Vol.2015-UBI-46 No.13
2015/5/12

HTML構造解析と機械学習に基づくイベント情報抽出システムの提案

廖　宸一[1]　廣井　慧[2]　梶　克彦[1]　河口　信夫[1,2]

概要：本研究は，店舗のホームページやブログからクーポンやキャンペーンなどのイベント情報を抽出する方法を提案する。この方法を利用してユーザはをひとつひとつの店舗のホームページの閲覧を必要とせず，イベント情報抽出の網羅性と効率性を支援できる。本提案は Web ページブロック分割およびイベント情報認識の二つのタスクから構成される。一つ目のタスクでは Web ページをタイトルや説明文や日付などのイベント情報を含むブロックに分割する。従来の研究は特定なタグ，画面構成あるいはブロックの機能などを特徴量として Web ページを分割することが多く，半構造化データのイベント情報抽出が難しかった。本研究では HTML 構造解析に基づいて Web ページをブロックに分割する。二つ目のタスクとは分割されたブロックから不用な情報を取り除くため、イベント情報を識別する。本研究では機械学習の手法を用いてイベント情報の識別を実現する。名古屋駅地下街「エスカ」と「ユニモール」にある店舗 96 軒を対象として行った検証実験とその結果を示す。

A Proportion of Event Data Extraction based on HTML Structure Analysis and Machine Learning

Chenyi Liao[1]　Kei Hiroi[2]　Katsuhiko Kaji[1]　Nobuo Kawaguchi[1,2]

1. はじめに

近年，展示会やパーティーポスターなどのイベント情報はインタネット上で蓄積されつつある。これまでの研究で時空間に基づくイベント情報を可視化するシステム「イベント.Locky[1][2]」を開発した。イベント.Locky は ATND や CONNPASS などのイベント検索 API からの構造化データを用いてイベント情報を可視化するシステムである。クーポンやキャンペーンなどのイベント情報は店舗のホームページやブログで掲載されることが多い。ユーザにとって，それらの情報は役に立つが，HTML ドキュメントの半構造化データで記載されるため，既存のイベント情報検索 API から取得できない。一方，ユーザはひとつひとつの店舗のホームページをアクセスしなければならないため，自律的なイベント情報抽出システムのない場合，効率性と網羅性を確保することが難しい。そのため, 本研究では店舗のホームページやブログから自律的なイベント情報抽出する方法を提案し，提案手法を評価するため，検証実験を行う。実験対象としての店舗のホームページの URL は名古屋駅地下街 *1 から取得する。本提案は Web ページブロック分割およびイベントデータ認識の二つのタスクで構成される。

一つ目のタスクは Web ページを HTML ドキュメントにある HTML コードごとのブロックに分割する。ブロックはタイトルや説明文および日付などのイベント情報を含んでいる。第 2 章で紹介する関連研究は特定のタグ，画面構成あるいは機能などを特徴量として Web ページを分割することが多い。本研究では HTML 構造解析に基づく Web ページをブロックに分割する。店舗のホームページのコンテンツマネージメントシステム（CMS）はデータベースからの検索結果を繰り返し，HTML コードテンプレートに埋め，HTML コードを生成する。そのため，一つの

[1] 名古屋大学大学院　工学研究科
Graduate School of Engineering, Nagoya University
[2] 名古屋大学　未来社会創造機構
Institute of Innovation for Future Society, Nagoya University

*1 名駅の地下街: http://www.meieki.com/station_sa.php

Web ページにあるブロックは近い HTML 構造を持つと考えられ，似ている HTML コードの構造をマッチングすれば Web ページをブロックに分割できる。

二つ目のタスクは分割されたブロックからイベント情報を識別する。分割されたブロックの中に，イベント情報と無関係な情報が含まれる。例えば，ログやナビゲーションバーや著作権情報などがある。そのため，分割されたブロックからイベント情報を認識する必要がある。本研究では機械学習の手法を用いてイベント情報の識別する分類器を提案する。単語ベクトル空間のような高次元（10,000 次元より多い）特徴空間に対し，Support Vector Machine(SVM) を採用する。

本論文では，第 2 章は関連研究を紹介し， それぞれの適用範囲を分析する。第 3 章は提案する Web ページ分割方法とアルゴリズムを説明する。第 4 章は SVM を用いてイベント情報の認識を実現する。第 5 章は提案する Web ページ分割方法とイベント情報の認識方法に対して実証実験を行う。

2. 関連研究

Web ページブロック分割技術ではひとつの Web ページを有意な複数のブロックに分割する。初期段階の研究 [3][4] では $\langle table \rangle$ や $\langle H1 \rangle$ などの特定のタグを特徴量として Web ページを分割していた。当時は，Web ページは Table などの標準的なレイアウトで開発されることが多かった。しかし，近年の Web ページは DIV+CSS のようなより適応性のあるレイアウトで開発されている。Web ページブロックは $\langle div \rangle$ のような一般的なタグで定義することが多い。そのため、タグに基づく Web ページブロック分割技術は現在の Web ページに適用することが難しい。

Kovacevic[5] らは，ページレイアウトに基づく Web ページブロック分割技術を提案した。ページレイアウトは Web ページの header，footer，left，right，center を指す。この方法は規則性のある一般的な Web ページに適用するが，すべての Web ページにのみ適用される。特に，ファッション分野の店舗のホームページは型破りなレイアウトデザインを採用することがある。そのため，店舗のイベント情報の抽出に対して，ページレイアウトに基づく Web ページブロック分割技術は適用することが難しい。

Cai[6][7] らは画面構成に基づく Web ページブロック分割技術（VIPS）を提案した。この提案は人間の視覚をシミュレーションし，Web ページにある境界線や空白や画像や色などの特徴量を認識して Web ページブロック分割をする。しかし，VIPS の処理する結果には木構造が残り，どのレベルがイベント情報のあるブロックであるかはまだ判別できない。一方，div にあるタイトルのリストはひとつのブロックとして判別されるため，各タイトルとして出力されることはない。そのため，イベント情報の抽出に適用することが難しい。

図 1　グルーポンのホームページ

分割されたブロックは，イベント情報と無関係な情報が残る。Lan[8] らは無関係な情報を除く提案をした。この方法でひとつの Web サイトにある複数の Web ページを比較する。無関係な情報は複数の Web ページに似た内容と HTML 構造を持つ。例えば，ログやナビゲーションバーや著作権情報などはどの Web ページでも同じ構造を持つ。しかし，この方法では Web サイトにある Web ページは同一のテンプレートを使うことを十分条件としている。さらに，クローラーは少なくとも二つの Web ページをダウンロードしなければならず，単一の Web ページはこの方法で処理できない。本研究ではイベント情報ではない情報を不必要なものとしている。テキスト分類器を適用する。この分類器は分割されたブロックからイベント情報を抽出する。

3. Web ページブロック分割

本研究では，Web 生成の視点から Web ページブロック分割を考える。データベースにあるイベント情報の構造化データは Web ページ上で HTML 構造の近いブロックに生成されるため，ブロックの HTML 構造のマッピングによって構造化されたイベント情報に復元することが可能と考えられる。第 3 章では Web ページブロック分割について述べる。そして，分割されたブロックにはイベントではない情報も残る。そのため，イベント情報のブロックを認識し，その他の無関係な情報を除く必要がある。第 4 章で，

イベント情報の認識機能を提案する。

図 1 に示すように，各イベント情報のブロック（矢印で示す）はイベントの場所やイメージや価格などの情報を含む。それらのブロックは同一の構造を持つ。そのため，同一の HTML 構造をマッチングしてイベント情報ブロックを抽出することが可能となる。本章では，まず用語を説明する。それから，提案する Web ページブロック分割方法を紹介する。最後，具体的なアルゴリズムを示す。

HTML コード：HTML コードは HTML ドキュメントの一部である。HTML ドキュメントが木構造で構成されるため，HTML コードは HTML ドキュメントでの副木になる。

ブロック：ブロックは Web ページ上でのエリアである。それに該当する HTML コードは HTML ドキュメントでの HTML コードである。ブロックは有意な情報を含まなければならない。イベント情報のブログはイベントに関する全ての情報を含む。例えば，住所，説明文，日付などがある。

HTML コードテンプレート：HTML コードテンプレートはコンテナである。これはテキストのない HTML コードである。これは HTML コードと同じ DOM[*2] 構造がある。

コンテンツマネージメントシステム (CMS)：CMS[9] は WordPress[*3] などのサーバー上でのアプリケーションである。これは Web ページの内容を管理する。CMS はデータベースを操作して Web ページを動的に生成する。ユーザがブラウザで Web サイトにアクセスする場合，CMS はユーザからのリクエストのパラメータを分析する。それから，CMS はデータベースからユーザのリクエストした情報を検索する。それらの検索結果を用いて HTML コードテンプレートに埋めて HTML コードを生成する。次に，CMS はそれらの HTML コードを組み立てて HTML ドキュメントを生成する。最後，Web ページとしてその HTML ドキュメントをユーザにプッシュする。

図 2 はブロックの生成する手順を示す。構造化データはデータテーブルの行としてデータベースに保存される。各行は有意な情報を持つ。CMS が情報を検索する場合，検索結果を同一の HTML コードテンプレートに注ぐ。生成されたブロックは内容が異なるが，同一の DOM 構造を持つ。そのため，Web ページにあるブロックは同じ DOM 構造を持つ可能性が高い。DOM 構造をマッチングしてブロックを分割することが可能である。

HTML2.0 基準から，ボディ要素はブロックレベル要素とインライン要素の二種類 [*4] に分けられる。ブロックレベル要素は $\langle P \rangle$，$\langle table \rangle$ などがある。インライン要素は $\langle a \rangle$，$\langle img \rangle$ などがある。ブロックレベル要素はブラウザでの表示に際してもひとつのブロック（通常改行を伴う表示上のまとまり）として扱われる。それは Web ページのレイアウトに影響を与える。インライン要素は Web ページのレイアウトに影響を与えない。本研究では，ブロックレベル要素しか扱わない。

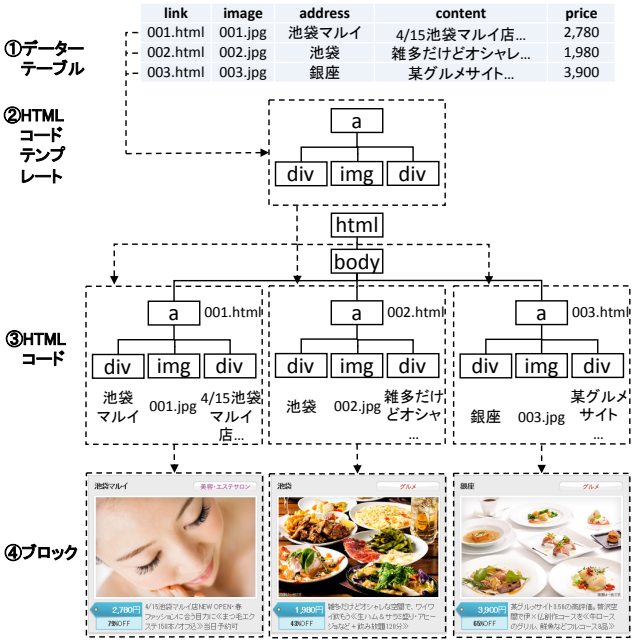

図 2　ブロック生成の例

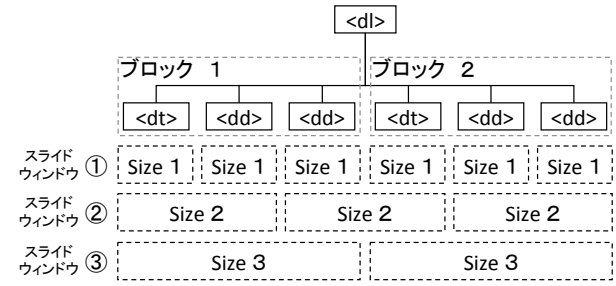

図 3　スライドウィンドウの例

HTML ドキュメントの構造は多階層木および順序木である。広いエリアから狭いエリアまでブロックを分割する手法を提案するため，根の要素から葉の要素まで，トップダウンによって木をスキャンする。そのため，幅優先探索を採用して木にある各要素をスキャンする。

HTML 木では，同じ深さにある複数の要素が一つのブロックに構成することがある。その問題を対応するため，スライドウィンドウメカニズムを提案する。図 3 に示すように，$\langle dl \rangle\langle dt \rangle\langle dd \rangle$ の HTML 構造がある。$\langle dt \rangle$ と $\langle dd \rangle$ は同じ深さにある。この例では一つの $\langle dt \rangle$ と二つの $\langle dd \rangle$ は一つのブロックを構成する。スライドウィンドウのメカニズムを導入する。ウィンドウのサイズ j は 1 からその深さにある要素の半分までインクリメントする。該当するウィンドウは $arrE_c$ で表示される。$arrE_l$ は $arrE_c$ 左隣のウィンドウを表す。$arrE_r$ は $arrE_c$ 右隣のウィンドウを表す。も

[*2] DOM: http://www.w3.org/DOM/
[*3] WordPress: https://wordpress.org/
[*4] ブロックレベル要素とインライン要素: http://www.w3schools.com/html/html_blocks.asp

表 1　Web ページブロック分割アルゴリズム

	Web Page Blocks Segmentation Algorithm
1	$Q = \{E_r\}, i = 0$
2	while $(i < Q.size())$ do
3	j = 1
4	while($j \leqslant \frac{Q_i.Level.Count()-i}{2} - 1$) do
5	$arrE_c = Q[i, i+j-1]$
6	$arrE_l = Q[i-j, i-1]$
7	$arrE_r = Q[i+j, i+2j-1]$
8	if($equal(arrE_c, arrE_l)$ OR $equal(arrE_c, arrE_r)$
9	OUTPUT $arrE_c$
10	i = i + j
11	break (goto 4)
12	else
13	if($j == \frac{Q_i.Level.Count()-i}{2} - 1$)
14	$Q_i.sons$ push into Q
15	$i = i + 1$
16	endif 13
17	endif 8
18	j = j + 1
19	endwhile 4
20	endwhile 2

表 2　関数 $equal(arrE_1, arrE_2)$

	Function $equal(arrE_1, arrE_2)$
1	String $S_1 = \sum_{i=1}^{n} BreadthFirstSearch(E_i \in arrE_1)$
2	String $S_2 = \sum_{i=1}^{n} BreadthFirstSearch(E_i \in arrE_2)$
3	Return $S_1 == S_2$

し $arrE_c$ の構造が $arrE_l$ または $arrE_r$ に等しい場合，ブロックとして出力する。

表 1 に示すように，幅優先探索をするため，待ち行列 $Q = \{E_R\}$ を初期化する。Q の中に一つの初期要素 E_R がある。E_R は HTML ドキュメントの根要素である。0 からの Q のカーソル i を初期化する。それから，ウィンドウのサイズ j を 1 に設定する。サイズ j を Q_i 深さの要素の数の半分 $\frac{Q_i.Level.Count()-i}{2} - 1$ まで繰り返す。$arrE_c$ と $arrE_l$ と $arrE_r$ をそれぞれに中，左，右の三つのウィンドウを設定する。それから，$arrE_c$ を用いて $arrE_l$ または $arrE_r$ と比較する。等しい場合，$arrE_c$ に含まれるすべての要素を一つのブロックとして出力する。そして，カーソルをウィンドウのサイズ j に加えて，行 4 に戻る。

等しくない場合，もしウィンドウのサイズ j が $\frac{Q_i.Level.Count()-i}{2} - 1$ になれば，すなわちこの深さにブロックのない場合，Q_i の息子要素を待ち行列 Q に追加する。カーソル i を 1 に加える。行 2 に戻る。

関数 $equal(arrE_1, arrE_2)$ を説明する。表 2 に示すように，$equal(arrE_1, arrE_2)$ は二つのウィンドウ $arrE_1$ と $arrE_1$ の副木の DOM 構造を比較する。この関数は $arrE_1$ と $arrE_2$ の副木をスキャンして文字列 S_1 と S_2 に転換する。S_1 と S_2 が同じ場合，関数は「真」をリターンする。それ以外は「偽」をリターンする。

その結果，HTML ドキュメントにあるすべてのブロックは出力される。しかし，イベント情報に除いて，ナビゲーションバーや著作権情報などの無益な情報も残る。第 4 章から，機械学習の手法を用いてイベント情報の認識する方法を説明する。

4. イベント情報認識

大量の無関係な情報が残るため，分割されたブロックを再分類する必要がある。本研究では，イベント情報の認識をイベント情報テキストの分類問題として処理する。すなわち，分割されたブロックは二種類（イベント情報と無関係なもの）になる。テキスト分類のタスクに対して，よく使われる方法は単純ベイズ分類器 [10] と k-最近傍分類器 (k-NN)[11] がある。単純ベイズ分類器はテキストがイベント情報に当たる確率を計算する。これはテキストがイベントに関係する確率である。単純ベイズ分類器の処理時間は訓練標本の数に線形関係がある。訓練標本が多い場合，処理時間は非常に長い。そして，確率の判別境界を決めるのは困難である。k-最近傍分類器は目標テキストと各訓練標本とのコサイン類似度を計算する。類似度の一番小さい訓練標本によって目標テキストのクラスを判別する。しかし，k-最近傍分類器の処理時間も訓練標本の数に線形関係がある。さらに，ノイズのある訓練標本に対し，k-最近傍分類器は過学習になる傾向がある。

多くの研究 [12][13] は Support Vector Machine(SVM) がテキスト分類問題での効率性を証明する。F 値によって SVM の性能は単純ベイズ分類器，k-最近傍分類器，決定木，ニューラルネットワークなどより高いことが指摘された。特に，単語ベクトル空間のような高次元（10,000 次元より多い）特徴空間に対し，SVM 分類器は優れたパフォーマンスを示す。さらに，SVM は過学習の問題が回避できる。本研究では SVM を用いてイベント情報の認識を実現する。SVM[14] は分類と回帰分析用の教師あり機械学習の一つである。特に，少ない標本での分類問題に高い性能を示す。そして，テキスト分類タスクのような高次元分類問題に適用できる [15]。

まず，SVM 分類器への入力が浮動小数点数で組み合わせた行ベクトルであるため，入力文字列を単語ベクトル空間にマッピングしなければならない。そのため，単語の辞書を導入する。その辞書はすべての単語を記載する。各単語は浮動小数点数の唯一の ID を持つ。本研究では，MeCab-IPADIC[16] と呼ばれる辞書を用いて単語ベクトル空間を構築する。式 1 に示すように，単語ベクトル $\vec{V}$ のサイズ n は辞書の容量 $size(D)$ に設定する。

図 4　単語ベクトル空間のマッピング

$$\vec{V} = [w_1, w_2, ..., w_n] \quad n = size(D) \tag{1}$$

式 1 に示すように，単語ベクトル空間にある一つの単語 D_i は入力文字列にもある場合，その値を 1.0 に設定し，ない場合 0.0 に設定する。

$$w_i = \begin{cases} 1.0 & D_i \in T \\ 0.0 & D_i \notin T \end{cases} \tag{2}$$

図 4 は単語ベクトル空間マッピングの例を示す。イベントブロックにある単語が単語ベクトル空間で 1.0 に設定され，その他は 0.0 に設定される。関連研究は $tf - idf$ のような重み付きの手法を用いて分類精度を上げることがある [17]。多値分類分問題に対して重み付きの手法は分類精度の向上に有益であることが指摘されるが，本研究は二値分類問題である。SVM 分類器は idf 方法の代わりに最適な分類超平面を計算できるため，idf を付ける必要はない。一方，イベントベクトルに含まれるテキストの長さは短いため，tf を付ける必要もない。そのため，本研究では重み付きの手法を用いない。

さらに，英語と異なり，日本語は接着型言語である。すなわち，単語の間にスペースが付いていない。日本語の文を単語に分割するため，形態素解析器が必要である。我々は KUROMOJI[*5] を用いて日本語単語の分割を処理する。

それから，SVM 分類器を実現する。Chang[18] らの開発した LIBSVM[*6] というオープンソースの SVM ライブラリがある。LIBSVM を実装する前，SVM 数式と SVM カーネル関数を事前に設定しなければいけない。SVM 数式は最適分類超平面を計算するための数式である。訓練標本の内にある正例と負例の数がほぼ同じ場合，式 3 に示す C-SVM を採用する。

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \\ & y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0, i = 1, \ldots, l. \end{aligned} \tag{3}$$

*5　kuromoji: http://www.atilika.org/
*6　LIBSVM: http://www.csie.ntu.edu.tw/ cjlin/libsvm/

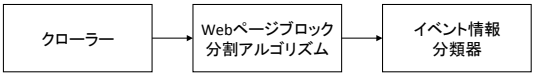

図 5　実験システムの構造

C-SVM はパラメータ C がある。それは外れ値のペナルティ係数である。第 5 章でパラメータ C についての最適なパラメータ検索を説明する。

カーネル関数は単語ベクトル空間をもっと高次元空間にマッピングする。多くのテキスト分類タスクは線形分離可能なので，式 4 の線形カーネル関数を採用する [15][19][20]。

$$K(x_i, x_j) = x_i^T x_j \tag{4}$$

線形カーネル関数は単語ベクトル空間を高次元空間にマッピングしない。それは一番速いカーネル関数である。

5. 検証実験

本章では，実験システムを設計して Web ページブロック分割アルゴリズムとイベント情報分類器を評価する。図 5 に示すように，実験システムは三つのステップがある。

クローラーは一般的な JAVA インタフェースで開発される。それはインタネットから店舗のホームページを HTML ドキュメントとしてダウンロードする。ダウンロードされた Web ページローカルで保存して Web ページブロック分割アルゴリズムに送る。

Web ページブロック分割アルゴリズムを評価するため，名古屋駅地下街「ユニモール」と「エスカ」にある店舗 96 軒を実験対象とする。式 5 に示すように，アルゴリズムの再現率を評価する。

$$\text{再現率} = \frac{\text{抽出されたイベントブロック数}}{\text{実際的なイベントブロック数}} \tag{5}$$

再現率は実際的なイベントブロック数にある抽出されたイベントブロック数の割合である。その結果は表 3 に示される。96 個ホームページのうち，30 個がオフラインまたはイベント情報がない。その他の 66 個が利用可能である。その結果，再現率は 91.35%になる。高い再現率を達成する。

23,761 個に分割されたブロックを，手動でフィルタリングする。三つの指標でイベント情報認識を評価する。式 6 に示すように，それぞれは適合率，再現率と F1 値がある。

$$\begin{aligned} \text{適合率} &= \frac{\text{正しく分類したイベント情報}}{\text{分類したイベント情報}} \\ \text{再現率} &= \frac{\text{正しく分類したイベント情報}}{\text{実際的なイベント情報}} \\ F_1 &= 2 \cdot \frac{\text{適合率} \cdot \text{再現率}}{\text{適合率} + \text{再現率}} \end{aligned} \tag{6}$$

C-SVM にあるパラメータ C に対する最適なパラメータ検索をする。C の値は， 2^{-5} から 2^5 まれそれぞれにループし，三つの指標を観察する。表 6 に示すように，左の Y

表 3　Web ページブロック分割の実験結果

url	イベント	再現率
http://www.komeda.co.jp/ (54 Homepages) http://www.akakura.jp/	425	100.00%
http://www.n-rs.co.jp/	30	96.67%
http://www.kikuchi-megane.co.jp/	11	90.91%
http://www.pokkacreate.co.jp/	913	89.05%
http://www.honeys.co.jp/	12	83.33%
http://www.hokennomadoguchi.com/	6	83.33%
http://www.world.co.jp/soup/	5	80.00%
http://www.hokennomadoguchi.com/	6	50.00%
https://www.facebook.com/	3	00.00%
http://www.riochain.co.jp/	1	00.00%
http://www.r-p-s.net/	3	00.00%
http://www.schiatti.co.jp/	1	00.00%
http://www.erina-t.com/	6	00.00%
	平均値	91.35%

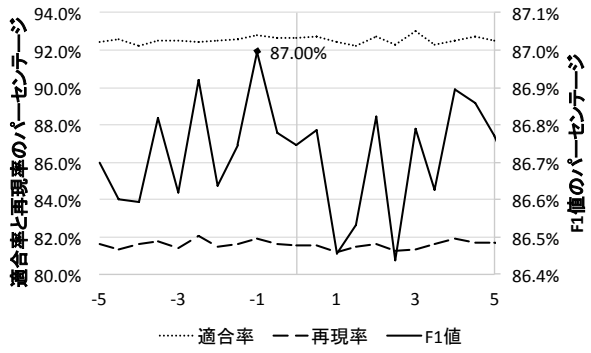

図 6　最適なパラメータ C 検索

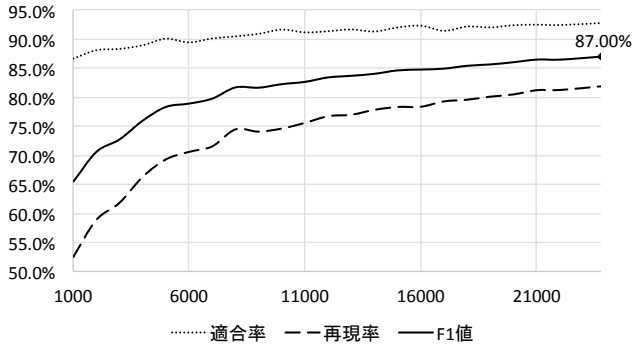

図 7　イベント情報分類器の評価

軸は適合率と再現率を表示する。右の Y 軸は F1 値を表示する。横軸 X は $\log C$ である。表 4 に示すように，最適なパラメータ C は 2^{-1} になる。

もう一つの実験は訓練標本容量と三つの指標との推移を表示する。この実験でパラメータ C が 2^{-1} に設定され

表 4　最適なパラメータ C 検索結果

C	適合率	再現率	F1 値
-5	92.41%	81.65%	86.70%
-4.5	92.61%	81.33%	86.60%
-4	92.23%	81.61%	86.59%
-3.5	92.53%	81.77%	86.82%
-3	92.53%	81.42%	86.62%
-2.5	92.40%	82.06%	86.92%
-2	92.49%	81.48%	86.64%
-1.5	92.55%	81.63%	86.75%
-1	92.79%	81.88%	**87.00**%
-0.5	92.62%	81.63%	86.78%
0	92.66%	81.54%	86.75%
0.5	92.74%	81.55%	86.79%
1	92.45%	81.19%	86.45%
1.5	92.23%	81.50%	86.53%
2	92.70%	81.64%	86.82%
2.5	92.29%	81.28%	86.44%
3	92.99%	81.36%	86.79%
3.5	92.30%	81.61%	86.63%
4	92.52%	81.91%	86.89%
4.5	92.72%	81.70%	86.86%
5	92.53%	81.69%	86.77%
5.5	92.43%	81.54%	86.64%

る。90%の訓練標本と 10%のテスト標本による相互検証を行う。標本のサイズを 1,000 から 23,761 まで 1,000 ずつ増やして実験をする。図 7 に示すように，横軸 X は標本のサイズを表示する。F1 は標本のサイズの増加に伴って増える。表 5 は具体的なデータを示す。標本のサイズは 23,761 になった場合，F1 値は最大値の 87.00%となる。高い分類精度を達成する。このイベント情報の分類器は利用可能であると証明される。さらに，図 7 に示すように，F1 は標本のサイズの増加に伴ってまた増加し続ける可能性がある。将来的には，クラウドソーシングを用いて訓練標本の補充とノイズ削減を考える予定である。

6. おわりに

本研究で提案するイベント情報の抽出する方法は，店舗のホームページやブログからクーポンやキャンペーンなどのイベント情報を抽出する方法である。この方法を利用してユーザはひとつひとつの店舗のホームページを閲覧を必要とせず，イベント情報抽出の網羅性と効率性が支援できる。本提案は Web ページブロック分割およびイベントデータ認識の二つのタスクを扱う。一つ目のタスクは Web ページをブロックに分割する。分割された各ブロックはタイトルや説明文や日付などのイベント情報を含むと考えられる。従来の研究は特定なタグ，画面構成あるいはブロッ

表 5 イベント情報分類器の評価結果

標本容量	適合率	再現率	F1 値
1000	86.60%	52.50%	65.37%
2000	88.11%	58.86%	70.57%
3000	88.33%	61.81%	72.73%
4000	88.96%	66.28%	75.96%
5000	90.09%	69.29%	78.33%
6000	89.45%	70.59%	78.91%
7000	90.13%	71.54%	79.76%
8000	90.49%	74.46%	81.70%
9000	90.94%	74.08%	81.65%
10000	91.66%	74.61%	82.26%
11000	91.18%	75.59%	82.66%
12000	91.37%	76.74%	83.42%
13000	91.69%	76.99%	83.70%
14000	91.32%	77.84%	84.04%
15000	92.03%	78.33%	84.63%
16000	92.33%	78.36%	84.78%
17000	91.44%	79.29%	84.93%
18000	92.20%	79.58%	85.43%
19000	92.03%	80.11%	85.66%
20000	92.41%	80.49%	86.04%
21000	92.50%	81.22%	86.49%
22000	92.44%	81.25%	86.49%
23000	93.01%	81.57%	86.91%
23761	92.79%	81.88%	**87.00**%

クの機能などを特徴量として Web ページを分割することが多い。本研究では HTML 構造解析に基づく Web ページをブロックに分割する。二つ目のタスクは分割されたブロックからイベント情報を識別する。本研究では機械学習の手法を用いてイベント情報の識別を実現する。検証実験は名古屋駅地下街「エスカ」と「ユニモール」にある店舗 96 軒を対象として行。検証実験の結果によって提案する Web ページブロックアルゴリズムとイベント情報識別アルゴリズムが利用可能とった分かった。

参考文献

[1] Chenyi L., Katsuhiko K., Kei H., Nobuo K.: Design and Implementation of Event Information Summarization System, CDS workshop of COMPSAC, 2014.
[2] 廖 宸一, 梶 克彦, 廣井 慧, 河口 信夫: 時空間情報に基づくイベント情報の集約システムの開発, DICOMO, pp.646-656, 2014.
[3] Lin, S.-H., Ho, J.-M., Discovering Informative Content Blocks from Web Documents, In Proceedings of ACM SIGKDD'02, 2002.
[4] Crivellari, F., Melucci, M., Web Document Retrieval Using Passage Retrieval, Connectivity Information, and Automatic Link Weighting–TREC-9 Report, In The Ninth Text REtrieval Conference (TREC 9), 2000.
[5] Kovacevic, Milos, et al. Recognition of Common Areas in a Web Page Using Visual Information: a possible application in a page classification. In: Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on. IEEE. p. 250-257, 2002.
[6] Cai D., Yu S., Wen J.-R., Ma W.-Y., VIPS: a vision-based page segmentation algorithm, Microsoft Technical Report, MSR-TR-2003-79, 2003.
[7] Cai, Deng, et al. Extracting content structure for web pages based on visual representation. In: Web Technologies and Applications. Springer Berlin Heidelberg. p. 406-417, 2003.
[8] Lan Y.; Bing L.; Xiaoli L.. Eliminating noisy information in web pages for data mining. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM. p. 296-305, 2003.
[9] Rockley, Ann; Kostur, Pamela; MANNING, Steve. Managing enterprise content: A unified content strategy. New Riders, 2003.
[10] Lewis, David D. Naive (Bayes) at forty: The independence assumption in information retrieval. In: Machine learning: ECML-98. Springer Berlin Heidelberg. p. 4-15, 1998.
[11] Yang, Yiming; LIU, Xin. A re-examination of text categorization methods. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. ACM. p. 42-49, 1999.
[12] Joachims, Thorsten. Text categorization with support vector machines: Learning with many relevant features. Springer Berlin Heidelberg, 1998.
[13] Serastiani, Fabrizio. Machine learning in automated text categorization. ACM computing surveys (CSUR), 34.1: 1-47, 2002.
[14] Cortes, Corinna; Vapnik, Vladimir. Support-vector networks. Machine learning, 20.3: 273-297, 1995.
[15] Joachims, Thorsten. Transductive inference for text classification using support vector machines. In: ICML. p. 200-209, 1999.
[16] Taku K., Kaoru Y., Yuji M.: Applying Conditional Random Fields to Japanese Morphological Analysis, Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP-2004), pp.230-237, 2004.
[17] Soucy, Pascal; MINEAU, Guy W. Beyond TFIDF weighting for text categorization in the vector space model. In: IJCAI. pp. 1130-1135, 2005.
[18] Chih-Chung C.; Chih-Jen L.: LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology (TIST), 2.3: 27, 2011.
[19] Huma L., Craig S., John Shawe-Taylor, Nello Cristianini, Chris Watkins: Text Classification using String Kernels, The Journal of Machine Learning Research, Volume 2, pp. 419-444, March 2002.
[20] Pilaszy, Istvn. Text categorization and support vector machines. In: The proceedings of the 6th international symposium of Hungarian researchers on computational intelligence. 2005.
[21] Bruns, Andreas; KORNSTADT, Andreas; WICHMANN, Dennis. Web application tests with selenium. Software, IEEE, 26.5: 88-91, 2009.