Vol.2009-DD-73 No.4
2009/9/25

Webページを対象としたXMLデータ抽出手法の検討

天笠 俊之[†1,†2] ゴー シー ヴィエト フー[†2]
北川 博之[†1,†2]

Webコンテンツなどの非構造情報からレコード情報を抽出する手法として情報抽出手法が注目され，この数年活発に研究されている．しかし，既存の手法は基本的に「(会社，所在地)」のような二項関係，あるいは単純な構造を持つレコード情報に特化しており，XMLのような複雑な構造を抽出する手法はあまり議論されていない．そこで本研究では，既存の情報抽出手法を組み合わせることで，複雑な情報を抽出する手法について議論する．種となるXMLデータからスキーマ情報を抽出し，XMLデータを関係表に写像する．写像した関係表を二項関係に分解し，既存のレコード抽出手法を適用する．得られた結果からXMLデータを再構成することによってXMLデータの抽出を実現する．予備実験により手法の妥当性を示すとともに，今後の研究課題について議論する．

A Scheme of XML Data Extraction from Web Pages

Toshiyuki Amagasa ,[†1,†2] Ngo Sy Viet Phu [†2]
and Hiroyuki Kitagawa [†1,†2]

Information extraction has been gaining increasing attention as a mean to extract (structured) record data out of (unstructured) documents such as Web pages. However, existing techniques are basically aiming at extracting simple records, such as binary relationship like "(company, location)," and extraction of complex records has not been deeply researched. For this reason, in this paper, we discuss an information extraction scheme that allows us to extract complex records like XML by combining conventional information extraction schemes and databases. Given a set of seed records mostly in the form of XML data, we firstly infer the schema information from XML data if it is not provided. Then, we transform the XML data to relational table by applying an existing technique. The obtained relational tables are decomposed into a set of binary relations, and they are forwarded to the record extraction system. We reconstruct XML data from the results obtained from the record the extraction system. We perform preliminary experiment to show the feasibility of the scheme, and discuss future research issues.

1. はじめに

構造を持たないテキストデータから，レコードデータのように構造化された情報を抽出する技術を情報抽出 (Information Extraction) と呼ぶ．古くは1980年代から，ニュース記事や文献データベースなどのテキストデータベースを対象とした研究が行われてきた[1)]．1990年代後半から，Webの爆発的な普及と検索エンジンの高性能化に伴い，Webを情報源とする情報抽出手法が脚光を浴び，盛んに研究が行われている[2),3)]．

情報抽出のタスクは，対象とする情報から大まかに固有表現抽出 (Named Entity Extraction)，レコード抽出 (Record Extraction)，トピック抽出 (Topic Extraction) に分類することができる．固有表現抽出は，主にテキスト中の実体情報に対するタグ付け，レコード抽出はテキストから二項関係，あるいはN項関係の抽出，トピック抽出は，テキストが表現しているトピック情報を抽出することを目的としている．本論文では，レコード抽出を目的とする情報抽出を対象に議論する．

レコード抽出手法は，大まかに，抽出すべき正解レコードを含む学習データが事前に与えられる教師あり学習手法とそれ以外に分けることができる．学習データ型の手法では，レコード情報に関する特徴を機械学習手法で学習し，得られた分類器をもとにテキストデータベースから実際のレコードを抽出する．主な手法としては，滑り窓 (Sliding Window) によって短いフレーズを多数抽出し，Naive Bayesなどを使って学習する手法や，隠れマルコフモデルなどの有限状態オートマトンによって情報を含むパターンを学習する手法などが代表的である．ただし，この手法は事前に正解集合を利用者の手によって準備しないといけないという問題点がある．

これに対し，学習に依存しないブートストラップ型の情報抽出手法が提案されている．これは，少数の種 (Seed) レコードを入力として与え，まず種レコードを含む文書を検索エンジンを使って取得する．得られた文書から，種レコード周辺の特徴的なパターンを抽出し，次はそのパターンを使って検索エンジンに再度問合せを行う．得られた文書から新たなレ

†1 筑波大学計算科学研究センター
Center for Computational Sciences, University of Tsukuba
†2 筑波大学大学院システム情報工学研究科コンピュータサイエンス専攻
Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba

コード情報を抽出するとともに，新たなパターンを取得する．この処理をパターンとレコードの双方を増やしながら繰り返すことによって，大量のレコード情報を抽出する．

既存の情報抽出手法において，抽出の対象とする情報の粒度は，固有表現 (Entity)，二項関係，N 項関係（レコード）などさまざまであるが，基本的には単純な構造を扱っている．N 項からなるレコードであっても，高々数属性を持つレコードを対象とすることがほとんどであり，抽出精度や効率の問題からあまり複雑なレコードは想定していない．しかしながら，現実の応用を考えると，ある程度複雑な構造を持ったレコードを Web から抽出することは大きなニーズを持っていると思われる．

そこで，本稿では複雑な構造を持ったレコードの抽出について議論する．複雑な構造を持つレコードの例として，ここでは XML データを取り上げる．すなわち，複数のレコードを含む XML データをシードとして与え，システムはそのシードから類推されるレコードを Web から抽出し，XML データに変換した上で利用者に返却する．このとき，複雑なレコード情報を抽出する目的のために新たな手法を開発するより，既存の手法を組み合わせる方が効果的であると考える．すなわち，例示レコード (XML) を単純な構造に分解し，既存の情報抽出手法の入力として与え，得られた結果を再構成するというものである．その際，以下のような技術的課題が考えられる．

- XML データから情報抽出の入力に用いるレコード型データへの変換をどのように行うか．
- 得られたレコード出たから，情報抽出モジュールへの与える例示レコードをどのように生成するか．

この問題に対し我々は，以下のアプローチを取る．

- シードとして与えられた XML データに対して，XML データから関係スキーマを導出する手法を利用して関係表への変換を行う。
- 得られた関係表を、主キーに着目しながら二項関係に分解する。
- 得られたそれぞれの二項関係について、既存のレコード抽出を行う。
- 結果として得られた二項関係から、今度は逆に関係表を再構成し、XML データへ変換する。

本稿の構成は以下の通りである．2 節では情報抽出手法の概要について述べる．3 節では提案手法を説明し、4 節の予備実験で提案手法の妥当性を検証する。5 節はまとめである。

2. 情報抽出手法の概要

これまで人間が創出してきた情報の多くが，電子データ，非電子データを問わず，テキストの形式でとして蓄積されている．テキストデータは構造化データのような明示的な構造を持たないため，電子的な処理を考えるとき，テキストデータから構造化されたデータを抽出することは重要である．このため，情報抽出の手法は古くから研究がなされてきた[1)–3)]．ここでは，紙数の都合から，ブートストラップ型の情報抽出手法について，その概要を説明する．

2.1 ブートストラップ型情報抽出手法

ブートストラップ型の情報抽出手法は，少数のレコードが入力として与えられる，準教師ありの情報抽出手法に分類される．教師ありの手法に比べて，大量の学習データを用意する必要がない，Web スケールの大規模なテキストデータを対象に効率的な情報抽出が可能であることなどから，近年注目されている手法である[4)–8)]．

以下，DIPRE[4)] を例に取って処理の概要を説明する．

(1) 少数の例示レコードを用意する．例えば，“(author, book)” のペアに関する情報を抽出するとすると，例えば

“Arthur Conan Doyle, The Adventures of Sherlock Holmes)”

が種レコードとなる．

(2) Web 検索エンジンを使い，種レコードを含む Web ページを探索する．さらに，種レコードを含む Web ページから，レコード周辺のタプルを導出する．DIPRE の場合，

```
[order, author, book, prefix, suffix, middle]
```

がタプルの例となる．ここで order とは author と book の文書中での出現順を示しており，1 の場合は author が先に表れることを，0 の場合はそれ以外を示している．prefix と suffix は book あるいは author にマッチする部分のそれぞれ前後 10 文字，middle は book と author の間の文字を示している．Web ページ中に

“Read The Adventures of Sherlock Holmes by Arthur Conan Doyle on line or in your email”

という表現があったとすると、

```
[0, Arthur Conan Doyle, The Adventures of Sherlock Holmes, Read,
on line or, by]
```

がタプルとして抽出される．別の出現箇所に，

"When Sir Arthur Conan Doyle wrote The Adventures of Sherlock Holmes in 1892 he was high"

のような場所があったとすると，ここからは，

```
[1, Arthur Conan Doyle, The Adventures of Sherlock Holmes, When Sir,
in 1892 he, wrote]
```

が抽出される．

(3) 得られたタプルを order と middle でグループ化し，共通のパターンを取得する．パターンは以下のように記述される．

```
[longest-common-suffix of prefix strings, author, middle, book,
longest-common-prefix of suffix strings]
```

例えば，上で示したタプルからは，

```
[Sir,.*?,wrote,.*?,in1892]
```

パターンとして抽出される．ここで "*?" はワイルドカードを示している．

(4) パターンを検索キーとして再度 Web 検索エンジンに問合せを行い，新たなレコードを取得する．以上の処理をある基準が満たされるまで繰り返すことによって，多くのレコードを取得する．

このように，DIPRE はパターンとレコードの双対性に着目して，繰り返しのプロセスの中で双方を増加させるというのが基本的な考え方である．

これに対して，Snowball[5] では，パターン抽出の精度を向上させるため，あらかじめ固有表現抽出器によって文書を処理し，固有表現に対してタグ付けを行っておく．また，タプルのグルーピングに類似度を導入し，より多くの単語を共有するタプルを類似していると判定する仕組みを導入し，パターンにも確信度を追加している．これによって抽出精度を向上している．

QXtract[6] では，レコード抽出の効率を改善するため，種レコードから生成する問合せについて，それらを全て処理するのではなく，有用である（多数のレコードを抽出できる）と推定される問合せのみを処理するという仕組みを取り入れている．このために，機械学習に基づく分類器を事前に学習しておき，検索エンジンに問合せを投入する前に，判別器でその有用性を判定する．

これまでに述べた手法では，利用者が事前に種レコードを与えるという前提であったが，KnowItAll[7] では，抽出したい実体のクラス，例えば "city", "scientist", "movie" などを与えるという問題に取り組んでいる．与えられたクラスに関するレコード情報を全て抽出するため，KnowItAll では，手作業で作成された汎用のパターンを用いている．例えば，"NP1 such as NP2", "NP1 is a NP2" などはそのパターンの例である．

張等[9] は，抽出されたレコードに対して，利用者が適合フィードバックを行うことによってパターンの評価を行う．これにより，より精度と効率の高い抽出を可能にしている．

このように，Web スケールの文書データ集合から効率的にレコード抽出を行なう手法が種々提案されており，それぞれについて，入力データ，前処理の要・不要などに細かい差異が存在する．また，これらの手法で抽出されるレコードは基本的には二項関係である．

2.2 複雑なレコードの抽出

二項関係以上の複雑なレコードの抽出に関する研究はこれまでにはあまり例がない．McDonald 等[10] は，多項関係を二項関係に分解して既存のレコード抽出を行い，得られた結果から多項関係を再構成するアプローチでこの問題に対処している．多項関係の分解には判別器を用いて適切な二項関係を選択する点，得られたレコード集合から多項レコードを再構成するために，実体グラフ上の極大クリークを探索する点などに特徴がある．

3. Web ページからの XML データの抽出

本稿では，例示レコードとして XML データが与えられたときに，それを種としていかに Web から類似する XML データを抽出するかを議論する．前提として，前述したいくつかのレコード抽出アルゴリズムが適用可能な状態であるとする．また，情報源として既存のデータベースを利用することも想定する。

3.1 XML に関するスキーマ情報の取得

システムへの入力は，種 XML データとそのデータ関するスキーマ情報 (DTD, W3C XML Schema, RELAX NG 等) であるとする．XML は必須であるが、スキーマ情報は必須ではなく，スキーマ情報が与えられない場合は，XML データから類推する．スキーマ情報の類推には，既存の手法を用いる[11),12)]．図 1 に種 XML データ、図 2 に種 XML データの文書型定義 (DTD) を示す．

Vol.2009-DD-73 No.4
2009/9/25

得られたスキーマ情報からスキーマグラフを導出する．スキーマグラフは，各ノードが要素あるいは属性を表し，枝は要素と要素の親子関係，もしくは要素と属性の所属関係を表している．これは後述の関係表への写像の処理に用いられる．図 3 に文書型定義から導出されたスキーマグラフの例を示す．

```
<CompanyInformation>
  <Company>
    <CompanyName>Apple</CompanyName>
    <CEO>Steve Jobs</CEO>
    <Products>
      <Name>MacBook</Name>
      <Name>MacBook Pro</Name>
    </Products>
  </Company>
  <Company>
    <CompanyName>Microsoft</CompanyName>
    <CEO>Steve Ballmer</CEO>
    ...
  </Company>
</CompanyInformation>
```

図 1 種 XML データの例.
Fig. 1 An example of seed XML data.

```
<!ELEMENT CompanyInformation (Company*)>
<!ELEMENT Compay (CompanyName, CEO?, Products?)>
<!ELEMENT CompanyName (#PCDATA)>
<!ELEMENT CEO (#PCDATA)>
<!ELEMENT Products (Name*)>
<!ELEMENT Name (#PCDATA)>
```

図 2 種 XML の DTD.
Fig. 2 DTD for the seed XML data.

3.2 XML データから関係表への写像

XML データを（スキーマ情報を利用しながら）関係表に写像する．XML データを関係表に写像する研究もこれまでに多数なされている．例えば，[11),12)] はその例である．[11)] では，Basic Inlining, Shared Inlining, Hybrid Inlining の方式を提案しており，それぞれによって得られる結果が異なる．それらの中から適切なものを選択して用いる．

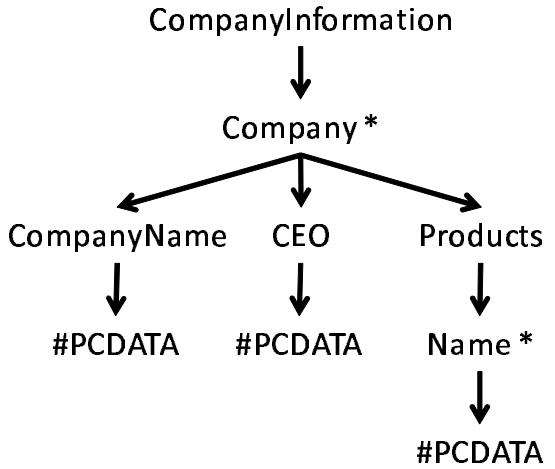

図 3 スキーマグラフ.
Fig. 3 Schema graph.

例えば，図 1 の XML は次のように変換される．ここで，各カラム名の下線はそのカラムが主キーであることを示している．また，CompanyInformation 要素の情報はここでは割愛している．

Table: Company

CompanyName	CEO
Apple	Steve Jobs
Microsoft	Steve Ballmer
...	...

Table: Products

CompanyName	Name
Apple	MacBook
Apple	MacBook Pro
...	...

図 4 種 XML データから写像して得られた関係表.
Fig. 4 Relations obtained from seed XML data.

Vol.2009-DD-73 No.4
2009/9/25

3.3 二項関係への分解

得られた関係表を二項関係に分解する．分解は基本的に各関係表の主キーを軸に，主キー属性以外の属性をそれぞれペアにして分解する．この例では，すでに二項関係になっているが，例えば，

Company (CompanyName, CEO, Location)

のような表があったとすると，

Company_CEO (CompanyName, CEO)

Company_Location (CompanyName, Location)

のように分解される．

3.4 レコード抽出の実行と XML の再構成

分解して得られた関係表を入力としてレコード抽出アルゴリズムを実行し，多数のレコードを取得する．得られた二項関係を，元の関係表を経由して XML データとして再構成し，利用者に返却する．

4. 予 備 実 験

提案手法の妥当性を評価するために予備実験を行った．検索エンジンには Yahoo! を用い，実装には PHP のフレームワークである Zend[⋆1] を用いた．実験に使用した PC の CPU は Pentium 4 (3.00GHz) であり，主記憶容量は 512MB である．

本予備実験では，会社名，所在地，最高営業責任者を記録した XML データを例示レコードとして与えた．例示レコードを図 5 に示す．この種データを関係スキーマに写像した結果，以下の二つの表が得られた．

Company_CEO (ComName, CEO)

Company_Location (ComName, Location)

この個別の関係表について DIPRE によってレコード抽出を行い，XML の再構成までの処理を行った．

4.1 結　　果

4 回のイテレーションを行った際のに抽出されたパターン数とレコード数を表 1 に示す．

- (ComName, CEO) の二項関係については，合計 414 レコードが抽出された．結果を人手で精査した結果 16.18%（67 レコード）はノイズであった．

⋆1 http://zendframework.com/

```
<CompanyList>
  <Company>
    <ComName>Microsoft</ComName>
    <CEO>
      <CEOName>Steve Ballmer</CEOName>
    </CEO>
    <Location>Redmond</Location>
  </Company>
  <Company>
    <ComName>Yahoo</ComName>
    <CEO>
      <CEOName>Jerry Yang</CEOName>
    </CEO>
    <Location>Sunnyvale</Location>
  </Company>
  <Company>
    <ComName>IBM</ComName>
    <CEO>
      <CEOName>Samuel Palmisano</CEOName>
    </CEO>
    <Location>Armonk</Location>
  </Company>
  <Company>
    <ComName>Apple</ComName>
    <CEO>
      <CEOName>Steve Jobs</CEOName>
    </CEO>
    <Location>Curpotino</Location>
  </Company>
  <Company>
    <ComName>Google</ComName>
    <CEO>
      <CEOName>Eric Schmidt</CEOName>
    </CEO>
    <Location>Mountain View</Location>
  </Company>
</CompanyList>
```

図 5 予備実験で用いた種 XML データ．

表 1 抽出パターン数とレコード数（左：(ComName, CEO)，右：(ComName, Location)）．
Table 1 # of patterns and records extracted (Left: (ComName, CEO), Right: (ComName, Location).

イテレーション	パターン数	レコード数
1	9	23
2	32	156
3	89	405
4	106	414

イテレーション	パターン数	レコード数
1	7	21
2	27	162
3	68	245
4	75	258

- (ComName, Location) の二項関係については，合計 258 のレコードが抽出され，そのうち 18.6%（48 レコード）はノイズであった．
- 個別に抽出された二項関係を元の関係表に復元する際，CEO, Location 双方の情報を兼ね備えていたレコード数は 59 であった．

結果として抽出された XML データのうち，興味深いものをいくつかを図 6 に示す．左側の (A), (B) では、CEO の情報が新たに追加されている．また，右側 (C), (D) は種 XML データには全く含まれていなかったデータである．

```
(A)
<Company>
  <ComName>Microsoft</ComName>
  <CEO>
    <CEOName>Steve Ballmer</CEOName>
    <CEOName>Bill Gates</CEOName>
  </CEO>
  <Location>Redmond</Location>
</Company>

(B)
<Company>
  <ComName>Yahoo</ComName>
  <CEO>
    <CEOName>Jerry Yang</CEOName>
    <CEOName>Carol Bartz</CEOName>
  </CEO>
  <Location>Sunnyvale</Location>
</Company>
```

```
(C)
<Company>
  <ComName>Sony</ComName>
  <CEO>
    <CEOName>Howard Stringer</CEOName>
  </CEO>
  <Location>Moutain View</Location>
</Company>

(D)
<Company>
  <ComName>Sony Ericsson</ComName>
  <CEO>
    <CEOName>Komiyama</CEOName>
  </CEO>
  <Location>London</Location>
</Company>
```

図 6 情報抽出によって得られた XML データ.
Fig. 6 A part of XML data generated by the proposed scheme.

4.2 考　　察

この予備実験の結果，以下のことが示唆された．まず，本実験で使用した単純な XML データに付いては，それを関係データに変換した上で従来の情報抽出を適用するアプローチは，それなりにうまく動作する．ただ，個別の二項関係のレコード数に比べて，最終的な XML データを再構成可能なレコードは 1 割程度と少なく，より最終的な XML データを作成するのに貢献するレコードを優先的に取得する仕組みが望まれる．例えば，

- 主キー属性で結合可能でないレコードの情報を優先的に検索エンジンの検索キーワードに投入する．
- スキーマ情報の出現回数 (‘?’, ‘*’) などに着目し，オプショナルな要素は空要素のまま出力する．あるいは，スキーマ定義そのものの制約を緩める (relaxation).

といった対応が考えられる．

これとは別の話題として，複数の情報源，レコード抽出アルゴリズムを組み合わせ，より精度／効率の高い XML データ抽出を目指すという方向性も考えられる．情報源としては，Web ページだけではなく，既存のデータベース等も考えられる．この場合，どのレコードを優先的に Web ページ取得のキーワードとして利用するかといった工夫や，二項関係を抽出する順番を入れ換え，より効率的な処理を目指すことも考えられる．

5. ま と め

本稿では，既存の情報抽出手法を組み合わせることで，XML データなど複雑な構造を持つレコードを抽出する手法を提案した．種となる XML データからスキーマ情報を抽出し，XML データを関係表に写像する．写像した関係表を二項関係に分解し，既存のレコード抽出手法を適用する．得られた結果から XML データを再構成することによって XML データの抽出を実現する．予備実験により，手法の妥当性を示すことができた．今後は，4.2 節で述べたような，複数のレコード抽出アルゴリズムのサポートや，情報源の特徴を考慮した精度／効率の高い XML データ抽出について取り組む予定である．

謝辞 この研究の一部は，科学研究費補助金 特定領域研究 (#21013004)，若手研究 (B) (#21700093) によるものである．ここに記して謝意を表す．

参 考 文 献

1) Appelt, D.E. and Israel, D.: Introduction to Information Extraction Technology, `http://www.ai.sri.com/~appelt/ie-tutorial/` (1999). IJCAI-99 Tutorial.
2) Chang, C.-H., Kayed, M., Girgis, M.R. and Shaalan:, K.F.: A Survey of Web Information Extraction Systems, *IEEE Trans. Knowl. Data Eng.*, Vol.18, No.10, pp. 1411–1428 (2006).
3) Etzioni, O., Banko, M., Soderland, S. and Weld, D.S.: Open information extraction from the web, *Communications of the ACM*, Vol.51, No.12, pp.68–74 (2008).
4) Brin, S.: Extracting Patterns and Relations from the World Wide Web., Technical Report 1999-65, Stanford InfoLab (1999). Previous number = SIDL-WP-1999-0119.
5) Agichtein, E. and Gravano, L.: Snowball: extracting relations from large plain-text collections, *Proc. ACM DL 2000*, pp.85–94 (2000).
6) Agichtein, E. and Gravano, L.: Querying Text Databases for Efficient Information

Vol.2009-DD-73 No.4
2009/9/25

Extraction, *Proc. ICDE 2003*, pp.113–124 (2003).
7) Etzioni, O., Cafarella, M.J., Downey, D., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D.S. and Yates, A.: Unsupervised named-entity extraction from the Web: An experimental study, *Artif. Intell*, Vol.165, No.1, pp.91–134 (2005).
8) Cafarella, M. J., Downey, D., Soderland, S. and Etzioni, O.: KnowItNow: Fast, Scalable Information Extraction from the Web, *Proc. HLT/EMNLP 2005* (2005).
9) Zhang, J., Ishikawa, Y. and Kitagawa, H.: Record Extraction Based on User Feedback and Document Selection, *Proc. APWeb/WAIM 2007*, pp.574–585 (2007).
10) McDonald, R., Pereira, F., Kulick, S., Winters, S., Jin, Y. and White, P.: Simple algorithms for complex relation extraction with applications to biomedical IE, *Proc. ACL 2005*, pp.491–498 (2005).
11) Shanmugasundaram, J., Tufte, K., Zhang, C., He, G., DeWitt, D.J. and Naughton, J.F.: Relational Databases for Querying XML Documents: Limitations and Opportunities, *Proc. VLDB 1999*, pp.302–314 (1999).
12) Bohannon, P., Freire, J., Haritsa, J.R., Ramanath, M., Roy, P. and Siméon, J.: LegoDB: Customizing Relational Storage for XML Documents, *Proc. VLDB 2002*, pp.1091–1094 (2002).