TR-IT-0122

相互情報量を用いた単語の分類の高速化

妹尾 正身　　隅田 英一郎　　飯田 仁

1995.8

概要

コーパスから知識を抽出する研究の一つに、相互情報量を使った単語の意味分類がある。従来の実現法では、大規模コーパスを対象とした場合の処理時間が問題であった。本稿では、並列化による高速化を提案し、新聞記事半年分に対する実験結果と大規模化した場合の予測について延べる。

目次

D 新聞記事半年分での分類結果 15

1 はじめに

コーパスから知識を抽出する研究の一つに、共起関係による単語の分類がある。単語の分類は、単語の共起頻度をクラスの共起頻度で近似することによる sparse data problem を解決することや、シソーラスの自動作成を目的として研究されている [1]。

単語の分類方法として、分類基準に客観的な尺度である相互情報量を使う方法があるが、従来の方法では、大規模コーパスを対象とした場合の処理時間に問題があった。本稿では、並列化による高速化を提案し、新聞記事半年分に対する実行結果を報告する。

2 分類の方法

本稿では、代表的な分類法である Brown et al. [2] の方法を、検討対象とする。まず、この方法について説明する。

2.1 相互情報量による融合対象決定

Brown et al. では、コーパス中の bigram を用いて 階層的分類を行なう。 全ての単語に対して各々一つのクラスを割り当て、そのうちの 2 つのクラスの融合を繰返して分類していく。

手順を以下に示す。コーパス中に現れる単語の集合を W、単語の個数を V とする。$V-k$ 回 $(0<k\leq V)$ の融合が行なわれた後のコーパス中に現れるクラスの集合を C_k とする。その要素の数は k である。前回の融合で C_{k+1} に属するクラス $C_{k+1}(1), C_{k+1}(2), \cdots, C_{k+1}(k+1)$ のうち、$C_{k+1}(i)$ と $C_{k+1}(j)$ とが融合されたとする $(i<j)$。$C_{k+1}(i)$, $C_{k+1}(j)$ を融合したクラスを $C_{k+1}(i+j)$ と表記する。

C_k に属するクラス $C_k(l)$ を、$l=i$ の時 $C_k(i)=C_{k+1}(i+j)$、$l=j$ の時 $C_k(j)=C_{k+1}(k+1)$、$1\leq l\leq k$ で $l\neq i,j$ の時 $C_k(l)=C_{k+1}(l)$ と定義する。

bigram の先行語が $C_k(l)$ に属し後続語が $C_k(m)$ に属する確率を $p_k(l,m)$ として、$pl_k(l)=\sum_m p_k(l,m)$,$pr_k(m)=\sum_l p_k(l,m)$, $q_k(l,m)=p_k(l,m)\log\frac{p_k(l,m)}{pl_k(l)pr_k(m)}$ とする。C_k における先行クラスと後続クラスの相互情報量 I_k は以下の式で表せる。

$$I_k=\sum_{x,y} q_k(x,y) \tag{1}$$

$C_k(l), C_k(m)$ を融合した後の相互情報量 $I_k(l,m)$ は

$$\begin{aligned} I_k(l,m) &= \sum_{x,y\neq l,m} q_k(x,y)+\sum_{y\neq l,m} q_k(l+m,y) \\ &\quad +\sum_{x\neq l,m} q_k(x,l+m)+q_k(l+m,l+m) \end{aligned} \tag{2}$$

となる。$I_k(l,m)$ が最大の $C_k(l), C_k(m)$ を融合する。以上を、全ての単語が唯一のクラスにまとまるまで繰返す。

確率はコーパスから $p_k(l,m) = \frac{b_k(l,m)}{C_b}, pl_k(l) = \frac{o_k(l)}{C_W}, pr_k(m) = \frac{o_k(m)}{C_W}$ として計算する。但し、$b_k(l,m)$ は先行クラスが $C_k(l)$ で後続クラスが $C_k(m)$ である bigram のコーパス中での出現頻度であり、C_b はコーパス中の bigram の延べ個数である。同様に、$o_k(l)$ は $C_k(l)$ のコーパス中での出現頻度であり、C_W はコーパス中の延べ語数である。

上記の確率の計算にかかる時間を考慮しないと、C_1 を求めるのに要する計算量は、$q_k(l,m)$ の計算回数で考えて、V^5 のオーダーである。

2.2 相互情報量減少分のみの計算による融合対象決定

C_k において $I_k(l,m)$ が最大の l,m を求める計算は、$I_k(l,m)$ の I_k からの 減少分が最も小さい l,m を求める計算で代用できる。$L_k(l,m) = I_k - I_k(l,m)$ とすると、(1)(2) より、

$$\begin{aligned} L_k(l,m) &= s_k(l) + s_k(m) - q_k(l,m) - q_k(m,l) - q_k(l+m,l+m) \\ &\quad - \sum_{x \neq l,m} q_k(x,l+m) - \sum_{y \neq l,m} q_k(l+m,y)。 \end{aligned} \tag{3}$$

ただし、$s_k(l) = \sum_x q_k(x,l) + \sum_y q_k(l,y) - q_k(l,l)$ 。$I_k(l,m)$ の計算量は V^2 のオーダーであったが、$L_k(l,m)$ では V に減る。

2.3 前回融合時の計算結果の再利用

$L_k(l,m)$ の計算は、前回 $C_{k+1}(l), C_{k+1}(m)$ が融合されていない場合は $L_{k+1}(l,m)$ との差分のみで計算できる。$C_k(1), C_k(2), \cdots, C_k(k)$ のうち、suffix だけでなく要素が前回と変化しているのは、$C_{k+1}(i+j) = C_k(i), C_{k+1}(k+1) = C_k(j)$ のみであるから、これらを先行クラスまたは後続クラスとする $q_{k+1}(l,m)$, $q_k(l,m)$ のみが前回との変分となる。これらのみを計算すれば $L_{k+1}\ (l,m)$ から $L_k(l,m)$ が求まる。

$l,m = i$ の時は 第2.2節の通り計算する。$l,m \neq i,j$ の時、

$$\begin{aligned} L_k(l,m) &= L_{k+1}(l,m) - q_{k+1}(l,i) - q_{k+1}(i,l) \\ &\quad -q_{k+1}(l,j) - q_{k+1}(j,l) + q_k(l,i) + q_k(i,l) \\ &\quad -q_{k+1}(m,i) - q_{k+1}(i,m) \\ &\quad -q_{k+1}(m,j) - q_{k+1}(j,m) + q_k(m,i) + q_k(i,m) \\ &\quad +q_{k+1}(l+m,i) + q_{k+1}(i,l+m) + q_{k+1}(l+m,j) \\ &\quad +q_{k+1}(j,l+m) - q_k(l+m,i) - q_k(i,l+m)。 \end{aligned} \tag{4}$$

$m = j$ の時、$L_k(l,j)$ は、(4) の右辺の m を $k+1$ に置き換えた式となる。$l = j$ の時、$L_k(j,m) = L_k(m,j)$。C_1 を求めるのに要する計算量は V^3 のオーダーとなる。

2.4 漸進的処理

第2.3節までと異なり第2.1節の解との一致は保証されなくなるが、ここまでの手順を一定の K 個の高頻度クラスに対して行ない、低頻度語を漸進的に融合対象に追加しつつ融合を進める手順で代用し高速化することができる。この手順を、以後漸進的処理と呼ぶ。

変更した手順を示す。$K < k \leq V$ の時、C_k に属するクラスを高頻度順に $C_k(1)$, $C_k(2), \cdots, C_k(K), \cdots, C_k(k)$ とする。相互情報量の計算を K 番目までの高頻度クラス同士に限定し、(1) を

$$I_k = \sum_{1 \leq x,y \leq K} q_k(x,y)$$

とする。同様に変更した $I_k(l,m)$ が最大の l,m を $1 \leq l < m \leq K$ の範囲で求め、融合させる。$C_k(K+1)$ を高頻度クラスに繰り入れる。以上を繰り返す。

$I_k(l,m)$,$L_k(l,m)$ は、$\sum$ が最大 K までの計算になり、省略する q_k は (単語の頻度が小さいことから) 値の小さなもののみで、高頻度クラス同士の q_k は計算することによって、漸進的処理を行なう以前の値に対する $I_k(l,m)$ の誤差の増大を押えている。

C_1 を求めるのに要する計算量は K^2V のオーダーなので、Brown et al. [2] では K として、$K \ll V$ の値を取って、計算量を大幅に減らしている。

3 分類の高速化

前節で述べた Brown et al. の分類法 (第2.4節の漸進的処理を行なう場合) について負荷分散を考慮して並列化した。

3.1 融合対象決定処理の並列化

第2.3,2.4節の計算法において、ある C_k における l,m に対する相互情報量減少分 $L_k(l,m)$ の計算に必要な情報は $C_k(1), \cdots, C_k(K)$ どうしの bigram の頻度と、$L_{k+1}(l,m)$ であり、他の l,m に対する $L_k(l,m)$ の計算と独立に計算できる。このことから、各プロセッサに異なる l,m に対する $L_k(l,m)$ の計算を分担させ、全ての $L_k(l,m)$ の計算が終るのを待ち合わせて最小値を取る組合せの l,m を選ぶ形に並列化した。

$L_k(l,m)$ は $l,m \neq i,j$ の場合と $l,m = i,j$ の場合とでは第2.3節から計算量は定数オーダー (q_k の計算回数は 24 回) か、K のオーダー (q_k の計算回数は $6K-7$ 回) か異なるが、それ以外は l,m に関わらず $L_k(l,m)$ の計算量は同じと考えて、各プロセッサの計算時間が平均するように、各プロセッサで計算する $L_k(l,m)$ の数を $l,m \neq i,j$ の場合と $l,m = i,j$ の場合を各々均一になるよう各プロセッサに割り当てた。

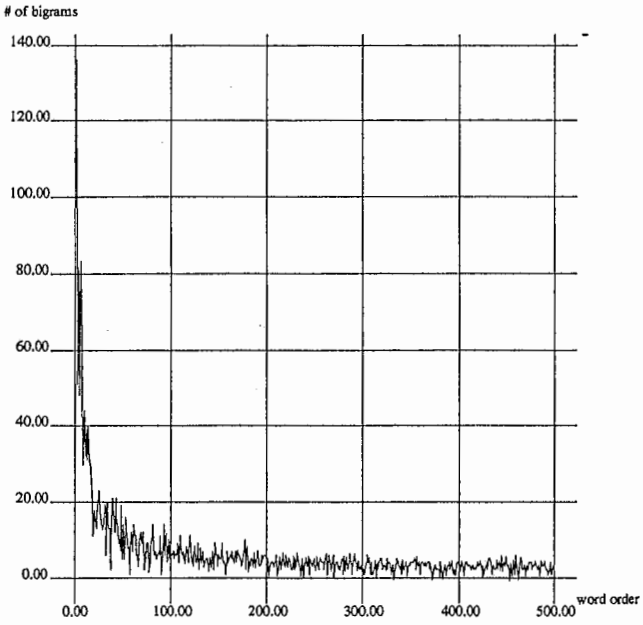


図 1: 高頻度 500 語間での先行語毎の bigram の数

3.2 bigram の分布を考慮した並列化

各 $L_k(l,m)$ の計算量を考えると、bigram の分布の偏りが問題となる。図 1に、the AP Newswire の英文プレーンテキストデータ[1] (LDC) の一部である延べ語数 =17,047・異なり語数 V=4,230 の英語 (表記) に対しての、$K = 500$ での偏りの様子を示す。また、この時の、高頻度 500 語間での可能な組合せに対する実際に存在する bigram の密度は 1.33% である。

分布の偏りのため、bigram が存在して計算が必要な $q_k(l,m)$ の数が $L_k(l,m)$ によって異なり、$L_k(l,m)$ の計算量が l,m によって異なる。$K = 500$、70 プロセッサの時の、上記データでのプロセッサ毎の q_k の偏りを第 3.1 節の並列化について表 1の 3.1 の欄に示す。

$r(k)$ を、C_k における先行クラス・後続クラスともに $C_k(1),\cdots,C_k(K)$ の組の中での実際に存在する bigram の密度とすると、

$$C_k\text{における融合の処理時間} = K^2 r(k)(2 - r(k))$$

となる。各プロセッサでの処理時間を平均化するため、bigram を多く持つクラスの各プロセッサへの割当を分散した。上記条件での、プロセッサ毎の bigram が存在する q_k の偏りが第 3.1 節の並列化に比べてどれだけ平均化されたか表 1の 3.2 の欄に示す。

[1] /DB/LTDB3/TIPSTER/vol_{1,2,3}.r/ap/ にあるファイルから TEXT フィールドのみを切り出したもの

表 1: プロセッサ毎の q_k の偏り

	q_k の計算回数比 (最大値 / 平均値)	
	bigram が存在しない場合を含む	bigram が存在しない場合を含まない
3.1	1.15	8.10
3.2	1.14	2.35

4 実装

以上で述べた方法による分類を実現するために作成したプログラムの各モジュールについて説明する。プログラムは KSR1 [3] 上で C によって実現した。

第 2 節で考慮しなかった、p_k, pl_k, pr_k の計算についても述べる。

4.1 単語集計

データ読み込み時に、単語切りと同時に各単語、単語 bigram の出現頻度を、集計しておく。ここで p_k, pl_k, pr_k のうち p_V, pl_V, pr_V が求まる。この処理の計算量は述べ語数 C_W のオーダーである。

4.2 ソート

漸進的処理のため、単語を出現頻度の大きい方からソートする。この処理の計算量は V のオーダーである。

4.3 初期設定

高頻度の K 語について、クラス、クラス bigram の出現頻度の計算などの初期設定を行なう。また、単語集計では、先行語のみから単語 bigram へのポインタをはっていたが、後続語から先行語へのポインタをここで設定する。この処理の計算量は、後続語から先行語へのポインタの設定で決まり、単語 bigram の異なり数のオーダーである。

4.4 分類

第 4.1 ～ 4.3 節の後、Brown et al. の方法による分類を行なう。融合を行なう度に第 4.5 節を行なってから次の融合を行なう。この処理の計算量は、第 3.2 節で述べたことから、bigram の密度と q_k の偏りが一定ならば K^2V のオーダーである。

4.5 クラス集計

各クラス、クラス bigram の出現頻度は、融合を行なう度に融合したクラス及び繰り入れられるクラスと他のクラスとの bigram の集計を行なう。ここで $p_{k+1}, pl_{k+1}, pr_{k+1}$

表 2: プロセッサ毎の分類の処理時間の偏り

	分類の処理時間比 (最大値 / 平均値)	分類の処理時間 (秒)	高速化率
3.1	1.649	1072.921	38.3 倍
3.2	1.228	851.698	48.2 倍

から p_k, pl_k, pr_k が求まる。

この処理の計算量は、融合 1 回に対し、繰り入れられる単語の異なり bigram 数のオーダーであり、全体では、全単語 bigram の異なり数のオーダーである。

5 実験結果

以上のプログラムでの実験結果を示す。

5.1 高速化率

$K = 500$、70 プロセッサの時の、第 3.2 節で用いたデータでの測定結果を第 3.1 節と第 3.2 節の並列化についてモジュール毎の処理時間を表 2に示す。第 3.2 節の方がプロセッサ毎の分類の処理時間の偏りが小さく、分類の処理時間が短くなっている。

第 3.2 節の並列化について、上記と同じデータを用いて、プロセッサ数を変化させて $K = 500$ での測定結果を表 3と図 2に示す。分類に要する時間は、1 プロセッサ時の約 41100 秒 (約 11 時間 25 分) から、70 プロセッサで約 850 秒 (約 14 分) へ、約 48 倍の高速化を達成している。全体としてみても、1 プロセッサ時の約 41150 秒 (約 11 時間 26 分) から、70 プロセッサで約 900 秒 (約 15 分) へ、約 46 倍の高速化を達成している。

5.2 大規模コーパスへの適用

新聞記事として、the AP Newswire の英文プレーンテキストデータ (LDC) のうち 1990 年 1 月 1 日〜1990 年 6 月 30 日分を使用した。述べ 21,807,580 語・異なり 198,290 語の規模である。表 4にモジュール毎の処理時間を示す。第 5.1 節の高速化率から考えると、これは逐次では推定 56 日間かかる。

結果の概要の一部を、付録 D に示す。品詞による分類、意味による階層的分類など興味深い分類が構成されている。

新聞記事 3 年分の場合の見通しを以下に示す。規模は延べ 1 億 3000 万語・異なり 53 万語である (付録 C)。

単語集計の処理時間は実験済みで 62 時間である。bigram の密度と q_k の偏りがが変わらないならば分類の計算量は異なり語数 V に比例するので、半年分の処理時間か

表 3: プロセッサ数毎の処理時間

プロセッサ数	処理時間 (秒)					
	単語集計	ソート	初期設定	分類	クラス集計	全体
1	2.914	0.399	7.301	41097.694	22.417	41145.718
	(0.007%)	(0.001%)	(0.02%)	(99.9%)	(0.05%)	(100%)
2	2.914	0.399	7.324	23788.393	24.304	23838.837
5	2.914	0.399	7.560	9700.642	14.500	9740.169
10	2.914	0.399	7.465	4871.341	12.133	4911.101
20	2.914	0.399	7.622	2483.132	12.716	2521.926
30	2.914	0.399	7.518	1886.068	14.924	1926.119
40	2.914	0.399	7.569	1304.970	15.414	1347.409
50	2.914	0.399	7.584	1072.874	17.156	1116.460
60	2.914	0.399	7.549	935.065	17.234	980.152
70	2.914	0.399	7.774	851.698	18.689	897.821

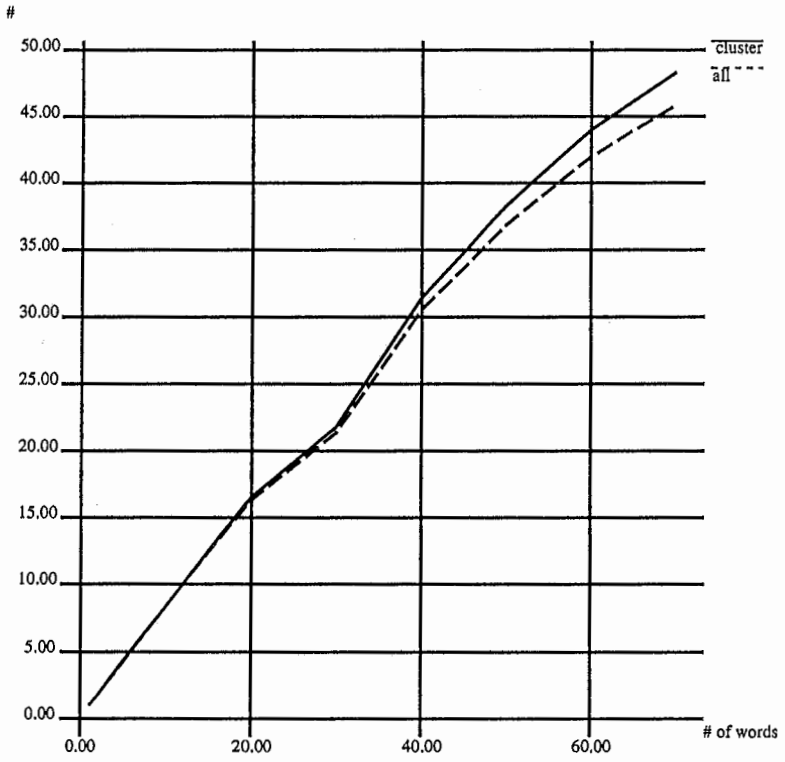


図 2: プロセッサ数毎の高速化率

表 4: 新聞記事半年分の処理時間

モジュール	処理時間 (秒)
単語集計	32,500.671
ソート	576.459
初期設定	8,130.527
分類	111,085.592
クラス集計	6,465.182
全体	152,291.431

ら考えて 3 年分の分類の処理時間は 75 時間程度と推定される。合わせて 6 日間で出来る見通しである。

6 おわりに

Brown et al. の単語の分類を対象として相互情報量の計算の並列化による高速化を行なった。bigram の分布を考慮して負荷を分散する方法を用いた。

その結果、延べ約 2 万語・異なり 4000 語のコーパスにおいて 70 プロセッサで 48 倍の高速化を達成した。

大規模コーパスを対象とした場合、延べ 2000 万語・異なり 20 万語の新聞記事半年分の処理を 42 時間で行なった。高速化率から考えると、逐次では推定 56 日間かかる。この分類結果には、品詞による分類、意味による階層的分類など興味深い分類が構成されている。

また、延べ 1 億 3000 万語・異なり 53 万語の新聞記事 3 年分の処理は、半年分の処理時間から考えて 6 日間で出来る見通しである。

本稿の並列化では相互情報量の計算対象が隣接語の bigram の場合だけでなく、Predicate-Argument などの bigram [4] でも適用できる。

参考文献

[1] 永田：“自然言語処理と学習理論,” 言語処理学会第 1 回年次大会チュートリアル資料, pp1 – 20 (1995).

[2] P. F. Brown, V. J. D. Pietra, P. V. deSouza, J. C. Lai, R. L. Mercer : “Class-based n-gram Models of Natural Language, ” *Computational Linguistics* 18, no. 4, pp467–479 (1992).

[3] Kendall Square Research Corporation : Technical Manuals, MA, USA (1994).

[4] D. Hindle : “Noun Classification from Predicate-Argument Structures”, Proc. of the 28th meeting of ACL (1990).

[5] 柏岡, E. W. Black：“相互情報量を用いた単語の分類手法,” 電子情報通信学会「自然言語における学習」シンポジウム論文集, pp104–111 (1994).

[6] 妹尾, 隅田, 飯田：“相互情報量を用いた単語の分類の高速化,” 言語処理学会第1回年次大会発表論文集, pp213 – 216 (1995).

A　プログラムの構成

A.1　外部仕様

当プログラムは、input、clusterという2つの実行ファイルからなる。全体に対する入力であるデータファイルをinputで処理してbigramファイルを出力し、bigramファイルをclusterで処理して分類結果を得る。

A.1.1　input

起動形式 input <データファイル> [<データファイル> *] < bigramファイル>

機能 単語集計

ソート

A.1.2　cluster

起動形式 cluster <並列プロセッサ数> <低頻度語切捨閾値> < bigramファイル>

<低頻度語切捨閾値>は、本稿では常に0である。

機能 初期設定

分類

クラス集計

分類結果ファイル出力(ファイル名はfile.outに固定)

A.2 ファイル構成

A.2.1 input

```
-rw-r--r--  1 senoo        348 May 23 13:41 Makefile
-rw-r--r--  1 senoo      19526 Aug 30 19:36 datain.c
-rw-r--r--  1 senoo        618 Aug 30 19:27 dataout.c
-rw-r--r--  1 senoo        880 Aug 30 19:36 main.c
-rw-r--r--  1 senoo       1032 Feb  9  1995 malloc1.c
-rw-r--r--  1 senoo        512 May 22 21:04 struct.h
```

コンパイル方法は、

```
%make "CFLAGS = -DTIMER -qdiv -O2"。
```

A.2.2 cluster

```
-rw-r--r--  1 senoo        502 Jun 30 13:40 Makefile
-rw-r--r--  1 senoo      16588 Aug 30 15:52 calc.c
-rw-r--r--  1 senoo      15452 Aug 30 16:58 datain.c
-rw-r--r--  1 senoo       4887 Aug 30 12:37 dataout.c
-rw-r--r--  1 senoo       2870 Aug 30 11:51 lib_class.c
-rw-r--r--  1 senoo      12707 Aug 31 12:59 main.c
-rw-r--r--  1 senoo       1114 Aug 30 12:32 malloc1.c
-rw-r--r--  1 senoo      24708 Aug 31 15:12 p_worker.c
-rw-r--r--  1 senoo       2226 Aug 30 16:45 struct.h
```

コンパイル方法は、

```
%cc -para -DTIMER -DBG -qdiv -c dataout.c -o dataout.o
```

の後

```
%make "CFLAGS = -DTIMER -DBG -qdiv -O2"。
```

B　入出力の形式

B.1　データファイル

1 行が 256 字以内の ASCII-code からなるテキストファイル

B.2　bigram ファイル

1 行で bigram 1 個を表す。1 行の構成は以下の通り。

＜先行語表記＞＜後続語表記＞＜*bigram*出現回数＞

先行語毎にまとめて表し、先行語の最後の bigram の行は、先行語出現回数も出力される。

＜先行語表記＞＜後続語表記＞＜*bigram*出現回数＞＜先行語出現回数＞

B.3　分類結果ファイル

1 行で 1 単語を表す。1 行の構成は以下の通り。

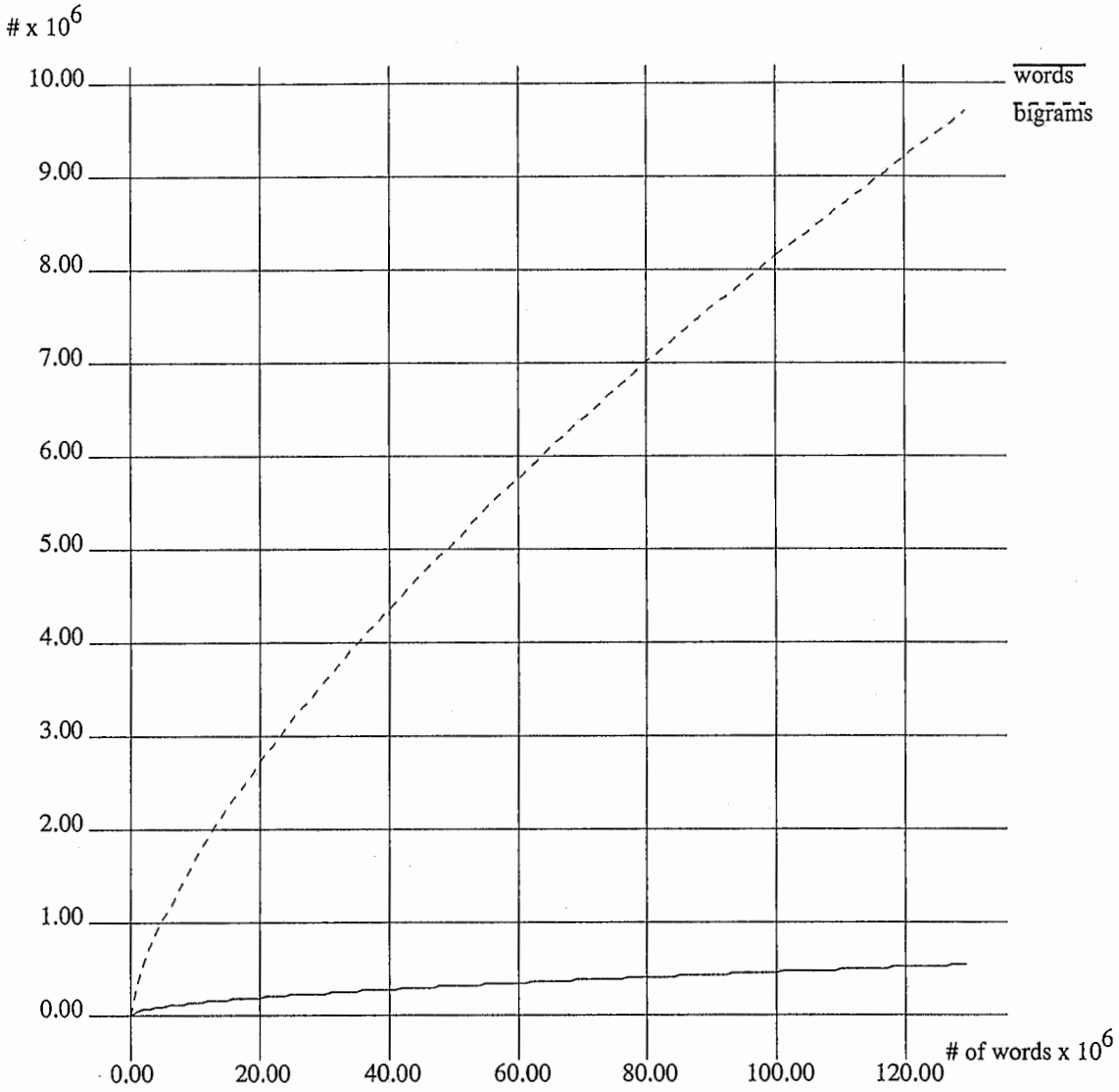
＜*id*1＞＜*id*2＞＜*id*3＞＜単語表記＞＜単語出現回数＞

id1 〜3 は、各々 10、100、K クラスに分類された時、その単語が属するクラスの id である。id が同じ単語はまとめて出力される。

C　新聞記事における延べ語数と異なり語数、異なり bigram 数の関係

新聞記事 3 年分における、延べ語数に対する異なり語数、異なり bigram 数の増え方を、以下の図に示す。

第 3 節で記述した the AP Newswire (LDC) の 1988 年 2 月 12 日～ 1990 年 12 月 31 日を使用した。



D　新聞記事半年分での分類結果

最終結果の500クラスの分類の一部を示す。全ての分類は

/dept3/work6/clustering/result/file.out.ap900a

に置いてある。

500クラスに分類された時のクラス毎に概要を以下に示す。id1、id2、id3は、各々10、100、500クラスに分類された時の、各クラスに対するidである。例えば、以下では、id3が2～11のクラスが100クラスに分類された時にはid2が2であるクラスに共通して属している。「延べ」はそのクラスに属する全ての単語の延べ語数である。「異なり」はそのクラスに属する全ての単語の異なり語数である。「語例」はそのクラスに属する単語から選んだ単語である。頻度が10より大きく、頻度の大きい順に最大10語選んだ。括弧内に頻度を併記した。

id3が2～11のクラスには副詞、13～14には単数のbe動詞、15～16には複数のbe動詞、35にはsay(言及する)に近い動詞、など興味深い分類が構成されていることがわかる。また、id3が280～281のクラスでは、280には曜日、281には最近の日を表す単語が分類されており、さらにこれらのクラスがまとまってid2が47という日を表すクラスになっており、階層的分類が構成されていることがわかる。

id1	id2	id3	異り	延べ	語例
0	0	0	24	76756	be(76729)
0	1	1	32	41813	been(41768)
0	2	2	75	21818	already(4509) often(2902) apparently(2234) allegedly(1366) generally(1314) usually(1245) sometimes(1177) reportedly(1161) finally(942) normally(638)
0	2	3	38	11563	never(5549) ever(2861) always(2388) personally(478) adversely(71) categorically(45) steadfastly(38) gotta(21) respectfully(18) gladly(15)
0	2	4	35	13599	just(11092) simply(1392) merely(396) necessarily(328) barely(300) scarcely(28)
0	2	5	50	21502	still(8853) really(3244) probably(2705) certainly(1113) actually(1054) clearly(919) obviously(388) basically(355) definitely(324) automatically(295)
0	2	6	626	13943	then(8040) eventually(1149) otherwise(562) thus(530) therefore(387) formerly(378) ultimately(351) somehow(222) please(209) presumably(141)
0	2	7	181	14199	now(12146) currently(966) indeed(350) nonetheless(84) nevertheless(79) presently(71) Maronites(44) regrettably(28) sadly(26) ironically(23)
0	2	8	131	30316	also(29881) subsequently(215) additionally(16)

id1	id2	id3	異り	延べ	語例
0	2	9	157	17566	immediately(2391) partly(2131) sharply(1451) slightly(1410) seriously(934) fully(833) completely(640) significantly(515) somewhat(495) badly(430)
0	2	10	299	30439	once(3635) recently(3170) previously(1261) twice(1145) repeatedly(988) strongly(861) originally(799) publicly(775) initially(655) formally(602)
0	2	11	139	13607	being(11024) widely(1081) fatally(249) forcibly(238) federally(212) constitutionally(118) popularly(74) genetically(72) loosely(60) freshly(45)
0	3	12	39	60341	not(60281)
0	4	13	45	138973	was(138917)
0	4	14	53	119814	is(119700) IS(13)
0	5	15	28	67220	were(67184)
0	5	16	44	64051	are(64001)
0	6	17	78	66161	has(66072)
0	6	18	20	47835	had(47808)
0	6	19	14	71672	have(71644)
0	7	20	152	30048	force(4775) change(4432) protest(2428) limit(1389) mark(1332) influence(1235) block(1086) delay(971) guard(935) combat(900)
0	7	21	88	30907	show(6437) offer(2812) play(2091) test(2064) claim(1898) fund(1601) launch(1484) answer(1168) estimate(1068) award(1000)
0	7	22	212	11416	comment(3257) ban(2080) crackdown(1102) focus(1022) elaborate(661) bail(529) monopoly(464) depend(312) rely(291) concentrate(279)
0	7	23	221	15432	work(9249) act(1728) experience(1213) eye(675) sleep(441) function(251) encounter(159) crest(140) engagement(126) sights(67)
0	7	24	148	32075	call(3709) run(3409) fight(2533) turn(2288) stand(1993) break(1797) drive(1389) push(989) watch(880) check(858)
0	7	25	122	19256	move(4342) visit(3779) step(2225) challenge(1251) venture(766) march(753) shift(673) link(512) blow(447) roll(420)
0	7	26	52	9746	close(5032) return(4184) foster(252) tow(51) poke(28) pep(25) subordinate(21) non-members(20) 12-(11) clutch(11)
0	7	27	95	20870	open(3627) lead(2681) present(1847) travel(1558) address(1489) spread(1366) complete(1336) finance(1320) slow(1042) correct(519)
0	7	28	115	14011	end(7702) start(2765) equivalent(585) edge(578) wake(469) overthrow(407) shore(373) array(197) outskirts(158) outline(140)

id1	id2	id3	異り	延べ	語例
0	7	29	103	28609	use(7527) face(3417) cause(2553) form(2363) sign(2043) review(1799) conduct(1091) purchase(1017) balance(755) speed(725)
0	7	30	161	10747	look(2991) talk(2242) looked(1098) speak(1031) talked(754) looks(723) worry(541) sounds(292) sounded(254) pray(190)
0	7	31	985	18682	live(3412) remain(3136) stay(2176) die(940) testify(715) participate(669) compete(584) cooperate(479) arrive(471) proceed(388)
0	7	32	113	15255	come(5783) appear(1443) happened(1281) happen(1154) grow(769) occur(603) sit(536) resign(530) happens(506) exist(494)
0	7	33	84	22135	go(7467) continue(3985) try(2895) agree(1321) fly(846) respond(712) apply(656) contribute(467) fail(452) refuse(424)
0	8	34	57	12971	do(12793) dare(76) reckon(12)
0	8	35	97	17958	say(13137) indicate(654) suggest(539) argue(484) contend(478) predict(359) admit(343) acknowledge(265) complain(264) insist(242)
0	8	36	49	18925	think(9415) believe(4299) feel(2580) am(1924) guess(440) suppose(75) Shrunk(43) resent(39) deplore(35)
0	8	37	116	11513	know(6682) understand(1248) remember(675) exactly(618) realize(567) forget(326) specify(295) wonder(248) imagine(219) anticipate(166)
0	8	38	41	11626	like(10050) mean(1510)
0	8	39	86	17099	hope(2899) thought(2792) felt(1720) knew(1652) love(1373) learned(1024) knows(811) wish(640) feared(617) loved(405)
0	8	40	58	5804	need(4553) yield(750) regard(343) Klugt(23) vie(21) uneasily(18)
0	8	41	60	13256	want(7955) expect(1703) seem(983) afford(542) tend(452) intend(405) prefer(307) deserve(162) owe(142) Wanna(121)
0	9	42	125	4698	become(4376) undergone(152) herald(16)
0	9	43	165	28023	take(9017) seek(2104) receive(1745) require(1596) spend(1379) lose(1185) affect(941) represent(723) add(663) contain(526)
0	9	44	166	42863	get(10153) find(3470) buy(2942) see(6396) tell(2000) determine(1484) ask(1476) decide(1440) carry(1344) hear(1180)
0	9	45	1024	105419	give(5399) keep(4119) make(10607) provide(3256) allow(2895) bring(2422) reduce(2259) discuss(2084) consider(2081) protect(2008)

(中略)

id1	id2	id3	異り	延べ	語例
3	46	279	55	8505	ago(7845) 2000(390) 2010(52) 2005(32) 2003(30) 2020(18) 2030(15)
3	47	280	241	85962	Sunday(9126) Saturday(6662) Thursday(14220) Wednesday(13340) Tuesday(14124) Friday(13718) Monday(13933) weekends(116) NBC-TV(82) horseback(58)
3	47	281	269	15889	today(14163) tonight(378) afterward(316) yesterday(244) beforehand(69) afterwards(57) autographs(32) Abroad(24) Brescia(15) Petrobras(15)
3	48	282	202	56927	week(14050) month(8351) year(30329) decade(1842) century(1548) minute(407) Crusade(75) juncture(38) quarter-century(37) WT(12)
3	49	283	250	41623	sales(6297) costs(3016) income(2614) earnings(2334) assets(1898) profits(1762) orders(1605) losses(1547) supplies(1491) gains(1387)

(中略)

id1	id2	id3	異り	延べ	語例
7	79	425	212	11537	Europe(7020) Airlines(1200) Ireland(1029) Salvador(868) Paso(167) Everest(104) Tiempo(93) Espectador(50) Faso(44) Chorrillo(40)
7	79	426	248	26430	America(4884) Panama(3458) Lebanon(2414) Mexico(2395) England(1781) Beirut(1657) Lake(1426) Kansas(1274) Jerusalem(1244) Oklahoma(1131)
7	79	427	1770	65034	Washington(8134) London(3838) Tokyo(2551) Earth(1451) Vietnam(1405) Beijing(1366) Paris(1266) Houston(1200) Manhattan(962) Hollywood(754)
7	79	428	1727	104815	Texas(4866) California(4186) 1990(3456) Chicago(3181) Florida(2930) Boston(2129) Miami(2111) Ohio(1957) Virginia(1805) Columbia(1789)
7	79	429	1332	91927	Japan(5592) China(4124) Israel(5902) Lithuania(3985) Moscow(4490) Britain(2831) Poland(2698) Iran(2555) France(2516) Nicaragua(2165)
7	79	430	660	35114	Congress(8939) Parliament(2361) NATO(2200) parliament(1628) God(1213) directors(1144) Federated(846) Life(816) Solidarity(660) Hezbollah(447)

(中略)

id1	id2	id3	異り	延べ	語例
8	97	497	118	46433	;(46273) es(15)
8	98	498	156	1079396	,(1079126) o(65)
9	99	499	264	1060065	.(1059459) mart(142) Vanilli(41) TO(18) PROGRAMMING(12)