行なう.

- 出現頻度が 30 未満 (コーパス全体に対する出現確率が約 0.01%以下) の単語を排除する.
- 二単語列の距離が 11 以上の組合せについては考慮しない.

これにより総組合せ数を約 1/20 以下に抑え, 実験を行なうことにした. ただし, この方法であると出現頻度が 30 未満の単語を含む単語列を抽出することは不可能である. よってその中に重要な単語列が含まれている可能性がなくもないが, 今回の実験ではそれらの単語列は無視することにする.

5.2 実験結果・考察

『距離反比例型スコア』を用いて抽出されたコロケーションの例を表 2に示す. 表 2中のコロケーションは, ほとんど連続型のコロケーションである. そこで平近距離が比較的大きく, それによって不連続なコロケーションであると思われるものを抽出してみた (表 3参照). 表 3から分かるように, スコアに加

表 2: 抽出されたコロケーションの例

Score	Dist	Collocation	Score	Dist	Collocation
652.1	2.27	of ; the	635.2	1.67	し ; て
521.1	2.45	the ; conference	629.5	1.06	です ; が
371.6	1.06	Thank ; you	570.8	2.28	し ; ます
182.7	1.19	would ; like to	484.2	1.01	でしょ う ; か
120.5	1.01	Thank you ; very much	352.0	1.43	て ; おり ます
119.0	1.70	Thank you ; for	295.9	1.13	いたし ; ます
96.6	2.58	the ; International Conference	295.0	1.00	の です ; が
89.5	1.96	I would like ; to	276.4	1.07	ます ; ので
88.0	1.39	I ; see	261.9	1.36	と ; 思い ます
87.6	1.94	at the ; conference	188.0	1.00	わかり まし ; た
79.0	1.00	application ; form	188.0	1.00	ございまし ; た
73.8	1.23	registration ; fee	170.0	1.00	ありがとう ; ございまし た
68.6	1.38	be able ; to	142.7	1.25	たい ; の です
65.0	1.00	May I ; help you	140.2	1.53	し ; て おり ます
62.5	1.09	looking forward ; to	140.2	1.04	失礼 ; します
60.0	1.21	I'm ; sorry	138.7	2.00	に ; なっ て
59.5	1.02	May I speak ; to	135.9	1.24	お 願い し ; ます
53.0	1.00	as soon ; as	121.5	1.24	あり ; ませ ん
370.7	3.03	for ; the	1261.7	1.21	です ; か
238.0	1.00	Is that ; so	1233.5	1.13	ん ; です
234.1	1.03	Thank ; you	812.7	4.14	の ; です
198.3	1.67	I ; see	792.1	1.44	そう ; です
158.4	1.32	would ; like to	781.0	1.28	です ; ね
116.8	1.33	a ; little	599.0	1.00	でしょう ; か
107.4	2.41	to go ; to	585.4	1.43	ます ; か
89.0	1.00	All ; right	573.8	2.29	し ; ます
88.0	1.00	Thank you ; very much	519.5	1.19	いたし ; ます
87.0	1.00	Good ; bye	497.5	3.41	の ; です が
85.9	1.50	My name ; is	381.6	1.56	そう ; です か
66.8	1.55	make ; reservations	318.5	1.19	なり ; ます
66.0	1.00	How ; much	314.0	1.33	な ; ん です
64.0	1.00	in that ; case	276.5	1.03	そう です ; ね
61.2	1.11	it would ; be	266.8	1.15	て おり ; ます
59.0	1.00	would you ; like	249.4	1.20	と いう ; こと
57.2	1.09	this is ; JTB	218.2	1.61	と ; 思い ます
50.3	1.15	Thank you ; for	203.2	1.10	お 願い し ; ます

表 3: 不連続なコロケーションの例

Score	Dist	Collocation	Meaning
18.1	2.16	as ～ as possible	できるだけ～
11.6	1.93	not ～ any	一つも～ない
8.3	1.60	put ～ together	～を集める
6.3	1.29	not ～ at all	全く～ない
6.1	3.31	inform ～ of ...	～に... を知らせる
4.8	5.23	not ～ but ...	～ではなく... である
4.3	4.08	take ～ for ...	～を... と思い違いする

えさらに平近距離を用いることにより, 不連続なコロケーションとして妥当なものを抽出することができた.

不連続なコロケーションは, コーパスに含まれている数が少なく, 連続なコロケーションに比べて実際に存在する数も少ない. 表 3からも分かるように, 連続なコロケーションよりもスコアがはるかに小さい. このため, スコアのみにより, 不連続なコロケーションを抽出するのは無理であり, 今回の実験では平近距離を併用した.

また, 連続なコロケーションの抽出に関しては, 仕事量基準の結果に準じた結果が得られた. よって, 提案した『距離反比例型スコア』は, 連続したコロケーションの抽出のための評価関数としても適用可能である.

6 今後の課題

不連続なコロケーションは連続したコロケーションに比べ, 出現頻度がはるかに少ない. よって, 評価関数に出現頻度が大きな影響を与えるような関数では, 抽出が困難である. 『距離反比例型スコア』も, 仕事量基準と同様に, 単語列の出現頻度に大きな重みを与えているので, 出現回数の少ない不連続なコロケーションを抽出するためには, 単語 (間) の平均距離を考慮する必要があった.

以上の考察から, 評価関数のスコアのみによる不連続なコロケーションの抽出では, 単語列の出現頻度に左右されずに, 離れた単語列と単語列の結びつきの強度を測るような関数が必要となってくる.

そこで次のような関数を不連続なコロケーションの自動抽出に適用できるのではないかと考えた.

$$I(\alpha;\beta) = \frac{1}{n}\sum_{r=1}^{n} \log_2 \frac{P_r(\alpha,\beta)}{P(\alpha)P(\beta)} \qquad (5)$$

$$P(\alpha) = f(\alpha)/N \quad P(\beta) = f(\beta)/N$$

$$P_r(\alpha,\beta) = f_r(\alpha,\beta)/N$$

ここで $P_r(\alpha,\beta)$ は, 二つの単語列α, βが距離 rだけ離れて同時に出現する確率を表している.

式 (5) は, 二つの単語列α, βが距離 1～n だけ離れている場合の相互情報量を合計し, n で正規化したものである.

この関数を用いることによって, 出現頻度の多少に関わらず, 単語列同士の結びつきが大きな単語列が抽出できる. しかし, この関数による値だけを用いると, 相互情報量を用いた場合のように, 無意味な単語列が多数抽出されてしまう恐れがある.

そこで距離反比例型スコアと併用し, いずれの関数に対しても閾値を設け, 両値が閾値以上の単語列をコロケーションと認定するような手法が考えられる.

7 おわりに

本稿では, 不連続なコロケーションの自動抽出のための評価関数である『距離反比例型スコア』を提案した. ATR 対話コーパスを用いた実験を行ない, 提案した評価関数から得られたスコアと単語 (列) 間の平均距離を同時に用いることにより, 不連続なコロケーションとして妥当なものを抽出することがで

きた. また, 提案した評価関数は, 仕事量基準と同様に, 連続したコロケーションの抽出にも適していることがわかった.

今後は, 最後に紹介した新しい相互情報量の式を用いた不連続なコロケーションの自動抽出について検討する予定である.

参考文献

[1] Basili, R., Pazienza, T. and Velardi, P.: A Shallow Syntactic Analyser to Extract Word Associations from Corpora, *Literary and Linguistic Computing*, Vol. 7, No. 2, pp. 113–123 (1992).

[2] Church, K. W. and Hanks, P.: "Word Association Norms, Mutual Information, and Lexicography", *Computational Linguistics*, Vol. 16, No. 1, pp. 22–29 (1990).

[3] Smadja, F.: "Retrieving Collocations from Text: Xtract", *Computational Linguistics*, Vol. 19, No. 1, pp. 143–177 (1993).

[4] 新納浩幸, 井佐原均: "疑似 N グラムを用いた助詞的定型表現の自動抽出", 情報処理学会論文誌, Vol. 36, No. 1, pp. 32–40 (1995).

[5] 新納浩幸, 井佐原均: "片方向の共起性による述語型定型表現の自動抽出", 自然言語処理学会, Vol. 2, No. 3, pp. 73–86 (1995).

[6] 池原悟, 白井諭, 河岡司: "N-gram を用いた連鎖型共起表現の自動抽出法", 言語処理学会第 1 回年次大会発表論文集, pp. 313–316 (1995).

[7] 白井諭, 池原悟, 小見佳恵: "N-gram を用いた離散型共起表現収集の一手法", 言語処理学会第 1 回年次大会発表論文集, pp. 317–320 (1995).

[8] 北研二, 小倉健太郎, 森元逞, 矢野米雄: "仕事量基準を用いたコーパスからの定型表現の自動抽出", 情報処理学会論文誌, Vol. 34, No. 9, pp. 1937–1943 (1993).

[9] 北研二, 加藤安彦, 尾本貴志, 矢野米雄: "コーパスからのコロケーションの自動抽出: 相互情報量および仕事量基準 – 言語学習の観点からの比較 –", 電子情報通信学会技術研究報告, ET93-128, pp. 31–38 (1994).

[10] Kita, K., Kato, Y., Omoto, T. and Yano. Y.: "Automatically Extracting Collocations from Corpora for Language Learning", *Corpora in Language Education and Research*, Wilson, A. and McEnery, T. (eds.), UCREL Technical Papers, Lancaster University, pp. 53–64 (1994).

[11] Kita, K. Omoto, T., Yano, Y. and Kato, Y.: "Application of Corpora in Second Language Learning — The Problem of Collocational Knowledge Acquisition —", *Proc. of the Second Annual Workshop on Very Large Corpora*, pp. 43–56 (1994).

[12] Kita, K., Kato, Y., Omoto, T. and Yano, Y: "A Comparative Study of Automatic Extraction of Collocations from Corpora: Mutual Information vs. Cost Criteria", *Natural Language Processing*, Vol. 1, No. 1, pp. 21–33 (1994).

[13] 尾本貴志, 北研二: "木構造によるコロケーションの自動抽出", 電子情報通信学会技術研究報告, ET94-148, pp. 141–148 (1995).

[14] Church, K. W., Gale, W., Hanks, P. and Hindle, D.: "Using statistics in lexical analysis", *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, Zernik U (ed.), Lawrence Erlbaum Associates, pp. 115–164 (1991).

[15] Smadja, F. A.: "Macrocoding the lexicon with co-occurrence knowledge", *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, Zernik, U. (ed.), Lawrence Erlbaum Associates, pp. 165–189 (1991).

[16] Ehara, T., Ogura, K. and Morimoto, T.: "ATR dialogue database", *Proc. of the 1990 International Conference on Spoken Language Processing* , pp. 1093–1096 (1990).