N-gramを用いた連鎖型共起表現の自動抽出法*1

池原　悟　　白井　諭　　河岡　司
（ＮＴＴコミュニケーション科学研究所）

１．まえがき

最近、自然言語処理において、大量のコーパスや用例の重要性が指摘され、それを分析する技術の必要性が増大している。言語データの中から、使用頻度の高いフレーズや表現のパターンを抽出する方法としては、２単語の結びつきの強度に着目した方法[1]や単語間の距離に着目した方法[2]、結合単語数と出現回数を考慮した方法[3,4]など[5]が提案されている。しかし、大量の言語データを対象とするとき、計算量の問題から、任意の長さで、出現頻度の高い表現を自動的に発見して、抽出することは困難であった。

これに対して、最近、大量の言語データを対象に、任意のｎに対するｎ-gram統計を高速に実行する方法が提案された[6]。この方法を用いれば、原文中に使用された文字列を、その長さ（文字数）の順かつ出現頻度の順に集計することが出来る。しかし、抽出する文字列間の相互関係が無視されているため、既に抽出された文字列の部分文字列が重複して抽出される。従って、抽出された文字列には、文法的、意味的にまとまりのない断片的な文字列が多数を占め、独立した表現の単位となるような文字列を選択する作業が大変であった。

本論文では、この問題を解決する方法として、言語データの中から、文字列を、それらが部分文字列として相互の重複のあるものは対象外とすると言う条件下で、文字数の多い順、かつ使用頻度の高い順に、漏れなく、自動的に抽出し、集計する方法を提案する。

２．従来の方法とその問題点

（１）文字列抽出の条件

自然言語表現に共起する表現としては、連語やフレーズのように連続した文字列を構成するもの（連鎖型共起表現）と、係り結び、呼応関係、特定の動詞と特定の名詞の組などのように、２種以上の文字列が、文中の離れた位置に現れるもの（離散型共起表現）がある。本報告では、前者の文字列を以下の条件で抽出することとする。

第１の条件：一致文字数の多い順に抽出する。

第２の条件：出現頻度の高い順に抽出する。

第３の条件：ある文字列が抽出されたとき、その文字列内に含まれる部分文字列は、既に抽出済みと考え、それ以降は抽出の対象としない。

（２）長尾・森の方法とその問題点

日本語文章データから、出現頻度の高い連続した文字列を抽出する長尾・森の方法[4]を要約すると以下の通りである。

［長尾・森の方法］

先頭文字から順に、０文字、１文字、２文字、・・・Ｎ－１文字を削除したＮ個の部分文字列（文字列単語と呼ぶ）に対して、各部分文字列の順に対応するＮ個のレコードを持った原文番地ファイルを作成し、文字列コード順にソートしたあと、前後の文字列の先頭からの一致文字数をカウントする方法で、以下の手順で構成される。

手順１：原文番地ファイルの作成
手順２：汎用ソートファイルの作成
手順３：一致文字数のカウント
手順４：文字列の抽出とカウント

この方法により、第１、第２の条件は満足されるが、目標とする第３の条件は満足されない。抽出された文字列の出現回数に対して、より長い文字列に含まれていた部分文字列の出現回数を差し引くなど、次数の異なる複数のｎ-gram集計表を組み合わせて計算する方法が考えられるが、集計表の生成された時点では、抽出された文字列の原文中での相互関係の情報が失われているため、計算は不可能である*2。

３．連鎖型共起表現の抽出法

３．１　重なる文字列の扱い

ｎ-gram文字列とｍ-gram文字列の抽出を考えたとき、問題となるのは、図１に示すように、両者に何らかのオーバーラップがあるときである。

---

*1 Automatic Extraction of Uninterrupted Collocation by n-gram Statistics, Satoru Ikehara, Satoshi Shirai and Tsukasa Kawaoka, NTT Communication Science Laboratories.

*2 最近、単語列の場合、ある単語列αの出現頻度を、その単語列の出現回数から、それを含むより大きな単語列βの出現回数を引くことにより評価する方法が使われていた例(4)もあるが、厳密には、この方法では正しい結果は得られない。

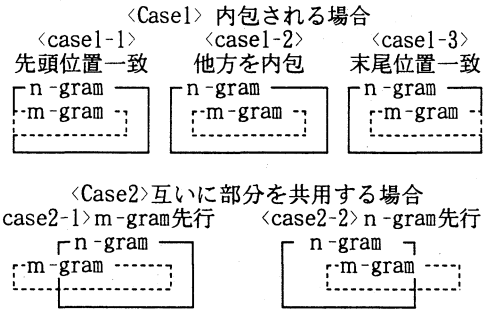

図１　抽出文字列の相互関係

（１）一方が他方を内包する場合

この場合は、さらに３つの場合に分けられるが、いずれの場合も、m-gram部分はｎ-gramとして抽出された文字列の部分に内包されるので、抽出の対象とならない。従って、ｎ-gram文字列を抽出するとき、ｎ-gram文字列として抽出した文字列の部分は、後の処理でm-gram文字列の対象としないよう「汎用ソートファイル」の該当レコードを無効化する必要がある。

（２）互いに部分を共用する場合

この場合は、さらに２つの場合に分けられるが、いずれの場合も、m-gram文字列とｎ-gram文字列は、原文上互いにその一部を共有するだけであるので、抽出対象文字列である。

３．２　文字列抽出アルゴリズム

（１）アルゴリズム

前章の議論をふまえ、原文データから、２回以上の出現回数を持つ固定的な（連続した）表現を文字列として、文字数の多い順に、かつ、重複なしに抽出するアルゴリズムを提案する。

＜前準備＞

まず、長尾等の方法で使用された「汎用ソートファイル」を拡張し、その各レコードに、「抽出文字数」の欄を設ける（図２参照）。

＜文字列抽出アルゴリズム＞

手順１：手順２：手順３：　長尾等の方法と同じ

手順４：「抽出文字数の記入」

上記で得た「拡張汎用ソートファイル」上の「抽出文字数」の欄に、自分のレコードの先頭から何文字までが抽出対象となっているかを記入する。具体的には、各レコードの抽出対象文字数の値を順に見て、自分のレコードの値が、直前のレコードと同一の数字か、それより大きい数字の場合は、一致文字数をコピーし、逆に、自分の値が直前のレコードの一致文字数より小さいときは、直前のレコードの値をコピーすればよい。

手順５：「拡張原文番地ファイルの作成」

手順４で得られた拡張汎用ソートファイルを原文番号順に再ソートしたあと、各レコードに「採否表示」の欄を設け、それを「拡張原文番地ファイル」とする。

手順６：「有効無効判定処理」

拡張原文ファイル上で、以下の方法で、先頭レコードから順に、各レコードの無効判定を行い、その結果を、「採否表示」欄に記入する。

すなわち、先頭レコードから「抽出文字数」の値を調べ、ｉレコード目で、始めて０でない値ｎを発見したとする。そのとき、ｉ番目のレコードの採否表示は「○（採用）」とし、それに続くｉ＋１，ｉ＋２，・・・，ｊ＋（ｎ－１）のレコードの抽出文字数の値を調べる。それらの値が、それぞれ、ｎ－１，ｎ－２，・・・，１以下であるレコード採否表示を「×（不採用）」とし、それ以外のレコードの無効表示は「○」とする。次に、新たに「○」と判定されたレコードの番号をｉとして、同様の処理を繰り返す。「○」と判定されたレコードのないときは、ｉ＋ｎのレコードまで進む。そのレコードの「抽出文字数」０の時は、はじめと同様、０出ない値を持つレコードまで進み、同様の手順を繰り返す。

手順７：「再拡張汎用ソートファイルの作成」

上記で得られた「拡張原文番地ファイル」を再度、「汎用ソートファイル」レコード順にソートし、「再拡張汎用ソートファイル」を作成する。

手順８：「抽出文字列集計処理」

「再拡張汎用ソートファイル」を参照して、以下の手順で、抽出する文字列を決定し、同時に、その出現回数を求める。

①採否表示が「○」となっているレコードを調べ、抽出文字数の最大のレコードを探す。そのレコードの一致文字数をｎとする。

②抽出文字数の値がｎの最初のレコードの番号をi_1とすると、i_1からから始まって、後方のレコードを見たとき、抽出文字数が連　続してｎとなるレコードの数が、そのｎ字の文字列の出現回数であるから、文字列と出現回数（但し、出現回数２以上）を集計表に記録する。次に現れた、抽出文字数がｎのレコードi_2に対して、前と同様にして文字列と出現回数を集計表に記録する。

③ｎ＝ｎ－１とし、②の手順を繰り返す。ｎ＝０になったら処理は終了する。

［原文データ］： むかし むかしの おかしなおかし。 おかしのはなしは おかしな おはなし。

（抽出対象箇所）｛ ⑤ ⑤ ① ③ ② ⑥ ② ③ ④ ① ④

○印：抽出される害の文字列

原文番地ファイル

原文番地	文字列単語［ソートなし］（先頭部分）
1	むかしむかし
2	かしむかしの
3	しむかしのお
4	むかしのおか
5	かしのおかし
6	しのおかしな
7	のおかしなお
8	おかしなおか
9	かしなおかし
10	しなおかし。
11	なおかし。お
12	おかし。おか
13	かし。おかし
14	し。おかしの
15	。おかしのは
16	おかしのはな
17	かしのはなし
18	しのはなしは
19	のはなしはお
20	はなしはおか
21	なしはおかし
22	しはおかしな
23	はおかしなお
24	おかしなおは
25	かしなおはな
26	しなおはなし
27	なおはなし。
28	おはなし。
29	はなし。
30	なし。
31	し。
32	。

［手順2］拡張汎用ソートファイルの作成 ⇒

［手順3］→一致文字数のカウント

［手順4］→抽出文字数の記入

拡張汎用ソートファイル

抽出文字数	一致文字数	レコード番号	原文番地	文字列単語［ソートあり］（先頭部分）
5	5	1	8	おかしなおか
5	3	2	24	おかしなおは
3	3	3	16	おかしのはな
3	1	4	12	おかし。おか
1	0	5	28	おはなし。
4	4	6	9	かしなおかし
4	2	7	25	かしなおはな
3	3	8	5	かしのおかし
3	2	9	17	かしのはなし
2	2	10	2	かしむかしの
2	0	11	13	かし。おかし
3	3	12	10	しなおかし。
3	1	13	26	しなおはなし
2	2	14	6	しのおかしな
2	1	15	18	しのはなしは
1	1	16	22	しはおかしな
1	1	17	3	しむかしのお
1	1	18	31	し。
1	0	19	14	し。おかしの
2	2	20	11	なおかし。お
2	1	21	27	なおはなし。
2	2	22	21	なしはおかし
2	0	23	30	なし。
1	1	24	7	のおかしなお
1	0	25	19	のはなしはお
1	1	26	23	はおかしなお
3	3	27	20	はなしはおか
3	0	28	29	はなし。
3	3	29	4	むかしのおか
3	0	30	1	むかしむかし
0	0	31	32	。
0	0	32	15	。おかしのは

［手順5］拡張原文番地ファイルの作成 ⇒

［手順6］→有効無効判定処理

拡張原文番地ファイル

採否表示	抽出文字数	一致文字数	レコード番号	原文番地	文字列単語
○	3	0	30	1	むかし
×	2	2	10	2	かしむ
×	1	1	17	3	しむか
○	3	3	29	4	むかし
○	3	3	8	5	かしの
×	2	2	14	6	しのお
×	1	1	24	7	のおか
○	5	5	1	8	おかし
×	4	4	6	9	かしな
×	3	3	12	10	しなお
×	2	2	20	11	なおか
○	3	1	4	12	おかし
×	2	0	11	13	かし。
×	1	0	19	14	し。お
×	0	0	32	15	。おか
○	3	3	3	16	おかし
○	3	2	9	17	かしの
×	2	1	15	18	しのは
×	1	0	25	19	のはな
○	3	3	27	20	はなし
×	2	2	22	21	なしは
×	1	1	16	22	しはお
○	1	1	26	23	はおか
○	5	3	2	24	おかし
×	4	2	7	25	かしな
×	3	1	13	26	しなお
×	2	1	21	27	なおは
×	1	0	5	28	おはな
○	3	0	28	29	はなし
×	2	0	23	30	なし。
×	1	1	18	31	し。
×	0	0	31	32	。

［手順7］再拡張汎用ソートファイルの作成 ⇒

［手順8］→抽出文字列集計処理

再拡張汎用ソートファイル

採否表示	抽出文字数	一致文字数	レコード番号	原文番地	文字列単語
○	5	5	1	8	おかし
○	5	3	2	24	おかし
○	3	3	3	16	おかし
○	3	1	4	12	おかし
×	1	0	5	28	おはな
×	4	4	6	9	かしな
×	4	2	7	25	かしな
○	3	3	8	5	かしの
○	3	2	9	17	かしの
×	2	2	10	2	かしむ
×	2	0	11	13	かし。
×	3	3	12	10	しなお
×	3	1	13	26	しなお
×	2	2	14	6	しのお
×	2	1	15	18	しのは
×	1	1	16	22	しはお
×	1	1	17	3	しむか
×	1	1	18	31	し。
×	1	0	19	14	し。お
×	2	2	20	11	なおか
×	2	1	21	27	なおは
×	2	2	22	21	なしは
×	2	0	23	30	なし。
×	1	1	24	7	のおか
×	1	0	25	19	のはな
○	1	1	26	23	はおか
○	3	3	27	20	はなし
○	3	0	28	29	はなし
○	3	3	29	4	むかし
○	3	0	30	1	むかし
×	0	0	31	32	。
×	0	0	32	15	。おか

最終結果 ⇒

［原文番地］ 自レコードが、原文の先頭から何番目の文字から始まる単語文字列に対応するかを表示。

［レコード番号］ 汎用ソートファイルのレコードの番号

［一致文字数］ 直後のレコードの文字列と比べて、先頭から一致する文字数

［抽出文字数］ 自分のレコードで抽出対象となる先頭からの文字数

［採否表示］ 部分文字列として抽出するか否かの判定結果

抽出された文字列

方法	本論文の方法		従来の方法（参考）	
gram数	文字列	出現回数	文字列	出現回数
5 gram	①おかしなお	2	おかしなお	2
4 gram	ーーーー	ー	おかしな かしなお	2 2
3 gram	②おかし ③かしの ーーー ④はなし ⑤むかし	2 2 ー 2 2	おかし かしな かしの しなお はなし むかし	4 2 2 2 2 2
2 gram	ーー ーー ーー ーー ーー ーー ーー	ー ー ー ー ー ー ー	おか かし しな しの なお なし はな むか	4 6 2 2 2 2 2 2
1 gram	ー ー ー ー ー ー	ー ー ー ー ー ー	お か し な の は む	5 6 8 4 2 3 2
ーーー	合　計	10	合　計	72

（２回以上出現した文字列で、部分的に相互重複のない文字列）

図２　連続型共起表現抽出アルゴリズム実施例

（２）例題検討

以上のアルゴリズムの適用例を図２に示す。この例では、従来のアルゴリズムで抽出される文字列の種類が24種類で、延べ抽出文字列数が72件であったのに対して、本論文の方法では、６種類、12件に絞られる。

４．共起表現の抽出実験

日経産業新聞記事３月分（892万字）を対象に、連鎖型共起文字列の抽出実験を行った。その結果を、表１～４に示す。使用した計算機は、XEROX ARGOSS 5270(SUN OS4.1.3)で、使用したメモリ量は、最大４８ＭＢである。

表１　抽出された文字列の種類と頻度(その１)

結果比較 / 抽出対象	本論文の方法 文字列種類数	本論文の方法 延べ出現回数	長尾・森の方法 文字列の種類数	長尾・森の方法 延べ出現回数
2文字以上	970,203	2.613,704	4,374,141	31,178,897
5文字以上	591,901	1,476,922	2,960,487	10,808,458
10文字以上	52,214	114,270	673,601	1,550,817
20文字以上	1,792	3,692	177,298	359,810

表２　抽出された文字列の種類と頻度(その２)

結果比較 / 抽出対象	本論文の方法 文字列種類数	本論文の方法 延べ出現回数	長尾・森の方法 文字列の種類数	長尾・森の方法 延べ出現回数
2回以上	970,203	2,613,704	4,377,087	39,588,291
5回以上	67,321	551,441	882,217	31,288,701
10回以上	12,351	217,934	372,291	28,050,199
20回以上	2,288	92,804	169,375	25,871,964
50回以上	285	37,850	62,991	22,209,875
100回以上	76	24,167	30,316	19,961,961
200回以上	20	16,771	14,363	17,759,432

表３　抽出された文字列の例（１０gramの例）

本論文の方法	長尾・森の方法
することになっている(44)	したところによると、(273)
第二次臨時行政調査会(35)	らかにしたところによ(223)
することになりそうだ(19)	明らかにしたところに(223)
８２ジャパンショップ(17)	かにしたところによる(222)
しているのではないか(16)	にしたところによると(222)
ワシントン一九日共同(14)	第二次臨時行政調査会(208)
サウジアラビア通過庁(14)	ることを明らかにした(191)
として注目されている(14)	日明らかにしたところ(173)
ニューヨーク二十五日(13)	たことを明らかにした(120)
を余儀なくされている(13)	ホテルニュージャパン(112)
［合計 21,155種 延べ 47,336回］	［合計 132,865種類 延べ 345,232回］

表４　出現頻度の高い文字列の例

高頻度文字列 200回以上	という(586), と述べた(512), としている(436), また(325), である(324), 写真(315), しかし、(302), と語った(283), 東京(281), 価格(278), 欧州共同体(277), しかし(274), ポイント(269), ひとこと(264), 発売期間(259), また、(236), これは(220), このため(204), ただ、(201)

これらの表から、以下のことが分かる。

①従来の方法に比べて、本論文の方法では、抽出される文字列の種類、出現回数共に大幅に抑制される。

例えば、２文字以上、２回以上の文字列では、抽出される種類が５分の１、延べ出現回数も約１０分の１に抑制される。この効果は、文字数の大きい文字列ほど大きく、２０文字以上の場合では，１００分の１にもなる。

②本論文の方法で抽出された文字列は、文法的、意味的に表現の単位になるものが多く、断片的な文字列の抽出は大幅に抑制される。

処理時間では、最初の文字列単語のソート（汎用ソートファイルの作成）に、最も多くの時間（数十時間）がかかったが、その後の処理は、それに比べて短時間（約１～２時間）であった。

５．あとがき

言語ｺｰﾊﾟｽなどの膨大な言語ﾃﾞｰﾀから、使用頻度の高い表現を自動的に、長い順かつ頻度の高い順に、漏れなく発見し、集計する方法を提案した。

約８９０万字の新聞記事データに適用した例によれば、従来の方法に比べて、断片的な文字列の抽出が抑制され、抽出される文字列の種類は２０～１%に、またそれらの述べ出現回数は１４～１%程度に削減されることが分かった。

このように、文法的、意味的な表現の単位としての連鎖型共起表現が得られるようになったため、これを応用すれば、離散型共起表現も自動的に抽出することが可能となった。それについては、別稿で述べる。

なお、本論文では、日本語文字列への適用例を示したが、単語列や形態素解析結果への適用も同様である。

参考文献

(1)Church,K.W. and Hanks,P.(1992): Word Association norms, Mutual Information and Lexicography, *Computational Linguistics*, Vol.16, No.1, pp.22−29

(2)Smadja,F.A. and MeKeown,K.R. (1992): Automatically Extracting and Representing Collocations for Language Generation, *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, pp.252−259

(3)北, 小倉, 森元, 矢野 (1993) : 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

(4)Kita,K., Kato,Y., Omoto,T. and Yano,Y.(1994): A Comparative Study of Autoatic Extraction of Collocations from Corpora: Mutual Information vs. Cost Criteria, *Jornal of Natural Language Processing*, Vol.1, No.1, pp.21−33

(5)Smadja,F.(1993): Retrieving Collocations from Text: Xtract, *Computational Linguistics*, Vol.19, No.9, pp.143−177

(6)Nagao,M and Mori,S (1994): A New Method of N−gram Statistics for Large Number of n and Automatic Extraction of Words and Phrases from Large Text Data of Japanese, *The Proceedings of the 15th International Conference on Computational Linguistics*, pp.611−615