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
ATR 音声翻訳通信研究所

1 はじめに

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分類がある。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。本稿では高速化の手法として 1) 低頻度語の処理、2) 並列化を検討する。1) は分類のための処理の一部を低頻度語に対して省略することであり誤差を伴うが、2) は誤差を伴わない。両者を合わせて検討しその結果を報告する。

2 分類の方法

本稿では、Brown et al. [1] 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 $\mathcal{C}_k$ とする。その要素の数は k である。前回の融合で $\mathcal{C}_{k+1}$ 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。

$\mathcal{C}_k$ 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。$\mathcal{C}_k$ 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。

$$I_k=\sum_{l,m} q_k(l,m) \tag{1}$$

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

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

となる。$I_k(i,j)$ が最大の $C_k(i), C_k(j)$ を融合する。以上を、全ての単語が属する唯一のクラスにまとまるまで繰返す。この定義をそのまま計算すると、$\mathcal{C}_1$ を求めるのに要する計算量は $\mathbf{V^5}$ のオーダーである。

確率はコーパスから $p_k(l,m)=\frac{b_k(l,m)}{C_b}, pl(l)=\frac{o_k(l)}{C_W}$, $pr(m)=\frac{o_k(m)}{C_W}$ として計算する。但し、$b_k(l,m)$ は先行語が $C_k(l)$ で後続語が $C_k(m)$ 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。

2.2 相互情報量減少分のみの計算による融合対象決定

$\mathcal{C}_k$ において $I_k(i,j)$ が最大の i,j を求める計算は、$I_k(i,j)$ の I_k からの減少分が最も小さい i,j を求める計算で代用できる。$L_k(i,j)=I_k-I_k(i,j)$ とすると、(1)(2) より、

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

ただし、$s_k(i)=\sum_l q_k(l,i)+\sum_m q_k(i,m)-q_k(i,i)$。$I_k(i,j)$ の計算量は V^2 のオーダーであったが、$L_k(i,j)$ 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、$L_k(j,m)=L_k(m,j)$。$\mathcal{C}_1$ を求めるのに要する計算量は $\mathbf{V^3}$ のオーダーとなる。

2.4 低頻度語の処理

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、低頻度語の融合を後回しにした手順で代用し高速化することができる。この手順を、以下で低頻度語の処理と呼ぶ。

変更した手順を示す。$K<k\leq V$ の時、$\mathcal{C}_k$ 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

$$I_k=\sum_{1\leq l,m\leq K} q_k(l,m)$$

とする。同様に変更した $I_k(i,j)$ が最大の i,j を $1\leq i<j\leq K$ 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。

$I_k(i,j)$,$L_k(i,j)$ 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、低頻度語の処理を行なう以前の値に対する $I_k(i,j)$ の誤差の増大を押えている。

$\mathcal{C}_1$ を求めるのに要する計算量は $\mathbf{K^2V}$ のオーダーなので、Brown et al. 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 低頻度語の処理の誤差と処理時間の検討

ここでは、低頻度語の処理による分類の、低頻度語の処理を行なう以前の分類に対する誤差を検討する。又、ノイズをさけるために低頻度の単語や bigram を切り捨てることが多いが、Brown et al. では切り捨てていない。その効果についてノイズ以外の点も含め、合わせて検討する。

3.1 誤差の評価

二つの W に属する単語の分類 $\mathcal{C}_n$ を、相互情報量の大きいと類似しているとする以下の方法で比較する [2]。A を比較元での $\mathcal{C}_n$、C を比較先での $\mathcal{C}_n$ とする。相互情報量 $I(C;A)=H(A)-H(A|C)$ で $H(A)$ は比較先に関係なく一定なので、$H(A|C)$ を使う。$H(A|C)$ が大きいほど A と C に共通する情報は少なく、誤差が大きいとみなす。

$$H(A|C)=-\sum_{c\in C}P(c)\sum_{a\in A}P(a|c)\log P(a|c)$$

を、$x\in X$ として、
$P(x)=\frac{x\text{に属する単語の延べ語数}}{X\text{に属するクラスに属する単語の延べ語数}}$ として計算する。

K と低頻度語の切捨ての二つを変更条件とする。低頻度語の切捨て頻度順を L とし、頻度順が高い方から L 番目以降の単語を切り捨てるとする。

ATR 対話データベース (ADD)[3] の電話会話の一部である延べ語数 =102,753・異なり語数 V=3,298 の英語 (表記) に対して、条件を変えて分類を行なった結果の $\mathcal{C}_{100}$ を比較した。$K=$ 3298/816/521/317/227/176/100, $L=$ 3298/816/521/317 の組合せを試した。頻度順が 3298/816/ 521/317/227/176/100 番目以降の単語は、各々、頻度が 0/8/16/32/48/64/144 以下の単語である。$K\geq L$ は、低頻度語の処理を行なわない場合に相当する。$K>L$ の時は K を L と等しくなるまで小さくしても、切り捨てる単語は出ず $H(A|C)$ は変わらないので省略した。

$K=L=816$ での結果を比較元 A とした (表 1)。$\mathcal{C}_n$ に含まれる単語についてだけを対象に比較するため、$H(A|C)$ は頻度順が 317(頻度 33 以上) までの単語に対して計算した。

まず、低頻度語の切捨てを行なわない $L=3298$ の場合を表 1で見ると、低頻度語の処理を行なう / 行なわないにかかわらず $L=521$ や $L=317$ の時より誤差が大きく、低頻度語の切捨てを行なうことによって安定した分類が得られることがわかる。

次に、$L=521$および317 で $K=L$ の場合は $H(A|C)$ は 0.11 程度で各々 $L=521$および317 で $K>L$ よりも、処理時間 (表 2) は大きく、$H(A|C)$ は小さい。これから、低頻度語の処理 (または、その低頻度語の切捨てとの組合せ) より低頻度語の切捨てのみの方が、同程度の処理時間の高速化に対して誤差が小さいという意味で良い結果を出している。

低頻度語の処理の問題点は、計算量を減らすために相互情報量の計算対象を高頻度語に絞った際に、融合する

表 1: $K = L = 816$ である A に対する $H(A|C)$

$K \setminus L$	3,298	816	521	317
3,298	0.126213	-	-	-
816	0.148292	0.000000	-	-
521	0.148292	0.130525	0.110547	-
317	0.147220	0.121829	0.119012	0.108007
227	0.141410	0.124021	0.098600	0.113765
176	0.154372	0.138586	0.137355	0.124629
100	0.164495	0.164495	0.164495	0.163860

表 2: 処理時間 (秒)

$K \setminus L$	3,298	816	521	317
3,298	61,077	-	-	-
816	3,596	330	-	-
521	1,331	234	110	-
317	642	141	82	43
227	418	98	62	38
176	309	76	51	33
100	186	50	36	32

対象も高頻度語に絞ったために、融合する順が低頻度語の処理をしない場合 (ほぼ低頻度語から融合していく) と異なる点にある。

4 並列化

低頻度語の処理を行なう場合について並列化した。

4.1 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 $L_k(l, m)$ の計算を分担させる形に並列化した。

$L_k(l, m)$ 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か、K のオーダーか異なるが、それ以外は l, m に関わらず同じ計算量と考えて、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。

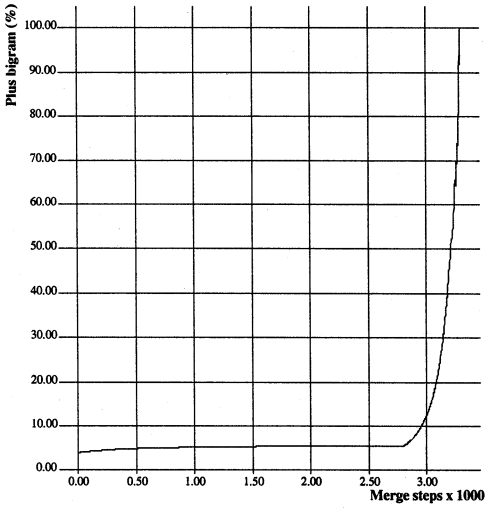

図 1: $K = 500, L = 3298$ の時の各 C_k における頻度が正の bigram の割合

4.2 頻度が正の bigram の分布を考慮した並列化

$L_k(l, m)$ の計算量を考えると、実際のデータでは、出現頻度 $b_k(l, m)$ が正の bigram は sparse であり、分布が偏っている。図1,2に、第3.1節と同じデータを用いて、$K = 500, L = 3298$ での様子を示す。そのため、bigram の頻度 $b_k(l, m)$ が正で計算が必要な $q_k(l, m)$ 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。

q_k の数で計算量を計ってみる。$r(k)$ 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 bigram 中での出現頻度が正の bigram の密度とすると、q_k の計算は主に $q_k(l+m, *), q_k(*, l+m)$ を計算する時に行なわれるので、

$$\text{融合1回の処理時間} = K^2 r(k)(2 - r(k))$$

となる。各プロセッサでの処理時間を平均化するため、出現頻度が正の 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。

$K = 500, L = 3298$,70 プロセッサの時のプロセッサ毎の p_k の不均衡を第 4.1 節と第 4.2 節の並列化について以下に示す。

	p_k の呼び出し回数比 (最大値 / 平均値)		全体の処理時間 (秒)
	$b_k = 0$ を含む	$b_k = 0$ を含まない	
4.1	1.09	4.13	923.841
4.2	1.15	2.85	856.172

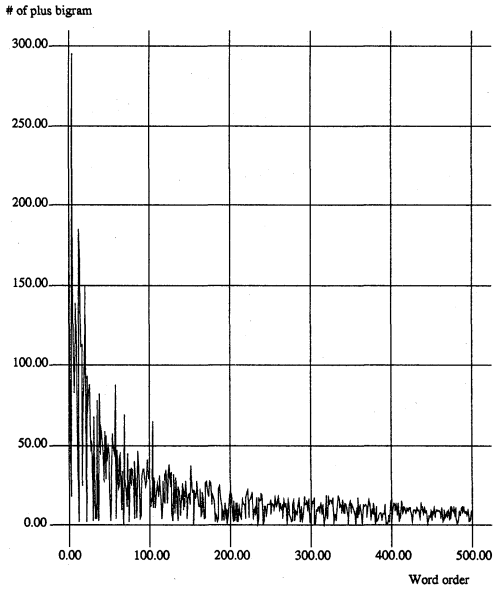

図 2: 高頻度 500 語間での先行語毎の正の bigram の数

第3.1節と同じデータを用いて、$K = 500, L = 3298$ での測定結果を図 3に示す。1 プロセッサ時の約 37000 秒 (約 10 時間 16 分) から、70 プロセッサで約 860 秒 (約 14 分) へ、約 43 倍の高速化を達成している。なお、プログラムは KSR1 [4] 上で C によって実現した。

5 おわりに

相互情報量を使った単語の分類に対して、低頻度語の処理 (または、それと低頻度語切捨てとの組合せ) より低頻度語切捨てのみの方が、ADD において同程度の高速化に対して誤差が小さいという意味で良い結果を出すことを示した。

また、並列化による高速化を bigram の分布を考慮して負荷を分散する方法で行ない、延べ約 10 万語のコーパスを用いて 70 プロセッサで 43 倍の高速化を達成した。

低頻度語に対する二つの処理と並列化は併用が可能であり、特に並列化は誤差が生じないため、他の処理の組み合わせやすい。

これらの手法は、相互情報量の計算対象が隣接語の bigram である Brown et al. の場合だけでなく、Predicate-Argument などの bigram [5] でも適用できる。

今後は、さらに大規模なコーパスへの適用を目指す。

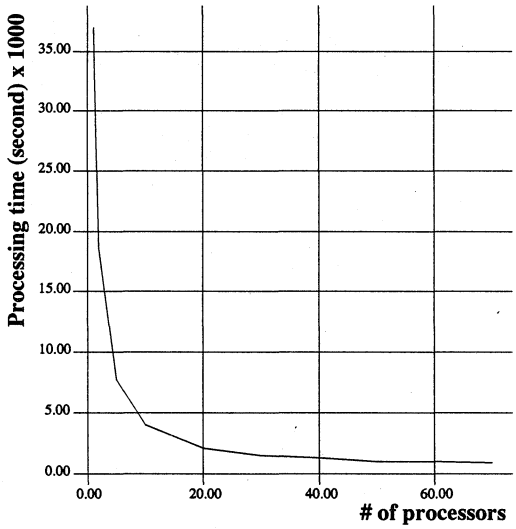

図 3: 並列化した時の処理時間

参考文献

[1] 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).

[2] M. Iwayama, T. Tokunaga : "Hierarchical Bayesian Clustering for Automatic Text Classification, " 人工知能学会研究会資料 SIG-J-9401-17, pp127–134 (1994).

[3] 江原, 小倉, 篠崎, 森元, 榑松 : "電話またはキーボードを介した対話に基づく対話データベース ADD の構築, " 情報処理学会論文誌, 33, no. 4, pp448–456 (1992).

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

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

[6] 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).