DEIM Forum 2009 C4-3

関連単語抽出アルゴリズムを用いたクエリ拡張

大石　哲也† 峯　恒憲†† 長谷川隆三†† 藤田　博†† 越村　三幸††

† 九州大学大学院システム情報科学府 〒 819–0395 福岡県福岡市西区元岡 744
†† 九州大学大学院システム情報科学研究院
E-mail: †oishi@ar.is.kyushu-u.ac.jp

あらまし Web 検索では，検索エンジンによって得られた結果がユーザの必要としている情報ではないことがしばしばある．この問題を解決する一つの方法として検索エンジンに与えるクエリを改善するクエリ拡張がある．クエリに含まれる語としてユーザの意図する適切な語を生成し追加することで Web 検索結果の改善に大きな効果がある．本稿では，ユーザの意図する語を生成するための方法として，センテンス間の距離に注目した関連単語抽出アルゴリズムを提案する．このアルゴリズムは重要な語の近くに出現する単語は重要であるという考えに基づいている．さらに，このアルゴリズムを適合性フィードバックの一手法である RSV と組み合わせることで，検索エンジンを利用した初期検索結果の検索精度の改善に役立つこと，その結果は RSV 単独で行った場合よりも検索精度が向上することを実験により示す．

キーワード 適合性フィードバック，クエリ拡張，Web 検索

A Method for Query Generation Using the Related Word Extraction Algorithm

Tetsuya OISHI†, Tsunenori MINE††, Ryuzo HASEGAWA††, Hiroshi FUJITA††, and Miyuki KOSHIMURA††

† Graduate School of Information Science and Electrical Engineering, Kyushu University
Motooka 744, Nishi-ku, Fukuoka-shi, Fukuoka-ken, 819–0395, Japan
†† Faculty of Information Science and Electrical Engineering, Kyushu University
E-mail: †oishi@ar.is.kyushu-u.ac.jp

Abstract When searching for information a user wants, search engines often return lots of results unintended by the user. Query expansion is a promising approach to solve this problem. In the query expansion research, one of big issues is to generate appropriate keywords representing the user's intention. This paper proposes the related–word-extraction-algorithm (RWEA) which pays attention to the distance between sentences where keywords in the original query appear. The RWEA is based on the idea that a word nearby important words is also important. We conducted several experiments to illustrate the validity of the RWEA. The results promise the effectiveness of the RWEA for improving search results.

Key words Relevance Feedback, Query Expansion, Web Retrieval

1. はじめに

有用な情報の宝庫である World Wide Web (以下，Web) は，現在，研究や仕事で利用するユーザだけでなく，一般のユーザによっても日常的に利用されている．たとえば何か物事を調べる際には，書籍を閲覧する前に，検索エンジンを利用することが一般的となってきている．

しかし検索エンジンを利用しても，ユーザが必要とする情報を取得できないことがしばしばある．その一つの理由に，ユーザが検索に利用するキーワード（以下，クエリ）に曖昧さがあり，そのため検索エンジンがユーザの意図を特定することが困難であることが挙げられる．さらに，検索エンジンから得られる情報が膨大で，ユーザがすべての情報を閲覧することが困難であるためである．このような事情から，ユーザが必要とする情報が，必要としない多くの情報の中に埋もれてしまい，必要とする情報を見つけることができない結果を招いている．

ユーザが求める検索結果を提示する方法の一つとして，検索エンジンに与えるキーワードを増やすクエリの拡張が考えられる．一般に検索エンジンに複数のキーワードを与えると AND 検索を行うので，検索結果のページ数を削減できる．もしユーザが調べたい事柄に関連する複数の適切なキーワードを検索エンジンに与えることができれば，適切なページをユーザに提示できる．しかし調べたい事柄に関連する適切なキーワードをユーザが検索エンジンに必ずしも与えることができるとは限らない．

そこで本稿では，クエリ拡張を行うための新たな手法として，クエリ中のキーワードと関連性の強い語を抽出する関連単語抽出アルゴリズムを提案する．つぎに，このアルゴリズムの有効性を示すために，新たなクエリ拡張システムを提案する．提案システムでは，適合性フィードバック手法である Robertson's Selection Value (RSV) [3] での単語の重みづけに，この関連単語抽出アルゴリズムを用いて単語の抽出と評価を行い，これを基にクエリ拡張を行う．

このシステムの有効性を示すため，2 つの比較実験を行った．まず，提案システムと RSV だけを利用したクエリ拡張を比較した．その結果，RSV だけを利用した場合よりも平均精度（MAP）が高くなった．次に，提案システムとオリジナルのクエリを利用した結果を比較した．このとき，オリジナルのクエリが，平均精度 0.8 以下と曖昧性が強い場合には，提案システムの方が平均精度が高くなるという結果を得た．

以下，2 節では関連研究について紹介し，3 節では提案システムの概要について述べる．4 節では関連単語抽出アルゴリズムとそれを用いた単語の有用度算出方法について述べ，5 節で実験結果について議論する．

2. 関 連 研 究

現在までに，関連単語の抽出やそれを利用した検索システムに関する研究がいくつか行われてきた．例えば，大塚と喜連川 [5] は大規模なアクセスログから関連語を抽出する研究を行った．彼らは，統計的に偏りなく抽出されたユーザの Web 検索の際の検索単語とそれによって閲覧したページを解析することで，関連単語を得た．この研究は，既存の検索エンジンで用いている不特定多数のユーザによって検索された頻度や，クリックされた結果などを組み合わせて関連語を提案する方法と同等またはそれ以上の精度を示している．

岡部と山田 [6] は，ユーザからの最小のフィードバックにより検索単語の拡張を行い，検索効率の向上を図ったシステムを提案した．このシステムはトランスダクティブ学習という機械学習法を用いて実現されている．この手法により，トランスダクティブ学習を用いない従来手法と比べ再現率が約 0.07 向上している．正田ら [2] は，ユーザが与えたクエリでの検索結果の上位 R 件を適合文書，それ以下を不適合文書として RSV を用いてクエリ拡張を行い，新たなクエリの重みにより初期の検索結果をソートする方法を提案している．この研究で，RSV など実験で用いたパラメータの最適な組み合わせを発見し，その組み合わせを用いた結果，精度は向上している．

大石ら [4] は，ユーザのスケジュールを考慮した Web 検索システムを提案した．このシステムは，ユーザがあらかじめ入力しておいたスケジュールを解析し，それを基に Web 検索を行い，ユーザの状況に合わせた検索を自動的に行う．この中で，スケジュールを解析する部分に本稿で提案する関連単語抽出アルゴリズムと類似のアルゴリズム（旧アルゴリズムと呼ぶ）が適用されている．具体的には，スケジュール入力の際，ユーザにはそのスケジュールのタイトルと詳細を入力させ，それを旧アルゴリズムによって解析して関連単語を発見する．その際，旧アルゴリズムで必要となるキーワード群にはスケジュールのタイトルを，関連するテキストにはその詳細を用いている．これにより，ユーザの状況に合わせた検索要求に合致した文書を検索結果の上位に表示することに成功している．本稿で提案するアルゴリズムと旧アルゴリズムとの違いは，旧アルゴリズムが語と語の間の距離に着目して，語の有用度評価を行っていたのに対して，提案アルゴリズムでは，文間の距離に着目していることが挙げられる．

3. システムの概要

我々の提案するシステムの処理の流れは以下の通りである．

3.1 クエリの入力

ユーザがシステムにクエリ（検索単語）を入力する．

3.2 適合文書，不適合文書の決定

ユーザにクエリを入力してもらい，クエリを既存の検索エンジンに与えることで，文書を取得する．ここで取得した文書を，ユーザに評価をしてもらい，適合文書，不適合文書を決定する．適合文書とはユーザの検索要求に合致している文書で，不適合文書とはユーザの検索要求に合致しない文書のことである．

3.3 関連単語の候補を抽出

3.2 節で取得した文書から関連単語の候補となる語を高速形態素解析システム MeCab [12] を用いて抽出する．MeCab は，文を与えると形態素に分解し，品詞，読み，活用などの情報を返すシステムである．

例えば,「関連単語を抽出」という文は “関連”，“単語”，“を”，“抽出” と分解される．この場合，本システムでは品詞が名詞である語のみを抽出する．つまり，この例では，“関連”，“単語”，“抽出” の 3 語が抽出される．しかし，MeCab では “関連単語” のような複合語は各単語に分解されてしまうため，それらの単語を再び結合し，連結語として抽出する．

3.4 関連語の有用度を算出

3.3 節で抽出した語に対して，関連単語抽出アルゴリズム（RWEA）と Robertson's Selection Value（RSV）を用いて関連語としての有用度を算出する．有用度の算出に用いられている RWEA は，各単語の抽出元となるセンテンス間の距離に着目している手法であり，4.1 節で詳しく説明する．また，RSV は適合文書，不適合文書それぞれの出現頻度に着目している手法であり，4.2 節で説明する．

3.5 拡張クエリの生成

3.4 節の 2 つの手法を用いて計算された有用度を基に拡張クエリを生成する．拡張クエリとは，ユーザが与えたクエリを基

に拡張されたクエリである．

例えば元のクエリが “A” と “B” であるときに，拡張クエリとは元のクエリに “C” を追加した “A”，“B”，“C” である．具体的には，有用度の高い語から順に，ユーザが与えたクエリに数語追加したものを拡張クエリとする．

3.6 検索結果を提示

3.5 節で得られた拡張クエリを既存の検索エンジンに与える．その検索結果をユーザに提示する．

4. 関連単語の有用度

本節では，関連単語の有用度の算出に用いる 2 つの方法について述べる．1 つは関連単語抽出アルゴリズムで，もう 1 つは Robertson’s Selection Value である．以下，これらの説明をする．

4.1 関連単語抽出アルゴリズム

4.1.1 アルゴリズムの概要

本アルゴリズムは，あるキーワード群 K とその K に関するテキスト T に現れる単語の中から，K に関連し，重要だと思われる単語群を抽出し，それらを出力するものである．単語群の抽出では，K や T で出現する単語間の距離に着目し，これを基準に各単語を評価し，結果として出力する単語群を形成する．

今回用いている提案アルゴリズムは旧アルゴリズムを少し改良し，単語一語ずつの距離ではなく，単語が含まれるセンテンス間での距離を用いている．センテンスとは，文を意味し，句点または終止符などで終わる，それだけで意味が通る語のかたまりである．

センテンス間の距離の重要性

上述の通り，本アルゴリズムでは単語が含まれるセンテンス間の距離を重要視している．具体的には，文書中で出現する単語が含まれるセンテンスの順番に注目しており,「ある単語 A が文書中に出現した場合，単語 A 付近のセンテンスに出現している単語ほど A に関連性が強い単語であろう」という考えに基づいている．

準備

アルゴリズムについて説明する前に，準備として以下の記号を定義する．

- K…関連単語を抽出するための基になるキーワード群
- T…キーワード群 K に関するテキスト
- $k_i(i = 1 \cdots m)$…キーワード群 K に含まれるテキスト T で出現する m 個のキーワード（テキスト T での出現順に $k_1, k_2, k_3, \cdots k_m$）
- $t_j(j = 1 \cdots n)$…テキスト T で出現する n 個のセンテンス（出現順に $t_1, t_2, t_3, \cdots t_n$）
- $w_l(l = 1 \cdots o)$…テキスト T で出現する o 個の単語（出現順に $w_1, w_2, w_3, \cdots w_o$）

但し，k_m はテキスト内に出現するキーワードは同一の語であっても区別する．w_l はテキスト T を高速形態素解析システム MeCab で形態素解析し，その結果得られた名詞のみを抽出し，それらを出現順に並べたものである．また，t_j はテキスト T をセンテンス毎に分け，各センテンスを 1 ブロックとし，それらを順に並べたものである．各ブロック内はそのセンテンス内に含まれる名詞のみを出現順に並べ，保持している．また，形態素解析した結果，同一単語が複数回出現してもそれらは別のものとみなす．

表 1　距離による単語の評価例

キーワード K	A B				
テキスト T	A G B	E G	A F C	F H	D E
$BV_A(t_j)$	5	4	3	2	1
$BV_B(t_j)$	5	4	3	2	1
$BV_A(t_j)$	3	4	5	4	3
$BV(t_j)$	13	12	11	8	5

表 2　出現位置 j での期待値 $EBV(j)$ とそれを用いた評価値 $EBV(t_j)$ の例

キーワード K	A B				
テキスト T	A G B	E G	A F C	F H	D E
$BV(t_j)$	13	12	11	8	5
$EBV(j)$	3	3.6	3.8	3.6	3
$EBV(t_j)$	4.33	3.33	2.89	2.22	1.67

4.1.2 単語の評価

テキスト T 内で出現する単語の評価は以下の手順で行う．

（1） $k_i(i = 1 \cdots m)$ を基準に $t_j(j = 1 \cdots n)$ の基礎評価値 $(BasicValue)BV(t_j)$ を計算．

（2） $BV(t_j)$ の平滑化．

（3） 単語の出現頻度による最終評価値 (Score)$S(w_l)$ を計算．

$BV(t_j)$ の算出

本アルゴリズムはキーワード群 K を基に，K とテキスト T で出現する単語間の距離を中心に単語の評価を行うことを目的としており，はじめにその基本となる評価値を求める．

まず，キーワード k_i による t_j の評価値を $BV_{k_i}(t_j)$ と表記する．この $BV_{k_i}(t_j)$ を次式により求める．ここで j_0 は，k_i を含むセンテンス t_{j_0} の添数とする．

$$BV_{k_i}(t_j) = n - |j - j_0| \tag{1}$$

つまり，$\exists w_r \in t_q(k_i = w_r)$ のとき $BV_{k_i}(t_q) = n$ とし，t_q の 1 つ隣のセンテンス t_{q-1},t_{q+1} は $BV_{k_i}(t_{q-1}) = BV_{k_i}(t_{q+1}) = n - 1$，$t_q$ の 2 つ隣のセンテンス t_{q-2},t_{q+2} は $BV_{k_i}(t_{q-2}) = BV_{k_i}(t_{q+2}) = n - 2$，…と各 t_j に対して k_i による基礎評価値 $BV_{k_i}(t_q)$ を与える．以上の計算をすべての $k_i(i = 1 \cdots m)$ で行う．

それぞれの t_j において $BV_{k_i}(t_j)$ の和を $BV(t_j)$ とする．即ち，$BV(t_j)$ は次式により求める．

$$BV(t_j) = \sum_{i=1}^{m} BV_{k_i}(t_j) \tag{2}$$

表 1 に $BV(t_j)$ を求める例を示す．例より，キーワード群 K に含まれている単語のあるセンテンスを中心に，それに近いセンテンスほど高評価を得ていることがわかる．

$BV(t_j)$ の平滑化

前節で求められた $BV(t_j)$ は，センテンス t_j がテキスト T で出現する位置を考えると不公平である．テキスト T の端に出

現するセンテンス $t_j(j=1,\ n)$ については $1 \leqq BV_{k_i}(t_j) \leqq n$ であるのに対し，テキスト T の中央に出現するセンテンス $t_{n/2}$ については $n/2 \leqq BV_{k_i}(t_{n/2}) \leqq n$ である．よって，$BV(t_j)$ のとり得る値の期待値はセンテンス t_j の出現位置 j によって異なり，このままでは T の中央に出現するセンテンスは必然的に与えられる評価値が大きくなってしまい，公平な評価はできない．

この問題を解決するために，求めた $BV(t_j)$ をセンテンス t_j の出現位置 j での評価値の期待値を用いて平滑化を行う．センテンス t_j の出現位置 j での評価値の期待値 (ExpectationBasicValue)$EBV(j)$ は以下の式で求められる．

$$EBV(j) = \frac{1}{2n}\{n(n+2j-1) - 2j(j-1)\} \tag{3}$$

ここで，n はテキスト T 内の全センテンス数である．

本アルゴリズムでは，平滑化後の評価値を $EBV(t_j)$ とし，以下の式で計算する．

$$EBV(t_j) = \frac{BV(t_j)}{EBV(j)} \tag{4}$$

表 2 に $EBV(j)$ を計算し，$EBV(t_j)$ を求めるまでの例を示す．$EBV(j)$ は出現位置 j での評価値の期待値であるので，中心付近 ($j = n/2$ 付近) の値が大きくなっている．当然だが $EBV(j)$ は中心をピークとして左右対称になっている．これにより $BV(t_j)$ では出現位置により不公平な評価を得ていたものが平滑化されている．

$S(w_l)$ の算出

本アルゴリズムはセンテンス間の距離を最重要視して評価値計算を行っているが，それに加えて TF/IDF 法などで用いられるテキスト内での単語の出現頻度 $TF(TermFrequency)$ の概念も考慮する．つまり，テキスト T 内で複数回出現した単語はある程度重要視すべきである，ということである．

まず，テキスト T 内で複数回出現した単語についてその単語の $EBV(w_l)$ の平均値 $AveEBV(w_l)$ を計算しておく．例えば，単語 w_a と単語 $w_b(a \neq b)$ が同じ単語だったとすると，$AveEBV(w_a) = AveEBV(w_b) = (EBV(w_a)+EBV(w_b))/2$ となる．無論，T 内で出現が 1 回のみの単語 w_l に関しては $AveEBV(w_l) = EBV(w_l)$ である．

さらに単語 w_l の tf 値 (T 内での出現回数) を用いて以下の式で表される重み $V(w_l)$ を計算する．

$$V(w_l) = 1 + \frac{tf(w_l)}{n}\log tf(w_l) \tag{5}$$

式 (5) (注1)で使われるパラメータは以下の通りである．

- $tf(w_l)$ … テキスト T 内での単語 w_l の出現回数

n の意味と対比させると，tf の値は本来「テキスト T 内での単語 w_l が出現するセンテンスの数」とすべきである．しかし，今回 tf の値を「テキスト T 内での単語 w_l の出現回数」とした．これは，単語 w_l が複数回出現したセンテンスはその語について重要なセンテンスであるので，そのセンテンスの tf を 1 とするのではなく，単語 w_l の出現回数とするべきと考えたからである．

(注1)：式 (5) は初めは tf の値のみを重みとしていた．しかし，これでは複数回出現した語が重要視され過ぎてほぼこの重みのみの評価値となる．次に，tf の値をセンテンスののべ数 n で割ったものを重みとした．この場合，重みが小さすぎ，ほとんど意味をなさなかった．このような試行錯誤の結果，上のような式となった．

表 4　単語毎の $S(w)$

w_l	A	B	C	D
$S(w)$	4.62	4.33	2.89	1.67
w_l	E	F	G	H
$S(w)$	3.19	3.27	4.90	2.22

そして，得られた $AveEBV(w_l)$ と $V(w_l)$ を用いて以下の式でテキスト T で出現する単語 w_l の評価値 $S(w_l)$ を計算する．

$$S(w_l) = AveEBV(w_l) \times V(w_l) \tag{6}$$

こうして求められた $S(w_l)$ を，本アルゴリズムでのテキスト T 内で出現する単語 w_l の評価値とする．

表 3 に $S(w_l)$ 算出までの例を示し，表 4 に各単語 w 毎の $S(w)$ を示す．テキスト T 内では単語 A，E，F，G が 2 回出現しているので $tf(w_l)$ は 2 となっており，その他の単語については tf 値は 1 となっている．重み $V(w_j)$ は tf 値を基に計算しているので，tf 値が 1 の単語については 1，2 回出現している単語については 1 より大きな値をとっている．そして最終的な単語の評価値 $S(w_l)$ が求められ，本アルゴリズムでは，テキスト T 内の単語は，評価値の大きい G，A，B，F，E，C，H，D の順でキーワード群 K へ関連度が高いとみなす．

4.2 Robertson's Selection Value

Robertson's Selection Value(RSV) では，単語 w が適合文書集合のみに偏って出現し，不適合文書集合にはほとんど出現しない場合ほど，評価値が高くなる特徴がある．

RSV は以下の式で求められる．

$$\begin{aligned} RSV_w = \quad & (\tfrac{df_w^+}{R^+} - \tfrac{df_w^+ + df_w^-}{R^+ + R^-})\{\alpha \log \tfrac{R^+ + R^-}{df_w^+ + df_w^-} \\ & + (1-\alpha)\log \tfrac{(df_w^+ + 0.5)/(R^+ - df_w^+ + 0.5)}{(df_w^- + 0.5)/(R^- - df_w^- + 0.5)}\} \end{aligned} \tag{7}$$

式 (7) で用いられているパラメータは以下の通りである．

- R^+ … 適合文書数
- df_w^+ … 語 w を含む適合文書数
- R^- … 不適合文書数
- df_w^- … 語 w を含む不適合文書数
- α … 制御パラメータ ($0 \leqq \alpha \leqq 1$)

式 (7) で用いられてる定数 0.5 は，対数が取れるようにするための調整値である．また，用いられている log は自然対数である．

4.3 有用度算出

RSV は文書毎の評価のみから計算される為，ある程度の文書数が必要となる問題がある．また，語 w が文書内のどの位置に出現しても同じ重みとなる問題もある．これらを解決するために関連単語抽出アルゴリズムを用いる．単語 w における RSV の評価値を RSV_w，関連単語抽出アルゴリズムの評価値を $S(w)$ とする．式としては以下のようになる．

$$Score(w) = S(w) \times RSV_w \tag{8}$$

表 3　$S(w_l)$ 算出までの各値の例

キーワード K	A	B										
テキスト T	A	G	B	E	G	A	F	C	F	H	D	E
$EBV(w_l)$	4.33	4.33	4.33	3.33	3.33	2.89	2.89	2.89	2.22	2.22	1.67	1.67
$AveEBV(w_l)$	3.61	3.83	4.33	2.50	3.83	3.61	2.56	2.89	2.56	2.22	1.67	2.50
$tf(w_l)$	2	2	1	2	2	2	2	1	2	1	1	2
$V(w_l)$	1.28	1.28	1	1.28	1.28	1.28	1.28	1	1.28	1	1	1.28
$S(w_l)$	4.62	4.90	4.33	3.19	4.90	4.62	3.27	2.89	3.27	2.22	1.67	3.19

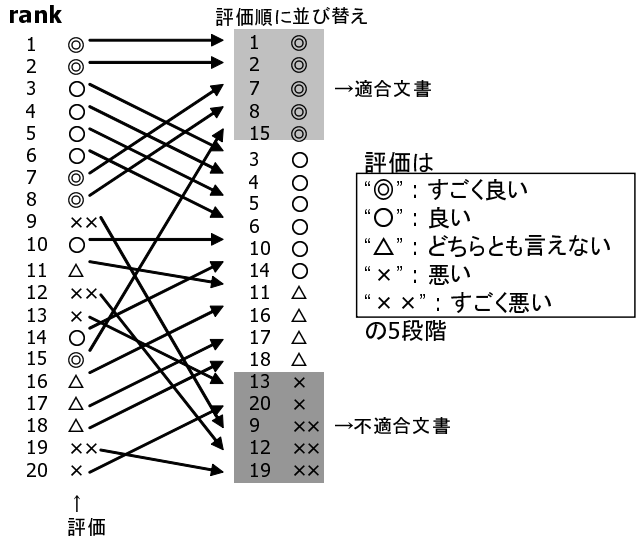

図 1　検索結果をユーザの評価順に並び替え

表 5　実 験 一 覧

語数		クエリ拡張手法	フィードバック	実験名
1 語	goo による検索			1:goo1
	クエリ拡張 1 語から 2 語	提案手法	明示的	2:pro1e
			擬似	3:pro1p
		RSV	明示的	4:rsv1e
			擬似	5:rsv1p
2 語	goo による検索			6:goo2
	クエリ拡張 2 語から 3 語	提案手法	明示的	7:pro2e
			擬似	8:pro2p
		RSV	明示的	9:rsv2e
			擬似	10:rsv2p
3 語	goo による検索			11:goo3
	クエリ拡張 3 語から 4 語	提案手法	明示的	12:pro3e
			擬似	13:pro3p
		RSV	明示的	14:rsv3e
			擬似	15:rsv3p

これにより同様の文書に出現している語が複数存在したとしてもクエリの語の近辺に出現する語の方が有用度は高くなる．

5. 評 価 実 験

5.1 実 験 方 法

1 語，2 語，3 語のクエリをそれぞれ 50 件ずつ用いて 3 人の被験者に Web 検索を行ってもらう．まず，検索エンジン goo を用いて検索結果を評価する．続いて，RSV によって拡張されたクエリと，提案手法によって拡張されたクエリを用いて，検索結果を評価する．以下に詳細を示す．

5.1.1　goo による検索

2 語で構成されるクエリを 50 件用意した．これらを用いた検索エンジン goo による検索は以下のように行う．

（1） Web ページの評価

クエリを検索エンジン goo に与えて得られた検索結果の上位 20 件の Web ページを被験者が評価する．被験者は Web ページを 5 段階（すごく良い，良い，どちらとも言えない，悪い，すごく悪い）で評価する．

（2） Web ページの並び替え

Web ページを評価の良い順で並びかえ，同じ評価の Web ページは，元の順位の昇順に並び替える．

（3） 適合文書，不適合文書の決定

ここで，擬似フィードバックと明示的フィードバックを用いて適合文書と不適合文書を決定する．前者は，並び替える前の Web ページの上位 5 件を適合文書，下位 5 件を不適合文書とする．後者は，図 1 に示すように，並び替えた Web ページの上位 5 件を適合文書，下位 5 件を不適合文書とする．

これらの作業を各クエリに対して，各被験者に行ってもらう．

5.1.2　RSV によるクエリ拡張

適合文書と不適合文書に対して RSV を適用して，5.1.1 節で使用したクエリを拡張する．RSV による評価で最も評価値の高い語を元のクエリに追加して拡張クエリを決定する．この拡張クエリを用いて Web 検索を行う．そして，5.1.1 節と同様に Web ページを評価する．

5.1.3　提案手法によるクエリ拡張

適合文書と不適合文書に対して提案手法を適用して，5.1.1 節で使用したクエリを拡張する．

ここで適合文書に対して関連単語抽出アルゴリズムを適用する．関連単語抽出アルゴリズムは単一文書に適用するアルゴリズムであるが，以下のように複数文書に適用する．適合文書を $d_k(k=1,2,\cdots,n)$ としたとき，語 w の各適合文書における関連単語抽出アルゴリズムの評価値を $S(w_{d_k})(k=1,2,\cdots,n)$ とすると，語 w の評価値 $S(w)$ は

$$S(w)=\frac{\sum_{k=1}^{n}S(w_{d_k})}{n} \tag{9}$$

として計算する．

適合文書と不適合文書に対して RSV を適用して，式 (9) を式 (8) に代入して得られた評価値の順番に語を並び替え，最も

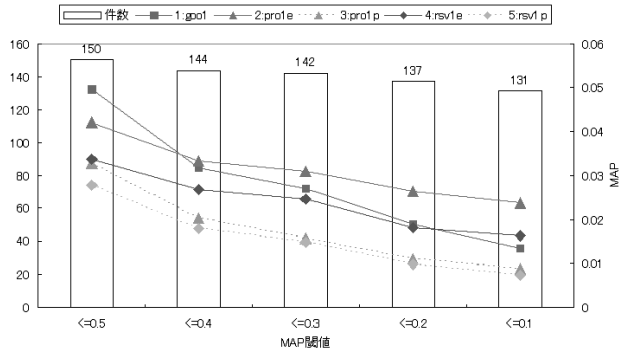

図 2　R10 と P1（1 語）の場合

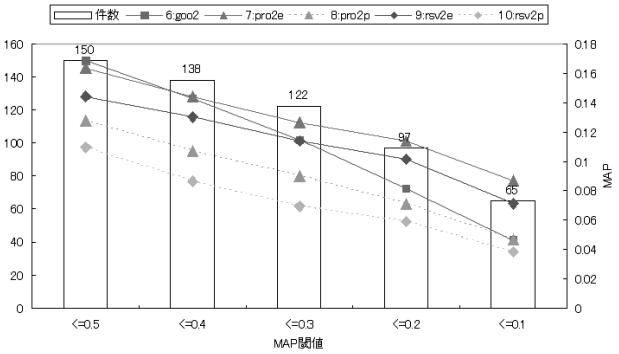

図 6　R10 と P1（2 語）の場合

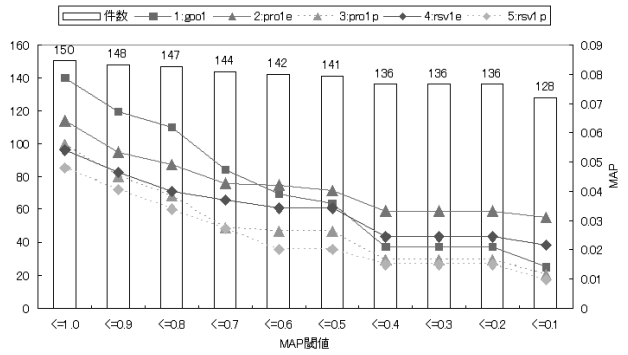

図 3　R20 と P1（1 語）の場合

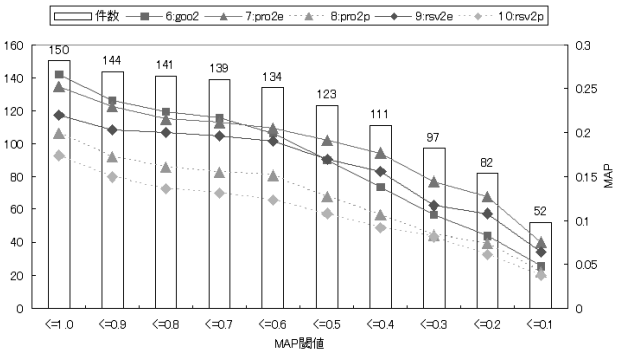

図 7　R20 と P1（2 語）の場合

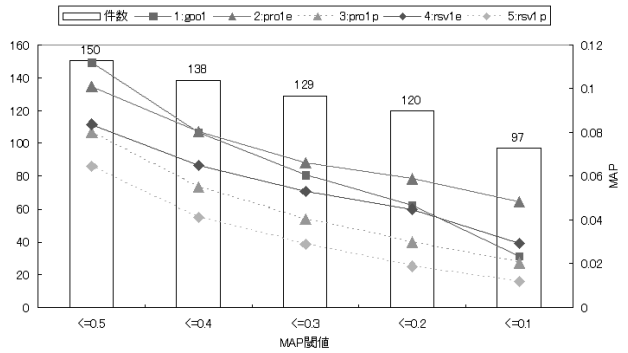

図 4　R10 と P2（1 語）の場合

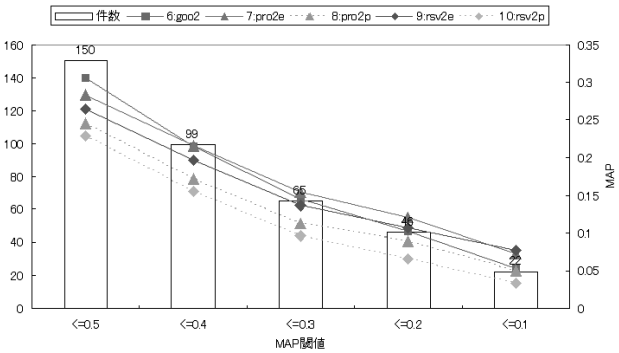

図 8　R10 と P2（2 語）の場合

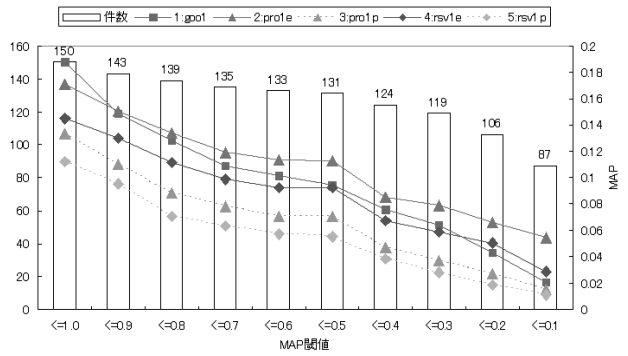

図 5　R20 と P2（1 語）の場合

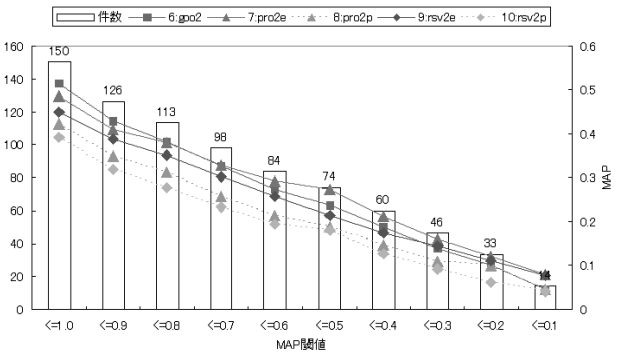

図 9　R20 と P2（2 語）の場合

評価値の高い語を元のクエリに追加して拡張クエリを決定する．この拡張クエリを用いて Web 検索を行う．そして，5.1.1 節と同様に Web ページを評価する．

5.2　結　　果

5.1 節で行った実験は表 5 に示す 15 個の実験である．これらの各実験結果に対して，各クエリにに対して平均精度（Average Precision: AP）を計算する．

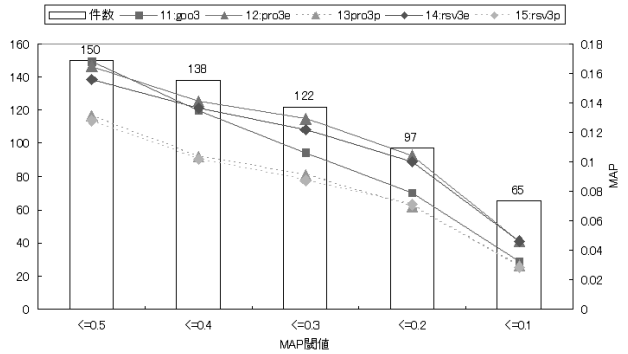

図 10　R10 と P1（3 語）の場合

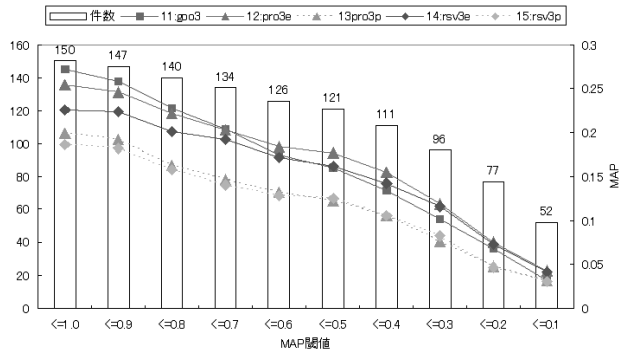

図 11　R20 と P1（3 語）の場合

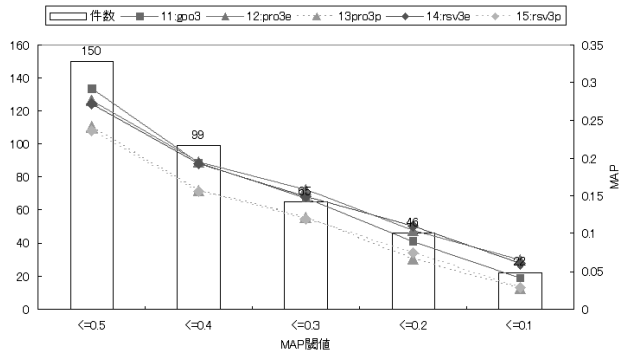

図 12　R10 と P2（3 語）の場合

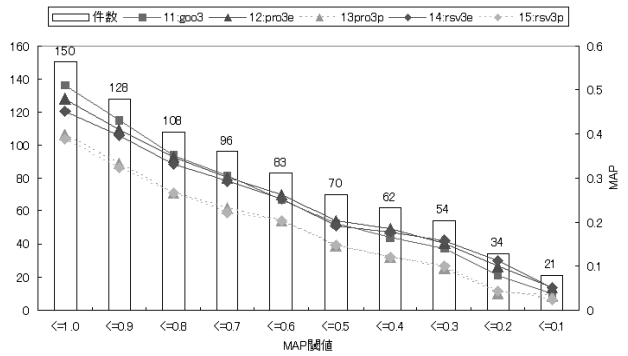

図 13　R20 と P2（3 語）の場合

このとき，平均精度を求める対象を以下の 2 パターンにする．

- 上位 10 件（R10）
- 上位 20 件（R20）

また，適合文書を以下の 2 パターンで定義する．

表 6　各手法の MAP 値の比較

	R10-P1	R20-P1	R10-P2	R20-P2
1:goo1	0.0496	0.0786	0.1120	0.1878
2:pro1e	0.0422	0.0641	0.1009	0.1707
3:pro1p	0.0327	0.0558	0.0799	0.1333
4:rsv1e	0.0338	0.0539	0.0835	0.1449
5:rsv1p	0.0278	0.0479	0.0646	0.1122
6:goo2	0.1687	0.2662	0.3057	0.5147
7:pro2e	0.1633	0.2521	0.2832	0.4856
8:pro2p	0.1279	0.1989	0.2458	0.4222
9:rsv2e	0.1442	0.2193	0.2652	0.4498
10:rsv2p	0.1092	0.1742	0.2289	0.3908
11:goo3	0.1677	0.2724	0.2918	0.5105
12:pro3e	0.1648	0.2544	0.2769	0.4800
13:pro3p	0.1311	0.1996	0.2412	0.3994
14:rsv3e	0.1558	0.2262	0.2716	0.4510
15:rsv3p	0.1279	0.1861	0.2369	0.3888

- 評価が「すごく良い」である文書を適合文書（P1）
- 評価が「すごく良い」と「良い」である文書を適合文書（P2）

これらのパターンの組み合わせて，R10-P1，R20-P1，R10-P2，R20-P2 の 4 つのパターンに関して MAP（Mean Average Precision）を計算する．MAP を計算する時に用いる適合文書数は各クエリとも 20 とする．

図 2 から図 13 は，goo による AP が変化するとき，各手法の MAP がどのように変化するか示した図である．各図の左縦軸は該当するクエリの件数，右縦軸は MAP，横軸は AP の閾値であり，棒グラフがクエリの件数，折れ線グラフが MAP を示している．例えば，図において，AP の閾値が「<= 0.8」であるとき，goo の AP が 0.8 より小さいようなすべてのクエリに対して，各手法の MAP をプロットしており，そのときの棒グラフの示す値が AP の閾値の条件を満たすクエリの件数である．R10 の AP は R20 の半分しか取りえない，すなわち，0.5 以下の値しか取れないので，R10 のグラフは 0.5 以下で表示している．

5. 2. 1　R10 と R20 の比較

各手法の R10 における MAP と R20 における MAP を比較する．

表 6 は AP の閾値が 1.0 以下の時の各実験結果の MAP を示している．図 2 から図 13 より，R20 における MAP が R10 における MAP より高いことがわかる．しかし，表 6 より，どの実験結果を見ても，R20 における MAP は R10 における MAP の 1.45 倍から 1.75 倍にしかならない．これは上位 10 件における適合率（P@10）が上位 20 件における適合率（P@20）よりも高いことを示している．もし逆であれば，R20 における MAP は R10 における MAP の 2 倍を超えるはずである．以上より，1 位から 10 位までの適合 Web ページ数と，11 位から 20 位までの適合 Web ページの数を比較すると，前者の方が多いことがわかる．

5.2.2 P1 と P2 の比較

P1 と P2 を用いたときの MAP の変化を比較する．図 2 から図 13 より，どの実験結果を見ても，P1 も P2 も似たような変化をする．同じ実験で P1 の MAP が P2 の MAP より低くなるのは，P1 では「すごく良い」と評価された文書のみを適合文書とするため，適合文書数が P2 と比較して少ないためである．これらの図は，厳しい評価をするユーザだけでなく，甘い評価をするユーザに対しても，goo による AP が，0.4 であるクエリに対して提案手法のクエリ拡張が効果的に働くことを示している．

5.2.3 RSV と提案手法の比較

提案手法を使用して拡張されたクエリによって得られた検索結果の MAP と，RSV によって得られた検索結果の MAP を比較する．

図 2 から図 13 と表 6 は，提案手法による MAP が RSV による MAP より高いことを示している．これは，関連単語抽出アルゴリズムがほぼ常に RSV を効果的に強めていることを表している．一部逆転している部分があるが，これは対象となるクエリ数が少なかったり，擬似フィードバックによってユーザの意志を反映した単語を適切に取り出すことができなかったためだと考えられる．

5.2.4 goo と提案手法の比較

クエリが 1 語の時（図 2 から図 5），どの実験結果も非常に低い MAP 値を取っている．これは goo による検索結果に適切な Web ページが含まれていないためであり，本研究で使用したどの手法も効果的には働かない．

クエリが 2 語の時（図 6 から図 9），goo による AP がある特定の値（各々0.4, 0.6, 0.4, 0.8）より小さいようなクエリに対して，明示的フィードバックを用いた提案手法（7:pro2e）による MAP が goo（6:goo2）による MAP より高くなった．表 6 を見ると “6:goo2” の MAP が “7:pro2e” の MAP より高い値を示しているが，この値にはもともと goo で高い評価をされているクエリも含まれている．そのようなクエリに対してクエリ拡張をする必要はないため，“6:goo2” で高い MAP 値を持つクエリを省くことで，上記のような結果となった．また，擬似フィードバックを用いた手法（8:pro2p，10:rsv2p）による MAP が goo による MAP より高くなることはなかった．

クエリが 3 語の時（図 10 から図 13），2 語の時と同様な傾向を示した．2 語の時と異なるのは，提案手法と RSV の差（12:pro3e と 14:rsv3e，13:pro3p と 15:rsv3p）が小さくなっており，提案手法だけが効果的に働いているとは言えない．

6. おわりに

本稿では，クエリ拡張を行うための一手法である関連単語抽出アルゴリズムを提案した．そして，関連単語抽出アルゴリズムと適合性フィードバック用アルゴリズムである RSV とを融合させて，初期クエリに対して検索エンジンから返された Web ページへのユーザの評価情報を元に，新たなクエリを生成し，そのクエリを利用して再検索を行うシステムを試作した．関連単語抽出アルゴリズムは，文書内頻度だけでなくクエリ内の語が出現するセンテンスと他のセンテンス間の距離に着目して，クエリ内の語との関連度を計算する．

提案システムの評価実験の結果，RSV だけを用いて再検索を行った場合よりも，提案システムは精度の高いクエリ拡張を行うことができることを示した．さらに検索エンジン goo での検索結果の MAP がある一定値よりも低い場合，すなわちクエリに曖昧さが強く goo の結果が思わしくない場合に，その検索精度を改善できることを示した．

これらの実験の結果，提案システムは，ユーザがクエリに適切な語を追加できない場合，即ち，クエリに用いられている語が少ない場合でかつその語が検索語として有用でない場合に最も精度が改善する，ということが確認できた．これは，当初の目的である，「ユーザが適切なクエリを追加できない場合にシステムが自動的に関連語を追加するシステムの実現」に向けた第一歩としては十分な結果を与えたと考える．

今回，予備実験として，ユーザに Web ページを評価してもらい，その評価情報を利用することで，適合文書と不適合文書に含まれる語を精度よく識別することができたが，Web ページの評価には多くの時間を要し，そのためのユーザの負担は無視できるものではない．そのため，このユーザの負担を軽減しつつ，適合文書と不適合文書の選択を自動的に行うことが望ましい．実験では，クエリの語数，クエリ拡張手法，フィードバック手法を変えて比較した．今後はクエリ拡張で関連単語抽出アルゴリズムのみを用いた場合も比較するために追加実験を行う．

謝辞 本研究の実験を行うにあたり，九州大学工学部電気情報工学科の倉門浩二氏，田代祐一氏，中村徹氏には多大な協力をして頂いた．ここに深く感謝します．また, 本研究は科研費（20240003）の助成を受けたものである．

文　献

[1] P. Lawrence, B. Sergey, M. Rajeev and W. Terry: “The PageRank Citation Ranking: Bringing Order to the Web”, Technical Report, Stanford University, 1998
[2] T. Masada, T. Kanazawa, A. Takasu and J. Adachi: “Improving Web Search by Query Expansion with a Small Number of Terms”, NTCIR-5 Workshop Meeting, (2005)
[3] S. E. Robertson, On term selection for query expansion, Journal of Documentation, v.46 n.4, p.359-364, Dec. 1990
[4] 大石 哲也, 峯 恒憲, 長谷川 隆三, 藤田 博, 越村 三幸, 倉元 俊介, 永田 廣人: “ユーザのスケジュールを考慮した Web 検索手法”, Joint Agent Workshop and Symposium 2007
[5] 大塚 真吾, 喜連川 優: “大規模アクセスログを用いた検索支援システムの提案”, 日本データベース学会 Letters Vol.5, No.1, 2006
[6] 岡部正幸, 山田誠二: “トランスダクティブ学習による最小文書判定からのクエリ拡張”, 人工知能学会論文誌 21 巻 4 号 I, pp.398-405, 2006
[7] 倉元 俊介, 永田 廣人, 大石 哲也, 峯 恒憲, 長谷川 隆三, 藤田 博, 越村 三幸: “単語間の距離を利用した関連単語抽出アルゴリズム”, 火の国情報シンポジウム 2007
[8] 馬場肇: “Google の秘密 - PageRank 徹底解説”, http://www.kusastro.kyoto-u.ac.jp/ ~baba/wais/pagerank.html
[9] goo, http://www.goo.ne.jp/
[10] Google, http://www.google.co.jp/
[11] Yahoo! JAPAN, http://www.yahoo.co.jp/
[12] 高速形態素解析システム “MeCab”, http://mecab.sourceforge.jp/