PDF 文書からの手順記述抽出に関する一検討

谷本 真吾*　黄瀬 浩一
大阪府立大学大学院工学研究科

Investigation of a System for Extracting Procedures from PDF Documents

Shingo Tanimoto*　Koichi Kise
Graduate School of Engineering, Osaka Prefecture University

キーワード 手順記述 How 型 情報抽出 PDF 文書

1　はじめに

現在，Web 上の電子文書群から情報検索を行う方法としては，検索エンジンの利用が一般的である．検索エンジンの応答は，検索質問のキーワードに対する Web ページの適合度のランキングである．ここでランキングの上位に希望するページが含まれず，所望のページが得られないという状況がしばしば発生する．情報検索の技術として，ここには改良の余地が残されている．

このような改良の一例として，検索質問に対する答えを電子文書から抜き出して直接返すシステムがあれば，Web ページのランキングが応答されるよりも利便性が高いと考えられる．このようなシステムは質問応答（QA）と呼ばれており，研究が進められている．

QA システムの現状としては，答えが短い単語列で表現できる場合に一定の精度で抽出することに成功している [1]．しかし手続きや物事の理由といった，より複雑な知識の応答は，今後の発展が望まれている QA の研究テーマである．

武智らにより，手順記述は箇条書きで表されることが多いという前提の下，箇条書きで表されている手順記述を抽出対象とする手法 [2] が提案されている，この手法では，HTML ファイルの UL・OL タグを手がかりとして箇条書きを安定に切り出す．そして検索ドメインをコンピュータ分野の文書としたときに，抽出した箇条書きを高精度で手順か否か識別することに成功している．しかし検索ドメインを限定しないときの性能については十分には議論されていない．

そこで本研究では，Web における文書検索技術を前提とした，検索ドメインを限定しない Web からの手順記述の抽出を目指す．戦略としては，ドメイン依存する識別器を大量に統合することによってドメイン非依存な識別器に近付けるのではなく，手順記述一般から共通に得られる手順識別に有効な特徴量を用いる．このとき，検索ドメインを限定したときと比べての計算負荷の増大，および検索意図と出力が真に適合しているか判定するための枠組みの必要性という 2 点が問題になってくると思われる．これらの問題を考慮して，抽出精度を保った上での計算負荷の低減を狙った識別器の多段階化，および手順の属性というものに焦点を当てた手順記述の抽出システムについて提案する．そして提案手法の予備実験として，PDF 文書を抽出元としたときの箇条書きの文書からの切り出し，および多段階手順識別の初期で必要となる高再現率のフィルタ識別器の予備実験を行った結果の性能評価を行ったので報告する．

2 では本稿に関連する先行研究について紹介する．3 では提案手法の概要を説明する．4 では，箇条書き切り出しの実験結果および箇条書き内に含まれる言語特徴を用いた手順識別の結果と考察について述べる．5 は本稿のまとめである．

2　関連研究

手順記述の抽出を，手続き的知識を回答するソフトウェアエージェントと捉えると，動的ないし自動的に手続き的知識を決定する小坂らの研究 [3] や丹羽らの研究 [4] が関連研究として挙げられる．

小坂らの研究は，GUI を持つソフトウェアの開発におけるテスト自動化について扱っている．これは Windows 環境を想定し，テスト手順を予めマクロスクリプトとして用意するのではなく，GUI のウインドウの状態遷移データを与えることにより，テスト項目（最終的な操作）を指定するだけで中間操作を自動補完して全探索的にテストを行うシステムである．また，GUI の挙動に関する即応的なルール (例：モーダルウィンドウはとにかくクリックをして消す等) を与えることにより例外が発生した時も可能な限り停止せずテストを継続する工夫がなされている．

丹羽らは原子力発電プラントの制御システムについて，事故発生の際にプラントの状態を随時監視し，事態を収拾する

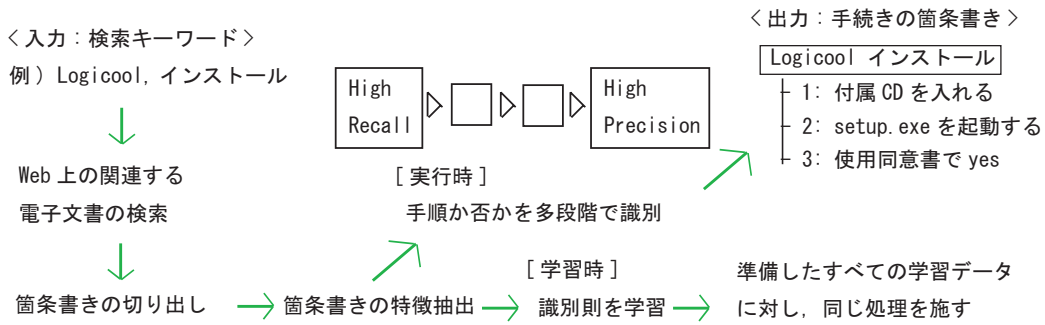

図 1　提案手法のシステム構成

手続きを自動更新して，必要なオペレーションを推奨するタスクについて述べている．プラントの状態は制御理論における状態方程式で管理されており，また各オペーレーションの与える外乱パラメータは事前に設計されている．

いずれのシステムにおいても，提示すべき知識は事前に設計されたシステム状態によって決定される．検索ドメインを問わない手順抽出というタスクを考えると，大規模な知識ベースを事前に人手によって設計することは難しく，大規模かつ常に変化する Web を知識ベースとして扱うアプローチがより適当であると考えられる．

これに相当する QA システムの先行研究としては，TREC[1] に参加している各システムが挙げられる．QA トラックに参加しているシステムは TREC の問題設定に照準を合わせて，問題解決のシステムを構成している．筆者の調査した範囲では，短い単語列による応答と単語列を複数列挙する応答と辞書の説明文のような語義定義文を応答する場合の 3 つに主眼が置かれており，手順記述を抽出するタスクは発展的課題とされている．

Web から手順を表す箇条書きを抽出するという，本研究に近いタスクを扱う先行研究として，武智らの手法 [2] が挙げられる．この研究では，知りたい手順の関連語を入力として，関連する HTML ファイルを収集し，その中から切り出した箇条書きを手順か否か識別する．HTML ファイル中の箇条書きとしては UL・OL タグで表された正規の箇条書きを抽出し，箇条書き中に含まれる言語特徴のみで手順か否かの 2 値識別を行う．この手法はコンピュータ分野の文書における手順抽出に特に有効であるという結果が示されているが，検索ドメインを限定しないときの性能については十分に議論がなされていない．

3　提案手法

3.1　システムの概要

本研究で構築を目指すシステムは，キーワードを入力すると関連する手順と手順の属性の組が出力される，というものである．ただし手順として一般のプレーンテキストレベルの記述を抽出するのではなく，手続きの各ステップが明確に分かるような箇条書き状の記述を抽出対象とする．

提案するシステムでは，識別器の多段階化，および手順の属性というものに着目する．識別器の多段階化では，複数種類の性能の識別器というものを考える．再現率重視の高速フィルタとなる識別器を初期段階として，徐々に適合率は高いが低速な識別器を適用することにより，抽出精度を保った上での計算負荷の低減を図る．手順の属性とは，その手順自身が何を表しているかという情報と定義する．より具体的には，手順を適用すべき状況や手順の目的といったものを表す記述が手順の属性を表す記述に相当する．

システムの構成を図 1 に示す．処理の流れは大きく学習と実行の 2 つに分けられる．

まず，システムの学習について述べる．学習時のシステムは，実行時に必要となる手順識別器の識別則の獲得のために動作する．始めに，設計者は学習サンプル収集用となる複数の検索質問を用意する．検索質問はドメイン不偏であり，かつ可能な限り多いほど望ましい．ドメインの偏りや学習サンプルの極端な不足は，大半の入力を負例と出力してしまう学習失敗につながり，本研究の趣旨と反するためである．次に各検索質問に対して，文書検索の技術を用いて学習サンプルとなる関連文書を収集する．本研究では GoogleAPI[*1] と wget[*2] を用いて，取得した URL から文書を得る．

文書検索で得られる各文書から，箇条書きを切り出す．切り出してきた箇条書き群を用いて，その箇条書きが手順を表しているか否かを識別するための識別則を学習する．

次にシステムの実行について述べる．実行時のシステムでは，ユーザが望む手順に関するキーワードを入力する．入力されたキーワードを用いて，関連する電子文書を収集する．文書検索で得られる各文書から箇条書きを切り出し，予め学習済みの識別器を用いて，手順か否かを識別する．次に手順であると識別された箇条書きに対して，その手順の適用対象を抽出する．実行時に適用する，文書検索，箇条書き切り出し，手順識別の特徴量計算，および手順の適用対象抽出の各手法は，学習時と同じものとする．最後に，手順と判断された箇条書きとその属性の組データを，何らかの方法でランキングした上でユーザに提示する．

[*1] http://www.google.com/apis/
[*2] ftp://prep.ai.mit.edu/pub/gnu/

表 1　ファイル拡張子から見た Web における各ファイル数の割合

拡張子	割合	拡張子	割合
html	36.12%	cgi	4.04%
gif	16.02%	php3	1.43%
jpg	11.02%	pdf	0.43%
php	10.10%	xml	0.08%
htm	8.11%	(others)	4.75%

表 2　HTML で記述されたファイル (HTML) と PDF 文書 (PDF) の各ファイルにおける，手順を表す箇条書きを含むファイルの割合と Web 中のファイル数の割合

ファイルの種類	手順記述の割合	Web 中の割合
HTML	1.4%	55.6%
PDF	88.9%	0.43%

3.2　処理の具体的実装

提案手法の中で実装が必要とされる事項は，以下の 5 つである．

- 情報源とする電子文書の選定
- 手順候補となる箇条書きの切り出し
- 切り出した箇条書きの手順識別
- 手順属性の抽出規則
- 手順のランキング方法

情報源の電子文書としては，PDF 文書を用いる．この狙いと妥当性を以下に示す．表 1 は手順記述をよく含むと期待される語を中心に生成した「料理 方法」,「HP 作り方」,「就職 仕方」,「車 選び方」,「自動車 乗り方」,「資格 取り方」,「パソコン 作り方」,「ディズニー 行き方」の 9 種類のクエリと GoogleAPI を用いて集めた 242,822 個のファイルの拡張子を調べた結果，1% 以上の割合となったファイルの拡張子ならびに PDF と XML のファイル数の割合を示したものである．表 1 は，Web 中の電子文書として最も一般的なフォーマットが HTML 形式であることを示しており，一見 PDF 文書は情報源として適でない様に見える．

しかし，手順記述の割合について調べたところ表 2 のようになった．表 2 の HTML の項目は，表 1 と同じデータ中の HTML で記述されたファイル群のデータについて示している．具体的には，Web 中のファイル数の割合と手順を表す箇条書きを少なくとも 1 つ含んでいるファイル数の割合を表す．PDF の項目は，後述の PDF データベースにおける手順を表す箇条書きを少なくとも 1 つ含む PDF 文書の比率と上述の Web 中の構成比率を並べたものである．表 2 から，PDF 文書は手順に関連の深いキーワードに対して，非常に高い比率で手順を含んでいるという結果を得た．Web 中のファイル数の割合と各ファイルにおける手順記述を含むファイルの割合から，PDF 文書と HTML ファイルの手順箇条書きを含むファイルは同数程度存在し，また HTML ファイルより PDF ファイルの方がノイズとなる手順箇条書きを含まないファイルの割合が少ないと期待される．よって，検索ドメインごとに文書を分類するといった処理を考えない本提案手法において，PDF 文書の方が情報源として適切であるといえる．

手順候補となる箇条書きの切り出しとしては，PDF 文書を平文化した上で，単語や文字表現の切り出しルールを用いる．PDF 文書の平文化には xpdf[*3] を用いる．この際，空隙や文の揃えといったレイアウトの情報の大半は失われる．次に平文からの箇条書き切り出しは文字表現のルールベースで行う．箇条書きの切り出しを文書の論理構造の推定の一処理と捉えると，主に単語や文字表現を用いる手法 [5] と主にレイアウトの位置情報を用いる手法 [6] がある．本研究の手法は前者に属するが，切り出しのルールは付録 A に示した独自のものを用いる．

切り出した箇条書きの手順識別としては，多段階の識別処理を考える．本稿で報告する実装としては，多段階識別の初期段階で必要となる高再現率の識別器について予備実験を行う．初期段階では計算コストを抑えつつ，手順記述をほとんど失うことなく，ノイズである手順でない記述を除外する識別器が求められる．具体的な実装としては，箇条書き内の表層特徴のみという，比較的簡易な特徴量を利用して学習させたサポートベクトルマシンを識別器とし，多段階識別の初期に適用する識別器としての性能を評価する．サポートベクトルマシンで用いる特徴ベクトルの各次元の特徴量としては，以下のものを採用する．

- 箇条書きの項目数
- 箇条書き全体の行数
- 各項目の単語数の差
- 学習サンプルにおける IDF と実行時の TF を用いた TFIDF 重み（学習時の TF は各学習サンプルに対する値）

日本語形態素解析には chasen[7]，サポートベクトルマシンの実装には svm-light[8] をそれぞれ用いる．

4　予備実験

提案手法の予備実験として，PDF 文書群からの箇条書き切り出しの実験と手順識別の多段階処理の初期段階で必要となる高再現率の識別器の性能評価実験を行った．

4.1　実験条件

実験用サンプルとなる PDF 文書を “setup”，“install” の 2 種類のクエリを用いて，GoogleAPI と wget によって収集した．オプションとして PDF 文書のみを検索し，得られた 1808 個の PDF 文書のうち，暗号化されていない 1538 個のファイルを PDF データベースとした．PDF データベース

[*3] http://www.foolabs.com/xpdf/

表 3　PDF データベースの詳細

検索質問	収集された全ファイル数	非暗号化ファイル数
“setup”	885	643 (72.7%)
“install”	923	895 (97.0%)
合計	1808	1538 (85.1%)

	個数	割合
何らかの手順を記述したファイル	1367	88.9%
それ以外の内容のファイル	171	9.2%
個数の合計	1538	-

表 4　箇条書き切り出しと手順識別の結果

実験の種類	適合率	再現率
箇条書き切り出し	1.4%	59.2%
手順識別	2.90%	96.7%

の内訳を表 3 に示す．ここで非暗号化ファイルとは，xpdf を用いてプレーンテキストを抽出できるファイルのことである．箇条書き切り出しの評価尺度には適合率と再現率を用いた．ここで適合率とは箇条書き切り出しの出力中の正しく切り出された箇条書きの割合であり，再現率とは全ての正しい箇条書き中の切り出しに成功したものの割合である．

手順識別では，箇条書き切り出しの結果を入力として，評価は箇条書き切り出しとは独立に行った．識別器として用いるサポートベクトルマシンのカーネルとしては，2 次の多項式カーネルを採用した．特徴ベクトルの単語特徴および箇条書きについては，10002 次元の特徴ベクトルを作成した．評価には 5 分割交差検定を用い，評価尺度としては適合率と再現率を求めた．ここで適合率とは手順識別の結果中の正しく識別できている手順記述の割合であり，再現率とは全ての正しい手順記述中の抽出に成功した手順記述の割合である．

4.2　結果・考察

実験結果を表 4 に示す．箇条書き切り出しにおいては，比較的簡易なルールを用いたにも関わらず再現率において一定の成果を得た．適合率が低めの値となった原因を分析した結果，xpdf が表や 2 段組といった複雑なレイアウトの平文化に失敗する例が比較的多いことが分かった．また箇条書きの項目の一部が画像データになっている箇条書きに対して，平文化した際に画像部分の表現を失うことにより，用意したルールでは抽出不能になるという例も少なくなかった．

また手順識別においては，箇条書き内の言語特徴を用いることにより，再現率重視の識別結果が得られた．この結果は，ほとんど正例を落とすことなく手順候補の箇条書きから全体の約半分に相当するノイズを除去できたことを示している．よって箇条書き内の言語表現のみを用いた識別器は，システムの最終形における手順識別の多段階処理の中で，初期のものとして適切であるという結果を得た．

5　まとめ

本稿では，PDF 文書からの手順を表す箇条書きの切り出しと，その手順識別について述べた．箇条書きの切り出しでは，xpdf の解析の影響を受けつつも，比較的簡易なルールで再現率において一定の成果が得られた．手順識別においては，箇条書き内に含まれる表層特徴のみを用いて，手順をほとんど失わずにノイズを除去できる，という知見が得られた．

今後の課題としては，多段階の手順識別の 2 段目以降で必要となる高適合率の識別器の実装および大規模データベース上での検証の必要性が考えられる．

参考文献

[1] J. H. Kil, L. Lloyd, S. Skina：“Question Answering with *Lydia* (TREC 2005 QA track)”, TREC-2005, Proceedings of the 2005 Edirion of the Text REtrieval Conference (2006).

[2] 武智峰樹, 徳永建伸, 松本裕治, 田中穂積：“WWW ページからの手順に関する箇条書きの抽出”, 情報処理学会論文誌 Vol.44, No.12, pp. 51–63 (2003).

[3] 小阪史, 山口聡之：“ダイナミックに操作を保管するテスト自動化について”, ソフトウェアテストシンポジウム 研究報告 (2003).

[4] 丹羽雄二, 寺邊正大, 鷲尾隆：“ソフトウェア・エージェントによる原子力発電プラントの事故時自動操作系の概念設計に関する研究”, INSS JOURNAL6 (1999).

[5] 土井美和子, 福井美佳, 山口浩司, 竹林洋一, 岩井勇：“文書構造抽出技法の開発”, 電子情報通信学会論文誌 Vol.J76-D- , No.9, pp. 2042–2052 (1993).

[6] 黄瀬浩一, 杉山淳一, 馬場口登, 手塚慶一：“レイアウトモデルに基づく文書構造解析”, 電子情報通信学会論文誌 Vol.J72-D- , No.7, pp. 1029–1039 (1989).

[7] 松本裕治, 北内啓, 山下達雄, 平野善隆, 松田寛, 高岡一馬, 浅原正幸：“日本語形態素解析システム『茶筌』version 2.2.9 使用説明書”, 奈良先端科学技術大学院大学 (2002).

[8] T. Joachims：“Making large-Scale SVM Learning Practical”, Advances in Kernel Methods - Support Vector Learning (MIT-Press) (1999).

付録 A 箇条書き切り出しのルール

このルールでは上段の文字列パタンにマッチしたものを箇条書きの項目として扱う．<>で囲んでいるパタンは，文字列パタンを表す．* は直前の文字パタンの 0 回以上の繰り返しを表す．::= は左辺から置換可能な文字パタンを右辺に表している．() 内は文字パタンそのものではなく，() 内の文字が表す文字パタンの条件を表す．適用方法は，1 から処

理を始めて，“何番へ”という指示がない場合には現在のステップ番号の次ステップに自動的に進む，という形式で表している．

ルール

1. (行の先頭)<空白>*<記号>*<空白>*<左括弧><空白>*<インデックス部><空白>*<右括弧>(<項目内容>に接続)
2. (行の先頭)<空白>*<記号>*<空白>*<インデックス部>(<項目内容>に接続)
3. (行の先頭)<空白>*<記号>(<項目内容>に接続)
4. (行の先頭)<空白>*<インデックス部><空白>(<項目内容>に接続)
5. <空白>*<記号>*<空白>*<左括弧><空白>*<インデックス部><空白>*<右括弧>(<項目内容>に接続)
6. <空白>*<記号>*<空白>*<インデックス部>(<項目内容>に接続)
7. <空白>*<記号>(<項目内容>に接続)
8. <空白>*<インデックス部><空白>(<項目内容>に接続)

ただし，
<インデックス部>::=<半角/全角アルファベット>｜<単一数字>｜<半角/全角数字>(<つなぎ記号><半角/全角数字>)*
<単一数字>::=<　-　>｜<全角/半角ローマ数字>
<項目内容>::=<(空文字以外かつインデックス部に用いる各種文字以外)>
<つなぎ記号>::=<全角/半角ハイフン>｜<全角/半角ピリオド>
<記号>::= ◆｜◇｜□｜■｜★｜☆｜◎｜○｜●｜・
<終端記号>::=<。>｜<全角/半角コロン>｜<全角/半角セミコロン>

適用方法

1. 文書から次の 1 行を読み込む．文書の終端ならば 10 へ．
2. ルール 1 から 2 を適用する．マッチした場合，7 へ．
3. ルール 3 から 4 を適用する．マッチした場合，7 へ．
4. ルール 5 から 6 を適用する．マッチした場合，7 へ．
5. ルール 7 から 8 を適用する．マッチした場合，7 へ．
6. 一つ前に登録されたハッシュキーで現在行を保存．終端記号がマッチする場合，ハッシュキーをクリアする．1 へ戻る．
7. インデックス部が数値の場合，最後にマッチした各種文字を先頭のパタンに置き換える．（例．　→　c → a 2.12 → 2.1）
8. 置き換え後のマッチング文字をキーとして，ハッシュを検索．置換した文字がある場合は置換した文字のインデックスが既に登録済みのパタンの次のインデックスになっているか調べる．キーが衝突した場合は追加で保存する．そうでない場合は新規保存.
9. 登録済みのインデックスを更新し，1 へ戻る．
10. 登録済みのパタンをチェックし，インデックス付きパタンの中で欠落があるものは除外する．
11. 次の文書へ．