情報処理学会論文誌 Vol.54 No.12 2502–2512 (Dec. 2013)

# ブログページ集合からのポストおよびコメント自動分離抽出手法

吉田 光男[1,a)] 乾 孝司[1] 山本 幹雄[1]

受付日 2013年4月13日, 採録日 2013年9月13日

**概要**：ブログページには，Web 検索エンジンなど機械的にページを処理するシステムにおいてノイズになる部分が含まれる．そのため，ブログのコンテンツを利用するためには，コンテンツの抽出処理が必要になる．さらに，ブログのコンテンツは，ポストと呼ばれるブログの書き手によるコンテンツと，コメントと呼ばれるブログの読み手によるコンテンツに二分できる．ポストとコメントの存在はブログの特性の 1 つであり，ブログの特性を活用するシステムや研究では，ポストおよびコメントを別々に抽出できていることが望ましい．本論文では，ブログページ集合を用いることにより，ポストとコメントを自動的に分離抽出する手法を提案する．複数のブログ記事ページを含むあるブログサイトにおいて，ポストはすべての記事ページに出現するが，コメントはいずれかの記事ページにしか出現しないという点に着目し考案した．また，本手法のアルゴリズムを実装したソフトウェアを用いて実験を行い，日本語ブログサイトに対しての有効性を検証し，コンテンツをポストおよびコメントに分離できることを確認した．

**キーワード**：コンテンツ抽出，ブログ，HTML，要素識別子

# Automatic Extraction of Blog Posts and Comments from Blog Pages

Mitsuo Yoshida[1,a)] Takashi Inui[1] Mikio Yamamoto[1]

Received: April 13, 2013, Accepted: September 13, 2013

***Abstract:*** Content extraction is necessary to use blogs as data for Web search engines, because blog pages are excessively added noisy parts such as menus, advertisements and copyright notices. Most of the blog contents are texts, and those can be divided in two parts, posts and comments. A post is a content written by the blog owner and a comment is piece of text written by readers in response to the owner's post. In this paper, we propose a simple method to extract the posts and comments separately from series of blog pages, whose posts are all written by the same owner. The proposed method is based on the assumption that although posts appear in all blog pages, comments do not. We describe experimental results to show good performance of the proposed method using real Web pages of the blog sites in Japanese.

***Keywords:*** content extraction, blog, HTML, element identifiers

## 1. はじめに

インターネット技術の発展とともに Web ページを作成する障壁が低くなり，様々なユーザが Web サイトを持つようになった．その結果，インターネット上には大量の情報があふれている．近年の Web ページの増加は，ブログの普及に一因がある．日本では，2004 年から 2005 年ごろにかけてブログおよびその記事ページが急増しており，現在も増加傾向が見られる [1]．ブログの普及にともない，ブログのコンテンツを利用する研究もさかんになってきている [2], [3]．

ブログに限らないが，Web ページには，ヘッダ，メニュー，広告，関連記事リストなど，そのページがまさに伝えたい情報（コンテンツ）以外の情報（不要部分）が含まれる．コンテンツ以外の不要部分は，Web 検索エンジンなど機械的

[1] 筑波大学大学院システム情報工学研究科
Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba, Ibaraki 305–8573, Japan
[a)] ceekz@mibel.cs.tsukuba.ac.jp

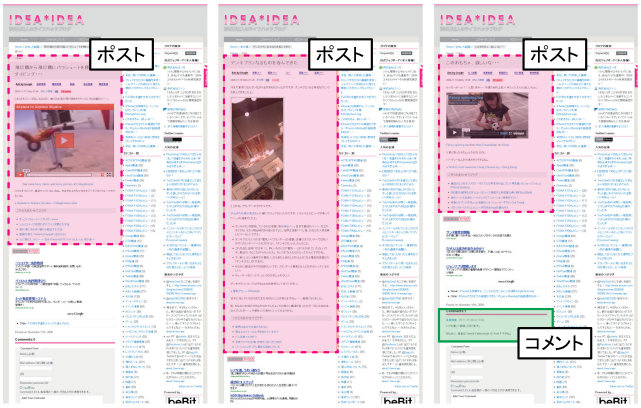



図 1 ブログ記事ページにはポストとコメントが存在する
**Fig. 1** An example of blog pages with posts and comments.

に Web ページを処理するシステムにおいてノイズになることが多い．そのため，ブログをはじめとする Web ページのコンテンツを利用するためには，コンテンツの抽出処理が必要になる．たとえば，Yi ら [4] は，Web ニュース記事ページのクラスタリングを行う際，事前にコンテンツ抽出を行うことにより，性能が大幅に向上することを報告している．

ブログのコンテンツは，ポストと呼ばれるブログの書き手によるコンテンツと，コメントと呼ばれるブログの読み手によるコンテンツに二分できる．たとえば，**図 1** に示す[*1]右のブログ記事ページには，上部にポスト，下部にコメントが存在する．従来のコンテンツ抽出（本文抽出，主要部分抽出などとも呼ばれる）は，ポストとコメントを区別せずに抽出したり，ポストのみを抽出したりしていた．しかし，ポストとコメントの存在はブログの特性の 1 つであり，ブログのコメントを利用する研究が行われ始めるなど，ブログのコンテンツ抽出はポストとコメントを分離抽出することが期待される．

本論文では，ブログページ集合を用いることにより，事前に学習データを準備する必要のないアルゴリズムで，ブログ記事ページからポストとコメントを自動的に分離抽出する手法を提案する．複数のブログ記事ページを含むあるブログサイトにおいて，図 1 のように，ポストはすべての記事ページに出現するが，コメントはいずれかの記事ページにしか出現しないという点に着目し考案した．

以降，2 章で関連研究について，3 章でポスト・コメントの定義および提案手法について，4 章で評価指標および日本語ブログサイトを対象とした実験結果について，最後の 5 章でまとめを述べる．

## 2. 関連研究

ポストとコメントを分離抽出する手法としては，まず，人手によって抽出ルールを記述する方法が考えられる．この方法の代表例として，正規表現や XPath [*2]があげられる．コンテンツ抽出を行う際は，ブログ記事ページごとにレイアウトが異なることに留意する必要がある．同一のブログホスティングサービスを利用している記事ページに関していえば，ポストやコメントの位置が同一であると考えられるため，ブログホスティングサービスごとに抽出ルールを準備する必要があるものの，人手によって抽出ルールを記述する方法は有効に機能する．しかし，筆者らによる予備調査では，人気のあるブログサイトの少なくとも 30%がブログホスティングサービスを利用していないことが分かった[*3]．このような状況下では，人手によって抽出ルールを記述する方法には多大な労力が必要となることは明らかであり，人手の負荷を要しない自動抽出手法を検討する必要がある．

Web ページのコンテンツを自動的に抽出する手法は数多く提案されている（文献 [5], [6], [7], [8], [9] ほか）．しかし，ブログ記事ページに適用するとなると，これらの手法はポストとコメントを区別せずに抽出することになる．これらの手法に対し，ブログのポストまたはコメントのみを抽出する手法も提案されている．Chang ら [10] は，Weninger ら [7] のコンテンツ抽出手法を改良したうえで，“comment” という文字列が出現する位置より後方のテキストを除外することでポストの抽出を行う手法を提案している．Kao ら [11] は，コメント部分は複数の投稿で構成されることに着目し，HTML の繰返しパターンとそのパターンに含まれる情報を用いた機械学習アプローチによってコメントを抽出する手法を提案している．Song ら [12] は，ブログサイトのトップページのような複数の記事が含まれるページから，それぞれの記事を抽出する手法を提案し，コメントの抽出に応用した結果も報告している．

本論文では，ポストとコメントの双方を活用するシステムや研究を想定し，ポストとコメントを同時に自動抽出する問題に取り組む．たとえば，宮田ら [13] はポストとコメントを活用する課題に取り組み，goo ブログ[*4]のブログサイトのみで評価を行っている．これは，ポストとコメントを準備する労力（抽出ルールの記述）による制約であると考えられ，ポストとコメントを同時に自動抽出できれば，ブログホスティングサービスを限定せず，より大規模な評価ができる可能性がある．このほかにも，Bhattarai ら [14] はブログに投稿されたコメントスパムに関する分析を行い，ポストとコメントとの類似度を用いたコメントスパム検出手法を提案している．Li ら [15] はブログ記事ページをクラスタリングするタスクにおいて，ポストとコメントを分割し，コメントにウェイトをかけることでクラスタリン

[*1] IDEA*IDEA（http://www.ideaxidea.com/）の 3 記事.
[*2] XML Path Language：XML 文書の特定部分を表現する言語.
[*3] livedoor Reader（http://reader.livedoor.com/）登録者数ランキング上位 1,000 サイトから推定した.
[*4] http://blog.goo.ne.jp/

グ精度が向上することを報告している．Parapar ら [16] と Hu ら [17] はブログのコメントに現れるポストに関するトピックなどをもとにポストを要約する手法を提案し，Ma ら [18] はニュース記事本文（ブログのポストに相当）のトピックなどをもとにコメントを要約する手法を提案している．なお，いずれの研究においても，ポストおよびコメントの準備は，その方法を明示しないか抽出ルールの記述を行うかしている．

Cao ら [19] は，ポストとコメントを分断する境界線を見つけることにより，ポストとコメントを自動的に分離抽出する手法を提案している．この手法は，視覚情報とテキスト情報をもとにブログ記事ページのコンテンツを特定する処理と，情報量をもとにコンテンツの中でポストとコメントを分断する境界線を見つける処理の 2 段階に分けられる．彼らの手法は，コンテンツを 1 カ所に特定するため，コンテンツの中に不要部分を多く含む傾向がある．また，コンテンツを特定する際，HTML ファイルのほかに，画像ファイルや外部スタイル情報（スタイルシート：Cascading Style Sheets）も必要になるため，情報検索システムなど，他のシステムに組み込むのは困難であると考えられる．

ポストとコメントを同時に自動抽出する手法は，筆者らが調査した限り，Cao ら以外に提案されていない．既存のポスト抽出手法とコメント抽出手法を組み合わせることにより，ポストおよびコメントを同時に抽出する方法が考えられるが，自動抽出が不完全であるがゆえに，ポスト抽出手法の結果とコメント抽出手法の結果とが一部重複する状況が考えられる．そのため，この重複部分をポストまたはコメントのいずれと見なすかの処理そのものが，ポストおよびコメントの分離抽出手法となりうる．また，コンテンツ抽出手法とコメント抽出手法を組み合わせた場合においても，コメント抽出手法でのみ抽出できた部分を，コメントと見なすか，不要部分と見なすかの判断の余地が残る．ほかにも，それぞれの抽出手法において，コンテンツ，ポスト，コメントの最小単位が異なる点も手法の組合せを困難にする要因である．たとえば，Cao ら [19] が提案する処理の一部（コンテンツ特定処理）を後述する吉田ら [8] が提案する手法に置き換えると，ポストとコメントを分断する境界線を見つける処理に必要な情報が欠けてしまい，ポストおよびコメントに分離できなくなる．

本論文では，吉田らが提案したコンテンツ自動抽出手法 [8] を拡張し，ポストとコメントを同時に自動抽出する手法を提案する．吉田らが提案したコンテンツ自動抽出手法は，HTML のブロックレベル要素をもとに定義した，コンテンツと不要部分のかたまり（ブロック）を単位として考え，ブロックごとにコンテンツ判定を行うため，Cao らのコンテンツ抽出手法よりも，コンテンツと不要部分を細かい粒度でかつ高精度に分離できるという特徴がある．また，HTML ファイルをテキスト処理するだけで抽出できるため，他のシステムへの組み込みが容易である．

## 3. 提案手法

### 3.1 コンテンツおよびポスト・コメントの定義

本論文では，ブログ記事ページの不要部分を除いた主要部分を**コンテンツ**とし，コンテンツをさらに**ポスト**と**コメント**に二分する．ポストは記事本文のほか，それに付随する記事タイトル，投稿日時，著者名，写真・図およびその説明文を指す．コメントは読者によるコメント本文およびトラックバック本文のほか，それらに付随するタイトル，投稿日時，コメント著者名（トラックバック送信元ブログ名）を指す．なお，単にコンテンツと記した場合は，ポストとコメントを区別しない．

### 3.2 ポスト・コメント自動分離抽出手法の概要

本論文で提案するポストとコメントを自動的に分離抽出する手法（以下，**提案手法**と呼ぶ）は，吉田らが提案したコンテンツ自動抽出手法 [8]（以下，**ExtractUniqueBlock 法**と呼ぶ）の拡張である．ExtractUniqueBlock 法は，ある Web ページのコンテンツは他の Web ページに出現しないという仮定をもとにした手法である．複数の HTML ファイルを与えさえすれば，抽出ルールを必要とせずにコンテンツを抽出できる．

今回，筆者らは，図 1 のように，複数のブログ記事ページを含むあるブログサイトにおいて，ポストはすべての記事ページに出現する一方で，コメントは一部の記事ページにしか出現しない傾向があることに着目した．この傾向を利用することで，ブロックの役割を識別して，すべての記事ページにおいて同じ役割を持つコンテンツとして認められるブロックをポストとして抽出し，残りのコンテンツとして認められるブロックをコメントとして抽出すれば，ポストとコメントを同時に抽出できるという仮説を立てた．

提案手法の概観を**図 2** に示す．ExtractUniqueBlock 法は「コンテンツ抽出」を行う手法であるが，提案手法は，続いて，コンテンツと認められたブロックのページ内で

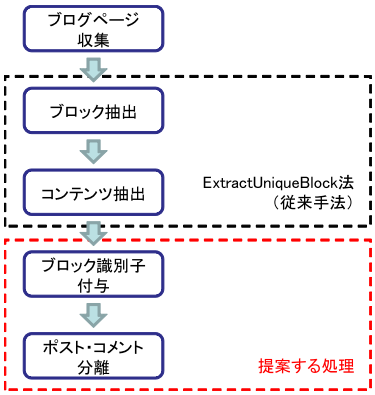



図 2 提案手法の概観
**Fig. 2** The overview of the proposed method.

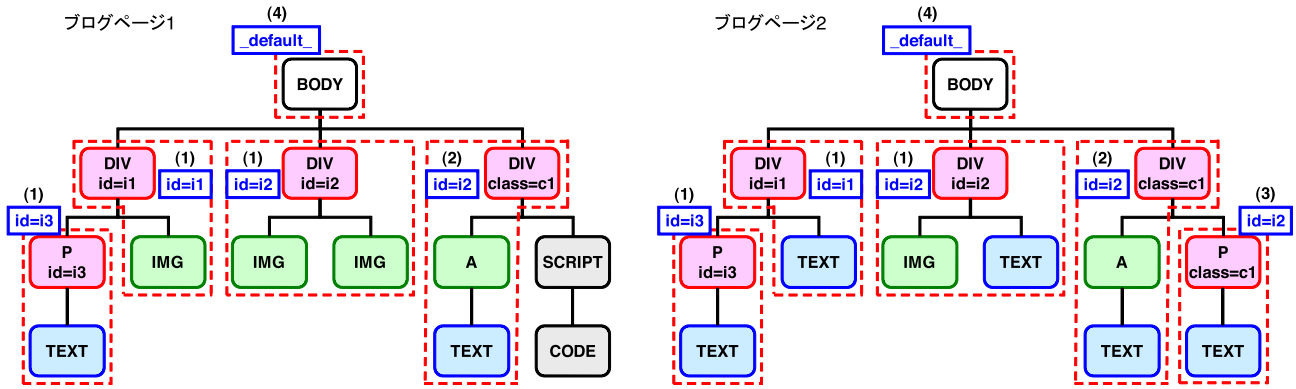



図 3 各ブロックにブロック識別子を付与した例
**Fig. 3** An example of blocks with block identifiers.

の役割を識別するために「ブロック識別子付与」を行い，先に述べた仮説をもとに「ポスト・コメント分離」を行う．ExtractUniqueBlock 法は，特定の言語に依存せず，W3C [*5]が定義する HTML タグの分類を利用してコンテンツ抽出を行う．提案手法による「ブロック識別子付与」においても，言語依存となる手がかり語（“ポスト”，“コメント”，“リプライ”，など）を使用せず，W3C によって定義されている情報のみを利用する．そのため，抽出ルールはもちろんのこと，手がかり語などの特殊な情報を準備することなく，抽出処理を行うことができる．

以降，図 2 に示した処理の詳細について説明する．3.5 節以降が本論文で新たに提案する処理である．3.4 節の詳細については文献 [8] を参照されたい．

### 3.3 ブログページの収集

ExtractUniqueBlock 法はブログ記事ページに限らず，一般の Web ページを処理対象とするが，本論文での課題はブログ記事ページに特化しているため，ブログ記事ページを処理対象として議論を進める．本論文において，処理対象となる**ブログページ集合**は，以下の集合とする．

(1) 同じブログサイトの記事ページのみで構成される．
(2) 2 記事ページ以上で構成される．
(3) コメントが付いていない記事ページが含まれる．

一般的にブログ記事に対するコメント率は低い．また，フィード[*6]を用いて最新の記事ページを投稿直後に収集すれば，少なくとも 1 ページはコメントが付く前に収集できると考える．そのため，上の条件 (3) は特殊事例とならない．

記事ページには画像ファイルやスタイルシートが含まれている場合も多く，Cao らの手法 [19] はそれらのファイルも必要とする．一方，提案手法はそれらのファイルを必要とせず，記事ページの HTML ファイルのみで抽出処理を行うことができる．そのため，処理に必要なデータの準備が比較的容易である．

### 3.4 ExtractUniqueBlock 法の概要

#### 3.4.1 ブロックの抽出

HTML のブロックレベル要素をもとに，コンテンツおよび不要部分の最小単位である**ブロック**の抽出を行う．W3C が定義するブロックレベル要素をもとに抽出することで，ページレイアウトの流行の影響を受けずにページを分割できる．

ブロックレベル要素をもとにブロックを抽出する際，ブロックがコンテンツおよび不要部分の最小単位となるよう，入れ子になっているブロックレベル要素を抜いて抽出する．たとえば，**図 3** では，破線で囲まれた部分がブロックを表し，DOM ツリー[*7]上の下位ノードにブロックレベル要素が存在しないように抽出する．なお，図 3 から分かるとおり，ブロックレベル要素とブロックは 1 対 1 で対応する．

#### 3.4.2 コンテンツの抽出

ブロックの一致を判断し，ページ集合内で 1 度だけ出現するブロックをコンテンツとして抽出する．本論文では，このブロックを**コンテンツブロック**と呼ぶ．

ブロックの一致判断は，ブロックに含まれる要素名，テキストおよび title，alt，src 属性の属性値を素性とするベクトルを用いて，ベクトルどうしのコサイン類似度を計算することにより行う．ここでは，ブロック間の類似度が 0.9 を上回ったとき，それらのブロックは一致したと認める．一致判断に類似度を導入することにより，レンダリングにほとんど影響を与えない微小な違いを許容できる．

### 3.5 ブロック識別子の付与

ブログページ集合内におけるブロックの役割を識別する**ブロック識別子**（提案手法のために定義する識別子）を付

[*5] World Wide Web で使用される技術の標準化を進める国際非営利団体．
[*6] RSS，Atom など Web サイトの更新情報を記述した XML 文書．
[*7] HTML（XML）文書の構造を木で表現したデータ構造．DOM は Document Object Model の略である．

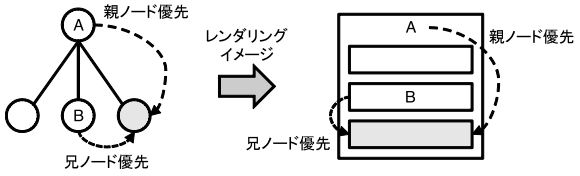



図 4 親ノード優先と兄ノード優先との違い

**Fig. 4** A difference of priorities between a parent node and a previous sibling node.

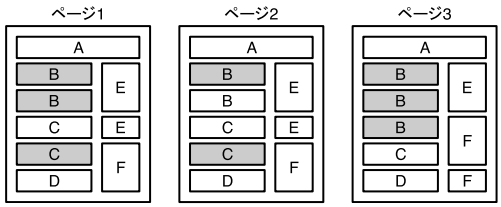



図 5 ポストとコメントの分離抽出の例

**Fig. 5** An example of extracting posts and comments.

与するために，あらかじめページ内の各ブロックレベル要素に付与された要素識別子（Element Identifiers）を利用する．要素識別子は id 属性および class 属性のことであり，スタイルシートのセレクタなど，要素を特定するために使われる識別子であることから，ブロックレベル要素をもとにしたブロックの役割を識別するのに準用できると考えられる．しかし，ブロック識別子に要素識別子をそのまま用いると問題が発生する．

ブロック識別子に要素識別子をそのまま用いることの最大の問題は，すべてのブロックレベル要素に要素識別子が付与されているとは限らない点である．この問題を解決するために，近隣のブロックは同じ役割を持つと仮定し，要素識別子を DOM ツリー上の兄ノードから弟ノードに伝播させる．親ノードよりも兄ノードを優先することにより，ブラウザでのレンダリングイメージにおいて，近隣から伝播される．たとえば，**図 4** に示すツリーにおいて親ノードを優先して伝搬させると（A，B が要素識別子である），下位ノードは左から A，B，A となる．この場合，レンダリングイメージで見ても分かるとおり，B のノードが孤立してしまう．対して兄ノードを優先して伝搬させると，左から A，B，B となり，孤立を防ぐことができる．

あらかじめ付与された要素識別子の中には，ブログページ集合内で一般化されたブロック識別子に適さない要素識別子が 2 種類含まれる．一方は，各ページにおいて何度も出現する要素識別子である．このような要素識別子を伝播させると，隣接していないにもかかわらず同じブロック識別子が付与されてしまう．他方は，ブログページ集合内において一部のページにしか出現しない要素識別子である[*8]．このような要素識別子は，個々のページ特有の情報を持つ可能性が高く，一般化するのに適さない．

以上をふまえ，ブロックに対し，ブログページ集合内で一般化されたブロック識別子の付与を行う．まず，ブロック識別子として伝播させる要素識別子として，次の条件を満たす要素識別子を選択する．

**a.** 1 ページ中に 1 度のみ出現する．

**b.** ブログページ集合すべてのページに出現する．

そして，各ブロックと対応するブロックレベル要素を DOM ツリー上で前順（preorder）に走査し，次の優先順位に従ってブロック識別子を付与する．

(1) 自身の要素識別子を付与する．

(2) 隣接する兄ノードのブロック識別子を付与する．

(3) 親ノードのブロック識別子を付与する．

(4) 標準値（_default_）を付与する．

図 3 は，DOM ツリー（要素識別子は要素内に記載）からブロック抽出を行い，各ブロック（破線枠部分）にブロック識別子（実線矩形枠部分）を付与した例である．ブロック識別子上部の括弧付き数字は，上で述べたブロック識別子付与規則を表す．たとえば，ブログページ 1 の「class=c1」は双方のブログページに出現しているが，ブログページ 2 において 2 度出現しているため，伝播させる要素識別子の条件 a を満たさず，このブロックに対しては規則 (2) が適用される．

### 3.6 ポストとコメントの分離抽出

3.1 節で定義したように，コンテンツはポストとコメントからなるため，ポストとコメントの分離抽出にあたり，コンテンツブロック以外のブロック（不要部分）を除外する．たとえば，**図 5** に示すレンダリングイメージにおいて，矩形をブロック，着色された矩形をコンテンツブロックとすると，着色されていないブロックを除外する．なお，A から F の英字は，各ブロックのブロック識別子を表す．

筆者らは，ポストはすべての記事ページに出現する一方で，コメントは一部の記事ページにしか付かない傾向があることに着目した．この傾向に従って，コンテンツブロックをポストとコメントに分類する．具体的には，すべてのページに出現するブロック識別子が付与されたコンテンツブロックをポストとし，残りのコンテンツブロックをコメントとして抽出する．図 5 では，ブロック識別子 B が付与されたコンテンツブロックがポストとして抽出され，ブロック識別子 C が付与されたコンテンツブロックがコメントとして抽出される．

## 4. 実験

### 4.1 評価指標

本実験の評価指標は，3.1 節の定義に従って人手により作成したポストとコメントの正解データに対する正確度（Accuracy），適合率（Precision），再現率（Recall），F 値（F-measure）である．

[*8] article_12345 のような記事固有の数値を含む要素識別子が，記事本文付近にしばしば出現する．

正確度は，各ブロックがどれだけ正確に判別できたかの指標であり，以下の式で計算される．

$$Accuracy = \frac{\text{正確に判別できたブロックの数}}{\text{正解データに含まれるブロックの数}}$$

なお，コンテンツ抽出性能の正確度は各ブロックをコンテンツまたは不要部分に分類した場合の正確度を表し，ポスト・コメント抽出性能の正確度は各ブロックをポスト，コメント，不要部分に分類した場合の正確度を表す．

適合率，再現率，F 値は，コンテンツ，ポスト，コメントがどれだけ抽出できたかの指標であり，たとえば，ポストに対するこれらの指標は以下の式で計算される．

$$Precision = \frac{\text{“ポスト” と正解した数}}{\text{“ポスト” と判定した数}}$$

$$Recall = \frac{\text{“ポスト” と正解した数}}{\text{正解データに含まれる “ポスト” の数}}$$

$$F\text{-}measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

コンテンツおよびコメントの適合率，再現率，F 値も同様に計算する．

### 4.2 データセット

実験に使用するブログ記事ページは，livedoor Reader の登録数ランキング上位からブログ形式の 9 サイト，100SHIKI[*9]，Engadget Japanese[*10]，ネタフル[*11]，404 Blog Not Found[*12]，IDEA*IDEA[*13]，TechCrunch Japan[*14]，Life is beautiful[*15]，My Life Between Silicon Valley and Japan[*16]，Going My Way[*17]から 2009 年 4 月 11 日に収集した計 206 ページである（**表 1**）．これらの 9 サイトは，提案手法の汎用性を示せるよう，ブログホスティングサービスが重複しないように選択されている．選択した 9 サイトのうち 6 サイトはブログホスティングサービスを利用せず，独自に開設されたブログサイトであった．なお，様々な観点で評価できるよう，すべてのページにコメントが付いているブログサイト（My Life Between Silicon Valley and Japan）およびコメントが 1 件も付いていないブログサイト（Going My Way）がデータセットに含まれる．

表 1 に示すとおり，ポストおよびコメントの候補となるブロックは 35,216 ブロック存在し，正解データを人手によって作成したところ，ポストと認められるブロックは 2,932 ブロック，コメントと認められるブロックは 973 ブロックであった．また，38%のページにコメントが付いていた．

[*9] http://www.100shiki.com/
[*10] http://japanese.engadget.com/
[*11] http://netafull.net/
[*12] http://blog.livedoor.jp/dankogai/
[*13] http://www.ideaxidea.com/
[*14] http://jp.techcrunch.com/
[*15] http://satoshi.blogs.com/
[*16] http://d.hatena.ne.jp/umedamochio/
[*17] http://kengo.preston-net.com/

**表 1** データセットの詳細
**Table 1** Details of the data set.

| ブログ名 | 記事数 | ブロック | ポスト | コメント |
|---|---|---|---|---|
| 100SHIKI | 10 | 725 | 153 | 12 |
| Engadget Japanese | 40 | 6,163 | 200 | 36 |
| ネタフル | 30 | 2,494 | 451 | 4 |
| 404 Blog Not Found | 50 | 19,264 | 1,008 | 750 |
| IDEA*IDEA | 9 | 569 | 121 | 6 |
| TechCrunch Japan | 20 | 1,951 | 182 | 8 |
| Life is beautiful | 12 | 1,525 | 130 | 111 |
| My Life Between ... | 5 | 575 | 166 | 46 |
| Going My Way | 30 | 1,950 | 521 | 0 |
| 合計 | 206 | 35,216 | 2,932 | 973 |

### 4.3 比較手法

提案手法の有効性を示すため，次の 3 つの手法と比較する．2 章で述べたように，提案手法と従来手法とではコンテンツの最小単位が異なるため，比較手法によって抽出されるコンテンツブロックは，コンテンツによって包含されるすべてのブロックのうち，テキストを含むブロックとする[*18]．

**比較 1（Cao ら [19]）**
Cao ら [19] によって提案された，ポストとコメントを同時に自動抽出する手法である．

**比較 2（Pappas ら [5]）**
Pappas ら [5] の手法はコンテンツ抽出手法であるが，彼らが公開している実装[*19]にはポストとコメントを同時に抽出する機能を有している．この実装では，抽出したコンテンツの中に “comment” や “reply” など，コメント出現特有の文字列が現れた場合，その周辺部分をコメントと見なし，それら以外をポストと見なしている．なお，この実装には特定の言語に依存する処理が含まれていたため，日本語のページにも適用できるように改良した[*20]．

**比較 3（Weninger ら [7] と Song ら [12]）**
コンテンツ抽出手法である Weninger ら [7] の手法[*21]とコメント抽出手法である Song ら [12] の手法を組み合わせる．Weninger らの手法による抽出分はポスト，Song らの手法による抽出分はコメント，双方の抽出が重複する場合はコメントと見なす[*22]．

[*18] テキストを含まないブロックもコンテンツブロックと見なした場合，再現率に変化が見られず，適合率が著しく低下した．
[*19] https://github.com/nik0spapp/webpage_segmentation
[*20] “comment”，“reply”，“.”，など特定の文字列を探索する処理をそれらの日本語訳（“コメント”，“リプライ”，“返信”，“。”，など）も探索するようにした．
[*21] 以下の Web ページで公開されている実装を利用した．
http://www.cse.nd.edu/~tweninge/cetr/
[*22] 予備実験を行ったところ，Song らの手法が Weninger らの手法よりも高い適合率を示したため，抽出結果の一部が不要部分であるか否かの判定は必要ないと判断した．

表 2 正確度（Accuracy）による分離抽出性能の比較
**Table 2** Accuracy for each algorithm in each blog.

| | コンテンツ抽出性能 | | | | ポスト・コメント抽出性能 | | | |
|---|---|---|---|---|---|---|---|---|
| ブログ名 | 提案 | 比較 1 | 比較 2 | 比較 3 | 提案 | 比較 1 | 比較 2 | 比較 3 |
| 100SHIKI | **94.2** | 85.9 | 78.6 | 84.6 | **92.8** | 85.9 | 77.5 | 82.3 |
| Engadget Japanese | **98.5** | 86.5 | 93.0 | 77.0 | **98.5** | 85.3 | 92.4 | 76.5 |
| ネタフル | **94.7** | 66.4 | 81.2 | 86.0 | **90.6** | 66.1 | 81.1 | 85.7 |
| 404 Blog Not Found | **98.8** | 96.6 | 91.6 | 81.5 | **98.5** | 87.2 | 91.2 | 75.2 |
| IDEA*IDEA | **94.7** | 77.9 | 73.8 | 85.4 | **93.7** | 77.9 | 73.8 | 85.1 |
| TechCrunch Japan | **97.0** | 81.3 | 90.1 | 94.9 | **96.8** | 68.3 | 90.1 | 94.6 |
| Life is beautiful | **97.3** | 89.0 | 87.1 | 91.1 | **96.0** | 75.7 | 81.1 | 88.5 |
| My Life Between ... | **95.0** | 79.7 | 61.7 | 83.5 | 80.3 | 70.6 | 58.3 | **83.1** |
| Going My Way | **94.4** | 85.0 | 68.5 | 80.3 | **89.7** | 45.6 | 68.5 | 80.3 |
| マイクロ平均 | **97.8** | 90.1 | 88.5 | 82.3 | **96.7** | 81.0 | 87.8 | 78.6 |
| マクロ平均 | **96.1** | 83.1 | 80.6 | 84.9 | **93.0** | 73.6 | 79.3 | 83.5 |

表 3 F 値による分離抽出性能の比較
**Table 3** F-measure for each algorithm in each blog.

| | コンテンツ抽出性能 | | | | ポスト抽出性能 | | | | コメント抽出性能 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ブログ名 | 提案 | 比較 1 | 比較 2 | 比較 3 | 提案 | 比較 1 | 比較 2 | 比較 3 | 提案 | 比較 1 | 比較 2 | 比較 3 |
| 100SHIKI | **87.3** | 75.1 | 28.6 | 72.3 | **87.8** | 86.6 | 26.3 | 70.4 | **48.3** | 29.3 | 0.0 | 0.0 |
| Engadget Japanese | **79.9** | 34.8 | 12.2 | 23.7 | **84.5** | 58.6 | 5.1 | 21.8 | **62.6** | 0.0 | 6.9 | 35.3 |
| ネタフル | **85.8** | 52.0 | 9.3 | 68.4 | **83.0** | 63.7 | 8.6 | 67.9 | **7.8** | 0.0 | 0.0 | 0.0 |
| 404 Blog Not Found | **93.3** | 84.0 | 28.2 | 43.2 | **90.0** | 20.7 | 36.1 | 27.4 | **92.8** | 47.2 | 0.0 | 0.0 |
| IDEA*IDEA | **88.7** | 65.8 | 0.0 | 73.0 | **90.0** | 79.3 | 0.0 | 73.8 | **35.3** | 15.4 | 0.0 | 0.0 |
| TechCrunch Japan | **83.9** | 49.6 | 26.1 | 73.7 | **84.1** | 31.4 | 26.9 | 73.7 | **60.0** | 3.2 | 0.0 | 0.0 |
| Life is beautiful | **92.1** | 72.7 | 40.6 | 76.1 | **82.5** | 58.4 | 20.3 | 60.1 | **95.2** | 0.0 | 0.0 | 86.2 |
| My Life Between ... | **93.5** | 66.5 | 8.3 | 73.8 | **83.2** | 59.6 | 0.0 | 69.0 | 0.0 | 26.2 | 0.0 | **87.8** |
| Going My Way | **89.3** | 76.0 | 41.0 | 47.3 | **84.1** | 24.6 | 41.0 | 47.3 | - | - | - | - |
| マイクロ平均 | **90.1** | 67.6 | 26.4 | 47.9 | **86.2** | 49.0 | 26.7 | 38.5 | **81.8** | 26.9 | 0.4 | 23.3 |
| マクロ平均 | **88.2** | 64.1 | 21.6 | 61.3 | **85.5** | 53.7 | 18.3 | 56.8 | **50.3** | 15.2 | 0.9 | 26.2 |

### 4.4 分離抽出の実験結果

まず，正確度の比較結果を**表 2** に示す．それぞれのブログサイトにおいて，コンテンツ抽出性能，ポスト・コメント抽出性能のそれぞれに対して符号検定を行ったところ，提案手法と各比較手法との間に危険率 1%で有意差を確認した．大半の局面で，提案手法の方が有意に正確度が高いことが確認できた．本指標は，不要部分の特定も含めた正確度を算出するため，ブロック数に対しコンテンツブロック数が少ない場合，すべてのブロックを不要部分とすることで高い性能を達成できる場合がある．たとえば，比較手法 2 は，多くのブロックを不要部分と判定することで，比較的高い正確度を達成していた．

抽出したポストおよびコメントの活用を想定すれば，それらに対する適合率，再現率，F 値の評価も必要である．適合率と再現率から算出した F 値による比較結果を**表 3** に示す．「Going My Way」は正解データにコメントが含まれないため，コメント抽出性能の F 値が計算できず「-」とした．提案手法の方が全体的に高い抽出性能を示していることから，提案手法の有効性が確認できた．提案手法のコメント抽出性能に関しては，正解ブロック数（コメント）が少ないため実験結果が安定していないが，「404 Blog Not Found」「Life is beautiful」から分かるとおり，十分なコメントがあれば高い性能で安定している．

提案手法による分離抽出性能の詳細を**表 4** に示す．「My Life Between Silicon Valley and Japan」はコメントが自動抽出されなかったため，コメント抽出性能の適合率が計算できず「-」とした．コメントが自動抽出されなかった原因は，3.3 節で述べた提案手法の前提による．「My Life Between Silicon Valley and Japan」にはすべてのページにコメントが付いており，自動的に分離抽出した結果，すべてのコメントがポストとして抽出された．表 2 の「ポスト・コメント抽出性能」，表 3 の「コメント抽出性能」において，提案手法が最高性能にならなかった原因も同様である．また，「Going My Way」には 1 件のコメントも付いておらず，仮説どおりであればすべてのコンテンツがポストとして抽出されるが，コンテンツ抽出の失敗により，コメントとして抽出されるブロックが存在した．

コンテンツ自動抽出に失敗したポストおよびコメント

表 4 提案手法によるポストとコメントの分離抽出性能詳細
Table 4 Details of the extraction results by the proposed method.

| | ポスト抽出性能 | | | コメント抽出性能 | | |
|---|---|---|---|---|---|---|
| ブログ名 | 抽出数 | 適合率 | 再現率 | 抽出数 | 適合率 | 再現率 |
| 100SHIKI | 150 | 88.7 | 86.9 | 17 | 41.2 | 58.3 |
| Engadget Japanese | 174 | 90.8 | 79.0 | 63 | 49.2 | 86.1 |
| ネタフル | 371 | 91.9 | 75.6 | 99 | 4.0 | 100.0 |
| 404 Blog Not Found | 832 | 99.5 | 82.1 | 715 | 95.1 | 90.7 |
| IDEA*IDEA | 128 | 87.5 | 92.6 | 11 | 27.3 | 50.0 |
| TechCrunch Japan | 158 | 90.5 | 78.6 | 12 | 50.0 | 75.0 |
| Life is beautiful | 156 | 75.6 | 90.8 | 120 | 91.7 | 99.1 |
| My Life Between ... | 233 | 71.2 | 100.0 | 0 | - | 0.0 |
| Going My Way | 456 | 90.1 | 78.9 | 46 | - | - |
| 合計（マイクロ平均） | 2,658 | 90.7 | 82.2 | 1,083 | 77.7 | 86.4 |

表 5 分離抽出処理のみの性能比較（F 値）
Table 5 Experimental results of only the separation.

| | ポスト抽出性能 | | | | コメント抽出性能 | | | |
|---|---|---|---|---|---|---|---|---|
| ブログ名 | 提案 | 比較 1 | 比較 2 | 比較 3 | 提案 | 比較 1 | 比較 2 | 比較 3 |
| 100SHIKI | 100.0 | 100.0 | 100.0 | 96.2 | 100.0 | 100.0 | 100.0 | 0.0 |
| Engadget Japanese | 100.0 | 91.5 | 0.0 | 93.7 | 100.0 | 0.0 | 26.5 | 40.0 |
| ネタフル | 99.6 | 99.6 | 100.0 | 99.6 | 0.0 | 0.0 | 100.0 | 0.0 |
| 404 Blog Not Found | 99.3 | 24.0 | 0.0 | 72.9 | 99.0 | 59.3 | 59.8 | 0.0 |
| IDEA*IDEA | 100.0 | 100.0 | 100.0 | 97.6 | 100.0 | 100.0 | 100.0 | 0.0 |
| TechCrunch Japan | 100.0 | 46.2 | 0.0 | 97.8 | 100.0 | 9.9 | 8.1 | 0.0 |
| Life is beautiful | 100.0 | 69.7 | 97.6 | 90.6 | 100.0 | 0.0 | 97.4 | 86.2 |
| My Life Between ... | 87.8 | 87.1 | 0.0 | 97.1 | 0.0 | 32.4 | 35.7 | 87.8 |
| Going My Way | 100.0 | 41.6 | 0.0 | 100.0 | - | - | - | - |

は，その後の処理である分離処理で抽出不可能となるため，表 4「ポスト抽出性能」「コメント抽出性能」の再現率には上限値が存在する．コンテンツ自動抽出の結果から正解データに含まれないブロック（不要部分）を除外し，残りのブロックを完璧に分離抽出した場合の再現率は，それぞれ 87.1%と 91.6%であった．提案手法による性能（再現率）は，それぞれ 82.2%と 86.4%である．このことから，提案する分離処理は，コンテンツ自動抽出が不完全であっても，その範囲内でかなり正確に分離できていることが分かる．

### 4.5 分離処理のみの性能評価実験

前節で評価した分離抽出性能はコンテンツ抽出性能に依存しているため，本節では，コンテンツを過不足なく抽出できたと仮定し，分離処理のみの評価を行う．比較手法 1 および 3 に関しては，それらの手法によって抽出されたコメントの集合を $C$，正解データを $T$ としたとき，ポストおよびコメントをそれぞれ $T-(T\cap C)$ および $T\cap C$ であると見なして評価する[*23]．比較手法 2 に関しては，表 3 が示すようにコメント抽出性能が低かったことをふまえ，HTML ファイルから “コメント” や “comment” などの文字列を探索して分割し，後方に含まれるブロックの集合を $C$ と見なして比較手法 1 および 3 と同様に評価する[*24]．

分離処理のみの比較結果（F 値）を**表 5** に示す．提案手法のための前提を満たさない「My Life Between Silicon Valley and Japan」およびコメントがポストの 1%未満しか存在しない「ネタフル」を除き，提案手法が最高性能を示している．コンテンツ自動抽出が理想的に行われた場合，提案手法は非常に高い性能を達成できることが明らかになった．

### 4.6 コンテンツ抽出の改善処理

ExtractUniqueBlock 法は，各々のブロックの位置を加味せずコンテンツを判定するため，連続するコンテンツブロックの一部を取りこぼすという欠点があった．そこで，コンテンツ抽出（3.4.2 項）で不要部分と判断されたブロックのうち，コンテンツブロックと同一のブロック識別子と要素名を持つブロックを新たにコンテンツブロックとして再抽出することにより，コンテンツ自動抽出の性能向上を試みた．

[*23] ポストの集合を $P$ とし，$T\cap P$ および $T-(T\cap P)$ と見なした場合も試したが，本文で述べた条件の方が F 値が高かった．

[*24] “コメント” や “comment” などの探索文字列が複数回出現する場合，最初の出現位置で分割する．探索文字列は 4.3 節と同じものを利用した．

表 6 コンテンツ再抽出の有無による平均抽出性能の比較（マイクロ平均）
**Table 6** Experimental results of the improvement by the content re-extraction.

| | | 適合率 | 再現率 | F 値 | 正確度 |
|---|---|---|---|---|---|
| コンテンツ再抽出あり（提案手法） | コンテンツ | 90.6 | 92.2 | 91.4 | 98.1 |
| | ポスト | 90.4 | 85.1 | 87.7 | 96.9 |
| | コメント | 74.6 | 93.2 | 82.9 | |
| コンテンツ再抽出なし（ExtractUniqueBlock 法） | コンテンツ | 92.1 | 88.2 | 90.1 | 97.8 |
| | ポスト | 90.7 | 82.2 | 86.2 | 96.7 |
| | コメント | 77.7 | 86.4 | 81.8 | |

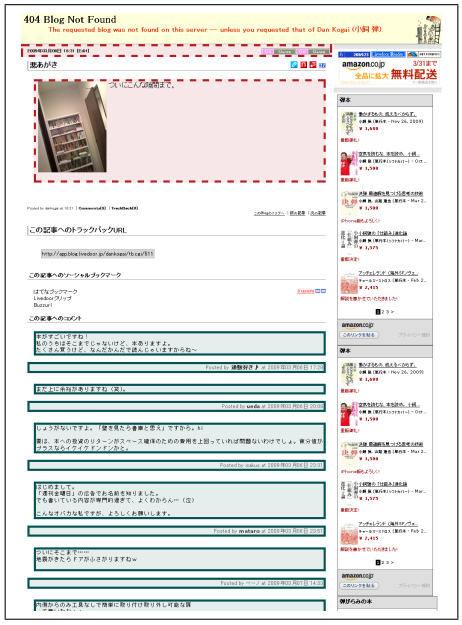



図 6 実験結果のブログ記事ページ例（分離抽出）
**Fig. 6** An example of a blog page and the extracted posts and comments.

**表 6** は平均抽出性能（マイクロ平均）を比較した結果である．この実験結果より，コンテンツ再抽出あり（提案手法）は再抽出なし（ExtractUniqueBlock 法）に比べて全体的に性能が向上していることが分かる．再抽出ありによるコンテンツ抽出性能は，再抽出なしに比べ，適合率は少々低下したが，再現率は大幅に向上し，F 値が向上している．正確度に関しても，コンテンツ抽出性能とポスト・コメント抽出性能のいずれも向上した[*25]．**図 6** はコンテンツ抽出の改善処理を含め，提案手法による抽出を行ったブログ記事ページ[*26]の例である（破線着色部分がポスト，実線着色部分がコメントを示す）．

### 4.7 考察

提案手法の性能は，ExtractUniqueBlock 法の性能に依存している．ポストの適合率に比べてコメントの適合率が低くなる傾向があるのは，ポストが完全に抽出されないこと，不要部分を誤って抽出することが影響している．提案手法の性能を改善するためには，ExtractUniqueBlock 法の性能を改善する必要がある．

ブログページ集合に構造が乱れた HTML（Valid でない HTML）で構成される記事ページが含まれている場合，ブロック識別子が適切に付与されない場合がある．4.5 節の実験で「404 Blog Not Found」が理想的な結果とならなかった原因は，そのような記事ページが 1 ページ含まれていたからである．構造が乱れた HTML であっても，Web ブラウザで表示すると適切に表示される場合が多い．これは，構造が乱れた HTML であってもブラウザが柔軟に解釈するからである．このことから，Web ブラウザのような HTML パーサを使用することにより，構造が乱れた HTML にも対応できると考えられる．

提案手法は，コメントが付いていない記事ページがブログページ集合に含まれることを前提としている．そのため，すべてのページにコメントが付いていると，コメントもポストとして抽出してしまう．4.5 節の実験で「My Life Between Silicon Valley and Japan」が理想的な結果とならなかった原因は，提案手法の前提に反し，すべてのページにコメントが付いていたからである．この問題は，3.3 節で述べたとおり，フィードを用いて記事ページを収集することで解決できるが，すでに収集済みの記事ページに適用するなど，過去の資源（データ）を有効活用するためには収集方法に依存せずに解決することも望まれる．コメント部分のブロック識別子はどのブログページ集合でも似ている傾向があること（ブロック識別子に「comment」を含むなど）に着目し，コメントが 1 件も抽出できない場合は，他のブログページ集合でコメントに分類されたブロックのブロック識別子を利用することにより，適切に抽出できる可能性がある．

スタイルシートを使用せずにデザインしているブログサイトに対しては，提案手法を適用することはできず，要素識別子を用いずにブロックの役割を識別する必要がある．コンテンツブロックを包含するような DOM ツリーの構造パターンを獲得し，その構造をブロック識別子と見なすことで，要素識別子に頼らずブロックの役割を識別できる可能性がある．Web ページ内において繰り返し出現する構造

[*25] 符号検定を行ったところ，危険率 1%において 5 サイトで有意差を確認した．
[*26] http://blog.livedoor.jp/dankogai/archives/51185176.html

パターンの獲得は，コメントを抽出する手法 [11], [12] でも用いられているが，獲得範囲をページ集合に拡張することで，提案手法を拡張できるかもしれない．なお，松本ら [20] の調査によれば，一般の Web ページも含めスタイルシートの利用率が年々増加しており（2002 年に 21.3%だった利用率が 2012 年には 80.9%まで増加），今後も要素識別子を用いる提案手法は有効に機能すると考えられる．

## 5. おわりに

本論文では，事前に抽出ルールや学習データを準備する必要のないアルゴリズムで，ブログページ集合の記事ページからポストとコメントを分離抽出する手法を提案した．日本語ブログサイトを対象とした実験により，提案手法の有効性を示した．提案手法は，ブログページ集合を与えさえすれば自動的にポストとコメントを分離抽出するため，非常に小さな労力で記事ページからポストとコメントを抽出できる．

提案手法の分離抽出性能は，コンテンツ抽出性能に依存しており，コンテンツ抽出性能の改善のみで分離抽出性能が改善できることが示唆された．さらに，分離抽出を実現するために提案したブロック識別子を活用することで，コンテンツ抽出性能が改善されることを示した．

今後，要素識別子が適切に付与されていないブログページ集合にも適用できるよう，より柔軟にブロック識別子を付与する方法を検討する．また，抽出ルールの自動獲得，ルール自動選択による抽出を検討し，より高速に分離抽出する手法を検討する．さらに，本論文の成果をモジュールおよびソフトウェアの形で公開し[*27]，ブログ記事ページを利用する研究の標準的な手法となることを目指す．

**謝辞** 本論文は JSPS 科研費・特別研究員奨励費 11J01016 の助成を受けた．

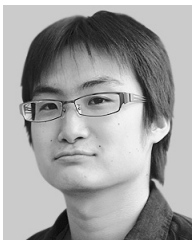
### 参考文献

[1] 佐伯千種，岩間健宏：ブログの実態に関する調査研究，技術報告，総務省情報通信政策研究所（IICP）(2009).
[2] 奥村 学：ブログマイニング技術の最新動向，電子情報通信学会誌，Vol.91, No.12, pp.1054–1059 (2008).
[3] Agarwal, N. and Liu, H.: Blogosphere: Research Issues, Tools, and Applications, *ACM SIGKDD Explorations Newsletter*, Vol.10, No.1, pp.18–31 (2008).
[4] Yi, L., Liu, B. and Li, X.: Eliminating Noisy Information in Web Pages for Data Mining, *Proc. 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *SIGKDD '03*, Washington, DC, USA, pp.296–305 (2003).
[5] Pappas, N., Katsimpras, G. and Stamatatos, E.: Extracting Informative Textual Parts fromWeb Pages Containing User-Generated Content, *Proc. 12th International Conference on Knowledge Management and Knowledge Technologies*, *i-KNOW '12*, Graz, Austria, pp.4:1–4:8 (2012).

[*27] https://github.com/ceekz/ContentExtraction

[6] 鶴田雅信，増山 繁：レイアウト情報を用いた Web ページの主要な DOM ノードの抽出法，人工知能学会論文誌，Vol.25, No.6, pp.742–756 (2010).
[7] Weninger, T., Hsu, W.H. and Han, J.: CETR: Content Extraction via Tag Ratios, *Proc. 19th International Conference on World Wide Web*, *WWW '10*, Raleigh, North Carolina, USA, pp.971–980 (2010).
[8] 吉田光男，山本幹雄：教師情報を必要としないニュースページ群からのコンテンツ自動抽出，日本データベース学会論文誌，Vol.8, No.1, pp.29–34 (2009).
[9] Debnath, S., Mitra, P., Pal, N. and Giles, C.L.: Automatic Identification of Informative Sections of Web Pages, *IEEE Trans. Knowledge and Data Engineering*, Vol.17, No.9, pp.1233–1246 (2005).
[10] Chang, C.-H. and Chen, J.-M.: Automatic Extraction of Blog Post from Diverse Blog Pages, *Proc. 2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, *WI-IAT '12*, Macau, China, pp.129–136 (2012).
[11] Kao, H.-A. and Chen, H.-H.: Comment Extraction from Blog Posts and Its Applications to Opinion Mining, *Proc. 7th International Conference on Language Resources and Evaluation*, *LREC '10*, Valletta, Malta, pp.1113–1120 (2010).
[12] Song, X., Liu, J., Cao, Y., Lin, C.-Y. and Hon, H.-W.: Automatic Extraction of Web Data Records Containing User-Generated Content, *Proc. 19th ACM International Conference on Information and Knowledge Management*, *CIKM '10*, Toronto, Ontario, Canada, pp.39–48 (2010).
[13] 宮田章裕，川島晴美，藤村 考，奥 雅博：コメント・トラックバック分析に基づくブログ記事の影響度判定，電子情報通信学会論文誌 D，情報・システム，Vol.93, No.6, pp.756–766 (2010).
[14] Bhattarai, A., Rus, V. and Dasgupta, D.: Characterizing Comment Spam in the Blogosphere through Content Analysis, *Proc. 2009 IEEE Symposium on Computational Intelligence in Cyber Security*, *CICS '09*, Nashville, Tennessee, USA, pp.37–44 (2009).
[15] Li, B., Xu, S. and Zhang, J.: Enhancing Clustering Blog Documents by Utilizing Author/Reader Comments, *Proc. 45th Annual Southeast Regional Conference*, *ACMSE '07*, Winston-Salem, North Carolina, USA, pp.94–99 (2007).
[16] Parapar, J., López-Castro, J. and Barreiro, A.: Blog Snippets: A Comments-Biased Approach, *Proc. 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, *SIGIR '10*, Geneva, Switzerland, pp.711–712 (2010).
[17] Hu, M., Sun, A. and Lim, E.-P.: Comments-Oriented Document Summarization: Understanding Documents with Readers' Feedback, *Proc. 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, *SIGIR '08*, Singapore, Singapore, pp.291–298 (2008).
[18] Ma, Z., Sun, A., Yuan, Q. and Cong, G.: Topic-Driven Reader Comments Summarization, *Proc. 21st ACM International Conference on Information and Knowledge Management*, *CIKM '12*, Maui, Hawaii, USA, pp.265–274 (2012).
[19] Cao, D., Liao, X., Xu, H. and Bai, S.: Blog Post and Comment Extraction Using Information Quantity of Web Format, *Proc. 4th Asia Infomation Retrieval Symposium*, *AIRS '08*, Harbin, China, pp.298–309 (2008).
[20] 松本章代，今村真浩，小西達裕，高木 朗，小山照夫，三宅

芳雄，伊東幸宏：日本語ウェブ文書を対象とした 10 年間の実態調査（2002 年～2012 年），第 5 回データ工学と情報マネジメントに関するフォーラム，*DEIM '13*，福島県郡山市 (2013).



**吉田 光男** （学生会員）

2011 年筑波大学大学院システム情報工学研究科博士前期課程修了．同年同大学院博士後期課程進学．2011 年より日本学術振興会特別研究員（DC）．自然言語処理，Web 検索エンジンに関する研究に従事．言語処理学会，人工知能学会，日本データベース学会各会員．

**乾 孝司** （正会員）

2004 年奈良先端科学技術大学院大学情報科学研究科博士課程修了．日本学術振興会特別研究員，東京工業大学統合研究院特任助教等を経て，2009 年より筑波大学大学院システム情報工学研究科助教．博士（工学）．自然言語処理，意見マイニングに関する研究に従事．言語処理学会，人工知能学会，ACL 各会員．

**山本 幹雄** （正会員）

1986 年豊橋技術科学大学大学院修士課程修了．同年（株）沖テクノシステムズラボラトリ研究開発員．1988 年豊橋技術科学大学情報工学系教務職員．1991 年同助手．1995 年筑波大学電子・情報工学系講師．1998 年同助教授．2008 年筑波大学大学院システム情報工学研究科教授．博士（工学）．自然言語処理の研究に従事．言語処理学会，人工知能学会，ACL 各会員．