DEWS2007 C9-2

周辺語による索引付けを用いた曖昧な記憶の想起支援手法

稲川　雅之† 　手塚　太郎†† 　田中　克己††

† 京都大学工学部情報学科　〒 606–8501 京都府京都市左京区吉田本町
†† 京都大学大学院情報学研究科社会情報学専攻　〒 606–8501 京都府京都市左京区吉田本町
E-mail: †{inagawa,tezuka,tanaka}@dl.kuis.kyoto-u.ac.jp

あらまし　うろ覚えの情報を検索したい場合，従来のキーワード検索では語句の完全な一致が求められるため，効率よく検索できないことが多い．この問題を解決するため，本稿では対象となるオブジェクトにあらかじめ索引付けを行うことにより，うろ覚えの状態でも求めるオブジェクトを検索することの出来る手法を提案する．これは，Web ページ中のオブジェクト名の周辺語を用いて索引付けを行うものである．具体的には周辺語を *tf-idf* に類似した手法を用いて重み付けし，この重みを元に索引を作成する．また，本稿では慣用的表現としてのことわざの検索に焦点をあて，実際に索引付けを行った．さらにこの索引を利用し，状況を表すフレーズを与えた時に，その状況を最も良く言い表すようなことわざを提示するシステムを構築し，実験を行った．この実験により，提案した手法が有効であることが示された．

キーワード　情報検索，索引付け，想起支援，Web マイニング

Memory Recall Support based on Indexing by Surrounding Terms of Candidates

Masayuki INAGAWA†, Taro TEZUKA††, and Katsumi TANAKA††

† Department of Informatics and Mathematical Science, Faculty of Engineering, Kyoto University,
Yoshida-honmachi, Sakyo-ku, Kyoto,606-8501 Japan
†† Department of Social Informatics, Graduate School of Informatics, Kyoto University,
Yoshida-honmachi, Sakyo-ku, Kyoto,606-8501 Japan
E-mail: †{inagawa,tezuka,tanaka}@dl.kuis.kyoto-u.ac.jp

Abstract　Since most information retrieval systems today require a complete match between the search query and result, it is difficult to obtain information when the user's memory is vague. In order to cope with this problem, we propose a technique that enables appropriate retrieval of information even when one's knowledge is vague, by utilizing surrounding texts of search targets for indexing. The indices are created from terms that surround the search target's appearance on the Web. Specifically, we use a technique similar to tf-idf to calculate weights for surrounding terms. We have implemented a system that creates indices for proverbs, enabling the user to find proverbs that best describes his/her situation, when it is expressed in a short phrase. The results of the experiment showed the effectiveness of our proposed method.

Key words　Infomation retrieval, Indexing, Memory recall support, Web mining

1. はじめに

近年，インターネットと Web 検索エンジンの普及により，多くの調べ物を検索エンジンを用いて手軽かつ迅速に行うことが可能になった．しかし，未だに既存の検索エンジンが苦手とする分野も多い．現在の検索エンジンは 1 つあるいは複数のキーワードを入力し，そのキーワードを含む Web ページを結果として返すものが一般的である．そのため，肝心のキーワードをはっきりと思い浮かべることが出来なければ求めるページを得ることが出来ない．

例えば，うろ覚えの名前，人物名や作品名などが思い出せない場合，その対象を的確に言い表すキーワードを思い付く事が出来れば，名前が載っているページを発見出来るかもしれない．しかし実際には適当なキーワードを思い付けない事も多い．

このようなうろ覚え状態での検索を実現するため，本研究ではオブジェクトの名称の想起を支援するための手法を提案する．具体的なキーワードを思いつくことの出来ないうろ覚えの状態でも，求めるオブジェクト名を得られるようにすることが本研究の目的である．

本研究におけるオブジェクトとは，一般名詞や固有名詞，定型句なども含む広い意味で用いている．これは「概念」とも呼べるものであるが，定義するならば「名前」が存在し，それによって識別することが可能なもの全体のことである．

具体的な手法としては，Web ページ中に出現するオブジェクト名の周辺語を用いてオブジェクトに索引付けし，その索引を用いてオブジェクト名の検索を実現しようというものである．

周辺語を索引付けに用いる理由は，オブジェクト名の周辺テキストという「文脈」の中に，そのオブジェクトを特徴付けるような語が現れるという仮定に基づくものである．

また，本稿ではオブジェクトの例としてことわざを取り上げる．状況を表す文章を与えた時，それを最も良く言い表すようなことわざを検索するシステムを実際に作成し，このシステムを使って提案手法の検証・考察を行った．

2. 節では，索引付けの具体的手法について詳しく述べる．3. 節では，2. 節で作成した索引を用いてオブジェクトの名称を検索する手法について述べる．4. 節では索引付けと検索のシステムを実装するに当たっての処理を述べる．5. 節では，実際に作成したシステムを用いて行ったことわざの検索の結果を示し，実験結果の検証と考察を行う．6. 節では関連研究を挙げる．7. 節では本研究のまとめと今後の課題について述べる．

2. 索引付け手法

本節では，オブジェクトに索引付けする具体的な手法を述べる．まず本研究におけるオブジェクト索引とは何かを説明し，次に索引付けの手法を計算式とともに示す．

一般的な索引とは，(索引語，ページアドレス）という形の索引レコードが集まったものである．本研究におけるオブジェクト索引とは，(索引語，オブジェクト名，重み）という形の索引レコードが集まったものであり，重みはオブジェクトと索引語の関連の強さを表す値である．また，このオブジェクト索引は各オブジェクトに対する特徴ベクトルの形を取り，このベクトルを索引ベクトルと呼ぶ．索引ベクトルは索引語に対する重みを各要素とするものであり，この各要素がそのまま各索引レコードに対応する．

索引付けは，大まかに次のような流れで行われる．

（1） Web 検索エンジンを用い，オブジェクト名が出現する Web ページ集合を得る．

（2） 各 Web ページに対し，オブジェクト名の周辺テキストを形態素解析器にかけ，索引語として適当な語を抽出する．

（3） 得られた語の重みを計算式に従って増加させる．

以下では，この各手順についてそれぞれ述べる．

2.1 Web ページの検索

まず始めに，オブジェクト名が出現する Web ページを得る必要がある．そのために，Web 検索エンジンでオブジェクト名をクエリとした検索を行う．本研究では Web 検索エンジンとして Google [1] を用いた．

検索を行う際，オブジェクト名をそのまま検索エンジンに与えると，オブジェクト名の中に現れる各語が分割して出現する Web ページも検索されてしまう場合がある．このような事を防ぐため，検索クエリとの完全マッチを取るようなオプションを設定する．Google の場合は，クエリを「“ 」「 ”」（ダブルクォート）で囲うと完全マッチを取るようになる．

オブジェクト名の出現する Web ページを検索エンジンを用いて得るのであるが，オブジェクト名の周辺テキストを取り出すための手法として，2 通りの方法が考えられる．1 つは Web ページそのものを取得してその中から周辺テキストを得る方法である．もう 1 つは，検索エンジンの検索結果から得られる各ページのスニペット（要約文）を用いる方法である．

検索エンジンのスニペットを用いる方法は，Web アクセスがオブジェクト名をクエリとする検索エンジンへのアクセスのみとなる．すなわち 1 つのオブジェクトにつき 1 度の Web アクセスで索引付けを行うことが出来，高速である．しかし，検索エンジンのスニペットは 1 つのページに対して一般に 100 文字程度であり，周辺テキストの量として十分であるとは言えない．

一方 Web ページそのものを取得する手法では，完全な周辺テキストを得ることが出来る．しかし，1 つのページにつき 1 回の Web アクセスが必要であり，オブジェクト名の出現するページを 100 件取得するとすれば，スニペットを用いる手法に比べて 100 倍の Web アクセス回数がかかってしまう．また，周辺テキストのテキスト量が増加すれば，後に述べる索引語の抽出において主に形態素解析にかかる時間も増加することになる．

索引付けはユーザの操作に応じてリアルタイムで行う処理ではなく，あらかじめ索引を用意しておく前処理の手法であるため，実行時間は最重要な要素ではない．だが，柔軟な実装を行うためには実行時間も重要なため，どちらが優れているとは一概に言うことが出来ない．そのため後に述べる実験においては，この 2 つの手法を共に用い，比較している．

2.2 索引語の抽出

前項によって得られた周辺テキストから，オブジェクト索引に用いる語として適当な語（索引語と呼ぶ）を抽出する手法について述べる．

まず，得られた周辺テキストを形態素解析器にかけ，日本語文の意味的な最小単位である形態素に分割する．本研究では形態素解析器として MeCab [2] を用いた．

分割された形態素のうち，品詞が名詞・形容詞・形容動詞・動詞のうちいずれかであるものを残す．ただし代名詞は除外する．また，活用のある語については活用を基本形に直した形にする．さらに「私」や「する」のように一般的過ぎる語をストップワードとして除外する．また，英字のみからなる語も同様に除外する．

2.3 索引ベクトルの作成

前項で抽出した索引語について，オブジェクトに対する索引ベクトルの重みを求める手法について述べる．

各オブジェクトに対する索引ベクトルの各語の重み計算には，

tf-idf [3] に類似した手法を用いる．

2.3.1 索引ベクトルの重み

オブジェクト O_k に対する索引ベクトル $\boldsymbol{iv_k}$ の語 w_i に対する重み $iv_k(w_i)$ は以下のように定義される．

$$iv_k(w_i) = stf_k(w_i) \times idf(w_i) \tag{1}$$

ここで，$\boldsymbol{stf_k}$ はオブジェクト O_k に対する周辺語頻度（surrounding term frequency）ベクトルであり，$stf_k(w_i)$ は周辺語頻度ベクトル $\boldsymbol{stf_k}$ の語 w_i に対する重みである．これは各語がオブジェクト名の周辺テキストにどの程度の頻度で現れるかを示す値である．また，$\boldsymbol{idf}$ は語の「珍しさ」を表すベクトルであり，$idf(w_i)$ はその語 w_i に対する重みである．

2.3.2 項および 2.3.3 項では，これらのベクトルについて説明する．

2.3.2 周辺語頻度ベクトルの重み

オブジェクト O_k に対する周辺語頻度ベクトルの語 w_i に対する重み $stf_k(w_i)$ はまず以下のように定義される．

$$stf_k(w_i) = \sum_{T_j \in P_k} tf(w_i, T_j) \tag{2}$$

ここで，P_k はオブジェクト O_k に対する周辺テキストの集合．$tf(w_i, T_j)$ はテキスト T_j における語 w_i の出現回数である．

この重み付けは最も単純であるが，周辺テキスト中に現れる語を全て等しく評価している．そのためスニペットを用いる手法ではあまり問題ないが，Web ページ本文を周辺テキストとして用いると，オブジェクト名が出現する部分とは離れた所にある語も等しく重みが増してしまうため，ノイズが多くなってしまうという問題がある．

この問題を解決するため，オブジェクト名との関連の強さによって重みを増す量を変化させる必要がある．オブジェクト名と索引語との関連の強さを評価する値として，ここではオブジェクト名と索引語との距離を用いることにする．

オブジェクト名との距離を周辺語頻度ベクトルに反映させるため，まず距離で重み付けされた頻度（$dwtf$）を定義する．距離の反映させ方として 2 通りの手法を用いた．上記の式 (2) を手法 (A) として，新たに定義する 2 手法をそれぞれ (B)，(C) とする．

$$\text{(A)}\, dwtf(w_i, T_j) = \sum_{a_l \in A_{ij}} 1 = tf(w_i, T_j) \tag{3}$$

$$\text{(B)}\, dwtf(w_i, T_j) = \sum_{a_l \in A_{ij}} \frac{1}{d_{il}} \tag{4}$$

$$\text{(C)}\, dwtf(w_i, T_j) = \sum_{a_l \in A_{ij}} \frac{1}{\log(d_{il} + 1)} \tag{5}$$

ここで，A_{ij} はテキスト T_j における語 w_i の出現の集合である．また，d_{il} は出現 a_l における語 w_i とオブジェクト名との距離である．具体的には，2.2 項で抽出した索引語をテキスト中での出現順に並べ，オブジェクト名に近い語から順に 1，2，3... と距離を与えていく．

手法 (A) は，上述の式 (2) における $tf(w_i, T_j)$ を書き直したものである．手法 (B) は重みを増す際に距離の逆数を加えている．また，距離による影響を減らした手法として，手法 (C) は距離の対数の逆数を加えている．

tf の代わりに $dwtf$ を用いた周辺語頻度ベクトルは式 (2) と同様に次のようになる．

$$stf_k(w_i) = \sum_{T_j \in P_k} dwtf(w_i, T_j) \tag{6}$$

後に述べる実験では，周辺テキストにスニペットを用いた場合は手法 (A) を，Web ページ本文を用いる場合は手法 (A)，(B)，(C) をそれぞれ用い，比較している．

2.3.3 $\boldsymbol{idf}$ の重み

$\boldsymbol{idf}$ を求めるため，まず $\boldsymbol{df}$ を求める．$\boldsymbol{df}$ の語 w_i に対する重み $df(w_i)$ は以下のように定義される．

$$df(w_i) = \sum_{O_k \in U} \delta(w_i, O_k) \tag{7}$$

ここで，U はオブジェクト全体の集合である．また，$\delta(w_i, O_k)$ はオブジェクト O_k に対する周辺テキスト集合 P_k に含まれる周辺テキストの何れかに語 w_i が含まれていれば 1，含まれていなければ 0 となるような関数である．言い換えればオブジェクト O_k に対する周辺語頻度ベクトルの語 w_i に対する重み $stf_k(w_i)$ が 0 なら 0，0 より大きければ 1 となるような関数である．

$\boldsymbol{idf}$ の語 w_i に対する重み $idf(w_i)$ は，$\boldsymbol{df}$ を用いて以下のように定義される．

$$idf(w_i) = \log \frac{N}{df(w_i)} \tag{8}$$

ここで，N はオブジェクトの個数，すなわち U の要素数である．

2.3.4 索引ベクトルの正規化

2.1 項で述べたように，Web 検索エンジンを用いてオブジェクト名の出現する Web ページを得るのであるが，オブジェクトの中には少数の Web ページしか得られないものがある．このような周辺テキストのテキスト量の違いによって生じる差異を取り除くため，得られた索引ベクトル $\boldsymbol{iv}$ は長さが 1 になるように正規化する．

2.3.5 tf-idf との共通点と相違点

tf-idf は主に，ベクトル空間モデルにおいて文書の特徴ベクトルを生成するために用いられる手法である．本研究ではオブジェクトの特徴ベクトルを生成するにあたって Web ページ中に出現するオブジェクト名の周辺テキストを用いているため，tf-idf に類似した手法が使えるのではないかと考えた．

tf-idf と異なる点としては，tf をとる際に複数のテキストの出現回数をマージしている点である．これは tf-idf における df にも類似するものであるが，df は 1 つのテキストに同じ語が何度出現しても重みは 1 しか増えないが，本研究では出現回数だけ重みが増える点がやはり異なっている．

また，文書ではなくオブジェクトの特徴ベクトルであることから，オブジェクト名と索引語との距離を重みの計算に用いている点も異なっている．

3. 検索システム

本節では，前節で作成した索引ベクトルを用いてオブジェクト名を検索する手法を述べる．

この検索システムにおける入力と出力は次のようになる．

- 入力: 文章またはキーワード列
- 出力: オブジェクト名のランキング

また，オブジェクト名の検索は以下のような流れで行われる．

（1） ユーザが入力した文章（クエリ）から，質問ベクトルを生成する．

（2） それぞれのオブジェクトに対する索引ベクトルと質問ベクトルとの類似度を計算する．

（3） 類似度に従ってランキングし，ユーザに提示する．

以下では，この各手順について述べる．

3.1 質問ベクトルの生成

まず，2.2 項と同様の手法でクエリから索引語を抽出する．次に，抽出された索引語に対する質問ベクトルの重みを 1 とする．クエリ Q_j に対する質問ベクトル $\boldsymbol{q_j}$ の語 w_i に対する重み $q_j(w_i)$ を式で表すと次のようになる．

$$q_j(w_i) = \delta'(w_i, Q_j) \tag{9}$$

ここで，$\delta'(w_i, Q_j)$ はクエリ Q_j に語 w_i が含まれれば 1，含まれなければ 0 となるような関数である．

3.2 類似度の計算

ベクトルの類似度を測る手法にはコサインを用いる手法や内積を用いる手法がある．本研究では索引ベクトルを作成する際に長さが 1 になるように正規化しているため，どちらの手法でも本質的な違いはない．ここでは内積を用いて類似度を計算することにする．

クエリ Q_j に対する質問ベクトルとオブジェクト O_k に対する索引ベクトルとの類似度 $Sim(Q_j, O_k)$ は次のようになる．

$$Sim(Q_j, O_k) = \sum_{w_i \in W} q_j(w_i) \times iv_k(w_i) \tag{10}$$

ここで，W は索引語全体の集合である．

3.3 ランキング

質問ベクトルとの類似度が高い索引ベクトルを持つオブジェクトほど，ユーザが求めるものである可能性が高いと考えられる．よって，前項で求めた類似度の高いものからオブジェクトをランキングし，ユーザに提示する．

4. 実　　装

本節では，実装上必要な処理や実装上の制限などについて述べる．

4.1 Web ページから取得する周辺テキストの文字数

Web ページ本文から周辺テキストを取得する場合，本来なら 1 つの Web ページ全体を使うべきである．しかし，取得した Web ページの中には 1 メガバイトを超えるものもあり，特に形態素解析器での処理に時間がかかってしまう．このため，周辺テキストをオブジェクト名の前後それぞれ 1000 文字の範囲とした．

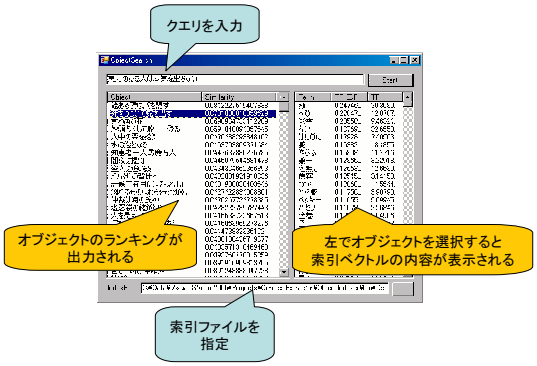

図 1　検索システムの動作例

この制限によって，意味の連続した長い文章の中でオブジェクト名が出現する場合，索引ベクトルに入るべき語を捨ててしまう場合がある．ただし，距離によって重み付けされた索引ベクトルを用いる場合は，オブジェクト名から離れた場所に出現する語は重みが低くなるため，影響は少ないと考えられる．

4.2 索引ベクトルの次元数の制限

索引ベクトルの次元（ランク数）は，特に Web ページから周辺テキストを取得した場合，非常に大きくなってしまうことがある．索引ベクトルの次元が大きくなると，索引ベクトルを保存した索引ファイルのサイズが大きくなり，また 1 度の検索にかかる時間も増えてしまう．

これを防ぐため，最終的な索引ベクトルは重みが大きい方から上位 1000 次元を使うこととする．これは，各オブジェクトに対する索引ベクトルに対して，重みの大きい上位 1000 語以外の語に対する重みを全て 0 にするという意味である．なお，索引ベクトルの正規化は次元を 1000 語にした後に行っている．

4.3 周辺テキストの取得件数

2.1 項において，オブジェクト名の周辺テキストを得るために Web 検索を行っているが，この検索結果のうち何件用いるかが問題となる．

索引ベクトルの作成に用いる周辺テキストの数を増やせば，より多くの用例を使うことが出来る．このため特殊な使われ方のノイズの影響を減らすことにより，より精度の高い索引ベクトルを得ることが出来ると考えられる．しかし，周辺テキストの数が増えればそれだけ索引ベクトルの作成に時間がかかることになる．両者の兼ね合いによる妥当な線として，後に述べる実験では検索結果の上位 100 件の Web ページを索引ベクトルの作成に用いた．

4.4 実装したシステム

2. 節で述べた各オブジェクトに対する索引ベクトルは，1 つの XML ファイルにまとめて書き出す．これを索引ファイルと呼ぶ．また，索引ファイルを読み込み，3. 節で述べた手法でオブジェクトの検索を行うシステムを Windows 上で実装した．このシステムの動作例を図 1 に示す．

これらのシステムは C#で実装した．

表 1　検索タスク

ことわざ	クエリ
溺れるものはわらをもつかむ	困っている人は頼りないものにもすがる
腐っても鯛	実力者は衰えても相応の力がある
棚からぼた餅	努力しないで利益を得る
能ある鷹は爪を隠す	実力のある人は本気を出さない
仏の顔も三度まで	穏やかな人もいつかは怒る
目くそ鼻くそを笑う	大して違わないのに偉そうにする
わざわい転じて福となす	不幸が幸福になることもある
猿も木から落ちる	名人も失敗することがある
石橋を叩いて渡る	慎重すぎるほど慎重に行動する
木を見て森を見ず	一部に気を取られて全体に目がいかない

表 2　検索タスクの結果

ことわざ	(1)	(2)	(3)	(4)
溺れるものはわらをもつかむ	**4**	56	**3**	**3**
腐っても鯛	35	圏外	圏外	圏外
棚からぼた餅	77	61	圏外	圏外
能ある鷹は爪を隠す	**1**	**1**	**4**	**1**
仏の顔も三度まで	**3**	**2**	**1**	**1**
目くそ鼻くそを笑う	圏外	圏外	圏外	圏外
わざわい転じて福となす	**2**	**1**	**1**	**1**
猿も木から落ちる	**4**	**2**	**2**	**2**
石橋を叩いて渡る	**1**	**1**	**1**	**1**
木を見て森を見ず	**1**	**2**	**3**	**2**

(1) スニペット　(2)Web ページ・手法 (A)
(3)Web ページ・手法 (B)　(4)Web ページ・手法 (C)

5. 実　　験

本節では，前節までに述べた手法によって構築したシステムを用いて行った実験とその結果について述べる．

実験の対象としては日本のことわざを用いた．これは，ことわざという定まった「フレーズ」の周囲に現れる語には，その「文脈」としてことわざを特徴付けるような語が出現するのではないかと考えたからである．

5.1　ことわざの集合

本研究の手法を適用するには，まずオブジェクトの集合を与える必要がある．今回の実験では，Web 上で名言やことわざを集めるプロジェクトであるウィキクォート [4] の「日本の諺」の項に掲載されていることわざをベースにした．

5.2　実験の方法

システムの有効性を検証する実験を行うにあたって，以下のような検索タスクを設定した．

まず，前述のウィキクォートの「日本の諺」の項に掲載されていることわざから 10 つ選択する．次に，そのことわざを端的に表す自然言語文をクエリとして設定する．ただし，クエリは検索したいことわざに含まれる語を含まないようにする．これは，後述するようにことわざに含まれる語をクエリに含めると類似度が大きくなるため，公平な評価が出来なくなると考えたためである．

タスクの一覧を表 1 に示す．

検索タスクを 3. 節で構築したシステムで実行し，結果を得る．得られた結果において目的のことわざが何位にランキングされているかを調べる．

索引ベクトルの作成手法としては，スニペットを用い，手法 (A) で重み付けしたもの，Web ページ本文を用い，手法 (A)，(B)，(C) で重み付けしたものを比較する．

また，索引付けすることわざの個数を変え，それぞれについて索引付けにかかる実行時間を計測した．この時間計測では周辺テキストとしてスニペットを用いるものと Web ページ本文を用いるものを比較する．索引ベクトルの重み付け手法は手法 (A) を用いた．手法 (A)，(B)，(C) の違いは，距離をどのように反映させるかだけであるため，索引付けの実行時間にはほとんど影響しないと考えられる．

実験を行った PC のスペックは CPU:PentiumM 1.50GHz, Memory: 760MB である．

5.3　実 験 結 果

設定したタスクを実行した結果を表 2 に示す．表中の数字が検索目的のことわざの順位である．「圏外」となっているものは検索目的のことわざがランキングに現れなかったものである．

表中のラベル (1)〜(4) の意味はそれぞれ次のようになる．スニペットを用いて手法 (A) で重み付けしたものが (1)，Web ページ本文を用いて手法 (A)，(B)，(C) で重み付けしたものがそれぞれ (2)，(3)，(4) である．

また，検索目的のことわざとして「溺れるものはわらをもつかむ」と「猿も木から落ちる」を設定した時の検索結果のランキングにおける上位のことわざと，そのことわざに対する索引ベクトルと質問ベクトルとの類似度を表 3 および表 4 示す．これらの表には，クエリに含まれる索引語のうち，ことわざに対する索引ベクトルでの重みが最も高い語（上位語），すなわち最も類似度に貢献している語とその重みを参考として示す．

索引付けを行うことわざの個数を変化させた時の実行時間をグラフ化したものをを図 2 に示す．横軸は索引付けしたことわざの個数，縦軸は索引付けにかかった実行時間（秒）である．大きくオーダーの異なるものを比較するために縦軸は対数目盛りになっている．

5.4　実験結果の考察

5.4.1　本研究の有効性

表 2 を見ると，7 割のタスクで目的のことわざが上位 5 位以内にランキングされており，本研究で提案した手法の有効性が示されたと言える．

本研究は，うろ覚えの情報から目的の対象を思い出す支援をすることが目的であり，必ずしも目的の対象が 1 位にランキングされる必要はない．5 件程度ならばチェックすることが容易であり，目的の対象が上位 5 件以内にランキングされれば想起支援の目的は達成されたと考えてよい．

5.4.2　失敗例の考察

表 2 を見ると，成功したタスクと失敗したタスクがはっきりと分かれている．この原因の一つとして，クエリの設定の仕方が考えられる．

表 3 「困っている人は頼りないものにもすがる」をクエリとした検索結果

(1) スニペット

	ことわざ	類似度	上位語と重み
1	困った時の神頼み	.2400	困る .2178
2	遠くの親類より近くの他人	.2269	頼り .2058
3	苦しい時の神頼み	.1637	すがる .1188
4	**溺れるものはわらをもつかむ**	.0572	困る .0572
5	渡る世間に鬼はない	.0547	困る .0332

(2) Web ページ・手法 (A)

	ことわざ	類似度	上位語と重み
1	遠くの親類より近くの他人	.1014	頼り .0706
2	困った時の神頼み	.0917	困る .0917
3	爪で拾って箕でこぼす	.0565	困る .0320
4	小姑鬼千匹	.0419	困る .0419
5	弱り目に祟り目	.0417	困る .0417

(3) Web ページ・手法 (B)

	ことわざ	類似度	上位語と重み
1	困った時の神頼み	.2231	困る .1846
2	遠くの親類より近くの他人	.1958	頼り .1796
3	**溺れるものはわらをもつかむ**	.0963	困る .0575
4	苦しい時の神頼み	.0701	困る .0303
5	田作の歯ぎしり	.0680	困る .0680

(4) Web ページ・手法 (C)

	ことわざ	類似度	上位語と重み
1	困った時の神頼み	.1599	困る .1375
2	遠くの親類より近くの他人	.1397	頼り .1106
3	**溺れるものはわらをもつかむ**	.0546	困る .0317
4	苦しい時の神頼み	.0526	困る .0254
5	田作の歯ぎしり	.0486	困る .0486

表 4 「名人も失敗することがある」をクエリとした検索結果

(1) スニペット

	ことわざ	類似度	上位語と重み
1	上手の手から水が漏れる	.2160	失敗 .1201
2	弘法にも筆の誤り	.2148	名人 .1252
3	するのは失敗何もしないのは大失敗	.2116	失敗 .2116
4	**猿も木から落ちる**	.1863	失敗 .1412
5	才子，才に倒れる	.1534	失敗 .1316

(2) Web ページ・手法 (A)

	ことわざ	類似度	上位語と重み
1	するのは失敗何もしないのは大失敗	.1295	失敗 .1295
2	**猿も木から落ちる**	.1050	失敗 .0716
3	下手な鉄砲も数打ちゃ当たる	.0953	失敗 .0953
4	弘法にも筆の誤り	.0875	失敗 .0452
5	巧遅は拙速にしかず	.0864	失敗 .0864

(3) Web ページ・手法 (B)

	ことわざ	類似度	上位語と重み
1	するのは失敗何もしないのは大失敗	.4735	失敗 .4735
2	**猿も木から落ちる**	.1565	失敗 .1101
3	弘法にも筆の誤り	.1498	名人 .1021
4	水火も辞せず	.1435	失敗 .1435
5	上手の手から水が漏れる	.1406	名人 .0753

(4) Web ページ・手法 (C)

	ことわざ	類似度	上位語と重み
1	するのは失敗何もしないのは大失敗	.2377	失敗 .2377
2	**猿も木から落ちる**	.1387	失敗 .0920
3	弘法にも筆の誤り	.1237	名人 .0726
4	上手の手から水が漏れる	.1201	名人 .0602
5	水火も辞せず	.1123	失敗 .1123

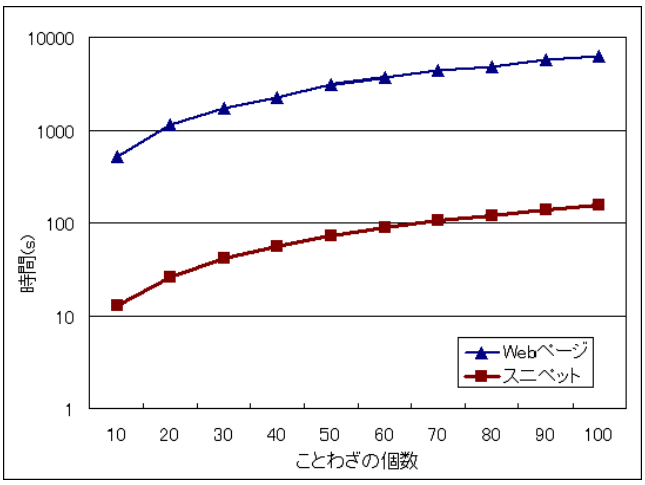

図 2 索引付けにかかる時間のグラフ

例えば,「腐っても鯛」を検索するためのクエリは「実力者は衰えても相応の力がある」であるが，このクエリで検索した結果のランキング上位には「衰える」や「相応」に対する重みが大きい索引ベクトルをもつことわざが多く現れている．一方で「腐っても鯛」に対する索引ベクトルは，これらの語に対する重みが小さいかあるいは 0 のため，ランキングに現れなかったと考えられる．

この問題は，多くの場合クエリを変えることで解決できる．例えば,「優れたものは落ちぶれてもそれなりの価値がある」というクエリで検索すれば「腐っても鯛」がトップにランキングされた結果が得られる．同様に,「棚からぼた餅」を検索する場合は「思いがけない幸運が降ってくる」,「目くそ鼻くそを笑う」を検索する場合は「五十歩百歩」などをクエリにすれば目的のことわざを得ることが出来る．

このように，検索に失敗した場合はクエリを変えることで目的の対象を得られる可能性が高い．しかし，本研究の目的はうろ覚えの状態でも求めるオブジェクトを得ることの出来るシステムを構築することであるので，そもそもユーザはいくつもクエリを入力することは困難であると考えられる．よって，目的の対象を正しく言い表しているクエリならばどのようなクエリでも検索できるべきであるため，索引ベクトルの精度を高めることが今後の課題である．

5.4.3 重み付け手法の比較

2. 節で述べたいくつかの索引付け手法について比較を行う．ここで比較するのは

（1） 周辺テキストとしてスニペットを用い，オブジェクト名との距離を考慮しない手法 (A) で重み付けする手法

（2） 周辺テキストとして Web ページ本文を用い，手法 (A) で重み付けする手法

（3） 周辺テキストとして Web ページ本文を用い，距離の逆数を加算する手法 (B) で重み付けする手法

（4） 周辺テキストとして Web ページ本文を用い，距離の対数の逆数を加算する手法 (C) で重み付けする手法

の 4 つの手法である．

まず，平均して良い結果を出しているのが (4) である．(3) は (4) とほぼ同じ傾向を示しているが,「能ある鷹は爪を隠す」を目的とするタスクなど一部で (3) とランキングの違いが見られた．(3) と (4) の相違点としては，(3) は (4) に比べて一部の語に対する重みが突出して大きい，つまり偏りが大きい点である．このことは表 3 および表 4 において，(4) よりも (3) の方が概して類似度が大きいことからも分かる．

周辺テキストとしてスニペットを用いるか Web ページ本文を用いるかの違いについては，それほど大きな差異は見られなかった．平均して Web ページを用いた方が結果が良いものの,「木を見て森を見ず」を目的とするタスクなど，スニペットを用いた方が良い結果となる場合もあり，一長一短である．ただし，Web ページを用いて距離を考慮しない (2) については「溺れるものはわらをもつかむ」を目的とするタスクなど，他に比べて大きく劣る場合があり，Web ページ本文を用いる場合は距離を考慮すべきであると言える．Web ページを用いる場合は，Web ページ中の HTML タグやスクリプトの処理など，特有のノイズを完全に取り除くことが出来なかったことも，Web ページを用いた手法の精度が上がらない原因の 1 つと考えられる．

5.4.4 上位にランク付けされるノイズの傾向

表 3,4 からも分かるように，ことわざに対する索引ベクトルはことわざの中に含まれる語に対する重みが大きくなる傾向がある．そのため，クエリ中の語を含むことわざが上位にランキングされる場合が多い．この傾向は特に Web ページ本文を用いる場合に強く出る．

表 4 における「弘法にも筆の誤り」など，目的のことわざではなくてもクエリに対して妥当なものが上位にランキングされることもある．一方で,「するのは失敗何もしないのは大失敗」や「水火も辞せず」など，妥当でないものも上位に現れる．この場合，クエリから抽出される索引語のうち，1 つの語のみが索引ベクトルで大きな重みを持っているために上位にランキングされていることが多い．前述の例では「失敗」に対する重みのみでランキングされている．このようなノイズを防ぐためには，クエリから抽出された複数の索引語に対する重みのバランスが良い索引ベクトルを持つものをより上位にランキングするなどの方法が考えられる．

もう一つのノイズのパターンとしては，オブジェクト名を Web 検索エンジンで検索した時の検索ヒット数が少ない場合が挙げられる．表 3 における「田作の歯ぎしり」などがこれにあたる．この場合，周辺テキストの数，すなわちオブジェクト名の用例の数が少ないため，索引ベクトルの精度が低くなってしまうためと考えられる．今回の実験では周辺テキストを 100 件取得しているが，前述の「田作の歯ぎしり」は検索エンジンでの検索ヒット数が 27 件であり，他のことわざに比べて 4 分の 1 程度の周辺テキストから索引ベクトルを作成しているため精度が低くなり，ノイズとなっているのではないかと思われる．

5.4.5 検索結果の意味的な成否判断に関する考察

今回の実験では，先に目的のことわざを決め，それに対してクエリを設定して検索し，何位にランキングされているかで評価するという方法を取った．しかし，前述の表 4 における「弘法にも筆の誤り」など，目的のものではなくても与えたクエリから検索されることわざとして適切なものもある．

本研究の提案手法を適切に評価するためには，あるクエリを与えた時の検索結果において，そのクエリから検索されることわざとして意味的に適切なものがどれだけ上位に来るかを見る必要がある．

しかし，検索されたことわざがクエリに対して適切かどうかを判断するのは基準が曖昧であり難しい．例えば,「名人も失敗することがある」というクエリに対して「才子，才に倒れる」が検索された場合，これを適切ととるかどうかは人によってばらつきがあると思われる．そのため，今回の実験では設定したことわざが何位にランキングされるかという，検索結果の評価において主観が入らない方法を取ったが，今後の課題として意味的な検索結果の評価も検討したいと考えている．

5.4.6 実行時間に関する考察

索引付けを行うことわざの個数を増加させた時，索引付けにかかる実行時間は線形に増加していくことが分かった．

また，周辺テキストとして Web ページ本文を用いたものは，スニペットを用いたものに比べておおむね 50 倍の実行時間がかかることが分かった．2.1 項で述べたように Web アクセス回数は 100 倍だが，実装したシステムでは Web アクセスを並列化しているため，実際には 100 倍の時間がかかるわけではない．しかし，周辺テキスト量の増加による形態素解析等の処理時間の増加も合わせて，実行時間には大きな差異が生じることが確認された．よって，実行時間の点からは Web ページよりもスニペットのほうが圧倒的に優れていると言える．

5.4.7 表記揺れに関する考察

5.4.4 項において,「田作の歯ぎしり」を Web 検索エンジンで検索した時の検索ヒット数が少ないため，索引ベクトルの精度が低くなっていると述べた．オブジェクト名をどのように Web 検索しても少数の Web ページしか得られないこともあるが,「田作の歯ぎしり」の場合は「ごまめの歯ぎしり」や「ごまめのはぎしり」といったクエリを用いて Web 検索を行えば，100 件以上の検索結果を得ることが出来る．

これは一般に「表記揺れ」と呼ばれる問題であるが，今回実装したシステムではこの表記揺れを考慮していない．

表記揺れの問題はオブジェクトを Web 検索エンジンで検索する時だけでなく，索引ベクトルを作成する際にも問題となる．

例えば,「鯨」「くじら」「クジラ」は全て同じ意味であるが，今回のシステムではそれぞれ別の索引語として扱われている．

このような表記揺れを吸収するためには辞書が必要であり，ある程度手間もかかる．また，索引付けにかかる計算時間も増大すると考えられる．表記揺れの影響を完全に取り除くことは困難であるため，手間や計算時間とのトレードオフになるが，今後の課題として検討したいと考えている．

6. 関 連 研 究

曖昧な記憶の想起支援を行う研究はユビキタス・ウェアラブルコンピューティングなどの分野で盛んに行われている [5]〜[7]．これらの分野における想起支援のアプローチは主に，実世界における人間の行動を記録し，その履歴を参照することでユーザの記憶想起を支援するというものである．

また，個人の記憶を三次元球面上に配置するという研究も行われている [8]．この研究では，データ化された個人の記憶を「知球」と呼ばれる三次元球面に配置し，直感的な記憶管理を可能にするというものである．

上記の研究では，いずれも個人の行動や記憶に焦点をあて，それを蓄積・管理することによりユーザに記憶の想起を促すものである．それに対して本研究では，オブジェクト名の周辺テキストを用いてオブジェクトに索引をつけ，それを用いて検索するという手法を用いているため，ユーザによらない汎用的なシステムとなっている．

今回の実験で行った，入力文からことわざを発見する手法の関連研究としては，言い換え表現（パラフレーズ）の抽出に関する研究がある [9], [10]．これらの研究では，シソーラスや言語処理の技術を用いて，ある文と同じ意味を表す別の文を得る手法を提案している．

今回行った実験は，与えた文と同じ意味の表現を発見するという点では同じであるが，これらの研究は慣用表現に限らず一般的な言い換えを発見することを目的としているのに対して，本研究では対象を「ことわざ」という慣用的表現に限定して検索を行えるようにしているという点で異なっている．

また，慣用表現を収集・整理し，慣用表現の同定を行う研究には [11] などがある．

7. まとめと今後の課題

本研究では，オブジェクトに対してあらかじめ索引付けしておくことで，うろ覚えの情報からユーザが求めるオブジェクト名を思い出す支援を行うことの出来る手法を提案した．

具体的には，Web ページ中に出現するオブジェクト名の周辺語を索引語として用い，オブジェクトに対する索引ベクトルを作成することで索引付けを行った．この際，オブジェクト名と索引語との距離を考慮した重み付け手法も提案した．

また，オブジェクトとしてことわざを選び，構築したシステムで実際に索引付けを行った．この索引を用いて検索タスクを実行し，どの程度想起支援が実現できるか実験を行った．この実験では，オブジェクトに対する索引ベクトルを作成する 4 つの手法を比較した．

今後の課題としては，まず索引ベクトルの精度向上が上げられる．今回の実験において失敗したタスクでも成功させることの出来るような索引ベクトルの作成手法の考案が課題となる．クエリを変えて再検索すれば求めるものを得られる場合が多いが，適切なクエリならばどのようなクエリでも検索できるべきであるため，索引ベクトルの精度を向上させる必要がある．その際，オブジェクト名の周辺テキストを全て同等に扱うのではなく，オブジェクトを特徴付けるような部分と無関係な部分を判別して索引ベクトルの重みに反映させるなどの方法が考えられる．また，今回はオブジェクトとしてことわざを用いて実験を行ったが，ことわざだけではなく別のオブジェクトでも実験し，提案手法の汎用性を検証することも今後の課題である．

謝　　辞

本研究の一部は，文部科学省 21 世紀 COE 拠点形成プログラム「知識社会基盤構築のための情報学拠点形成」（リーダー：田中克己，平成 14〜18 年度），文部科学省科学研究費補助金特定領域研究「情報爆発時代に向けた新しい IT 基盤技術の研究」，計画研究「情報爆発時代に対応するコンテンツ融合と操作環境融合に関する研究」（研究代表者：田中克己，A01-00-02，課題番号 18049041），文部科学省研究委託事業「知的資産の電子的な保存・活用を支援するソフトウェア技術基盤の構築」，異メディア・アーカイブの横断的検索・統合ソフトウェア開発（研究代表者：田中克己），および，文部科学省科学研究費補助金特定領域研究「情報爆発時代に向けた新しい IT 基盤技術の研究」，計画研究「情報爆発に対応する新 IT 基盤研究支援プラットフォームの構築」（研究代表者：安達淳，Y00-01，課題番号：18049073）によるものです．ここに記して謝意を表すものとします．

文　　献

[1] Google. http://www.google.com/.
[2] “MeCab: Yet Another Part-of-Speech and Morphological Analyzer”. http://mecab.sourceforge.jp/.
[3] G. Salton: “Automatic Information Organizations and Retrieval”, McGraw-Hill (1968).
[4] ウィキクォート日本語版. http://ja.wikiquote.org/wiki/.
[5] M. Lamming and M. Flynn: “Forget-me-not: Intimate computing in support of human memory”, Proceedings of FRIEND21, International Symposium on Next Generation Human Interface, pp. 125–128 (1994).
[6] B. Rhodes: “The wearable remembrance agent: a system for augmented memory”, Proceedings of The First International Symposium on Wearable Computers, pp. 123–128 (1997).
[7] 河野恭之, 河村竜幸, 上岡隆宏, 村田賢, 浮田宗伯, 木戸出正継：“ウェアラブル日記の実現に向けて－日常記憶の検索・編集・整理・共有機構－”, 電子情報通信学会 パターン認識・メディア理解 (PRMU) 研究会, pp. 55–60 (2003).
[8] 久保田秀和, 角康之, 西田豊明：“知球を用いた個人記憶支援”, 人工知能学会全国大会 (第 19 回) 論文集, 2G1-05 (2005).
[9] 村田真樹, 井佐原均：“同義テキストの照合に基づくパラフレーズに関する知識の自動獲得”, 情報処理学会 自然言語処理研究会 2001-NL-142 (2001).
[10] 木村健司, 徳永健伸, 田中穂積：“漢字インデックスを利用したパラフレーズの抽出”, 情報処理学会研究報告 NL-146-7 (2001).
[11] 田中康仁, 吉田将：“慣用表現について －収集と整理－”, 情報処理学会研究報告「情報学基礎」, No. 39 (1987).