社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－DBS－140 (II) (82)
2006／7／14

データクリーニングを統合した情報抽出システムの提案

石川 佳治[†], 黒川 沙弓[††], 張 建偉[††], 北川 博之[††,§]
[†]名古屋大学情報連携基盤センター
[††]筑波大学システム情報工学研究科コンピュータサイエンス専攻
[§]筑波大学計算科学研究センター
ishikawa@itc.nagoya-u.ac.jp,{zjw,saku39}@kde.cs.tsukuba.ac.jp, kitagawa@cs.tsukuba.ac.jp

概要

ウェブなどの大量のテキスト情報源から有用な情報を取得するための情報抽出は，データ工学における重要な研究課題の一つとなっている．有用な情報抽出のためには，抽出結果に含まれる誤りやノイズを削減することが求められる．そこで，本研究では情報抽出にデータクリーニングを統合し，ユーザによる対話的なフィードバックを利用することにより，精度の高い情報抽出システムの提案を行う．ブートストラップ型のレコード抽出手法の処理プロセスにデータクリーニング処理も含め，ユーザからのフィードバックを抽出レコードや抽出パターンの評価に反映させる．

A Proposal of Information Extraction System with Data Cleaning Facility

Yoshiharu Ishikawa[†], Sayumi Kurokawa[††], Jianwei Zhang[††],
Hiroyuki Kitagawa[††,§]
[†]Information Technology Center, Nagoya University,
[††]Department of Computer Science,
Graduate School of Systems and Information Engineering,
University of Tsukuba
[§]Center for Computational Sciences, University of Tsukuba

Abstract

Information extraction to acquire useful information from a large amount of text sources such as Web is one of the important research topics in data engineering. For useful information extraction, errors and noises included in extraction results should be reduced. In this paper, we propose an approach to an information extraction system with high accuracy by integrating data cleaning into information extraction and using interactive feedbacks from users. The approach is based on the bootstrap record extraction method and includes data cleaning in the process of record extraction. User feedbacks are reflected in the evaluation of the extracted records and the extraction patterns.

1 はじめに

ウェブの拡大や様々な情報発信手段の発展により，今日の情報社会ではテキストデータが大量に溢れている．テキストデータに含まれる有用な情報を利用することが求められているが，大量のテキストからユーザが必要とする情報を手作業で見出すことは膨大な作業を要する．そこで，大量のテキストデータから有用な情報を自動的に，あるいは半自動的に抽出する**情報抽出**（information extraction）の技術がますます重要となっている．

一方，自動的に抽出されたデータにはノイズが多く含まれる可能性が高いという問題が存在する．データベース中に存在するノイズデータの除去については，近年**データクリーニング**（data cleaning）に関する研究および開発が盛んに進められている [8]．精度のよい情報抽出を行うアプローチの一つとして，情報抽出の結果に対してこのようなデータクリーニング手法を適用し，誤りの検出や修正を実現することが考えられる．しかし，このアプローチでは情報抽出のプロセスにはフィードバックがなされないことから，情報抽出システムの抽出処理自体には改善がなされず，情報抽出の質を向上させる機会を十分生かしていない．

そこで，本研究では情報抽出のプロセスにデータクリーニングを統合し，抽出結果のみならず情報抽出処理自体の質を向上させるための実現技術を提案する．ブートストラップ型の情報抽出機構をサブシステムとして含み，データクリーニング機構と連携する情報抽出システムのアーキテクチャを提案し，それらの役割と処理内容について述べる．

2 関連研究

2.1 情報抽出

テキストからの情報抽出にはさまざまなアプローチがあるが，本研究では，抽出される情報は自由な形式をとるのではなく，指定された属性値の集まりからなる**レコード**（タプル）の形態をとるものとする．また，特に大量のテキストデータを対象とした情報抽出を対象とし，**ブートストラップ**（bootstrap）型と呼ばれる情報抽出手法の利用を想定している [6]．ブートストラップ型のレコード抽出法の一般的な処理は次のようになる．

1. **例示レコード**の集合（シード集合）がユーザから提示される．
2. 情報抽出システムは，レコード集合をもとに文書リポジトリをスキャンし，例示レコードの**オカレンス**（occurrence；テキスト内のレコード出現）を発見する．
3. レコードのオカレンスの出現コンテクストを分析することで，レコード抽出のための**抽出パターン**を生成する．
4. 抽出パターンを用いて文書リポジトリからのレコード抽出を試みる．その結果得られたレコード集合は既存のレコード集合に統合される．
5. 抽出されたレコード数が収束するまで (2) ～ (4) のステップを繰り返す．

このような繰り返し処理により，小さい例示レコード集合から大量のレコードを抽出することがブートストラップ型の手法の特徴であるが，個々の手法においてはさまざまな工夫がなされている．

ブートストラップ型のレコード抽出法の一つに**DIPRE**（Dual Iterative Pattern Relation Expansion）[4] がある．もともとはHTML文書群からレコード集合を抽出するための手法として提案された，比較的単純な手法である．基本的には上記のブートストラップ型の処理に基づいている．抽出パターンの生成では，レコードのオカレンスの出現コンテクストだけでなく，レコードが発見されたURLのパターンも分析する点に特徴がある．レコードが抽出されたURLを一般化することにより，レコードを含むと想定されるURLのパターンを生成することで，抽出対象とするHTML文書の絞込みを可能とする．

Snowball[1] はDIPREを拡張したアプローチである．DIPREとは異なり，主として構造化されていないテキストからのレコード抽出を対象としている．特徴の一つは，固有表現（named entity）の抽出システムを利用することである．固有表現を用いた抽出パターンを生成することにより，精度の高い抽出処理を実現する．

QXtract[2] は，これらのレコード抽出法を内部で利用する，メタなレコード抽出法である．特徴は，レコード抽出における処理コストの削減に着目した点である．訓練データをもとに，レコードが抽出できる文書群とできない文書群の特徴を，文書分類手法をもとに抽出する点である．この結果を利用して，レコードが抽出できる可能性が高い文書群を中心に抽出処理を行うことで，比較的少ない処理時間で多くのレコードの抽出を目指している．

上記のような自動的なレコード抽出の処理では，誤りやノイズを含んだレコードが抽出されることは避けられない．そのため，たとえばSnowballでは，抽出パターンの有用性に関する評価尺度を導入しており，抽出精度が低いパターンを自動的に棄却する方式が提案されている．しかし，データクリーニングの処理を実際に行うわけではないため，たとえば単純な表記のゆ

れなどのノイズなどにも適応的に対応することが難しい．また，抽出されたレコードに対するユーザの評価を導入する仕組みがないことから，抽出精度の向上の余地を生かしきれていないといえる．

2.2 データクリーニング

大量のデータに含まれるノイズを削減し，質の高いデータを得る処理は**データクリーニング**（data cleaning）[5] と呼ばれる．データクリーニングは，情報統合（レコード同定 [3]，名寄せなどの処理なども含む）やデータマイニングの要素技術であり，実世界の統合処理やマイニング処理では，大半の処理がデータクリーニングに費やされると言われる．

データクリーニングに関する一つの研究の方向としては，レコード同定処理を効果的に支援するための，あいまいさを考慮したテキストの結合処理技術の開発などがある．たとえば，[7] では，RDBMS 中のテキスト値を外部のテキストと結合（join）するための手法が提案されている．コサイン類似度を用いて，類似度が高いテキストを結合するアプローチがとられている．

他の方面の研究としては，大規模データを対話的にクリーニングするためのシステム技術の開発がある．たとえば，**Potter's Wheel**[9] では，大量のデータをクリーニングするために，クリーニング対象のレコードをユーザが対話的にブラウズし特定することを可能とするためのシステム技術が開発されている．また，あるレコードに対する修正処理を他のレコードの修正に波及させるための技術や，属性のドメイン情報の利用手法なども提案されている．

レコード抽出の精度を向上するためのアプローチとして，抽出されたレコード集合に対し，このようなデータクリーニングの手法を事後に適用することが考えられる．これにより，抽出の誤りの除去やノイズの削減などが行えると考えられるが，レコード抽出の処理プロセス自体が本質的に改善されるわけではない．既存の抽出レコードはクリーニングされても，抽出パターンの洗練は行われないため，新たなレコード抽出処理において，やはりノイズが多数生み出されることになる．データクリーニング処理を融合することによるレコード抽出処理の本質的な改善が重要な課題であるといえる．

3 本研究のアプローチ

3.1 研究の目的

本研究では，データクリーニング手法を統合したブートストラップ型のレコード抽出手法を開発する．研究の目的は以下のようになる．

- **レコード集約・選択手法の開発**：レコード抽出を行うユーザに対し，対話的なクリーニング処理機能を提供する際に重要となるのが，ユーザの負担の軽減である．抽出された大量のレコードをユーザがすべてブラウジングして判断を行うことは現実的に不可能であるため，抽出されたレコード集合を集約して，重複やノイズを削減する技術を開発する．また，真に人手による検証が求められるようなレコードを優先してユーザに提示する手法を開発する．

- **ユーザによるフィードバック機構の開発**：本手法では，対話的なクリーニング処理により，ユーザが抽出レコードの一部に対して抽出の妥当性に関するフィードバックを行うことを想定する．レコードに対するフィードバック結果を，上記のレコード集約・選択処理のみならず，レコード抽出処理自体に反映することにより，レコード抽出の精度の向上や抽出処理コストの削減を図る．ブートストラップ型のレコード抽出では繰り返しにより抽出レコード集合を拡大していくが，早い段階で適切なフィードバックを行うことにより，ノイズの削減を行うことが可能となると考える．

3.2 想定するシステム構成

図 1 に想定するシステムの構成を示す．

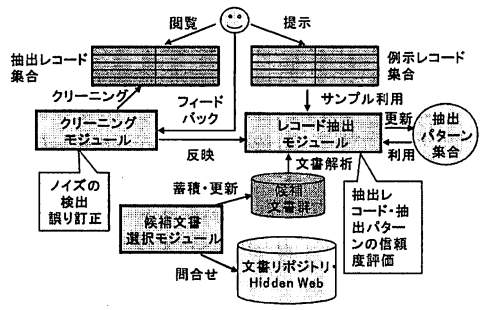

図 1: システム構成

文書リポジトリには大量のテキストデータが格納されている．本稿では特に議論しないが，Hidden Web のような外部の文書情報源も対象にすることを考えている．

候補文書抽出モジュールは，文書リポジトリから，抽出に役立つと考えられる一部の文書を取り出し，一時的に蓄積する役割を果たす．抽出処理の繰り返しの過程で，候補文書集合の更新なども行う．候補文書の取

得は，初期時点ではランダム，その後は過去の履歴を踏まえて行う（後述）．

ユーザは，抽出したいレコードの例としての**例示レコード集合**をシステムに与える．**レコード抽出モジュール**は，候補文書群の中の文書を解析し，例示レコード集合にマッチする文書中のレコードのオカレンスを特定する．オカレンスの出現コンテクストを分析することで，レコード抽出モジュールは，レコード抽出に役立つと考えられる**抽出パターン**を構築する．次いで抽出パターンを用いて，候補文書群から新たなレコードを抽出する．

ブートストラップ型のレコード抽出の繰り返しの過程で，ユーザには抽出されたレコードが提示される．その前に，抽出されたレコードは**クリーニングモジュール**によりクリーニングされる．ノイズ・誤り，データの重複などの検出や，類似レコードの発見などが行われ，ユーザに判断を仰ぐべきレコードを優先的に選択する．ユーザは，提示されたレコードを閲覧し，その内容の妥当性について判断を下す．

フィードバックの結果は，クリーニングモジュールを介してレコード抽出モジュールに渡される．レコード抽出モジュールは，フィードバック結果をもとに，抽出レコードおよび抽出パターンの信頼性について評価を行う．この信頼性の評価をもとに，以降のレコード抽出の処理における抽出精度の向上を図る．

以上の抽出処理の詳細を次節で述べる．

4 レコード抽出手法

この節では，レコード抽出処理にデータクリーニング処理を統合したアプローチについてその詳細を述べる．ここでは，DIPRE, Snowball などの特定のレコード抽出手法にできるだけ依存しない形で，汎化したブートストラップ型のレコード抽出法を示す．以下で提示する関数の具体的な処理は，対象等に応じてより詳細化される．たとえば，HTML 文書を対象とする場合には DIPRE のように URL の情報などを用いることが考えられ，抽出の対象に人名，社名などが含まれる場合には，固有表現の抽出を交えた Snowball のようなアプローチで具体化される．

4.1 初期フェーズ

まず，レコード抽出処理の初期フェーズについて述べる．ユーザから与えられた例示レコード集合をもとに，文書リポジトリを探索し，第 1 回目の抽出結果を提示するまでの処理を述べる．

処理の過程は次のようになる．

Algorithm 1 初期フェーズ

1: $Seed$：例示レコード集合
2: Doc：文書集合
3: $D = select\_documents(Doc)$ { 対象文書を選択 }
4: $Occ = find\_occurrences(D, Seed)$ { オカレンスを発見 }
5: $Pat = gen\_patterns(Occ)$ { パターンを生成 }
6: $D' = select\_documents(Doc)$
7: $Rec = find\_records(D', Pat)$

1. **例示レコード集合の提示**：ユーザから例示レコード集合 $Seed$ が与えられる．$Seed$ はテーブル構造をもち，そのエントリは

$$Seed(id, attrs) = Seed(id, a_1, a_2, \ldots, a_n)$$

というスキーマを持つとする*．id はシステムより順次割り当てられるレコードの識別子，$a_1, a_2, \ldots, a_n$ はレコードの属性名のリストであり，$attrs$ はその略記である．

2. **対象文書群の選択**：候補文書選択モジュールにより，文書集合 Doc からレコード抽出の初期的な対象とする文書群 D を選択する．特に他の情報がなければランダムに選択を行う．選択する文書数 $|D|$ の設定などは，実現環境などに応じて具体的に決定する．Hidden Web など，外部情報源からの文書抽出の場合はとくに，選択した文書をローカルに蓄積して，抽出処理を容易にする．

3. **オカレンスの発見**：$Seed$ に含まれる例示レコードが，D に含まれるかを $find\_occurrences$ 関数で探索する．Occ は

$$Occ(id, sid, doc\_info^*, context^*, pid)$$

というスキーマで表されるテーブルである．id はシステムから順次付与されるオカレンスの識別子，sid はマッチした例示レコードの識別子（$Seed$ テーブルの id 属性）である．doc_info は，オカレンスが含まれている文書の情報であり，文書名や URL などが該当する．また，$context$ は例示レコードが文書内でマッチした際の文脈（コンテクスト）情報である（例：前後に出現した単語リストなど）．$doc\_info, context$ に付与した "*" は，これらが下位のアルゴリズムで具体化されることを意味している．また，その値は必ずしも単純値であるとは限らず，集合値やリスト値をとることも許す．pid は後述するパターン識別子である．この時点ではその値は空値（NULL）とする．

*ここでは，説明を容易にするため，リレーショナルデータベース風の説明を行う．ただし，実際の実装では必ずしもリレーショナルデータベースを用いるわけではない．

4. **抽出パターンの生成**：次いで，*gen_patterns* 関数によりオカレンス集合 *Occ* よりパターン集合を生成する．生成したパターンは

$$Pat(id, doc\_pat^*, context\_pat^*)$$

というスキーマを有する．*id* は自動的に付与されるパターン識別子であり，*doc_pat* はそのパターンが有効な文書のパターンである．たとえば，DIPRE のオリジナル論文 [4] のように，パターンがマッチした文書群の URL を一般化して，抽出対象の文書群を選択するための URL のパターンを生成する場合などに活用される．一方，Snowball のオリジナルのアプローチ [1] では，HTML を対象とはしていないため，*doc_pat* の値は空値となる．*context_pat* は，コンテクスト情報を一般化したもので，実際に照合処理に用いられるパターンを表す．

なお，この処理の過程において，どのオカレンスからどのパターンが抽出されたかという情報を，*Occ* テーブルの *pid* 属性に記載する．各パターンについて一般には複数の *Occ* のエントリが対応するため，同じパターン識別子が複数の *pid* 属性に現れうることになる．

5. **レコードの抽出**：まず，レコード抽出候補の文書群 D' を *select_documents* 関数で求める．具体的な D' の選択方式は対象などに応じて詳細化する．その際には，オカレンスの発見処理の時に用いた文書集合 D 中の各文書の有用性の評価を反映することも可能であると考える．

次いで，抽出パターンの集合 *Pat* をもとに，D' からレコード抽出を試みる．ただし，*Pat* 中のパターンで有用性が低いと思われるものを事前に排除する．具体的には，テーブル *Occ* の pid 属性中に θ 回以上（典型的には $\theta = 2$）識別子が出現したパターンのみを使用する．抽出した結果は，

$$Rec(id, pid, a_1, a_2, \ldots, a_n, doc\_info^*, score^*, match, sid)$$

というスキーマ構造を持つ．*id* は順次割り当てられる識別子，*pid* は抽出に用いられたパターンの識別子，$a_1, a_2, \ldots, a_n$ は各属性の値，*doc_info* はレコードが抽出された文書の情報を表す．*score* ($0 \leq score \leq 1$) は，このレコードの出現コンテクストに対する抽出パターン *pid* の一致度である．*match*, *sid* は後の処理で利用する．この時点では空値とする．

4.2 フィードバックフェーズ

このフェーズでは，ユーザに修正候補のレコードを提示し，ユーザからのフィードバックを受ける．処理の概要をアルゴリズム 2 に示す．

Algorithm 2 フィードバックフェーズ

1: *Seed*：例示レコード集合
2: *Rec*：抽出されたレコードの集合
3: μ：「照合可」と判断する類似度の閾値
4: ν：「照合可能性あり」と判断する類似度の閾値 ($\nu < \mu$)
5: $(Match, PMatch) = find\_matches(Seed, Rec, \mu, \nu)$
6: ユーザへ情報提示し，フィードバックを受ける．
7: 各パターンの信頼度を評価する．
8: ユーザが判定を下していない各レコードについて，信頼度を評価する．

処理過程は以下のようになる．

1. **レコードのマッチング処理**：まず，与えられた例示レコード集合 *Seed* と抽出されたレコード集合 *Rec* のマッチングを図る．*find_matches* 関数では，*Rec* テーブルの各レコード *r.attrs* と *Seed* テーブルの各レコード *s.attrs* について，類似度関数 *sim* に基づく類似結合（similarity join）を行い，その結果を *Match*, *PMatch* テーブルに入れる．*Match* テーブルは，

$$Match(sid, rid, score)$$

というスキーマを有し，類似度が閾値 μ を超えているレコード対の情報を蓄積する．μ は「同一レコードとみなせるほど十分類似している」ことを保障する類似度の閾値である（ユーザにより指定）．*PMatch* テーブルは同様のスキーマを有し，「同一レコードであるとはいえないが，同一レコードである可能性がある」レコード対の情報（possible match）を保持する．閾値 ν ($\nu < \mu$) はこの構築に用いられる．具体的には，両テーブルは以下のように定義できる．

$$\begin{aligned} Match &= \{\langle s.id, r.id, score\rangle \mid s \in Seed, r \in Rec, \\ &\quad score = sim(s.attrs, r.attrs), \\ &\quad score \geq \mu\} \\ PMatch &= \{\langle s.id, r.id, score\rangle \mid s \in Seed, r \in Rec, \\ &\quad score = sim(s.attrs, r.attrs), \\ &\quad \nu \leq score < \mu\} \end{aligned}$$

Match テーブルの *rid* は，例示レコード *sid* と一致した（とみなしてよい）レコードの識別子であり，それらのレコードについてユーザによる検

証は不要である．一方，$PMatch$ は，例示レコードと類似しているが一致しているとは言い切れないレコードのペアである．

2. **ユーザによるフィードバック**：次いで，以下のようにユーザからのフィードバックを受ける．

   - $PMatch$ に含まれる抽出レコードへのフィードバック：$PMatch$ の情報を類似度値によりランク付けして提示する．ユーザからは，レコードについて

     (a) 例示レコードと同一視してよい

     (b) 例示レコードと異なるが正しい

     (c) 誤っている

     のいずれかの評価が与えられる．ここでは記述の簡単化のため，ユーザがすべてのペアに対して評価を下すことを想定する．ただし，一般的には負荷の軽減のため，一部のペアだけについて評価することも考えられる．

   - $Match, PMatch$ に含まれない抽出レコードについては，状況に応じてユーザが適宜ブラウジングを行う．すなわち，この処理はオプションである．ユーザから

     (a) 正しい

     (b) 誤っている

     のいずれかのフィードバックを受ける．

抽出レコードに対するフィードバックを受ける際，そのレコードを抽出するために用いられた抽出パターンも閲覧可能とし，ユーザからパターンの妥当性を判断してもらうことも考えられる．また，抽出レコードおよび抽出パターンについて，フィードバックの際にユーザが直接修正を加えることも考えられる．これらの機能についての議論はここでは省略するが，抽出結果の洗練や抽出精度の向上などについて役立つと考えられる．

なお，抽出レコードされたレコードにユーザが満足できない場合には，本研究が想定しているようなノイズや誤りだけではなく，トピックが合致しない場合も考えられる．たとえば，例示データとして，ユーザが「自動車メーカー名」と「その本社の所在地」という 2 属性からなる例示レコード集合を与えた場合，得られるレコードは必ずしも自動車メーカーのものとは限らず，他の業種のものも得られると考えられる．このような現象は，抽出パターンは誤っていないが，対象の文書の選択が適切でないことにあると考えられる．このような問題に対し，我々の研究グループでは，分類処理を統合したレコード抽出のアプローチを別個に検討し研究を進めている [11]．本手法と相補的に利用できるものと考える．

3. **フィードバックの反映**:ユーザからのフィードバックをの結果，抽出レコードは

   - **正しいレコード**：$Match$ に含まれているレコード，および，それ以外のレコードでユーザにより「正しい」とフィードバックを受けたレコード
   - **誤りレコード**：$Match$ に含まれないレコードのうち，ユーザから「誤り」とフィードバックを受けたレコード
   - **評価を受けていないレコード**:$Match, PMatch$ に含まれないレコードのうち，ユーザが評価しなかったもの

   に分類できる．

   これを受けて，各抽出パターン p の**確信度**（confidence）を求める．

$$conf(p) = \frac{p.pos}{p.pos + p.neg} \cdot \log_2(p.pos) \quad (1)$$

ここで，$p.pos$ はそのパターンで抽出されたレコードのうち「正しい」と評価されたレコードの数，$p.neg$ は「誤っている」と評価されたレコードの数である．この指標は，$\frac{p.pos}{p.pos+p.neg}$ の部分でパターンの精度を求め，$\log_2(p.pos)$ の部分でパターンの有用性（抽出可能なレコード数）を反映する．この評価尺度は [10] で提案され，Snowball [1] でも用いられたもので，**RlogF 確信度**と呼ばれる．

次いで，パターンの評価をもとに抽出されたレコードのうち，ユーザが評価を与えなかったものについて，確信度を求める．レコード r の**確信度**を以下のように定義する．この確信度の定義は Snowball [1] でも用いられている．

$$conf(r) = 1 - \prod_{i=1}^{|P|}(1 - conf(p_i) \cdot match(p_i, r)) \quad (2)$$

ここで $P = \{p_1, \ldots, p_n\}$ は抽出パターンの集合であり，$match(p_i, r)$ は Rec テーブルに含まれる $score$ 属性の値を用いる．これにより，r の確信度は $0 \leq r \leq 1$ の値をとり，大きいほどそのレコードの質が高いことになる．

4.3 繰り返し処理

これまで述べてきた処理は繰り返し処理に先立つ初期化処理に該当する．その後の繰り返し処理において

は，ユーザからのフィードバックを踏まえて以下のようにブートストラップ型のレコード抽出を進める．

1. **例示レコード集合 *Seed* の内容の更新**：本来の *Seed* の内容に加え，先のレコード抽出およびフィードバックの過程において「正しい」と判定されたレコードの集合をマージする．ただし，物理的にマージするのではなく，本来の *Seed* の情報はそのまま保存する．同様に，前回の抽出処理の過程で得られた情報についても，単に廃棄するのではなく，記録しておくことで重複したレコードやパターンを再び抽出することなどを省略することが考えられる．

   さらに，ユーザから判定が下されなかったレコードのうち，$conf(r) \geq \theta$ を満たすレコード r をすべて *Seed* にマージする．以上の処理により，*Seed* の内容は比較的質の高いレコード集合で拡張されることになる．

2. **対象文書群の選択**：再度，対象文書群の選択を行う．注意しなければならないのは，それまでの抽出処理においてすでに抽出を行った文書群であっても，新たな抽出パターンを用いれば，さらに新たなレコードが抽出される可能性があるということである．一方，いくつかの抽出パターンで抽出を行ってもまったくマッチしなかった文書は，再度抽出を試みる価値は高くないといえる．もちろん，多くの文書を対象文書としてすれば，取りこぼしの危険はないかわりに処理のオーバヘッドが大きくなる．このため，トレードオフをとることが重要となる．

   なお，DIPRE [4] のように，URL のパターンをも考慮する場合においては，質の高いレコードが得られた URL のパターン，および，ノイズが多く含まれたレコードが得られた URL のパターンなどを求めることで，候補文書の選択に役立てることが可能であると考えられる．この点については今後の課題とする．

3. **オカレンスの発見**：これは初期化の際の処理に準じる．過去の履歴情報を用いて，同じオカレンスを再度発見しないようにすることが重要なポイントである．

4. **抽出パターンの生成**：初期化の場合と異なり，前回までに得られた抽出パターンの集合が存在している．また，過去のフィードバックの過程において，ユーザから「妥当でない」と判定を受けている抽出パターンの集合も存在している可能性がある．そこで，抽出パターンを生成する際には，過去の「妥当な」抽出パターンと比較し，パターンがすでに得られたものであるかどうかの検証が必要である．また，「妥当でない」と判定済みのパターンと一致した場合には，即座にそのパターンを廃棄することになる．

   問題となるのは，「妥当な」抽出パターンと完全一致はしないが類似しているパターンが得られた場合である．この場合のパターンの一般化処理は，実装レベルのより詳細なレコード抽出法において具体化されることになる．

5. **レコードの抽出**：レコードの抽出処理も，基本的には初期化処理と同様である．ただし，抽出パターンに信頼度のスコアが付与されているため，多数の抽出パターンが得られている場合には，高いスコアのパターンを優先的に適用することが考えられる．

以上のような処理を繰り返し行い，抽出レコード集合の拡充と抽出パターンの洗練を進めていく．

5 まとめと今後の課題

本稿では，ユーザが与えた例示レコードをもとに大規模文書リポジトリから繰り返し処理でレコード抽出を行うブートストラップ型のレコード抽出手法に対し，データクリーニングのプロセスを統合することにより抽出精度を向上させるアプローチについて述べた．ユーザとの対話を抽出処理の過程に統合し，ユーザからのフィードバックを抽出されたレコードのクリーニングに用いるだけでなく，抽出パターン自体の評価に用いる点が特徴である．

今後の課題としては以下の点が挙げられる．

1. **実装レベルの抽出アルゴリズムの詳細化**：本稿で示したアルゴリズムは，DIPRE や Snowball などのブートストラップ型のレコード抽出手法を一般化した抽象レベルのアルゴリズムである．フィードバックなどの反映においては，実装レベルのアルゴリズムで対応する部分もあるため，それらの詳細化を勧める．

2. **クリーニング手法の詳細化**：Potter's Wheel [9] のような既存のクリーニングシステムのアプローチを参考にして，抽出レコードのクリーニング処理の具体化を進める．

3. **システムの実装と実験**：提案したシステム構成をもとに実装を行い実験を行う．精度の向上の度合いとユーザの負担のトレードオフをとることが問

題となるため，どの程度の労力をかければどの程度精度が向上するかなどの実験を行う必要がある．

また，このようなレコード抽出処理は，対象となる文書の種類と抽出したいレコードの種別に大いに依存するものと考えられる．そのため，いくつかのシナリオで異なる文書集合に対する実験も必要であると考える．

謝辞

本研究の一部は，日本学術振興会科学研究費補助金基盤研究 (C)(16500048)，文部科学省科学研究費補助金特定領域研究 (18049005) 及び科学技術振興機構 CREST「自律連合型基盤システムの構築」による．

参考文献

[1] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *Proc. ACM DL*, 2000.

[2] E. Agichtein and L. Gravano. Querying text databases for efficient information extraction. In *Proc. ICDE*, 2003.

[3] 相澤, 大山, 高須, 安達. レコード同定問題に関する研究の課題と現状. 電子情報通信学会論文誌, Vol. J88-DI(3):576–589, 2005.

[4] S. Brin. Extracting patterns and relations from the World Wide Web. In *Proc. WebDB*, 1998.

[5] T. Dasu and T. Johnson. *Exploratory Data Mining and Data Cleaning*. Wiley, 2003.

[6] O. Etzioni, M. Cafarella, and D. Downeym. Web-scale information extraction in KnowItAll. In *Proc. WWW*, 2004.

[7] L. Gravano, P. G. Ipeirotis, N. Koudas, and D. Srivastava. Text joins in an RDBMS for Web data integration. In *Proc. WWW*, 2003.

[8] E. Rahm and H. H. Do. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, 23, 2000.

[9] V. Raman and J. M. Hellerstein. Potter's Wheel: An interactive data cleaning system. In *Proc. VLDB*, 2001.

[10] E. Riloff and R. Jones. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proc. 16th National Conf. on Artificial Intelligence*, 1999.

[11] 張, 黒川, 石川, 北川. フィードバックを利用した文書の選択に基づくレコード抽出手法. 夏のデータベースワークショップ（DBWS 2006）, 2006.