2次元クラスタリングに基づく高頻度語の検索手法

楠村　幸貴†　　水口　弘紀†　　久寿居　大†

† NEC 共通基盤ソフトウェア研究所　〒 630–0101 奈良県生駒市高山町 8916–47
E-mail: †kusumura@az.jp.nec.com, ††hironori@ab.jp.nec.com, †††kusui@ct.jp.nec.com

あらまし　近年，データ量の増大により，テキストマイニングの分野では処理の高速化が求められている．本論文では，テキストマイニング技術の中でも，最も基本的な機能の一つである高頻度語の検索に焦点を当てる．高頻度語の検索とは，ある概念についての検索クエリで得た文書集合内で，頻度の高い上位 K 個の単語を返す処理である．しかし，高頻度の検索を行うためには長い時間がかかるという問題がある．そこで本論文では，厳密解を出力するのではなく，短時間でより精度の高い近似解を計算することを目的とした，２次元クラスタリング処理を用いた検索手法を提案する．この手法は，単語と文書の関係を表現した行列に対して，予め２次元クラスタリング処理を加え，行列を密な領域と疎な領域に分割しておく．そして検索時に，より高密度な領域に絞った参照を行い，出力すべき高頻度語となりそうな単語から順に調べることで，高速に精度の良い結果を返す手法である．評価実験では，本手法は同程度の近似解を出す従来手法に比べ，約 9 倍高速であるという結果が得られた．

キーワード　連想検索, テキストマイニング, 高頻度語，共クラスタリング，NMF

Top K Term Processing based on Two-dimensional Clustering

Yukitaka KUSUMURA†, Hironori MIZUGUCHI†, and Dai KUSUI†

† NEC, Common Platform Software Research Laboratories
Takayama-cho 8916–47, Ikoma-shi, 631–0101 Japan
E-mail: †kusumura@az.jp.nec.com, ††hironori@ab.jp.nec.com, †††kusui@ct.jp.nec.com

Abstract　With the rapid growth of text data, speed of processing becomes more important in the text mining. This paper focuses on a simple function of text mining, which is to search high frequent words in a set of documents. High frequent word search takes long time, when the size of the documents is large. This paper proposes an approximate method of high frequent word search based on two-dimensional clustering. This method makes two-dimensional clusters for all words and all documents in advance, to identify dense and sparse regions in word-document matrix. Given a subset of the documents in searching, the method predicts the frequency of words for each region from the density information in the matrix, to avoid reading low frequent regions. As a experimental result, the proposed method shows nine times faster than the conventional approximate methods.

Key words　Semantic associative search, text mining, top k processing, co-clustering and NMF

1. はじめに

近年，データ量の増大により，テキストマイニングの分野では，システムの大規模化・高速化が重要な課題となっている．本論文では，テキストマイニング技術の中でも，最も基本的な機能の一つである高頻度語の検索処理に焦点を当てる．高頻度語の検索処理とは，ある文書集合内でその集合を特徴付ける高頻度な単語をK個返却する処理である．文書集合を特徴付ける単語を高速に取り出すことができれば，文書検索におけるキーワード拡張 [1] や，ある概念に対する理由や原因の分析を実現することができる．例えば,「ラーメン」という検索クエリによって得た文書集合内で高頻度な単語を取り出すことで,「とんこつ」や「つけ麺」などの関連語を得ることができ，ユーザに次の検索語として提供できる．また,「裁判員 AND 反対」という検索で得た文書集合に対して高頻度な語を取り出すことで，裁判員制度に反対する理由や意見を表現した単語を得ることができる．

高頻度語の検索を実現する最も単純な方法は，入力された文書番号のリストを元に，各文書をすべて読み込み含まれる単語の数をすべて数える方法である．しかし，各文書の読み込み処理はランダムディスクアクセスであるため，読み込む文書の数が多くなると処理に長い時間がかかる．さらに，従来の手法は

厳密解を出すことに執着し，長い時間がかかっても解を正確に出すことを目的としている．しかし，キーワード拡張や対話的な分析を実現するようなアプリケーションでは，短い時間で近似的な解を返すことも重要である．

そこで本論文では，厳密解を出すのではなく，短い時間でより精度の良い近似解を出力することを目的とする．本論文では，高頻度語の検索を高速かつ高精度に実現する方法として，２次元クラスタリングを用いた絞込み手法を提案する．これは単語と文書の間の関係を保持した単語文書マトリックスに対して，予め単語間と文書間の２つの方向に対してクラスタリングを行うことで，行列を密な領域と疎な領域に分割しておき，検索時に高密度な領域に絞った参照を行うことで，検索の高速化を図る方法である．さらに本論文では，その評価実験について報告する．評価実験によると，提案手法は同程度の近似解を出す従来手法に比べ約 9 倍高速に検索が可能であった．

本論文の構成は次の通りである．本論文ではまず，2. 節において，高頻度語の検索問題を定義する．次に，3. 節において，従来の関連研究について述べる．さらに，4. 節において，提案手法を説明する．そして，5. 節では，提案手法に対する評価実験の結果について報告し，最後にまとめを行う．

2. 問題定義

ある文書集合 D_{all} が与えられた場合，D_{all} 内に登場する単語の集合を W_{all} とする．そして，文書集合 D_{all} と単語集合 W_{all} の間の共起関係を表現した行列を単語文書マトリックス M として定義する．つまり，単語文書マトリックス M は，文書数 $|D_{all}|$ ×単語数 $|W_{all}|$ の大きさを持つ行列である．行列 M の各要素 m_{ij} は文書 $d_i(d_i \in D_{all})$ が単語 $w_j(w_j \in W_{all})$ を含むか否かを表すバイナリの値 (１：含む，０：含まない) とする．

次に，単語 w_j の文書集合 D 内の単語の文書頻度 $f(w_j, D)$ を式 1 によって算出される値と定義する．

$$f(w_j, D) = \sum_{k=0}^{|D|} m_{kj} \tag{1}$$

高頻度語の検索問題とは，$D \subset D_{all}$ となる D を表現する文書の識別子（文書番号）の列を入力とし，単語集合 W_{all} の中から，指定された D 内の頻度 $f(w_j, D)$ が高い上位 k 個の単語のリストを出力する問題である．

3. 関連研究

高頻度語の検索問題に対する最も単純な解法は，入力された文書番号のリストを元に，各文書をすべて読み込み含まれる単語の頻度をすべて数える方法である．しかし，文書集合の読み込み処理はランダムディスクアクセスであり，処理に長い時間がかかってしまう．また，近似解を出すために文書をサンプリングし，指定された文書集合のうち一部の文書集合だけを読み込むというアプローチも考えられるが．この方法は大きく精度が下がるという問題がある．

そこで，吉田ら [2] は文書を元に参照するのではなく，単語側から参照するというアプローチを取っている．この手法では，行列 M 内の各単語に対する列を転置インデックス [3] として保持し，キーとなる単語を全文書集合に対する頻度 $f(w_j, D_{all})$ でソートしておく．検索時には，高頻度な単語から順に，転置インデックス内の文書集合と入力文書集合 D とを比較することで，$f(w_j, D)$ を算出していく．そして，ある時点までで得た頻度第 k 位の単語 w_x の頻度 $f(w_x, D)$ が，次に読む単語 w_y の頻度 $f(w_y, D_{all})$ の頻度より大きくなった時点で終了する．この参照は毎回同じ順序で行われるため，M 内の高頻度な単語に対する転置インデックスをキャッシュとしてメモリ上に保持できる点と，読み込み処理をシーケンシャルディスクアクセスで実現できる点から，高速に処理を行うことができる．しかし，この手法は，全文書集合 D_{all} 内で高頻度な単語は，入力文書集合 D 内でも高頻度である，という仮定に基づく方法である．よって，D 内の単語分布と D_{all} 内の単語分布が大きく異なる場合に，単語とその登場位置を大量に読み込む必要があり大きな時間がかかるという問題がある．

また，高野ら [4] は，単語と文書で構成される単語文書マトリックスを転置インデックスの形式で圧縮し，メモリ上に保持する方法を提案している．これにより，すべての計算をメモリ上で行うことができるため，高速に関連語を返却することができる．しかし，このアプローチはメモリを大量に消費するため，データ量の制約があり大規模なデータを扱うためには大規模な分散計算環境が必要となる，という問題がある．

また，分散環境で上位 k 個の解を算出する問題は，”top k query processing”として知られており，多くの研究 [5]〜[7] が行われている．これらは検索エンジンの検索結果のようにランキングを回答とする問い合わせに対して，上位 k 個さえ返せば良いという観点に基づき処理を高速化するアルゴリズムの研究である．本研究は，単一のサーバ内での計算の高速化を目指す研究である．よって本研究はこれらの研究を否定するものではなく，本研究とこれらの研究を統合させることで，より大規模なデータに対しより小さな計算環境でより高速な検索が可能となると考えている．

4. 提案手法

本節ではまず，4.1 節において，提案手法を説明するために必要な語を定義する．次に，4.2 節において，提案手法の概要を説明する．さらに，4.3 節において，提案手法が用いている２次元クラスタリング処理を詳細に説明し，4.4 節において，領域内の頻度の予測方法を説明する．

4.1 語の定義

本節では図 1 を用いて領域，静的領域頻度，動的領域頻度，領域内文書長を定義する．

領域とは，単語文書マトリックス M 内において，文書集合を g 個の集合に分割し，単語集合を h 個の集合に分割することで得られる，M の一部を意味する．なお，分割された g 個の文書集合それぞれを文書クラスタ ($DC[p]$ $\{1 < p \leqq g\}$)，分割された h 個の単語集合それぞれを単語クラスタ ($WC[q]$ $\{1 < q \leqq h\}$)

と呼ぶ．また，領域の集合を R とするとき，各領域を $R[p][q]$ $\{1 < p \leqq g, 1 < q \leqq h\}$ として表現する．図 1 では，単語集合を単語クラスタ $WC[1]$ と $WC[2]$ に，文書集合を文書クラスタ $DC[1]$ と $DC[2]$ に分割することで，単語文書マトリックス全体を $R[1][1]$，$R[1][2]$，$R[2][1]$，$R[2][2]$ の４つの領域に分割している．

静的領域頻度とは，単語と領域によって決まる値である．単語 w_j の領域 $R[p][q]$ に対する静的領域頻度 $fs(w_j, R[p][q])$ とは，領域 $R[p][q]$ 内での単語 w_j の頻度である．例えば，図 1 において，単語「殺人」に対する領域 $R[1][1]$ 内の静的領域頻度は４，領域 $R[2][1]$ 内の静的領域頻度は１となる．

動的領域頻度とは，単語と領域と，入力文書集合によって決まる値である．単語 w_j の領域 $R[p][q]$ に対する動的領域頻度 $fd(w_j, R[p][q])$ とは，ユーザに指定された文書集合 D における，領域 $R[p][q]$ 内での単語 w_j の頻度である．例えば，図 1 においてユーザから文書番号の列として「１，４，５，７」という文書集合が指定された場合，単語「殺人」に対する領域 $R[1][1]$ 内の動的領域頻度は３，領域 $R[2][1]$ 内の動的領域頻度は０となる．

また，領域内文書長とは，文書と領域によって決まる値である．文書 d_i の領域 $R[p][q]$ に対する領域内文書長 $dl(d_i, R[p][q])$ とは，文書 d_i が持つ単語のうち単語クラスタ WC[q] に含まれる単語の数である．例えば，図 1 において文書３番の $R[1][1]$ に対する領域内文書長は３である．

4.2 概　　要

図 2 に提案手法の概要を示す．提案手法による処理は，事前におこなうオフライン処理と，検索時に実行するオンライン処理から成る．

オフライン処理は次の手順で行われる．システムはまず，単語文書マトリックス M に対し，２次元クラスタリング処理を行い，M を単語方向と文書方向にそれぞれ分割する（１）．なお，この処理の詳細については 4.3 節で述べるためここでは割愛する．

次に，システムは，１で生成された各領域に対して，要約情報を作成する（２）．なお，図 2 では，表 (a) がこれにあたる．表 (a) は各領域内の静的領域頻度の最大値を示している．例えば，領域 $R[1][1]$ の静的領域頻度の最大値は２００である．これは，領域 $R[1][1]$ に含まれる単語は，領域内にたかだか２００回しか登場しないことを意味する．

オンライン処理は次の手順で行われる．ユーザから文書番号の列を入力されると，システムはまず，入力された文書番号それぞれを，各文書領域に割り振る（３）．どの文書がどの文書クラスタに入るかは予め１において計算済みであるため，ここではメモリ上の変換行列を参照するのみで割り当ての計算が行われる．

次にシステムは，各領域に対して動的領域頻度の最大値を予測する（４）．領域 $R[p][q]$ に対する動的領域頻度の最大値 $mfd(R[p][q])$ とは次の式 2 で定義される値である．

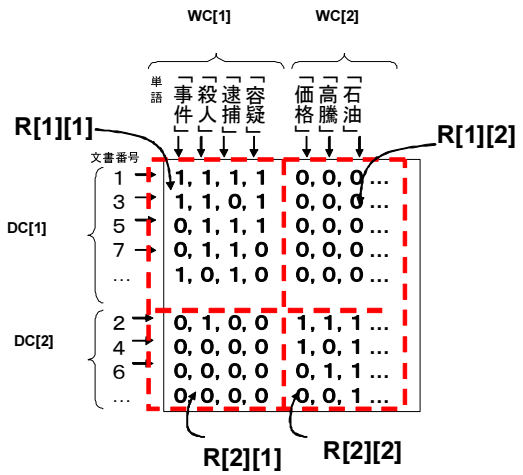

図 1　単語文書マトリックス内の領域
Fig. 1　Regions in a word-document matrix

$$mfd(R[p][q]) = max\{fd(w_j, R[p][q]), w_j \in WC[q]\} \quad (2)$$

領域に対する動的領域頻度の最大値がわかれば，領域内に出力すべき単語があるかどうかを事前に判定できる．しかし，真の $mfd(R[p][q])$ は領域内のベクトルをすべて読み込まなければ計算できないため，システムは $mfd(R[p][q])$ の予測値として $pfd(R[p][q])$ を算出する．$pfd(R[p][q])$ の詳細については 4.4 節で述べるためここでは割愛する．

さらに，システムは，各領域において算出された予測値 $pfd(R[p][q])$ を各単語クラスタごとに和算することで，各単語クラスタに対する動的領域頻度の最大値を予測する（５）．この予測値 $pfd(WC[q])$ は次の式 3 で定義できる．

$$pfd(WC[q]) = \sum_{s=1}^{g} pfd(R[s][q]) \quad (3)$$

例えば，図 2 では，単語クラスタ $WC[1]$ に対する予測値 $pfd(WC[1])$ を 5300 として算出している．

最後に，システムは，５において算出された予測値 $pfd(WC[q])$ が大きい単語クラスタを一つ選択し，単語クラスタ内の単語に対して読み込み処理を行う（６）．この読み込みに関しては吉田ら [2] の方法と同様である．つまり，単語クラスタ内の単語に対して，転置インデックスを参照し，真の $f(w, D)$ を算出する．ある単語クラスタに対する処理が完了したら，その時点での頻度第 k 位の単語 w_x の頻度 $f(w_x, D)$ と，未処理の単語クラスタの動的領域頻度の予測値 $pfd(WC[q])$ を比較し，$f(w_x, D) > pfd(WC[q])$ であれば，処理を停止する．そうでなければ次に $pfd(WC[q])$ が大きい未処理の単語クラスタを一つ選択し，同様の処理を続ける．

以上の処理により，システムは，指定された文書集合 D にとって関連の高い単語クラスタから順に読み込むことができる．

4.3　２次元クラスタリング

２次元クラスタリング処理とは，単語文書マトリックス M を利用し，文書集合を g 個の文書クラスタに分割するクラスタリングと，単語の集合を h 個の単語クラスタに分割するクラスタ

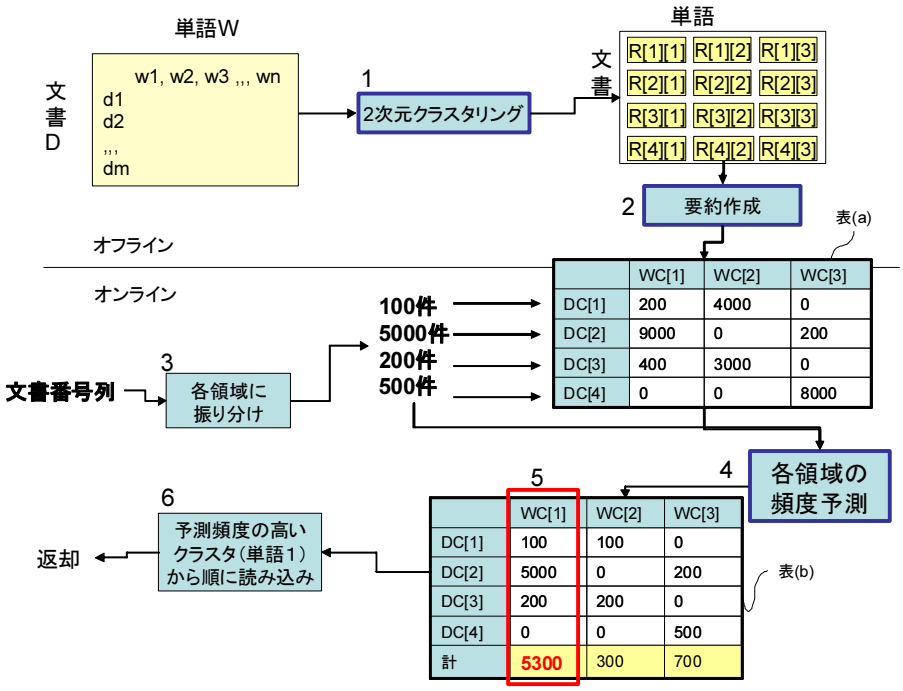

図 2　提案手法の処理手順

Fig. 2　The proposal method

リングとを同時に行う処理である．この処理により単語文書マトリックス M を領域に区切ると，$g \times h$ 個の領域に分割できる．

２次元クラスタリング処理を実現する代表的なアルゴリズムとしては，共クラスタリング [8] や NMF (non-negative matrix factorization) [9] [10]，PLSI [11] が知られている．

共クラスタリングは，単語クラスタ数 g と文書クラスタ数 h を入力とし，M 上の列が類似する単語同士を同じクラスタにまとめ，M 上の行が類似する文書同士を同じクラスタにまとめるアルゴリズムである．よって処理結果のクラスタを元に単語文書マトリックスを領域に分割すると，単語と文書の相関性が高く１が多く含まれる高密度な領域と，単語と文書の相関性が低く０ばかりが含まれる低密度な領域が生成される．

NMF と PLSI は単語と文書に共通する l 個の概念の因子があることを仮定し，単語，文書をそれぞれ l 次元のベクトル空間に射影するアルゴリズムである(注1)．これらはいずれも次元縮約のアルゴリズムではあるが，射影された l 次元を l 個のクラスタと捉えることで，２次元クラスタリング処理と見なすことができる．つまり，各単語と各文書それぞれに作成された l 次元のベクトルを各概念クラスタに対する重みとして捉え，最大の重みを持つ次元 (クラスタ) に割り当てることで，各単語/文書を１個のクラスタに分類できる(注2)．この結果に基づき，単語文書マトリックスを領域に分割すると，対角する位置にある領域 $R[p][p] \quad \{1 < p \leqq l\}$ に各次元の概念を表現した高密度な領域が作成され，それ以外に疎な領域が作成される．

これらの処理の結果をより直感的に説明するために図 3 に３つの３次元グラフを示す．これらは約 50 万記事の新聞データ(注3)を元に作成した単語文書マトリックス M を，30×30 の領域に分割し，各領域内で「１」となる要素の数を z 方向に示したものである．図 3 の (a) はランダムに 30×30 の領域に分割したものである．各領域に対して単語と文書の共起が均等に散らばっていることがわかる．図 3 の (b) は共クラスタリングの結果に基づき領域を作成したものである．共クラスタリングにより，疎な領域と密な領域がうまく分かれていることがわかる．図 3 の (c) は NMF に基づき領域を作成したものである．次元圧縮により対角成分に密な領域が作成されていることと，疎な領域と密な領域がうまく分かれていることが確認できる．また，(b) と (c) を比較すると，(b) は文書と単語の関係を対角化できていないものの，Z 方向により高い領域が存在する．これは，共クラスタリングが対角化という制約を負わないことにより，より高密度な領域を作成できることを意味する．

なお，共クラスタリングと NMF・PLSI の理論上の計算時間は，$O(g \times h \times |D_{all}| \times |W_{all}|)$ である．しかし，単語文書マトリックスは非常に疎であり，「0」の要素に関する計算を省くよう実装上の工夫を加えることで，$O(g \times h \times |D_{all}| \times |$ 平均文書長 $|) = O(g \times h \times$ 全文書長$)$ の計算時間で処理が可能である．本研究でもこのような実装を用いているが詳細は割愛する．

(注1)：NMF と PLSI は異なるアルゴリズムであるが，理論的には目的関数が同じであり同様の結果が得られることが知られている ([12])．

(注2)：次元縮約を行うアルゴリズムとしては他にも LSI [13] が知られている．しかし LSI で作成される l 次元の軸は意味空間上の単語・文書を良く分散させるために決定された相互に直交する軸であり，各軸が何を表すかは自明ではなく，単純にはクラスタリングアルゴリズムとして使用できない．これに対し NMF と PLSI が作成する軸は，意味空間上での単語と文書のクラスタの重心であり，各次元をクラスタとして扱うことができる．この解釈の詳細については論文 [10] を参照されたい．

(注3)：CD 毎日新聞データ集 [14] の 2003 年版から 2007 年版

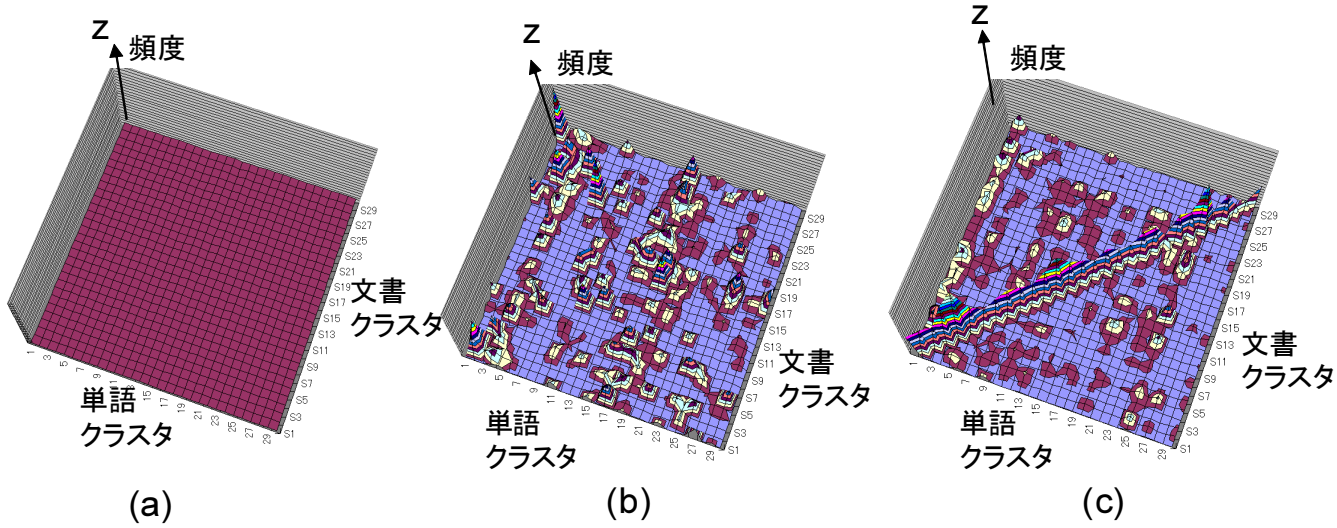

図 3　2次元クラスタリングの効果
Fig. 3　The effect of two-demention clustering

4.4　動的領域頻度の予測

我々は各領域に対する動的領域頻度の予測値 $pfd(R[p][q])$ として，予測値 $pfd_a(R[p][q])$，予測値 $pfd_b(R[p][q])$，予測値 $pfd_c(R[p][q])$ の3種類の予測値を考案した．以下，それぞれについて説明する．

予測値 $pfd_a(R[p][q])$ は式4によって算出される値である．

$$pfd_a(R[p][q]) = min\{|D \cap DC[p]|, mfs(R[p][q])\} \qquad (4)$$

なお，$|D \cap DC[p]|$ は，文書クラスタ $DC[p]$ に含まれ，かつ，入力文書集合 D に含まれる文書の数（領域 $R[p][q]$ において選ばれた文書の数）を表す． $mfs(R[p][q])$ は領域内の単語の静的領域頻度の最大値であり，式5により算出される値である．$mfs(R[p][q])$ は入力文書集合 D に無関係であるため，事前に算出しておくものとする．

$$mfs(R[p][q]) = max\{fs(w_j, R[p][q]), w_j \in WC[q]\} \qquad (5)$$

$pfd_a(R[p][q])$ は，選択された文書の数と領域内の静的領域頻度の最大値のうち小さい方の値である．これは図2の例で用いた値である．例えば，ある領域内で100個の文書が選択された場合，領域内の単語の動的領域頻度が100を超えることはない．また，領域内の単語の静的領域頻度の最大値が200である（領域内の単語は，領域内の文書に対してたかだか200回しか登場しない）場合，領域内の単語の動的領域頻度が200を超えることはない．$pfd_a(R[p][q])$ はこのような二つの制約に基づき，最大値を見積もった値である．このため，予測値 $pfd_a(R[p][q])$ と真の値 $mfd(R[p][q])$ の間には常に $pfd_a(R[p][q]) > mfd(R[p][q])$ の関係が成り立つ．

次に，予測値 $pfd_b(R[p][q])$ について述べる．予測値 $pfd_b(R[p][q])$ は式6によって算出される値である．

$$pfd_b(R[p][q]) = mfs(R[p][q]) \times \frac{|D \cap DC[p]|}{|DC[p]|} \qquad (6)$$

$pfd_b(R[p][q])$ は，領域内のすべての文書が選ばれた場合の領域内の頻度 ($mfs(R[p][q])$) に対し，領域内で選択された文書の割合を乗じた値である．つまり，入力された文書集合が，領域内の文書をランダムに選び出したことを仮定し，$mfd(R[p][q])$ を見積もっている．しかし，この仮定は常に正しくはないため，実際には予測値が真の値を下回る場合がある．

次に，予測値 $pfd_c(R[p][q])$ について述べる．予測値 $pfd_c(R[p][q])$ は式7によって算出される値である．

$$pfd_c(R[p][q]) = mfs(R[p][q]) \times \frac{\sum^{d_i \in DC[p] \cap D} dl(d_i, R[p][q])}{\sum^{d_i \in DC[p]} dl(d_i, R[p][q])} \qquad (7)$$

右側の分数の分母 $\sum^{d_i \in DC[p]} dl(d_i, R[p][q])$ は，領域内の文書の領域内文書長の合計であり，事前に算出される．$\sum^{d_i \in DC[p] \cap D} dl(d_i, R[p][q])$ は，選択された文書集合における領域内文書長の合計である．この値はオンラインで算出される．このために，システムは各文書の領域内文書長 $dl(d_i, R[p][q])$ を予め算出しメモリ上に保持しておく．そしてシステムは検索時に，選択された文書に対する領域内文書長の和を算出することで $\sum^{d_i \in DC[p] \cap D} dl(d_i, R[p][q])$ を算出する．

$pfd_c(R[p][q])$ は，領域内ですべての文書集合が選ばれた場合の領域内の頻度 ($mfs(R[p][q])$) と，領域内にある頻度の総和に対する選択された文書集合内の頻度の総和の割合を乗じた値である．$pfd_c(R[p][q])$ は，各文書の領域内文書長を用いることで，各文書の領域への貢献度を考慮した計算が可能であり，より正確に $mfd(R[p][q])$ を見積もることができる．しかし，その反面，他の予測値に比べ検索時の計算量が大きいという特徴がある．

5. 評　　価

我々は提案手法の有効性を検証・分析するために３つの実験を行った．一つ目の実験は，提案手法の有効性を確認するために従来手法との比較を行った実験である．二つ目の実験は，2 次元クラスタリング処理におけるアルゴリズムやクラスタ数が，検索性能にどう影響を及ぼすかを調査するために行った実験である．三つ目の実験は，4.4 節で提案した３つの予測値のうちどれが良いのかを調査するために行った実験である．

本節ではまず，5.1 節において，実験に用いる評価指標と実験条件について述べる．そして，5.2 節から 5.4 節において，３つの実験についてそれぞれ述べ，最後に 5.5 節において，実験結果のまとめと考察を行う．

5.1　評価指標と実験条件

提案する手法は高頻度になりそうな単語から順に真の頻度を調べることで，検索時間あたりの精度を向上させる技術である．そこで，ある時点での検索結果がどの程度正しいかを表す指標として，精度 $Accuracy(k)$ を用いる．精度 $Accuracy(k)$ とは，出力された上位 k 語の単語のうち，真の上位 k 語に含まれる単語の割合を意味する．

実験条件は次の通りである．文書集合 D_{all} は，CD 毎日新聞 [14] の 2003 年版から 2007 年版 (約 50 万記事 データサイズ 1.6GB) を使用した．単語集合 W_{all} は，D_{all} 内に含まれるすべての単語において，上位 100 位までの高頻度語を除いたもの (約 50000 語) とした．入力文書集合 D は次のように作成した．予め頻度が 3000 回〜１万回程度の単語 100 個をランダムに取り出し，それぞれの語が登場する文書集合を１セットとし，計 100 セットの文書集合を用意した．実験に用いた計算機は CPU Pentium4 3.6GHz，メモリ 1GB で構成される(注4)．また，検索システムは J2SE1.5 を用いて実装した．

5.2　従来手法との比較

本実験では，提案手法の有効性を確認するため，文書ベース，単語ベース，KMeans ベース，ICC ベース，NMF ベースという５つの手法の精度を比較した．

文書ベースとは，入力文書集合を順に読み込み単語の頻度を数える方法である．最も単純な従来手法として用いた．

単語ベースとは，吉田ら [2] が用いた方法であり，D_{all} に対し高頻度な単語から順に調べる方法である．なお，単語ベースでは，キャッシュの効果が重要であるため，$f(w_j, D_{all})$ の高い上位 1000 個の単語に対する転置インデックスをメモリ上に保持した．

KMeans ベースは，M を 30×30 の領域に分割する際に単語についての 1 次元クラスタリング処理のみを行う手法である．この手法では予め KMeans 法 [15] を用いて，単語集合を 30 個のクラスタに分割する．そして，文書についてはランダムに 30 個の領域に分類する，という処理を行った．以降の処理は提案手法と同様である．この手法は，クラスタリングを 2 次元方向に対して行う効果を計るために用意した．

ICC ベースと NMF ベースは提案手法である．ICC ベースは，共クラスタリングを用いて M を 30×30 の領域に分割し，検索に利用する提案手法である．NMF ベースは，NMF を用いて M を 30×30 の領域に分割し検索に利用する手法である．なお，これらの２次元クラスタリング処理にかかった時間はそれぞれ約 3 時間程度であった．

KMeans ベース，ICC ベース，NMF ベースの手法で算出する予測値 $pfd(R[p][q])$ としては，$pfd_c(R[p][q])$ を用いた．

また，単語ベースにおけるキャッシュの効果の影響が大きいため，キャッシュ効果の比較にならないよう，KMeans ベース，ICC ベース，NMF ベースにおいては，単語ベースと同様に $f(w_j, D_{all})$ の高い上位 1000 個の単語に対して転置インデックスをメモリ上に保持しておき，残りの単語集合に対して領域の分割を行っている．

実験手順は次の通りである．各手法の１回の検索において，検索開始 0.3 秒後，0.5 秒後，1 秒後，2 秒後，5 秒後，10 秒後の時点で検索を停止し，その時点までで読み込んだ文書/単語を元に検索結果を作成し，Accuracy(10) と Accuracy(100) を計測した．これを 100 セットの入力文書集合に対して行い，100 回の検索により得た 100 個の精度の平均値を算出した．

この実験結果を図 4 と図 5 に示す．文書ベースは，どちらの精度においても，最終的に精度は向上するものの，検索初期（2 秒まで）において精度が低い．また，単語ベースはキャッシュの効果により，最初から 90%程度の精度を確保できているものの，それ以降の精度の上昇が遅い．これは全文書集合と入力文書集合の間で単語分布が異なる場合に，一般的には低頻度な単語が上位 k 個に入る場合があるためである．

提案手法である ICC ベースと NMF ベースの精度は他の手法を上回っている．特に，ICC ベースの手法は，1 秒後の時点で，従来の単語ベースの手法に比べ，Accuracy(10) で 7%程度，Accuracy(100) で 5%の差がある．また，Accuracy(10) で精度 95%以上達成するために，従来の文書ベースの手法が 5 秒かかっていたのに対し，提案手法では 0.6 秒程度の時点で 95%に達している．このため，提案手法は従来手法に比べ近似解を出すために約 9 倍の高速化を実現したと言える．

また，KMeans ベースは，どちらの精度においても，従来手法よりも精度が高いものの，2 次元クラスタリング処理を用いた ICC ベースと NMF ベースよりも精度が低い．これにより，2 次元クラスタリング処理の効果が確認できる．

5.3　２次元クラスタリング処理の比較

本実験では，２次元クラスタリングにおけるクラスタ数を変えた場合の比較を行う．このため本実験では，5.2 節で用いた ICC ベースと NMF ベースの手法について，分割するクラスタ数を 10×10，30×30，50×50 に変化させ，それぞれの場合に対して，5.2 節の実験と同様に Accuracy(100) を計測した．

この結果を図 6 に示す．なお，図 6 には，5.2 節で計測した KMeans ベースの精度も比較のために示している．

この図から，クラスタ数が大きければ大きいほど精度が高いことが読み取れる．また，ICC ベースの 10×10 の場合は，

(注4)：ただし，提案手法はメモリを使用しないアプローチであるため，実験では最小限のメモリ (200MB 程度) しか使用していない

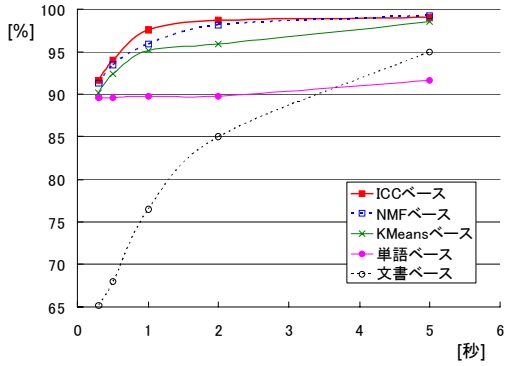

図 4　Accuracy(10) による従来手法との比較
Fig. 4　Accuracy(10)

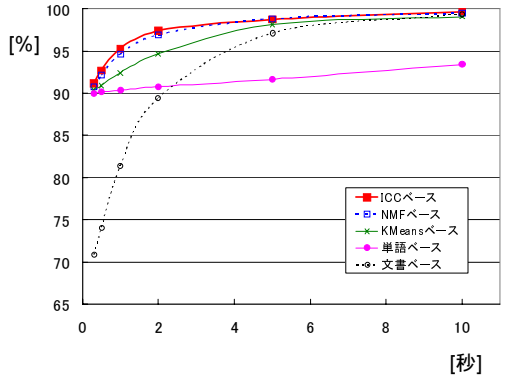

図 5　Accuracy(100) による従来手法との比較
Fig. 5　Accuracy(100)

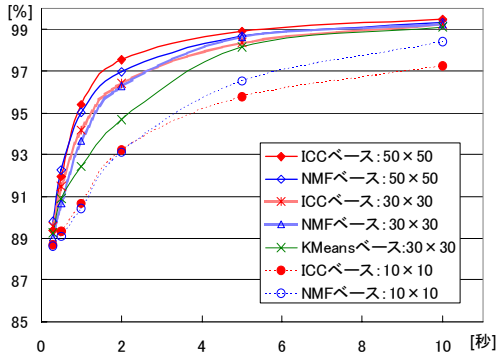

図 6　クラスタ数の影響
Fig. 6　The effect of number of clusters

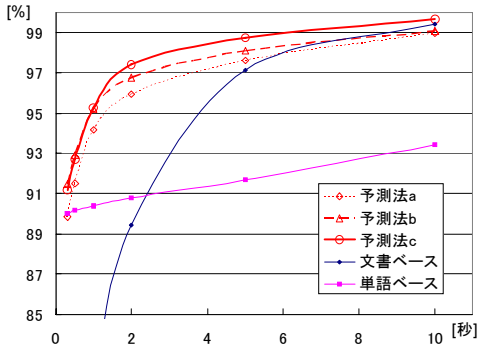

図 7　予測方法の影響
Fig. 7　The effect of $pfd(R[p][q])$

NMF ベースの 30 × 30 あるいは Kmeans ベース (30 × 30) の場合に比べ，精度が低い．このことにより，クラスタリング・アルゴリズムの差よりもクラスタ数が結果に大きく影響を及ぼしていることがわかる．

ICC ベースのみに着目したとき，クラスタ数を 10 × 10 から 30 × 30 に変えた場合の精度の上昇は 3%程度あるのに対し，クラスタ数を 30 × 30 から 50 × 50 に変えた場合の精度の上昇は 1%以内にとどまっている．このことは，クラスタ数と精度の関係に頭打ちがあることを意味している．

また，ICC ベースと NMF ベースの差に注目すると，クラスタ数が 30 × 30 と 50 × 50 の場合は ICC ベースの精度が高いもののその差はあまり大きくない．クラスタ数が 10 × 10 の場合は，検索時間の後半において NMF ベースの精度が高い．これは，検索時に最も関連する単語クラスタを決める上では，共クラスタリングを用いた場合と NMF を用いた場合の差はあまり大きくないものの，２番目，３番目に関連する単語クラスタを決める上で，NMF を用いた場合が有効であったことを意味している．

5.4　予測法の比較

本実験では，提案手法が行う $mfd(R[p][q])$ の予測処理において，予測値 $pfd_a(R[p][q])$，$pfd_b(R[p][q])$，$pfd_c(R[p][q])$ を用いた場合をそれぞれ，予測法 a，予測法 b，予測法 c とし，5.2 節の実験と同様の手順で Accuracy(100) を計測した．なお，領域の分割には共クラスタリングを用い，30 × 30 個の領域を作成した．この結果を図 7 に示す．なお，図 7 には，5.2 節で計測した文書ベースと単語ベースの結果も比較のために示している．

この結果によると，予測法 c が最も精度が高く，予測法 a の精度が最も低い．これは，予測法 c が他の予測法に比べ計算量が大きいものの，その影響は小さく，より正確に動的領域頻度を見積もれることが有効であったことを意味する．ただし，その差はあまり大きくなく，予測法 c と予測法 b の差，予測法 b と予測法 a の差はいずれも 1%以内であった．

5.5　実験結果のまとめ

本節の実験結果によると，次の点が明らかになった．

- ２次元クラスタリング処理を用いた高頻度の検索アルゴリズムは，検索時間と精度の関係において，従来の手法に比べ効率がよい．
- クラスタ数を大きくすると時間当たりの精度は向上するものの，頭打ちが存在する．
- クラスタ数を変えた場合とクラスタリング・アルゴリズムの種類を変えた場合では，クラスタ数を変えた場合の方が性能への影響が強い．
- 動的領域頻度の予測値としては，$pfd_c(R[p][q])$ が最も良い結果をもたらすものの，その差は小さい．

一点目の知見から，キーワード拡張や対話的な分析など，解

の厳密性よりも高速性が求められるアプリケーションにおいて，提案手法は有効であることが言える．二点目の知見からは，クラスタ数を闇雲に大きく設定することは必ずしも良い結果にはつながらないことがわかる．クラスタ数を大きくすると各領域に対する動的領域頻度の予測値を算出するために，より大きな処理が必要となる．このためクラスタ数の設定には注意が必要である．また，三点目と四点目の知見により，提案手法における各種パラメータ（クラスタリングアルゴリズムの種類，クラスタ数，動的頻度の予測方法の種類）のうち，最も影響が大きいのはクラスタ数であることがわかる．これは，各単語クラスタに対する動的領域頻度の予測値を算出した後の，読み込みアルゴリズムの影響が強いためと考察される．つまり，提案手法の読み込みアルゴリズムは，より予測値の高いクラスタから順に読み込むため，クラスタの粒度が重要になると思われる．

なお，共クラスタリングと NMF の比較に関しては，今回の実験では有意な差を観察できなかった．この点に関しては今後の課題であり，検索アルゴリズムの各段階での詳細な分析を行う必要がある．また，共クラスタリングは NMF と異なり単語クラスタ数と文書クラスタ数に異なる数値を設定できる．このため，単語文書マトリックスを 100×10 の領域に分割するなど，文書数と単語数に合わせたクラスタ数を設定することで共クラスタリングを用いた手法の有効性が高まるのではないかと予想している．この実験についても今後の課題である．

6. ま と め

本論文では，テキストマイニングにおける基本的な処理の一つとして高頻度語の検索処理に注目し，この処理を高速化する方法について述べた．我々は，厳密解を出力する時間を短縮する手法ではなく，指定された時間あたりでより精度の高い結果を返す方法として，2 次元クラスタリングにより単語文書マトリックスを領域に分割し，高頻度語が含まれていそうな領域から読み込む手法を提案した．評価実験によると，本手法は従来手法に比べ，約 9 倍の高速な検索が可能であった．

今後は，提案手法の詳細な分析を進めた上で，より効率の良いアルゴリズムの開発を進める予定である．また，提案手法は事前に２次元クラスタリング処理を行う必要があるため，実運用のためには２次元クラスタリング処理の高速化が求められる．この点も今後の課題である．

文　　献

[1] E.N.Efthimiadis: “Query expansion”, Annual Review of Information Systems and Technology, **31**, pp. 121–187 (1996).

[2] 吉田一星, 宅間大介：“対話的テキストマイニングのためのソフトウェアアーキテクチャー”, ProVISION, 52.

[3] J. Zobel and A.Moffat: “Inverted files for text search engines”, ACM Computing Surveys, **38**, 2 (2006).

[4] 高野明彦, 西岡真吾, 丹羽芳樹, 他：“汎用連想計算エンジンの開発と大規模文書分析への応用”, IPA 2001 年度成果報告集 (2001).

[5] 松波秀和, 寺田努, 西尾章治郎：“P2p 型コンテンツ検索システムのための効率的な top-k 検索処理手法”, 情報処理学会論文誌, **47**, 9, pp. 2850–2859 (2006).

[6] S.Michel, P.Triantafillou and G.Weikum: “Klee: a framework for distributed top-k query algorithms”, Proceedings of VLDB2005, pp. 637–648 (2005).

[7] K.Zhao, Y.Tao and S.Zhou: “Efficient top-k processing in large-scaled distributed environments”, Data and Knowledge Engineering, **63**, 2, pp. 315–335 (2007).

[8] I.S.Dhillon, S.Mallela and D.S.Modha: “Information-theoretic co-clustering”, Proceedings of ACM SIGKDD, pp. 89–98 (2003).

[9] D.D.Lee and H.S.Seung: “Algorithms for non-negative matrix factorization”, Proceedings of NIPS (Conferece on Neural Information Processing Systems), pp. 556–562 (2000).

[10] W.Xu, X.Liu and Y.Gong: “Document clustering based on non-negative matrix factorization”, Proceedings of ACM SIGIR, pp. 267–273 (2003).

[11] T.Hofmann: “Probabilistic latent semantic analysis”, Proceedings of UAI (Conference on Uncertainty in Artificial Intelligence), pp. 289–296 (1999).

[12] C.Ding, T.Li and W.Peng: “Nmf and plsi: equivalence and a hybrid algorithm”, Proceedings of SIGIR2006, pp. 641–642 (2006).

[13] S.Deerwester, S.T.Dumais, G.W.Furnas, T.K.Landauer and R.Harshman: “Indexing by latent semantic analysis”, Journal of the American Society for Information Science, **41**, pp. 391–407 (1990).

[14] CD-毎日新聞データ集：“http: //www.nichigai.co.jp /sales /mainichi /mainichi-data.html”.

[15] J.A.Hartigan and M.A.Wong: “A k-means clustering algorithm”, Applied Statistics, **28**, 1, pp. 100–108 (1979).