Vol.2013-DD-90 No.2
2013/7/26

ブートストラップ型文書構造化のための Label 学習方式の開発

藤尾正和[†1]　　平山淳一[†1]

医療，金融，製造で扱われる業務情報は文書の形で記録，閲覧，活用される。業務情報の分析・整理に携わる知識労働者の生産性向上のためには，大量の非構造文書を利活用可能な形式へ構造化する技術が必須である。文書構造化においては，本文領域の他，表の構造化が有効だが，表構造化のための Label(項目名）辞書を業務に合わせて構築する必要があり，辞書構築のコストが導入ネックとなる。本研究では，Label 辞書構築のコスト削減のため，既知の Label 辞書を基に新規 Label 単語を獲得し，Label 辞書を随時更新していく，Label 学習型の表構造化技術を開発した。これにより，表構造の抽出率 80%を達成した。

Label Learning Technique for Bootstrapping Document Structure Analysis

MASAKAZU FUJIO[†1]　JUNICHI HIRAYAMA[†1]

1. はじめに

医療，金融，製造など，業務を進める上で必要な情報は，文書の形で記録・閲覧・活用される。例えば，医療の場合は電子カルテや薬品の安全性情報，金融の場合は税公金振込に用いられる帳票類，製造の場合は設計仕様書などが文書として数多くやりとりされる。社内で流通する文書数は増加の一途を辿っており，その勢いは今後も加速し続けることが予想される（IDC の調査では，企業内のデータ量は，2011～2020 年の 10 年で 50 倍に増加することが報告されている[1]）。このような文書数の増加に伴い，業務の分析・整理・報告に携わる知識労働者らは，必要な情報が記載された文書の収集・分析作業に多大な時間を費やしている。文書管理システムの導入により，文書の収集作業は効率よく行えるようになった。しかし，収集した文書から必要な情報を読み取り，分析する作業には，いまだ知識労働者の作業時間の多くが費やされている。この問題は，文書の大半が非構造文書であり，文書を閲覧するまでは文書内の情報を知ることができないことに起因している。知識労働者の業務効率向上のためには，大量の非構造文書を適切に構造化し，分析可能な形式で提供することが不可欠である。

我々は，文書内から業務に必要な情報を抽出する文書理解技術の開発をこれまでに行ってきた。一例として，金融機関で扱われる紙帳票の上に記載された文字列とその属性を認識し，コード化する帳票認識技術[2][3]や，ECM(Enterprise Contents Management)上で扱われる文書に，文書の構造を解析して新たに抽出したメタデータを付与する，メタ情報生成技術[4][5]がある。これらの技術は，抽出すべき属性とその手掛かりとなる特徴をあらかじめシステム側で定義することで，文書内から所望の属性を持つ文字列を抽出する。例えば帳票認識技術[2][3]の場合，金額や日付，口座番号といった属性を表す文字列である項目名（例えば，金額属性：“税額”，“納付額”など，日付属性：“納期限”，“支払期限”など，口座番号属性：“振替口座” “振込口座”など）をユーザがあらかじめ辞書として作成し，項目名を起点にその周辺から項目値を抽出する。また，メタ情報生成技術[4][5]の場合，上述のコンセプトに加えて，明確な項目名となる文字列が存在せずとも，項目値が属性ごとにある固有表現を持つ場合や，ある正規表現に一致するケースでは，それらの固有表現および正規表現を辞書として与えることで，項目名とその属性を抽出する。

上記の技術はいずれも，業務で扱われる文書の種類がある程度決まっており，文書理解に必要な辞書を事前に構築することが可能な場合に効果的な文書理解方式である。しかしながら，社内に存在する大量の非構造文書を構造化し，知識労働者が分析等に利活用できる形式で提供するといったサービス形態においては，文書の種類が未知かつ多岐に渡るため，事前の辞書構築が不可能となり，上記の技術の適用は困難となる。

2. 表構造解析の概要と課題

本章では，本報告で述べる Label 学習型表構造解析の要素技術となる表構造解析技術の概要と基本フロー，およびその課題について述べる。

(1) 表構造解析技術の概要

表構造解析とは，表内に記載された文字列間の関係を抽出し，構造化する技術である。構造化された表情報は，RDB（Relational DataBase）に格納されたり，RDF/XML（Resource Description Framework / eXtensive Markup Language）にて記述されたりすることで，利活用可能な形式で提供される。本報告で述べる表構造解析では，具体的には，表内から以下の情報を抽出することで表の構造化を行う。

†1 株式会社日立製作所中央研究所
Hitachi Ltd., Central Research Laboratory

図 2.1 に併せて表構造解析の結果例を示す（一部のみ）。

- 各文字列の種別（“Label” or “Value”）
- 文字列間の関係（“Lable-to-Label” or “Label-to-Value”）
- Value となる文字列が属するレコード

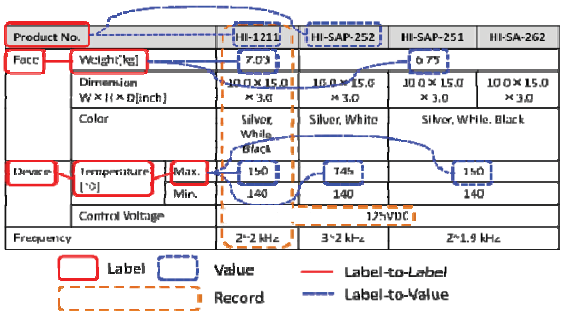


図 2.1 表構造解析結果例(※模擬サンプル)

Label とは，“Weight”や“Color”といった属性を示す文字列，Value とは “7.03”や “Silver”といった値そのものを表す文字列である。

図 2.1 の場合，“Face”の中の“Weight”が“7.03kg”であり，“Device” の “Temperature” の “Max” 値 が “150” で あ り ，“Product No.”が“HI-1211”であることが表現される。また，これらの Value が同一のレコードに属し，互いに関連づくことで，“Product No.”が“HI-1211”であるものの“Weight”が“7.03kg”であり，“Temperature”の“Max”が“150”であることが表現される。

(2) 表構造解析の基本フロー

表構造解析では，PDF や PowerPoint などの電子文書に埋め込まれている文字や罫線といった要素情報を抽出し，それらを元に表の物理構造解析，論理構造解析を行う。物理構造解析では，文字や罫線の要素情報から枠構造の抽出，文字列の抽出を行い，論理構造解析では，文字列種別の判定，文字列間関係の判定を行う。表構造解析の基本フローは以下のようになる。

①文書から罫線，文字情報などの要素情報を抽出する。
② 罫線情報から枠を抽出。また，枠同士の隣接関係および枠と文字の包含関係を解析する。
③ 文字列を抽出する。
④ 表内の文字列から Label となる文字列を抽出する。
⑤ 表内の文字列のうち，④で Label とならなかった文字列を Value 候補の文字列とする。
⑥ Label となった文字列同士の枠の隣接関係や，文字列の位置関係から Label-Label 関係を決定する。
⑦ Label となった文字列と Value 候補となった文字列の枠隣接関係，および文字列の位置関係から，Value となる文字列を抽出し，Label-Value 関係を決定する。

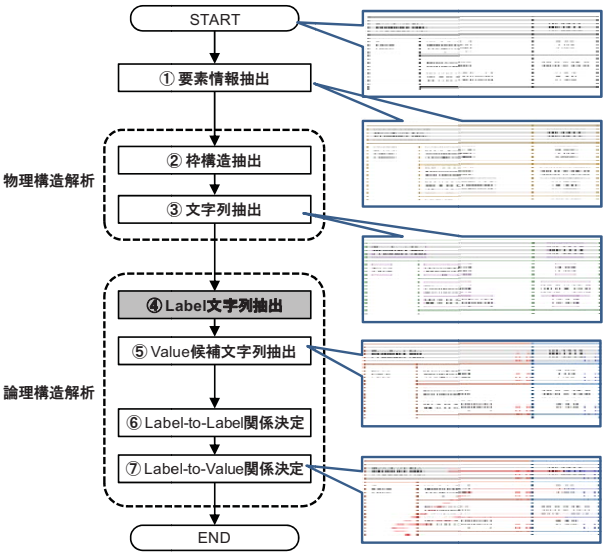


図 2.2 表構造解析の基本フロー

(3) Label 辞書構築の課題

図 2.2 の「④Label 文字列の抽出」において，従来は，Label となる文字列のリストを辞書としてあらかじめ与え，表内の文字列と辞書の文字列を照合し，一致した場合に表内の文字列を Label と判定していた。しかしながら 1 章で述べたように，社内の大量文書の構造化といった利用ケースにおいては，業務に特化した辞書の事前準備には大きなコストが生じる。そのため，Label 辞書に依存しない，辞書非依存型の表構造解析技術が必要となる。これに鑑み，報告者らは，Label 推定型の表構造解析技術を開発した。次章に概要を述べる。

3. Label 推定型表構造解析の概要

3.1 Label 推定方式の概要と処理フロー

Label 推定型表構造解析では，図 3.1 に示すように，ある罫線を境界として，表が Label 領域と Value 領域に分割されることを利用する。Label 領域と Value 領域の境界となる罫線を判定することで，Label となる文字列を判定する。Label 推定方式の処理フローを図 3.2 に示す。次節以降に各処理ブロックの詳細を示す。

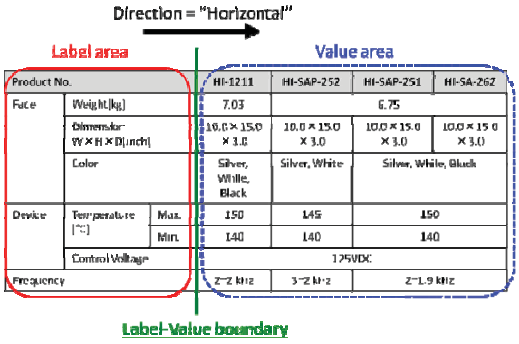


図 3.1 Label-Value 境界線の例

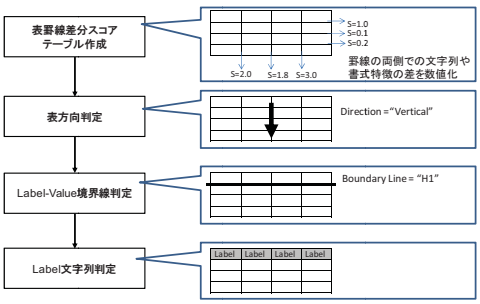


図 3.2 Label 推定方式の処理フロー

3.2 表罫線差分スコアテーブルの作成

表内の縦罫線V_iと横罫線H_jに対して，表罫線の差分スコア${D_V}^{Text}(V_i)$，${D_V}^{Layout}(V_i)$，${D_H}^{Text}(H_j)$，${D_H}^{Layout}(H_j)$を計算する。表罫線の差分スコアとは，その罫線の両側で互いに隣接する枠および枠内文字列の特徴にどの程度差があるかを表した数値であり，大きいほど差があることを示す。${D_V}^{Text}(V_i)$，${D_H}^{Text}(H_j)$は特徴をテキスト情報からのみ求めたスコア，${D_V}^{Layout}(V_i)$，${D_H}^{Layout}(H_j)$は特徴をテキスト情報＋書式情報（色やフォント，右/左寄せなど）から求めたスコアである。まず，隣接する２つの枠のペア$C_{n,}$，$C_{m,}$ごとに差分スコアを求め，２つの枠$C_{n,}$，$C_{m,}$の接線$L_{n,m}$にその値を付与する。その後，枠接線$L_{n,m}$が属する表罫線に対して，接線$L_{n,m}$の差分スコアを加算することで，表罫線の差分スコア

${D_V}^{Text}(V_i)$，${D_V}^{Layout}(V_i)$と${D_H}^{Text}(H_j)$，${D_H}^{Layout}(H_j)$

を次式(1)～(4)により計算する。

$$ {D_V}^{Text}(V_i) = \sum_{n,m} ED(S_{n,}S_m)\ \{\text{if } L_{n,m} \in V_i\} \quad \cdots \quad (1) $$

$$ {D_V}^{Layout}(V_i) = \sum_{n,m} [ED(S_{n,}S_m) + LD(C_{n,}C_m) + LD(S_{n,}S_m)]\ \{\text{if } L_{n,m} \in V_i\} \quad \cdots \quad (2) $$

$$ {D_H}^{Text}(H_j) = \sum_{n,m} ED(S_{n,}S_m)\ \{\text{if } L_{n,m} \in H_j\} \quad \cdots \quad (3) $$

$$ {D_H}^{Layout}(H_j) = \sum_{n,m} [ED(S_{n,}S_m) + LD(C_{n,}C_m) + LD(S_{n,}S_m)]\ \{\text{if } L_{n,m} \in H_j\} \quad \cdots \quad (4) $$

$ED(S_{n,}S_m)$は，文字列$S_{n,}$，S_mの編集距離であり，数字は"n"と置き換えて計算した。$LD(C_{n,}C_m)$は枠$C_{n,}$，C_mの書式特徴の差を表した距離，$LD(S_{n,}S_m)$は文字列$S_{n,}$，S_mの書式特徴の差を表した距離であり，下記に示す書式特徴の差分値の和と定義した。

- 文字列に関する特徴
  - フォントサイズ：差分値　$|Size_1 - Size_2|$
  - フォントタイプ：差分値　1.0（タイプが異なる場合）
  - Bold，Italic 設定：差分値　1.0（設定の有無が異なる場合）
  - 色：差分値
    $\sqrt{(R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2/(3\alpha)}$
    ※$\alpha = 128$とした
- 枠に関する特徴
  - 寄せ位置：差分値 1.0（右/左/中央 寄せのタイプが異なる場合）
  - 色：差分値
    $\sqrt{(R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2/(3\alpha)}$
    ※$\alpha = 128$とした

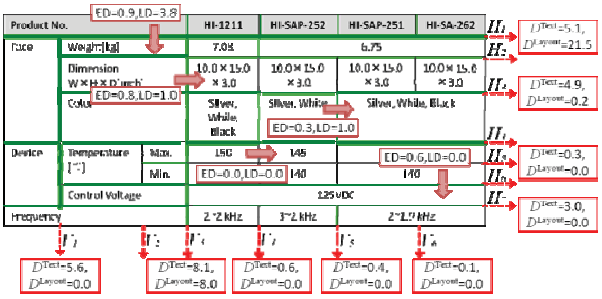


図 3.3 表罫線差分スコアテーブル

3.3 表方向判定

表の方向とは，Label となる文字列から見た，それに関連付く Value 文字列の方向である。Label 文字列から Value 文字列の方向は，複数の Value が構成するレコードが並ぶ方向と一致するため，類似した枠や文字列が並ぶ方向を，表の方向を判断することができる。これらの特徴は，隣接する枠内文字列のテキスト編集距離が小さいと言い換えることができる。よって，

$$ \mathrm{DirScore}_H = \frac{\sum_{j=P}^{N_H} {D_H}^{Text}(H_j)}{N_H - P} \quad \cdots \quad (5) $$

$$ \mathrm{DirScore}_V = \frac{\sum_{i=P}^{N_V} {D_V}^{Text}(V_i)}{N_V - P} \quad \cdots \quad (6) $$

とし，$\mathrm{DirScore}_V < \mathrm{DirScore}_H$ のときに表方向＝縦，$\mathrm{DirScore}_V > \mathrm{DirScore}_H$のときに表方向＝横，$\mathrm{DirScore}_V = 0$ もしくは$\mathrm{DirScore}_H = 0$ のとき，表方向＝不明とする。また，上式において，pは表方向判定に使う表罫線の開始番号（上，左ほど番号は小さい）である。これは表の上部や左部が Label 領域になり得るため，レコードの並び方向を判定するには適さないためである。本報告ではp=2 を用いた（Label の階層数の平均値 1.83 に最も近い整数値を用いた）。

3.4 Label-Value 境界線判定

Label 文字列の判定は，表罫線の中から Label-Value 境界線を判定することでできる。すなわち，

$D_V^{Layout}(V_i)$，$D_H^{Layout}(H_j)$ が最大となる罫線 V_i，H_j が Label-Value 境界線であると言える。よって，下記のように Label-Value 境界線を決定する。

- 表方向=縦のとき，

$$BoundLine = \arg_{H_j} \max\{D_H^{Layout}(H_j)\} \quad \cdots \quad (7)$$

- 表方向=横のとき，

$$BoundLine = \arg_{V_i} \max\{D_V^{Layout}(V_i)\} \quad \cdots \quad (8)$$

- 表方向=不明のとき，

$$BoundLine = \arg_{V_i H_j} \max\{D_V^{Layout}(V_i), D_H^{Layout}(H_j)\} \quad \cdots \quad (9)$$

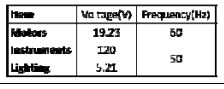
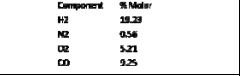
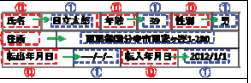
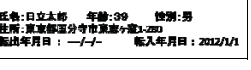
3.5 Label 文字列決定

見つかった Label-Value 境界線が縦罫線のときは，境界線より左に位置する枠内文字列を Label 文字列，横罫線のときは，境界線より上に位置する枠内文字列を Label 文字列とする。

3.6 Label 推定型表構造解析の課題

表 3.1 は，表レイアウトを[A].Label-Value 関係の整列性，[B].文字列を分割する罫線の有無によって分類したものである。前報告[1]でも述べているが，分類パターン[A1-B1]および[A1-B2]については，Label と Value の文字列領域が，表内のある罫線を境界として分割されるため，前節で述べた **Label 推定型表構造解析**が適用可能である。一方で，分類 [A1-B3]，[A2-B1]，[A2-B2]，[A2-B3] については，Label-Value の境界を示す罫線が存在しないことや，Label-Value 関係が整列しないため Label 領域を特定できないことにより，Label 推定型の表構造解析は適用できない。これらの表レイアウトに関しては，次章で述べる **Label 学習型表構造解析**により，構造化を図る。

表 3.1 表レイアウトパターンの分類

[分類A]Label-Value関係の整列性で分類	[分類B]論理文字列を分割する罫線の有無で分類	レイアウトの例
[A1]レコード表 Label-Value関係が互いに整列，グループ(=レコード)を構成している	[B1]枠あり型 文字列が罫線により全て分割されている	
	[B2]枠なし(外枠あり)型 文字列が罫線により分割されていない，表領域を特定する外枠はある	
	[B3]枠なし(外枠なし)型 文字列が罫線により分割されていない，表領域を特定する外枠もなし	
[A2]非レコード表 Label-Value関係が個々に並ぶ，レコードが整列していない形式	[B1]枠あり型 文字列が罫線により全て分割されている	
	[B2]枠なし(外枠あり)型 文字列が罫線により分割されていない，表領域を特定する外枠はある	
	[B3]枠なし(外枠なし)型 文字列が罫線により分割されていない，表領域を特定する外枠もなし	

4. Label 学習型表構造解析技術

4.1 Label 学習方式の処理フロー

Label 学習型表構造解析は，以下のフローから成る

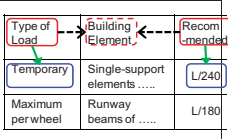
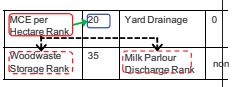
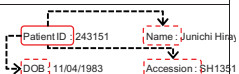
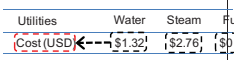
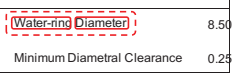
① Label 推定型表構造解析を行い，文書サンプル
② から Label 辞書を作成する
③ 現在の Label 辞書を元に表構造解析を行い，新規 Label 追加ルールに基づき Label を追加することで，Label 辞書を更新する（4.2 節で説明）
④ 現在の Label 辞書を元に Label 推定リトライ型表構造解析を行い，新たに Label を抽出することで Label 辞書を更新する（4.3 節で説明）
⑤ ②&③を繰り返し，随時 Label 辞書を更新

②の新規 Label 追加ルール，および③の Label 推定リトライ方式について，次節，次々節に述べる。

4.2 新規 Label 追加ルールによる Label 辞書の更新

表 4.1 新規 Label 追加ルールを示す。主として，現在わかっている Label 文字列配置関係を利用して，その周辺の文字列から条件を満たす文字列を新規 Label と判定する。このとき，既に Value として対応付いている文字列や，文字列長の短い数字列等は，Label らしさが低いとして，新規 Label 登録は行わない。

表 4.1 新規 Label 追加ルール

新規 Label 追加条件	追加例
両隣接する枠内の文字列が，同一方向に Value 対応を持つ Label である場合，当該枠内の文字列を新規 Label とする	
同表内の文字列において，同一の徹頭語もしくは徹尾語を持つ文字列群のうち，ある複数個が Label と判定されている場合，残りの文字列を新規 Label とする	
同一階層となる Description 型リストにおいて，いずれか 1 つのリストタイトルが Label と判定されている場合，残りの文字列を新規 Label とする	
同表内の文字列において，同一の数値正規表現を持ち，Label-Value 対応していない文字列群が整列する場合，整列方向の左 or 上に存在する文字列を新規 Label とする	
複数の Label 文字列からなる複合語が，表内に Label-Value 関係を持たずに存在する場合に，複合語を新規 Label とする	

4.3 Label 推定リトライ方式による Label 辞書の更新

3 章で述べた Label 推定型表構造解析は，Label 辞書が与えられていない状態にて，表内の Label-Value 境界線を判定し，表内の Label 領域を判断していた。一方で，Label 辞書が与えられている状態においては，既知の Label 辞書を用いることで，さらに高精度に Label 領域を判断することができる。具体的には，3.2 節の「表罫線差分スコアテ

ーブル」の計算式，および 3.3 節の「表方向判定」の判定式に，既存の Label 辞書を考慮した項を導入する。以下に詳細に述べる。

4.3.1 表罫線差分スコアテーブルの計算式

3.2 節の式(1)～(4)をそれぞれ下記の式(10)～(13)のように変更する。

$$D_V{}^{Text}(V_i) = \sum_{n,m}\left[ED(S_n,S_m) + VD(S_n,S_m)\right] \{if\ L_{n,m}^{n,m} \in V_i\} \quad \cdots \quad (10)$$

$$D_V{}^{Layout}(V_i) = \sum_{n,m}\left[ED(S_n,S_m) + LD(C_n,C_m) + LD(S_n,S_m) + VD(S_n,S_m)\right] \{if\ L_{n,m} \in V_i\} \quad \cdots \quad (11)$$

$$D_H{}^{Text}(H_j) = \sum_{n,m}[ED(S_n,S_m) + VD(S_n,S_m)] \{if\ L_{n,m} \in H_j\} \quad \cdots \quad (12)$$

$$D_H{}^{Layout}(H_j) = \sum_{n,m}\left[ED(S_n,S_m) + LD(C_n,C_m) + LD(S_n,S_m) + VD(S_n,S_m)\right] \{if\ L_{n,m} \in H_j\} \quad \cdots \quad (13)$$

$VD(S_n,S_m)$ は文字列S_n，S_mが Label-Value 関係に成り得る程度をスコアリングした項であり，文字列 S_nがマッチする Label 単語のスコアをL^nとしたとき，$VD(S_n,S_m) = L^n - L^m$で定義される。つまり，$VD(S_n,S_m)$は S_n が Label で S_m が Label でない場合に大きな値となり，逆の場合には負値となり，S_m，S_mともに Label でない場合には 0 となる項である。この項の導入により，表罫線差分スコアがより強く，Label-Value の境界を表す数値となる。

4.3.2 表方向判定の判定式

3.3 節の式(5)，(6)をそれぞれ下記の(14)，(15)のように変更する。

$$DirScore_H = \frac{\sum_{j=P}^{N_H} D_H{}^{Text}(H_j)}{N_H - P} + \alpha\frac{P}{\sum_{j=1}^{P} D_H{}^{Text}(H_j)} \quad \cdots \quad (14)$$

$$DirScore_V = \frac{\sum_{i=P}^{N_V} D_V{}^{Text}(V_i)}{N_V - P} + \alpha\frac{P}{\sum_{j=1}^{P} D_V{}^{Text}(V_i)} \quad \cdots \quad (15)$$

第 1 項は Value となりそうな領域（縦横 P 番目の罫線以後の領域）において，隣接する文字列の類似性を調べることで，同一表現を持つ文字列並びの方向を示す項である。一方，第 2 項は Label となりそうな領域（縦横 P 番目の罫線以前の領域）において，既存の Label 辞書に基づいて得られている Label-Value 関係の方向(の逆数)を示す項である。すなわち，式(14)，(15)は既存の Label 辞書を考慮して，表方向を判定する式になっている。αは係数であり，辞書更新が進むにつれて徐々に増加するように設定する（αが大きいほど，Label 辞書を信用して表方向の判定を行うことになる）。本研究では，（辞書更新進捗回数）／（目的の辞書更新回数）と設定した。

5. 評価実験

5.1 実験条件

Label 学習型表構造解析の Label-Value 関係文字列抽出精度評価のため，ICDAR2009 公開の PDF 文書サンプルである「PDF-Trex dataset」[7]を対象に行った。全 100 サンプルのうち，英語での調査レポート文書 26 サンプルを選んだ。調査レポートには，水質調査，電子製品市場動向，インフラ市場動向，主要経済指標調査などの文書カテゴリが存在する。抽出すべき Label-Value 数は 3,744 であった。実験は，CPU Intel Core-i7 3.40GHz，メモリ 16GB の PC 上で行った。

5.2 実験結果

抽出すべき Label-Value 関係 3,744 個に対する再現率(Recall)と適合率(Precision)は図 5.1 のようになった。横軸は，辞書更新回数であり，回数＝1 が Label 辞書なしで Label 推定型表構造解析を行った結果，回数＝2～4 が Label 辞書更新後の結果である。繰り返し回数 4 回における再現率は 79.68%となった。また，自動で抽出した Label の延べ数は 387 であった。これは手作業での定義コスト換算で約 0.6 [人日/文書]に相当する。

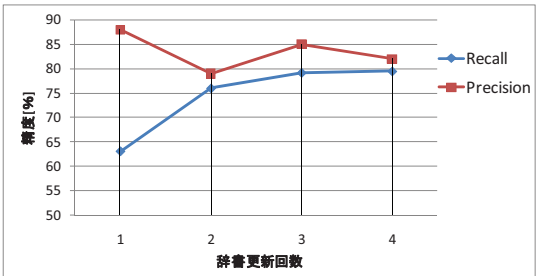


図 5.1 辞書更新回数ごとの Label-Value 関係抽出精度

5.3 解析結果考察

辞書更新が進むにつれて，正しく抽出できる Label-Value 関係が増加することが分かる。特に，第 1 ステップの Label 推定型解析では表方向判定誤りにより正しく解析できなかった表が，Label 辞書が更新されるにつれ，表方向が正しく判定できるようになるケースが目立った。また，Label 学習型の解析によって，Label 推定型解析では正しく解析できなかった，複合表や枠なし表の解析が可能になっていることが確認できた。

5.4 抽出失敗原因の分析

Label-Value 関係の抽出失敗の要因を下記にまとめる。これらは今後の課題であり，継続して精度向上に取り組む。

(1) Label 辞書更新の不十分性

Label 単語を十分に収集することができず，下図のように，本来 Label と判定すべき文字列が未判定となってしまったケースが多くみられた。図 5.2 は，Label-Value 境界線の誤判定により，“Total Aluminum”や“Total Mercury”といった文字列を Value と誤判定している例である。表解析結果

の確からしさを，表内の Label-Value 関係の自然さや，大局的な表の流れの自然さを元にスコア化し，最も確からしい表解析結果を最終結果とするような，解析結果の検定処理が必要である。

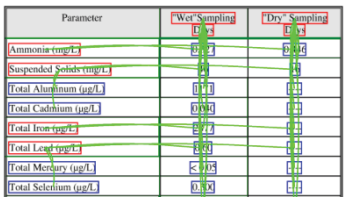


図 5.2 境界線判定失敗の例

(2) 文書構成要素の未抽出による物理構造解析失敗

PDF 文書では，内部の文字や罫線情報の表現形式が様々である。例えば下図の場合，文字座標の見た目と PDF 内の実際の座標（灰色矩形）にずれが生じている。また，中央の横罫線も PDF 文書内には罫線として存在していない。大量サンプルによる文書構成要素の表現形式分析など，地道なブラッシュアップを継続して行う必要がある。

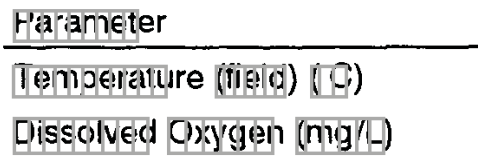


図 5.6 文書構成要素抽出失敗の例

6. まとめと課題

非構造文書内の情報分析の効率化には，文書構造化技術の開発が不可欠である。本研究では，文書内でも特に多くの情報が集中する表の構造化技術の開発に取り組んだ。従来の表構造解析では，Label 辞書を事前に構築し，表の構造化を行っていた。しかし，医療やインフラ製造分野など，大量かつ多様な文書の構造化が求められる利用形態においては，Label 辞書の事前構築のコストが導入ネックとなっていた。

本方式では，既存の Label 辞書を基に，表内から新規に Label と成り得る文字列を獲得し，随時 Label 辞書を更新していく Label 学習型の表構造解析技術を開発した。公開サンプル PDF-Trex dataset を対象に評価を行い，表構造の抽出率において 80%を得た。これにより，’13/3 の目標値を達成した。PDF-Trex dataset を用いた評価[7]では，表エリア抽出，セル抽出の評価のみで Label-Value 関係の評価は行っておらず，タスク設定が異なるため，単純な比較はできない。

今後の課題として，以下２点を挙げる。

(1) 階層辞書利用型表構造解析

本研究で述べた表構造解析技術は，文字列の隣接関係に Label-Label や Label-Value といった解釈を与えるものである。一方で図 6.1 のような表においては，「“2012”→“Sumas River”→“pH”→“7.5”」といった関係や，「“2013”→“Amiskwi River”→“Temperature (c)”→“8.0”」といった関係のように，必ずしも関係性を持つ文字列が隣接しないケースもある。このようなケースにおいては，Label 辞書にて Label 単語の階層性を記述し，それらを目的の表から探索するような階層辞書利用型の表構造解析技術が必要となる。

2012		2013	
Sumas River	Temperature (c) 3.9 pH 7.5 Dissolved Oxygen (mg/L) 12.0	**Sumas River**	Temperature (c) 4.0 pH 7.1 Dissolved Oxygen (mg/L) 11.2
Amiskwi River	Temperature (c) 9.8 pH 6.1 Dissolved Oxygen (mg/L) 7.8	**Amiskwi River**	Temperature (c) 8.0 pH 6.4 Dissolved Oxygen (mg/L) 9.1

図 6.1 階層辞書および意味構造解析な必要な表の例

(2) 意味構造解析

これまで開発してきた表構造解析技術は主に，枠-枠の隣接関係や，枠-文字列の包含関係，論理文字列の区切りを判定する物理構造解析と，物理構造解析結果を元に，隣接する文字列の論理関係を判定する論理構造解析に分けられる。一方で，図 6.1 の表から得られた論理構造解析結果「“2013”→ “Amiskwi River”→ “Temperature (c)”→“8.0”」は，Value となった”8.0”の意味を説明するにはまだ不十分である。例えば，「“8.0”は“Temperature”を表す数値であり，“2013”年にサンプリングされた“Amiskwi River”のものである。」と解釈できなければ，文書から抜き出したデータの利活用は困難である。そのためには，「“2013”は年度を表す数値，“Amiskwi River”は河川名，と理解した上で，“8.0”の属性を表す Label は“Temperature”であり，その他の 2 つの Label は属性ではなく条件を表す Label」と理解する必要がある。今後は，得られた論理構造解析結果のリンクにさらに意味を持たせる意味構造解析技術が必要となってくる。

謝辞 文書構造解析技術に関して議論頂いた多くの方々に，謹んで感謝の意を表します。

参考文献

[1] “Worldwide File-Based Storage 2010-2014 Forecast Update,” IDC,Dec. 2010.
[2] 新庄広，他，“DP マッチングを用いた帳票枠構造照合方式,” 電子情報通信学会，パターン認識・メディア理解（PRMU），2003.
[3] A. Minagawa, Y. Fujii, H. Takebe, and K. Fujimoto, “Logical Structure Analysis for Form Images with Arbitrary Layout by Belief Propagation,” Proc. ICDAR ‘07, pp. 714-718, 2007.
[4] M. Yoshida, K. Torikawa, and J. Tsuji, “Extracting Attributes and Their Values from Web Pages,” Machine Perception and Artificial Intelligence, World Scientific, pp. 179-200, 2004.
[5] H.H.Chen, S.C.Tsai and J.H.Tsai,“Mining Tables from Large Scale HTML Texts,” Proc. International Conference on Computational Linguistics, pp. 166-172, 2000.
[6] E.Bart, “Information extraction by finding repeated structure,” Proc. DAS 08, pp.175-182, 2010.
[7] Ermelinda Oro, and Massimo Ruffolo, “PDF-TREX: An Approach for Recognizing and Extracting Tables from PDF Documents,” Proc. ICDAR’09, pp. 906-910, 2009