DEWS2007 C7-5

アンカー関連テキストを用いたWebページ分類方式の実装

大坪　正典† BuiQuang Hung† 土方　嘉徳† 西田　正吾†

† 大阪大学大学院 基礎工学研究科
〒 560-8531 大阪府豊中市待兼山町 1 − 3
E-mail: †{otsubo,bqhung}@nishilab.sys.es.osaka-u.ac.jp, ††{hijikata,nishida}@sys.es.osaka-u.ac.jp

あらまし Web 上の情報が増加し続ける中で，Yahoo!や Excite などのような Web ページをカテゴリ分類しているポータルサイトの需要が高まっている．これらのサイトにおける Web ページの分類は，人手で行われてきた．しかし，人手では膨大な量の Web ページを処理することはできないため，Web ページの自動分類が必要となる．近年では，分類対象のページそのものを利用するのではなく，そのページにリンクしているページを利用して分類を行う研究が注目されている．これまでの研究では，ページの形式に関わらず一定のアンカー周辺のテキストを抽出し分類に用いていた．本研究では，ページの形式によってテキストの抽出方法を変えることで，より意味のあるテキスト部分を抽出し，それを分類に用いることを試みる．

キーワード Web ページ分類，アンカー関連テキスト，LSP, USP

Implementing Web Page Classification Method by Anchor-related Text

Masanori OTSUBO†, Bui QUANG HUNG†, Yoshinori HIJIKATA†, and Shogo NISHIDA†

† Graduate School of Engineering Science, Osaka University
1-3 Machikaneyama, Toyonaka, Osaka 560-8531, JAPAN
E-mail: †{otsubo,bqhung}@nishilab.sys.es.osaka-u.ac.jp, ††{hijikata,nishida}@sys.es.osaka-u.ac.jp

Abstract With the exponential growth of information on the Internet, we need to categorize web pages automatically. Many studies extract keywords from web pages to classify them by using the keywords. Recently, they extract keywords not only from a target page which they want to categorize, but also from the pages which link to the target page. However these approaches conduct the same extraction method even if the format of web pages differs. In our research, we change extraction method by the format of web pages in order to adapt each web page.

Key words Web page classification, Anchor-related text, Local Semantic Portion, Upper-level Semantic Portion

1. はじめに

近年，インターネット上の情報は増加の一途をたどっており，2005 年現在 Google が持つデータベースには 80 億もの Web ページが登録されている．情報の増加に伴い，ユーザの要求する検索結果を提供することが難しくなっているのが現状である．膨大な情報の中からユーザが要求する情報を見つけるのは困難であるため，Yahoo!や Excite に代表される Web ディレクトリがよく利用されている．しかしこれらの Web ディレクトリにおける Web ページの分類は人手でなされており，80 億ものページを分類するには限界がある．そこで，Web ページを解析し自動分類しようとする研究が行われてきた．

従来の方法では，分類対象のページ (以下，ターゲットページ) からそのページに関するテキスト部分を抽出し，その中の単語を用いて Web ページを自動分類していた．これは「ターゲットページを最もよく表している単語はターゲットページ中にある」という考えに基づいた手法である．しかしターゲットページ中には，必ずしもそのページを説明するような記述があるとは限らないことが分かっている [1]．例えば，TOYOTA のホームページ (http://toyota.com/) には，画像中の表記で “Cars” や “Trucks” といった単語は存在するものの，テキストで表現された自動車に関する記述はない．また，Aflac(アメリカンファミリー保険会社) のホームページ (http://www.aflac.com/) 中には “insurance” という単語はあるが “health” や “cancer” などの単語がないので，生命保険会社なのか自動車保険会社なのかは分からない．

そこでターゲットページではなく，そのページにリンクしているページ (以下，リンク元ページ) に含まれる単語を用いて分類する方法が近年注目されている．以下に，リンク元ページ中の単語を用いて分類を行う研究事例を挙げる．Bulm ら [2] は，

アンカーテキスト(注1)のみを用いて分類しているが，リンク元ページ全体のテキストを使う場合とその精度はほぼ変わらなかった．また，Chakrabarti ら [3] や Furnkranz ら [4] は，アンカーテキストを含む段落全体やリンク元ページの見出しを使って分類しているが，段落が長くなると分類精度が悪くなっていた．Glover ら [1] は，リンク元ページのアンカー前後 25 単語を拡張アンカーテキストと定義し，アンカーテキスト，拡張アンカーテキスト，全テキストのうち，いずれが有効であるかを比較検討している．彼らは拡張アンカーテキストが最も精度がよいという結果を示した．

しかしこれらの研究は，フォーマットに関わらず常に同じルールでアンカー周辺テキストを抽出しており，リンク元ページのフォーマットが均一であることを前提としている．このため，ブログや掲示板など様々なスタイルで書かれた Web ページに適用すると，無関係なテキスト部分を抽出してしまう可能性が高い．

本研究では，アンカーに関連している可能性の高いテキスト部分のみを抽出し分類に用いることで，より精度の高い自動分類を目指す．実現方法の 1 つとして，Web ページの DOM 構造を見ることが挙げられる．DOM は文書の論理的構造や文書へのアクセス方法，文書部分への操作の方法を定義するもの [5] で，そのオブジェクトの区切りは文脈の区切りとある程度関連があると思われる．我々は多様なフォーマットに適応するために，DOM レベルの文書構造からアンカーに関連するテキスト部分 (以下，アンカー関連テキスト) を抽出することを提案する．

以降，2. 章で提案するアンカー関連テキストとその抽出方法について述べ，3. 章で本研究で用いる SVM について簡単に説明する．また，4. 章で本手法を評価するために実装した分類システムの流れを明示する．5. 章で作成した分類システムによる評価実験の結果を示し，6. 章でまとめと今後の課題を挙げ，本稿を締めくくる．

2. アンカー関連テキスト

2.1 事 前 調 査

アンカー関連テキストの抽出方式を決めるにあたり，事前調査 [6] を行った．調査は，表 1 のように実際の Web ページを数多く調べることで行われた．

	Official Page	Personal Page
ターゲットページ	50	50
リンク元ページ	752	356
合計	1108	

表 1　事前調査の Web ページ数

具体的に述べると，まずターゲットページとして Official Page(注2) と Personal Page(注3) を，それぞれ 50 ページずつ Open Directory [7] からランダムに収集した．これら 100 ページのターゲットページそれぞれに対するリンク元ページを，Google のリンクページ検索 [8] を用いて集めた．この際，集めるリンク元ページの上限を各ターゲットページに対し 20 ページにしている．

このようにして集められたリンク元ページ 1108 ページに対し，どの部位がターゲットページに関する記述があるのか調査した．この調査により，アンカー関連テキストには “Local Semantic Portion(以下「LSP」)” と “Upper-level Semantic Portion(以下「USP」)” の大きく 2 種類があることが分かった．以下，調査結果から決定した，LSP と USP の抽出方法を述べる．

2.2 Local Semantic Portion (LSP)

LSP はアンカーを含むアンカー周辺テキスト部分であり，文書構造上，アンカーノードと同レベルにあるものを指す．以下では，4 つある LSP の抽出方法を紹介する．

2.2.1 アンカーが「段落 $< P >$」内にある場合

アンカーが段落中にあるとき，段落中に改行タグがなければ，段落全体を抽出する．段落中に改行タグがあれば，段落中の改行タグの位置関係を考慮して抽出する．位置関係は，改行タグの直後にアンカーがある場合と改行タグの直後にテキストがある場合の 2 種類がある．改行タグの直後にアンカーがある場合は，アンカーの前にある改行タグから次のアンカーの前にある改行タグまでを抽出する (図 1)．改行タグの直後にテキストがある場合は，前のアンカーの後にある改行タグからアンカーの後にある改行タグまでを抽出する (図 2)．

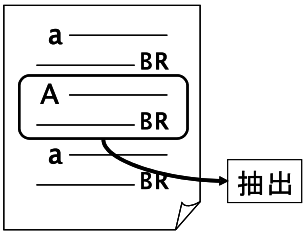

図 1　LSP がアンカーで始まる場合の抽出例

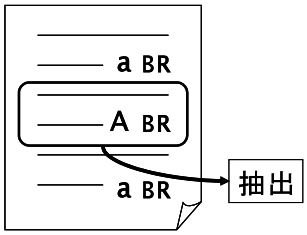

図 2　LSP がテキストで始まる場合の抽出例

ここで，‘A’ はターゲットページのアンカー，‘a’ はその他のアンカー，‘BR’ は改行を表す．また，段落がテーブルのセルに属する場合もあったが，極めて稀にしか登場しないため，考慮しないことにした．

2.2.2 アンカーが「リスト $< OL >, < UL >, < DL >$」内にある場合

アンカーが Ordered List(OL) や Unordered List(UL) のアイテムであるとき，<LI>タグで分けられた項目全体を抽出し (図 3)，Definition List(DL) のアイテムであるとき，<DT>と<DD>で分けられた項目全体を抽出する (図 4)．Ordered List のアイテムの中に，いくつか改行タグがある場合があったが，極めて稀にしか登場しないため，特別な考慮はせず，先ほどと同様に，項目全体を抽出する (図 5)．

(注1)：リンク元ページの中で，ターゲットページへリンクしているテキスト部分のこと．具体的には $< Ahref = "xxx.html" > \bigcirc\bigcirc\bigcirc < /A >$ で示される○○○部分．

(注2)：政府の Web ページのような公的なページのこと

(注3)：自分の趣味などが書かれた個人的な Web ページのこと

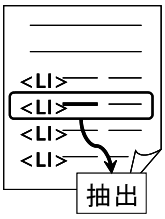

図 3　OL や UL での抽出例

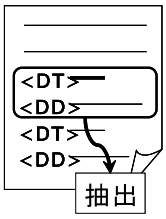

図 4　DL での抽出例

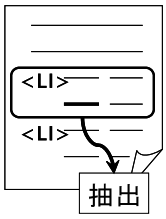

図 5　改行タグを含む場合の抽出例

2.2.3　アンカーが「テーブル $<TABLE>$」内にある場合

アンカーがテーブル内にあるとき，どのくらいのセルに広がっているのか考慮する必要がある．そこで他のアンカーを見つけるまでセルを拡張し，パターンにより抽出範囲を変えることにした．

まずターゲットへのアンカーを含むセルを見つけたら，その左右のセルにその他のアンカーがあるか調べる．その他のアンカーがなければ，更に隣のセルにその他のアンカーがあるかどうかを調べる．もし同じ行のセルにその他のアンカーがなければ，上下のセルへも拡張する．こういった拡張をその他のアンカーが見つかるまで繰り返し，ターゲットページへのアンカーとそれに対応するテキスト部分を抽出する．具体例を図 6 と図 7 に挙げる．'A' はターゲットページのアンカー，'a' はその他のアンカー，'T' はテキストを表す．

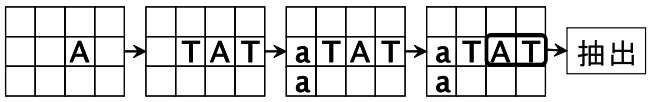

図 6　テーブルでの抽出例

図 6 の場合，2 行 3 列目にアンカーがある．まずアンカーの左右のセルを調べ，どちらもテキストであることが判った．さらにその隣のセルを調べ (端列に到達した場合は次の行へ進む)，他のアンカーを見つけた．ここで，2 行目を見てみると「アンカー→テキスト」の順番である．よって，ターゲットページへのアンカーを含むセルと，その右にあるテキストセルが抽出対象となる．

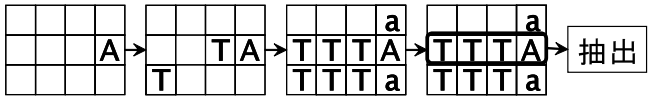

図 7　テーブルでの抽出例

次に，図 7 を見てみると，アンカーは 2 行 4 列目にある．先ほどと同様にその左右のセルを調べ，他のアンカーを見つけるまで，拡張していく．この例の場合は「テキスト→アンカー」の順番になっている．よって，2 行目全てを抽出することになる．

2.2.4　アンカーが「ブロック $<DIV>$」内にある場合

LSP がブロック中にあるときは，2.2.1 節と同様の処理を行う．

2.3　Upper-level Semantic Portion (USP)

USP はアンカーに接していないテキスト部分で，文書構造上，アンカーノードよりも上位レベルに位置するものを指す．以下，USP の抽出方式について述べる．

事前調査の結果より，「ページタイトル」はアンカーに関連していることが判った．また，H1 や H2,…,H6 のような「ヘッダー」もアンカーに関連があり，もし同レベルのヘッダーがいくつかある場合は，アンカーのすぐ上にあるヘッダーがアンカーに関係があるということが判った．図 8 に例を挙げる．この例では<H2>が 2 つあるが，アンカー (<A>) のすぐ上にある方を抽出している．

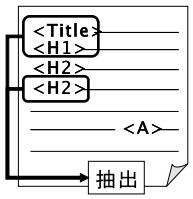

図 8　USP におけるタイトルとヘッダーの抽出例

次にアンカーが「テーブル<TABLE>」内にある場合の抽出方法を述べる．テーブルヘッダーはアンカーと関係があるが，リンク元ページの著者はテーブルヘッダーをあまり使わないことが事前調査で分かった．通常，著者はテーブルヘッダーの代わりに，そのテーブルの 1 行目や，上位レベルのテーブルを用いる．よって，もしテーブルのセルにアンカーがあれば，その最上段もアンカーに関係のあるテキスト部分であると考えられる．

これらの調査より，テーブル内にアンカーがある場合，次のように抽出することとする．

（1）　テーブルヘッダーがあれば抽出する (図 9)

（2）　テーブルの 1 行目が他の段よりも少なければ，1 行目を抽出する (図 10 左)

（3）　テーブルの各行にアンカーがあり，1 行目にのみアンカーがない場合，1 段目を抽出する (図 10 右)

（4）　上記 1〜3 の作業を更に上位レベルのテーブルについて行う．

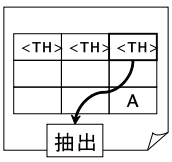

図 9　テーブルヘッダーの抽出例

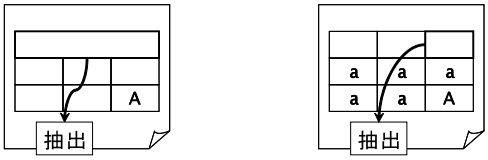

図 10　テーブル 1 段目の抽出例

3. SVM

SVM [9] [10] とは，Support Vector Machine の略で，学習アルゴリズムの 1 つである．学習アルゴリズムは他に Naive Bayes や決定木などが知られているが，SVM は数ある学習アルゴリズムの中でも，最も「高次元に強い」という特徴をもっている．本研究の分類対象は Web 上の文書であり，そこに出現する多様な単語をベクトルの要素とするため，その次元は非常に高いものとなる．よって，本研究には SVM が最も適していると言える．

SVM が行う大まかな流れとして以下のものが挙げられる．

（1） 学習用データの特徴量を SVM に学習させる
（2） マージン最大化を行って，決定境界を引く
（3） 得られた学習モデルにテスト用データをあてる
（4） テスト用データを分類する

この流れは，多くの学習アルゴリズムに共通のものであるが，２つめの項目にある「マージン最大化」は，SVM の最大の特徴ともいえる．マージン最大化とは，学習データの中で最も他のクラスと近い位置にいるもの（サポートベクトル）を基準として，サポートベクトルから他クラスまでのマージンが最大になるような決定境界を設定することである．(図 11)

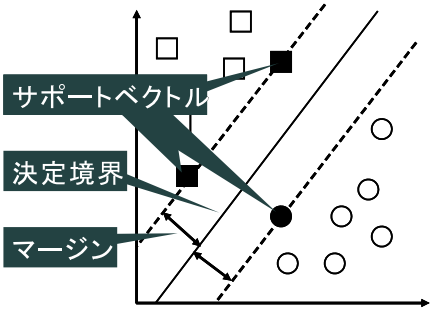

図 11　マージン最大化

SVM ではマージン最大化の他，非線形の分類問題に対して “ソフトマージン” や “カーネルトリック” などの解法が用いられる．ソフトマージンは決定境界からある程度の距離までをグレーゾーンとし，どちらのクラスが属してもよい領域を設ける解法である．また，カーネルトリックは特徴ベクトルに非線形変換を施し，線形分離問題に変換する解法である．

4. 分類システム

本研究で評価実験に用いるシステムの流れを図 12 に示す．本システムは，正ページ (ある分類したいカテゴリに属するページ) とそれ以外の負ページの URL リストを入力データとする．入力データには学習用データとテスト用データが必要で，学習用データの学習結果を元にテスト用データの分類精度・再現率を出力する．以降はこのシステムの流れにそって，実際の実験で行う作業を述べる．

4.1 リンク元ページの収集

ターゲットページの URL のリストが入力されると，Google SOAP Search API (Beta) [11](以下，GoogleAPI) を利用して

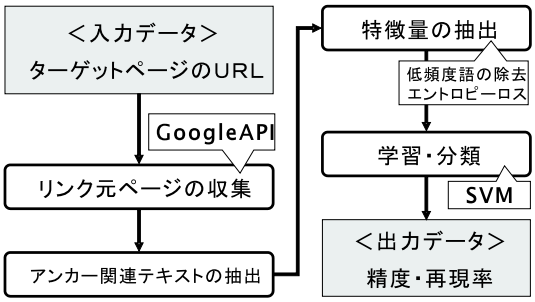

図 12　システムの流れ

それぞれのターゲットページに対するリンク元ページを収集する．この際収集するリンク元ページは，Google が返すリストをそのまま用いることはせず，以下のフィルタを用いて有益なリンク元ページのみを収集する．

このフィルタには，ターゲットページへのアンカーが存在しない Web ページを取得してしまうことを避ける意味 (手順 4a) と，ページ内リンクをしている Web ページや同種の Web ページ(注4) を多数取得してしまうことを避ける意味 (手順 3a,5a) がある．ページ内リンクや同種ページは，Web でよく発生する重要な問題である．

1) ターゲットページ T のドメイン D_T(注5) を抽出
2) GoogleAPI が返すリンク元ページ $X(i)$(注6) のドメイン $D_{X(i)}$ を抽出
3a) $D_T = D_{X(i)}$ ならば，ページ内リンク(注7) とみなし $X_{(i)}$ は**取得せず**，i に 1 を加えて 2) へ
3b) $D_T \neq D_{X(i)}$ ならば，$X(i)$ のソース $S_{X(i)}$ を Google キャッシュ(注8) から取得→ 4) へ (キャッシュを用いる理由：リンク元ページが編集され，ターゲットページへのアンカーが無くなっていることがある．キャッシュを使い編集前の情報を取得すれば，これを避けることができる．)
4a) $S_{X(i)}$ が Google キャッシュに存在しないか，$S_{X(i)}$ 中に D_T の縮小ドメイン (本節末参照)SD_T が無ければ $X(i)$ は**取得せず**，i に 1 を加えて 2) へ
4b) SD_T が $S_{X(i)}$ のキャッシュ中に存在すれば 5) へ
5a) $D_{X(i)} = D_{X(k)}$ ならば $X(i)$ を**取得せず**，i に 1 を加えて 2) へ
5b) $D_{X(i)}$ が $D_{X(1)}$〜$D_{X(i-1)}$ のどれとも一致しなければ $X(i)$ を**取得し**，i に 1 を加えて 2) へ

縮小ドメイン

手順 4a) で登場した縮小ドメインについて詳しく述べる．縮小ドメインとは「ドメイン中の会社名や団体名を抽出したもの」といえる．原則として，“www.” を除いた URL 中の最初のピ

(注4)：BLOG やショッピングサイトなどに多く見られる，日記や商品の内容が変わっただけで，アンカー周辺が全く同一の Web ページを指す．
(注5)：ここでは，URL からサーバ名 (“www” から始まるもの) を除いたドメイン部分のことを指す．例：“http://www.google.com” ならば “google.com” の部分
(注6)：i は GoogleAPI が返す順番
(注7)：ターゲットページからターゲットページへのリンクのこと．“home へ戻る” というリンクであるケースが多い．
(注8)：http://www.google.co.jp/help/features.html#cached を参照

リオドまでの部分である (例 : www.abc.com であれば，abc の部分)．しかし，このルールだけではうまく抽出することはできない．何故ならサーバ名が “www” であるとは限らないからだ．例えば “http://news.abc.com/” のようなページの場合，サーバ名は “news” である．

そこで，サーバ名によく用いられる名前の調査を行った．調査は，Yahoo ディレクトリ [12] の第 3 階層までに登録されている 40,190 ページを対象に行った．調査の結果，10 ページ以上で使われているサーバ名のリストを表 2 に示す．本研究では，これらのサーバ名を除いた後の URL のうち，最初のピリオドまでの部分を縮小ドメインと定義した．

表 2　よく用いられているサーバ名

arts, astro, chem, cs, education, fan, health, home, homepage, honors, info, law, lib, library, math, med, members, music, news, people, pharmacy, physics, psych, sci, science, state, usembassy, users, web

4.2　アンカー関連テキストの抽出

4.1 節で取得したリンク元ページのソースの解析は，“CyberNeko HTML Parser [13](注9)” が生成した DOM 木を探索することで行った．このパーザは，HTML タグの不備を保管する機能(注10)を持っているので，正しく書かれていない HTML ソースにも対応できる．

まず DOM 木の中からターゲットページへのアンカーを探索する．この際，ターゲットページの URL とアンカーの URL を単純に比較すると URL が一致せず，アンカーを発見できない場合が多いことが調査より分かった．ターゲットページの URL を “http://www.abc.com” として以下に URL の単純比較で一致しない例を挙げる．

- http://www.abc.com/index.html
  (同じ URL を示すファイルを記述している)
- http://abc.com
  (サーバ名を省略している)
- http://dinamic.cgi?url=http://www.abc.com
  (他のサイトを経由する)
- http://www.abc.org
  (ドメインの尻尾部分が異なる)

本研究ではアンカーを探索する際，比較する URL を段階的に縮小していくことでこれらの問題を解決することにした．“段階的に” とは以下に示す 5 つの比較手法それぞれに対して，1) から順に DOM 木内の全てのアンカーを探索することを意味する．

1)　ターゲットページの URL との単純比較
2)　“http://” より後方にある ‘/’ 以降を切り捨て比較
3)　2) の手法に加え，サーバ名 (www 等) の除去を行い比較
4)　‘?’ があれば URL 全体からターゲット URL を探索
5)　4.1 節で述べた縮小ドメインを比較

(注9)：使用には以下が必要：Java 1.1 以上，Xerces 2.0.0 [14] 以上
(注10)：例：Web ページの著者が書き忘れた閉じタグを保管。

5) の比較手法は少し強引な比較であるように思えるが，実際は 1)〜4) の比較を終えた後の比較であるため約 95 %の精度があり十分利用可能な比較手法である．また，1)〜5) 全体のアンカー探索精度は約 97 %であった．

以上の手順(注11)を経て得られたターゲットページへのアンカーを元に，2. 章で述べた手法を用いてアンカー関連テキストが抽出される．

4.3　特徴量の抽出

4.2 節の作業で得られたアンカー関連テキストを SVM へ学習させるために，それらの特徴量を抽出する．特徴量の抽出作業は “基底単語リスト作成” と “特徴量抽出” の 2 段階に大きく分けられる．図 13 に特徴量抽出の流れを示す．

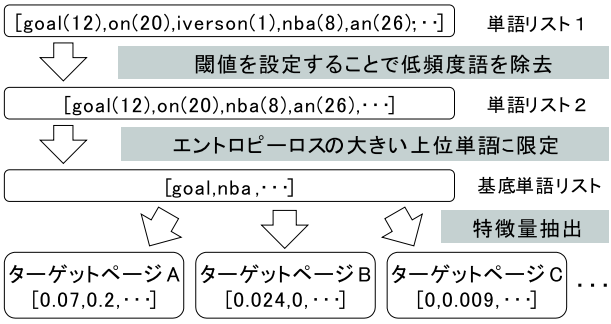

図 13　特徴量の抽出

a)　基底単語リスト作成

まず，基底単語リストの元となる単語リスト 1 を作成する．これは学習に用いる全てのリンク元ページにある，アンカー関連テキスト中の単語を重複なく並べ，各単語の頻度を付加したリストである．次に，単語リスト 1 中の低頻度語を除去し，単語リスト 2 を得る．更にエントロピーロス [1] を計算し，エントロピーロスの大きい方から上位 t 単語を重要単語として，t 次元の基底単語リストを抽出する．この基底単語リストは，SVM の学習・テストの両方に用いる．

b)　特徴量抽出

次に，SVM に読み込ませる特徴量を抽出する．先程とは異なり，各リンク元ページごとに特徴量を抽出し，同じターゲットページへの特徴量を合計する．つまり基底単語リストは全体で 1 つだが，特徴量はターゲットページの数だけ作成される．

具体的な特徴量抽出方法を述べる．まず，リンク元ページ中の基底単語リストに一致する単語の頻度を数える．各単語の頻度をページ中の全単語数で割って正規化し，単語ベクトルを得る．先ほども述べた通り，同じターゲットページへのリンク元ページの単語ベクトルは全て加算する．こうして得られた各ターゲットページに対する単語ベクトルが特徴量となる(注12)．

以下，4.3.1 節で低頻度語を除去する方法を述べ，4.3.2 節

(注11)：4.1 節，4.2 節と同じような作業を行うが，4.1 節は “有益でない Web ページを除く” ことを目的とした作業であり，4.2 節は “必要な部分を見つけ出す” ことを目的とした作業で，主旨が異なる．
(注12)：今回のシステムでは英語ページのみを対象としているが、単語の頻度のみを用いている（意味的要素は用いていない）ため、形態素解析などを用いれば日本語にも対応できる

ではエントロピーロスの定義を説明する．

4.3.1 低頻度語の除去

まず，閾値を設定することにより，低頻度語を除去する．以下の式を満たす単語 f を低頻度語とする．

$$\frac{f\text{を含む正データ}}{\text{正の全データ}} < \tau \quad \cap \quad \frac{f\text{を含む負データ}}{\text{負の全データ}} < \tau$$

ここで τ は閾値である．低頻度語を除去することで，計算コストを下げると同時に分類精度も上がると考えられる．

4.3.2 エントロピーロス

C と f を，

- C : 文書があるカテゴリに属する事象
- f : 文書がある単語を含む事象

とし，$Pr()$ を括弧の中の事象が起きる確率とすると，以下のように定義できる．

- $Pr(C)$: 文書があるカテゴリに属する確率
- $Pr(\bar{C})$: 文書があるカテゴリに属さない確率
- $Pr(f)$: 文書がある単語を含む確率
- $Pr(\bar{f})$: 文書がある単語を含まない確率

これらは，以下のように求められる．

$$Pr(C) = \frac{\text{正のデータ数}}{\text{全データ数}}$$

$$Pr(\bar{C}) = \frac{\text{負のデータ数}}{\text{全データ数}} = 1 - Pr(C)$$

$$Pr(f) = \frac{f\text{を含むデータ数}}{\text{全データ数}}$$

$$Pr(\bar{f}) = \frac{f\text{を含まないデータ数}}{\text{全データ数}} = 1 - Pr(f)$$

これらを定義したとき，各単語の事前エントロピーは次のように計算される．

$$e = -Pr(C)\log Pr(C) - Pr(\bar{C})\log Pr(\bar{C}) \qquad (1)$$

一方，$Pr(C|f), Pr(\bar{C}|f), Pr(C|\bar{f}), Pr(\bar{C}|\bar{f})$ が

$$Pr(C|f) = \frac{f\text{を含む正データ}}{f\text{を含むデータ}}$$

$$Pr(\bar{C}|f) = \frac{f\text{を含む負データ}}{f\text{を含むデータ}}$$

$$Pr(C|\bar{f}) = \frac{f\text{を含まない正データ}}{f\text{を含まないデータ}}$$

$$Pr(\bar{C}|\bar{f}) = \frac{f\text{を含まない負データ}}{f\text{を含まないデータ}}$$

により計算されるとすると，また，f が含まれるときの事後エントロピー e_f と，含まれないときの事後エントロピー $e_{\bar{f}}$ は次のように計算される．

$$e_f = -Pr(C|f)\log Pr(C|f) - Pr(\bar{C}|f)\log Pr(\bar{C}|f)$$

$$e_{\bar{f}} = -Pr(C|\bar{f})\log Pr(C|\bar{f}) - Pr(\bar{C}|\bar{f})\log Pr(\bar{C}|\bar{f})$$

よって，事後エントロピーは

$$\acute{e} = e_f Pr(f) + e_{\bar{f}} Pr(\bar{f}) \qquad (2)$$

となり，式 (1) と式 (2) より，エントロピーロスは

$$E = (\text{事前エントロピー}) - (\text{事後エントロピー})$$
$$= e - \acute{e}$$

で求められる．

エントロピーロスは，正データ・負データの両方に頻出するような単語に対しては，小さな値となる．つまり，4.3 節で述べた「大きいものから順に並べ，上位単語に絞る」というのは「正データにも負データにもよく登場する単語は，分類に役立たない」という考えが元になっている．

低頻度語の除去でめったに出てこない語を除去し，エントロピーロスの小さいものを除くことでストップワードのような頻出しすぎる単語も除去した．この 2 段階のベクトル次元削減により，分類精度は向上すると考えられる．

4.4 SVM による学習・分類

以上の過程を経て得られた特徴量を用いて，学習・分類を行う．ただし，テスト時には特徴量の抽出のときに基底単語リストの作成は行わず，学習時に作成した基底単語リストを用いる．学習・分類には SVM を用いるが，本システムでは “LIBSVM [15]” を用いた．

5. 評 価 実 験

4. 章で述べたシステムを用いて，評価実験を行った．本章では，実験に用いたデータセットと実験条件，実験結果を述べる．

5.1 データセット

実験に用いる正のターゲットページとして，Yahoo! Directory [12] の以下に示すカテゴリ下から Web ページを収集した (カッコ内は集めたページ数)．

データ 1　Science/Biology (634)
データ 2　Science/Biology/Zoology/Animals (1594)
データ 3　Entertainment/Music (757)
データ 4　Entertainment/Music/Genres/Rock&Pop (634)
データ 5　Recreation/Sports (1090)
データ 6　Recreation/Sports/Baseball/MajorLeague (713)
データ 7　Computers/Internet (953)
データ 8　Computers/Internet/WWW/Weblogs (698)

収集方針は「同じジャンルのカテゴリを 2 つずつ計 4 組収集し，データ 1,3,5,7 は深さ 2 から，データ 2,4,6,8 は深さ 4 から収集すること」である．ジャンルによって分類精度に差が出るか，カテゴリの深さによって分類精度に差が出るか，を確かめることができるように収集した．また，負のターゲットページとして，Yahoo! Directory 全体から上記のカテゴリ避けて 4942 ページを収集した．

次に Google SOAP Search API (Beta) [11] を用いて，収集した 9 データ (データ 1〜8 ＋負データ) それぞれのターゲットページ (約 12 千ページ) に対するリンク元ページ (約 200 万ページ) を収集した．

5.2 実 験 条 件

今回の実験では，5.1 節で述べた，収集したターゲットペー

ジから，それぞれランダムに 150 ページを選び，50 ページを学習用データ，100 ページをテスト用データとして用いた．またリンク元ページは，各ターゲットページに対し最大 50 ページとした．

4.3.1 節で述べた低頻度語除去の閾値は 0.07 とし，4.3.2 節で述べたエントロピーロスの大きい上位 1000 単語を基底単語リストとした．これらは Glover ら [1] が実験を行った時と同じ値である．

5.3 実験結果

提案手法は，5.1 節で述べたリンク元ページから「アンカー関連テキスト」を抽出し分類に用いている．提案手法を以下「STP」と呼ぶ．また，Glover ら [1] が行った「アンカー前後 25 単語」を用いて分類する手法を比較手法とする．彼らの手法は，既存の Web ページ分類手法の中で，最も良い分類精度を出している．比較手法を以下「Fix」と呼ぶ．

図 14 に，各データごとの分類 F 値とその平均を示す．ここで，F 値は (2 × 精度 × 再現率)/(精度＋再現率) である (精度は「システムによって正に分類されたページの内，正のターゲットページはいくらか」を示し，再現率は「正のターゲットページの内，システムによって正に分類されたページはいくらか」を示す)．

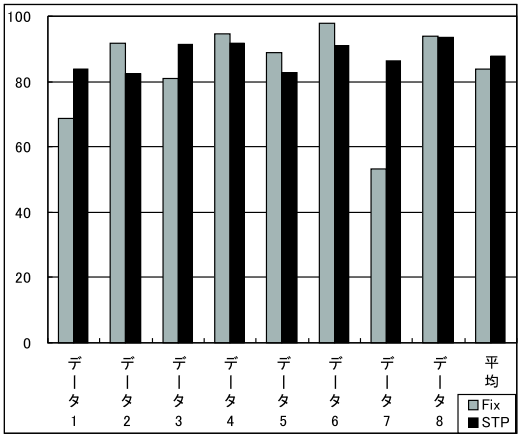

図 14　データ別分類結果 - F 値

STP に関してはどのデータについても F 値 80 以上であるが，Fix についてはデータによってばらつきがあることが見受けられる．さらに細かく見ていくと，Fix の結果について，データ 1 よりもデータ 2，データ 3 よりもデータ 4，といった具合に，データ 1,3,5,7 よりもデータ 2,4,6,8 の方が F 値が良いことが分かった．5.1 節でも述べたが，データ 1,3,5,7 はカテゴリの深さが 2，データ 2,4,6,8 はカテゴリの深さが 4 である．このことから「Fix は STP に比べ浅いカテゴリの分類が苦手である」という知見が得られた．

この理由を検討してみる．図 15 に STP(LSP,USP) と Fix(前後 25 単語) の抽出範囲例を示す．この図から分かるように，Fix は LSP よりも大きい範囲で抽出することが多い．つまり Fix は，最も関連している最小部分 (LSP) の前後テキストも抽出するので，この前後テキスト部分にどれだけのノイズが入り込むかが分類精度に影響を及ぼす．

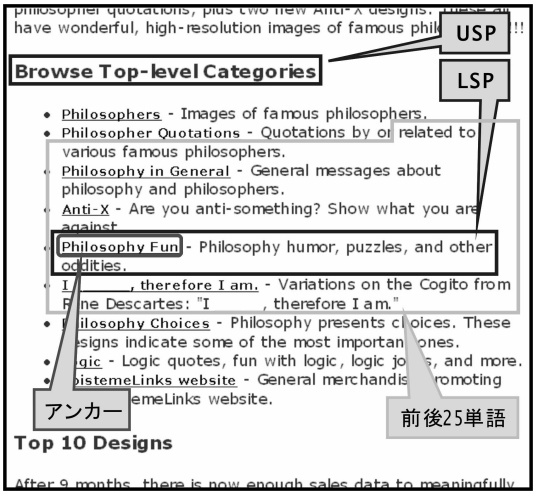

図 15　抽出範囲

アンカーが “深いカテゴリに属するターゲットページ” を指している場合，LSP の前後テキストは，ターゲットページの内容に比較的近いものであるが，逆に “浅いカテゴリに属するターゲットページ” を指している場合，LSP の前後には，ターゲットページの内容から遠いテキストが含まれると考えられる．

例えば，ターゲットページが階層的に深い “Jazz に関するページ” であった場合，その前後のリンクは Pops であったり Rock であったりするだろう．しかし，ターゲットページが階層的に浅い “Music 全般に関するページ” であった場合，その前後のリンクは Movie であったり Game であったりする．

全てがこのような性質を持っているとは限らないが，こうした傾向があるといえるだろう．これらのことから提案手法は，カテゴリ階層が深いターゲットページにも浅いターゲットページにも安定した分類を行うことができるといえる．

5.4 詳細実験

次に，STP についてより詳細に実験を行う．2.1 節でも述べた通り，STP は LSP と USP からなるが，分類にどちらが大きく寄与しているのかを確認する．

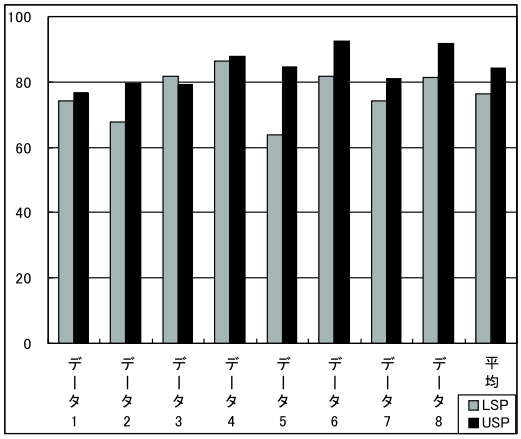

図 16　LSP/USP 別 分類結果 - F 値

図 16 に LSP/USP を分けて抽出し分類した結果を示す．デー

タ 3 では逆転しているものの，LSP よりも USP の方が分類精度が良いといえる．この理由として，USP が “Title” や “Header” など，ノイズの入りにくい部分を抽出しているのに対し，LSP がノイズを入れずに抽出するのが比較的困難である部分から抽出していることが考えられる．

ではそのノイズが，LSP 抽出手法 (2.2 節参照) の内，どの抽出部分に含まれているのかを検証する．今回は LSP 抽出手法を以下の 5 つに分割した．

- DIV : アンカーがブロック内にある場合
- TBL : アンカーがテーブル内にある場合
- DLI : アンカーが定義リスト内にある場合
- LST : アンカーがリスト内にある場合
- PGF : アンカーが段落内にある場合

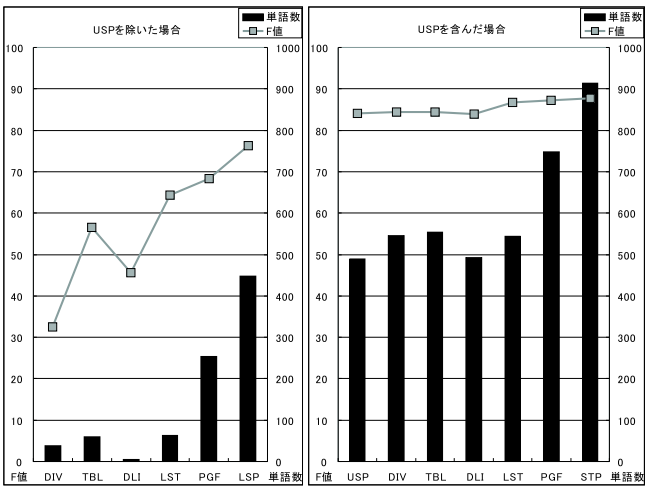

図 17　LSP 詳細 分類結果

図 17(左) に，LSP の各抽出手法のみを用いた場合の分類結果を示す．“LSP” は 5 つの抽出手法を全て使った場合の分類結果である (DIV〜PGF までの単語数を全て足し合わせると LSP の単語数に等しくなる)．この図を見ると，LSP 抽出手法の違いが分類精度に与える影響よりも，DIV や DLI に見られるように，分類に用いる単語量の違いが分類精度に与える影響の方が大きいことが分かる．

そこで今度は，LSP の各抽出手法に USP を加えることで最低単語数を確保し，その分類精度を検証してみる．その結果が図 17(右) である．DLI が僅かながら分類精度を下げているものの，基本的に LSP 抽出手法の内どの手法も，分類精度を良くすることはあっても悪影響を及ぼすことはないことが分かる．

本節の詳細実験を通して得られた知見は以下の通りである．

- LSP と USP を分けて抽出し，それぞれの分類精度を比較した場合，USP の方が良い精度で分類ができる．
- LSP 抽出手法の中で悪影響を及ぼしているものはない．

これらのことから，Web ページ分類において「分類に用いる単語を抽出する場合，ノイズが入らないように抽出することも重要であるが，なるべく単語量を多めに保つことも重要である」ということがいえるのではないか．より詳しい検証が必要ではあるが，このことは “SVM を用いて 2 値に分類する” という Web ページ分類特有の特徴だといえるだろう．

6. おわりに

本稿では，アンカー関連テキストを用いた Web ページ分類手法を提案し，アンカー関連テキストの定義とその抽出手法を述べた．アンカー関連テキストには LSP と USP の 2 種類があり，それぞれ事前調査に基づき抽出方法を決定している．また，リンク元ページの収集方法とページ中でのアンカー部分の特定方法について述べ，抽出したアンカー関連テキストから特徴量を計算する方法を示した．

評価実験より，Fix 手法は浅いカテゴリに属するページの分類が苦手であるのに対し，提案手法は安定した分類ができることが分かった．提案手法の LSP/USP を分けて分類に用いた場合，USP の方が分類精度が良いことが示されたが，LSP が分類に悪影響を与えている訳ではない．

今後は，5.4 節でも述べたが “単語量と分類精度の関係” をより詳細に調べてみる必要がある．また 5.2 節で述べた “低頻度除去のための閾値” と “エントロピーロスの上位単語に絞る際の単語数” を変化させた場合に，分類精度がどのように変化するのかを検証したい．

文　献

[1] Eric J.Glover, Kostas Tsioutsiouliklis, Steve Lawrence, David M.Pennock, Gary W.Flake: Using Web Structure for Classifying and Describing Web Pages, WWW2002, pages 562-569, Hawaii, USA, 2002.

[2] Avrim Blum, Tom Mitchell: Combining Labeled and Unlabeld Data with Co-Training, COLT98, pages 92-100, Madison, USA, 1998.

[3] Soumen Chakrabarti, Byron Dom, Piotr Indyk: Enhanced hypertext categorization using hyperlinks, SIGMOD’98, pages 307-318, Seattle, USA, 1998.

[4] Johannes Furnkranz: Exploiting Structural Information for Text Classification on the WWW, IDA’99, pages 487-497, Berlin, Germany, 1999.

[5] World Wide Web Consortium: Document Object Model (DOM) Level 1 Specification Version 1.0, 1998 http://www.w3.org/TR/REC-DOM-Level-1/

[6] Bui Quang Hung, Masanori Otsubo, Yoshinori Hijikata, Shogo Nishida: Extraction of Semantic Text Portion Related to Anchor Link, IEICE Transactions on Information and Systems, pages 1834-1847, VOL.E89D, NO.6 JUNE 2006.

[7] Netscape: dmoz - Open Directory Project (ODP), http://dmoz.org/

[8] Google Advanced Search, http://www.google.com/advanced_search

[9] 栗田 多喜夫: サポートベクターマシン入門, http://www.neurosci.aist.go.jp/~kurita/lecture/svm/svm.html

[10] 石田貴士: Support Vector Machine, http://www.bi.a.u-tokyo.ac.jp/~tak/svm.html

[11] Google SOAP Search API (Beta), http://code.google.com/apis/soapsearch/

[12] Yahoo! Directory, http://dir.yahoo.com/

[13] Andy Clark: CyberNeko HTML Parser, http://people.apache.org/~andyc/neko/doc/html/

[14] The Apache Software Foundation: Xerces2 Java Parser, http://xerces.apache.org/xerces2-j/

[15] Chih-Chung Chang, Chih-Jen Lin: LIBSVM – A Library for Support Vector Machines, http://www.csie.ntu.edu.tw/~cjlin/libsvm/