自然言語処理 118－2
（1997. 3. 21）

構造化テンプレートを用いた新聞記事からの製品情報抽出

井出 裕二　藤吉 誠　永井 秀利　中村 貞吾　野村 浩郷

九州工業大学 情報工学部 知能情報工学科

E-mail: {ide,fujiyosi,nagai,teigo,nomura}
@dumbo.ai.kyutech.ac.jp

我々は，抽出すべき情報とその周辺の文字列との関係を記した“テンプレート”を用いて，字面処理による情報抽出処理の研究を行っており，新聞の製品紹介記事を題材とした実験で，テンプレートを用いた抽出処理の有効性を確認してきた．しかし，従来の抽出処理は，個々のテンプレートと入力文を決められた順にマッチさせていくだけの単純なものであったため，多数のテンプレートを用いて情報抽出処理を行うと処理時間が非常にかかるという問題があった．

本論文では，構造化したテンプレートを用いることよって，パターンマッチングの処理効率を向上させる方法を提案する．構造化テンプレートは，パターンマッチングの対象となる部分を共通のノードとして持つことでマッチングの回数を減らすことができる．実験により，構造化テンプレートを用いることにより，処理効率が向上することを示す．

Information Extraction with TRIE Structure Template from Newspaper Articles

Yuji Ide　Makoto Fujiyoshi　Hidetoshi Nagai
Teigo Nakamura　Hirosato Nomura

Department of Artificial Intelligence, Kyushu Institute of Technology

E-mail: {ide,fujiyosi,nagai,teigo,nomura}
@dumbo.ai.kyutech.ac.jp

We have researched a textual analysis method for extracting information from newspaper articles with templates which describe the relationship between information to be extracted and its surrounding strings and have confirmed the effectiveness of our method. However, as the set of templates learned from many examples articles becomes large, it takes a long time to match an input text with all templates. This paper describes an efficient method of pattern matching with TRIE structure templates. TRIE structure templates share the common parts of templates and can reduce the number of times of pattern matching. We show the efficiency of our method of pattern matching with TRIE structure templates by experiment.

1 はじめに

電子化された文書の増加により，必要とする情報をそれらの文書の中から見つけ出すことが困難になりつつある．人手のみによる情報の検索・抽出は，もはや不可能に近く，計算機を用いた情報の検索および抽出の支援技術は今後必要不可欠なものとなるであろう．このような大量の文書データの中から，あらかじめ目的とした情報のみを取り出してくる技術は，文書情報の整理やデータベースの自動的な構築，要約文の生成など応用範囲の広い技術であり，近年では盛んに研究されている [1][2].

自然言語処理システムでは，形態素解析，構文解析，意味解析などの解析をカスケードに接続したシステムが一般的である．多段の解析を行うほど意味や内容に踏み込んだ処理が可能となり，より高い精度の抽出結果を得ることができると期待できる．しかし，それぞれの解析の段階で多くの曖昧性が生じること，前段階の曖昧性のある解が次の段階へも引き継がれることによりさらに多くの曖昧性を生じることは処理量の増大につながるため大きな問題である．また，解析処理には一般に膨大な時間がかかることや，解析の失敗により次の段階の処理を続行できないこともある．さらに，それぞれの解析で用いる辞書や規則の整備に多くの労力を要することになる．

一方，分野が限定された文書からの情報抽出処理に対しては，パターンマッチングが有効であると言われている [1]．従来，文書から情報を抽出するには構文解析が必須と考えられていたが，処理文書の分野を限定した場合，全文の構文要素を解析せず，表層の単語列の並びに現る特定のパターンを認識するパターンマッチング手法が情報の抽出に適していることが確かめられている [1].

我々の研究室では，テンプレートを用いた情報抽出処理の研究を行っている [3][4]．この方式は，解析を行わない完全な字面処理による情報抽出処理である．

本研究は，新聞の製品紹介記事を題材として，表層処理による情報抽出処理システムを構築することを目的とする．また，本研究では，新聞の製品紹介記事に関するテンプレートの集合の特性を利用することによってテンプレートを構造化し，テンプレートを用いた情報抽出におけるパターンマッチングの効率を向上させることを目標とする．

2 テンプレートを用いた情報抽出

2.1 テンプレート

以下では，情報抽出処理のためのテンプレートを定義する．まず，テンプレートの定義のための用語を定める．

抽出項目: 抽出を試みる情報の内容を表すラベル (販売元，製品名など)

抽出情報: 抽出項目に対応する，情報を表す文字列

パターン: パターンマッチングの対象となる文字列長 1 文字以上の文字列

ワイルドカード: 文字列長 0 以上の任意の文字列

テンプレートは次のように定義する [4].

L を抽出項目，P をパターン，W をワイルドカードとしたとき，テンプレート T を

$$T = C_0L_1C_1L_2\cdots C_{n-1}L_nC_n$$
$$(C_i = P_0W_1P_1W_2\cdots P_{m-1}W_mP_m)$$

と定める．なお，文頭の C_0 および文末の C_n は空文字列であってもよい．テンプレートは 1 文単位で作成し，C と L は必ず交互に現るものとする．

ワイルドカードを導入することにより，テンプレートをより一般化することができる．テンプレートの一般化の長所は，テンプレート集合 (後述) の縮小につながり，抽出処理に要する時間を短縮することができる点である．

以後，テンプレートを表現する場合，抽出項目は {item} の形式で表す．item は抽出項目名である．ワイルドカードは * で表す．

2.2 従来の研究

我々の研究室では，抽出する項目とその周辺の文字列を記述した" テンプレート" を用いて情報を抽出する研究を行ってきた．方針は，解析処理を行わず表層的な情報 (字面情報) のみを用いて高速な抽出処理を行うというものである．新聞記事からの製品情報の抽出実験においては高い精度で正しい抽出結果を得ることができ，字面処理による手法の有効性を確認することができた．すなわち，定型性のある文章に対しては，テンプレートを用意することによって簡易な方法で情報の抽出処理を行うことができると言える．

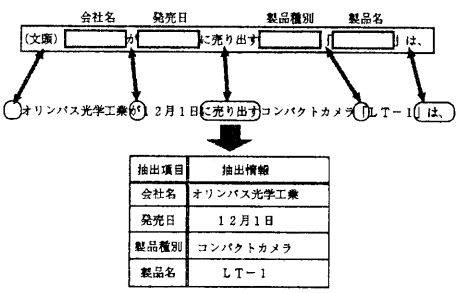

図 1: パターンマッチングによる情報の抽出

2.3 従来の研究の問題点

これまでに行ってきた方法では，テンプレートの数の増加に伴って抽出処理にかかる時間が増大するという問題がある．これは，テンプレートと入力文とのパターンマッチングを行う際に，無駄なマッチングを重複して行っていることが一つの要因であると考えられる．大量のテキストデータを対象とした情報抽出処理では，処理スピードの改善は重要な問題であり，改善されなければ字面処理による情報抽出処理のメリットは著しく低下する．

また，テンプレートと入力文とのマッチングを行うにあたっての，マッチングの制御法にも問題があった．従来はマッチングの成功により処理を中断する戦略 (stop-on-success 戦略) であったが，今後はすべての可能なマッチングを行った後に適当な解を見つける戦略 (try-all-possibility 戦略) へと移行すべきであろう．なぜなら，stop-on-success 戦略で正しい解を得るためには，複数テンプレートの配置を動的に変化させるなどの工夫が必要であるが，テンプレートの配置を決定することは容易ではない．その点，try-all-possibility 戦略では適切なテンプレートが整備されていれば，少なくとも1つは正しい解を抽出することができる．しかし，try-all-possibility 戦略を用いると解の候補が多数になることが問題である．そのため，正しい解を含む候補を除くことなく解候補を減らし，複数ある解候補の中から正しい解を選択する必要がある．

3 構造化テンプレートを用いた情報抽出

ここでは構造化テンプレートを用いることによって，前に述べた問題点を解決する方法を提案する．テンプレートを構造化することによって無駄なマッチングを減らすことができ，処理の高速化を図ることができる．本論文で提案する構造化テンプレートは，高速な語彙検索のための構造として自然言語処理システムで用いられてきたTRIE構造と類似した構造を持つので，TRIE構造化テンプレートと呼ぶものとする．

3.1 テンプレート集合の性質

{販売元}は、{製品種別}を発売した。
{販売元}は、{製品種別}を発売する。
{販売元}は、{製品種別}を販売する。
{販売元}は、{製品種別}を発売すると発表した。
{販売元}は、{発売日}から{製品種別}「{製品名}」を発売する。
* の{販売元}は、{製品種別}を販売する。

テンプレートファイルが上記のようになっているとすると，入力文「A は，B を販売する．」とのマッチングは，まずテンプレート「{販売元}は，{製品種別}を発売した．」とのマッチングを行うが，文末までのマッチングには失敗する．続いて「{販売元}は，{製品種別}を発売する．」にも，マッチング処理では失敗する．ここで，文頭と『は,』とのマッチングが重複して行われることに注目したい．さらに，「{販売元}は,{製品種別}を販売する．」とのマッチングで，文字列の抽出に成功し，処理は終了する．

ここで，図3.1の中で，上から5つ目までのテンプレートのパターンの列は，「(文頭)」→「は，」となっており，テンプレートとのマッチングが失敗する度に再度同じマッチングをすることは，効率が悪い．そこで，同様のマッチングを再び行わないようにすることで，TRIE を用いることにより処理効率が向上する．

3.2 TRIE 構造化テンプレート

TRIE は，キー集合 K の各キーの共通接頭辞を併合して作られる木構造であり，キーの文字 (あるいは値) の桁単位を比較の対象とすることを特徴とする．この特徴により，TRIE はプログラミング言語処理系や自然言語処理システムにおける文字列を対象とした語彙の検索に向いている [5].

テンプレート集合の共通する部分をまとめることで，TRIE と同じ利点が得られる．そこで，テンプレート集合中の要素を結合することにより，TRIE 構造化テンプレートを作成する．テンプレート集合 T に対する TRIE 構

造化テンプレートを図2に示す．なお，このときTRIEの葉とキーを1対1に対応させるために'#'をキーの最後に付ける．

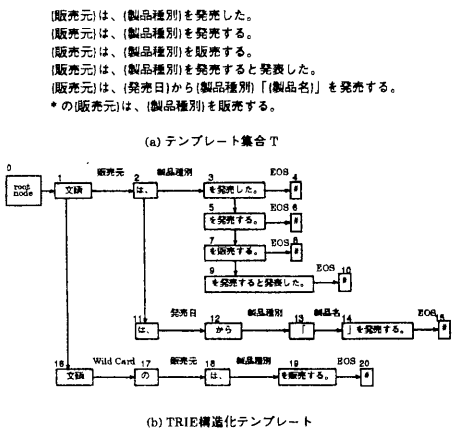

図 2: TRIE構造化テンプレート

TRIE構造化テンプレートは，テンプレート集合の最左部分列を共通のノードとして括ったものである．共通部分のパターンマッチングを重複して行わないことによって，パターンマッチング処理の高速化を実現できる．

TRIE構造化テンプレートのノードは，入力文とのパターンマッチングで用いられるパターン，次のパターンマッチの対象となるパターンを持つノードへ向かう右リンク，次のノードがパターンマッチに成功した場合に確定する文字列に対する抽出項目，右リンクで辿ったノードのパターンがマッチングに失敗した場合に辿る下リンクから構成される (図3)．

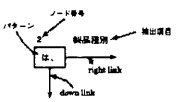

図 3: TRIE構造化テンプレートのノード

3.3 情報抽出アルゴリズム

TRIE構造化テンプレートを用いれば，入力文とのパターンマッチング処理を高速に行うことができる．入力文にマッチするテンプレートが存在するかどうかを調べるだけならば，ノードに記されたパターンとのマッチングが成功する度に右リンクを辿って，最終的にパターン'#'とマッチすれば，入力文にマッチするテンプレートが存在したことになる．パターンマッチングに失敗すると，現在のノードに下リンクがあれば，下リンクの指すノードへと移る (図4)．

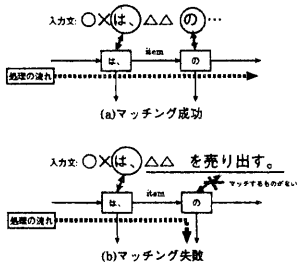

図 4: TRIE構造化テンプレートとのパターンマッチ

1文からの情報抽出アルゴリズムは，再帰的に定義できる．以下に，TRIE構造化テンプレートと入力文とのマッチングによる情報抽出アルゴリズムを示す．以下のアルゴリズムは，すべてのテンプレートとのパターンマッチングを行って，1文からの抽出処理に成功すると抽出項目と抽出項目に対応する文字列を出力するものである．パターンマッチの戦略はすべての可能なマッチングの結果を得る戦略を取る．図2(b)のTRIE構造化テンプレートを用いてすべての可能なマッチングを行うと，入力文「AのBのCのDは，Eを販売する．」に対する抽出結果は，以下のようになる．

1. 販売元: AのBのCのD，製品種別：E
2. ワイルドカード：AのBのC，販売元：D，製品種別：E
3. ワイルドカード：AのB，販売元：CのD，製品種別：E
4. ワイルドカード：A，販売元：BのCのD，製品種別：E

[情報抽出アルゴリズム]

```
procedure match (current,line,stack);
begin
  空のスタック stack2 を用意する;
  if pattern[current] が '#' であるならば
    stack の内容を出力する; (一文からの抽出成功)
  elseif right[cur] が存在するならば,
```

```
if line 中に pattern[current] が含まれるならば,
次のことを行う :
begin
   Y := '';
   while line 中に pattern[current] が含まれる do
   begin
      if Y = '' ではない do
      begin
         X := pattern[current];
         push(stack2,YX);
         Y := YX;
      end
      else
      begin
         X = pattern[current] であるとき,
         line = α X β となっている.
         push(stack2, α);
         Y := α
      end
      push(stack2, β);
      line := β;
   end
   while stack2 が空でない do
   begin
      post := pop(stack2);
      pre := pop(stack2);
      push(stack,pre);
      push(stack,item[cur]);
      match(right[cur],post,stack);
      pop(stack);
      pop(stack);
   end
end
elsif pattern[current] が" 文頭" である do
begin
   push(stack,item[current]);
   match(right[current],line,stack);
   pop(stack);
end
if down[cur] が定義されている do
   match (down[cur],line,stack);
end.
```

ここで, current は現在注目しているノードを指すものとし, pattern[current] とは, 現在注目しているノードのパターンを指すものとする. 同様に, item[current] は現在注目している抽出項目であり, right[current] は現在注目しているノードから右リンクで辿れるノード, down[current] は現在注目しているノードから下リンクで辿れるノードを指す. stack は, 抽出処理の途中経過を保持するためのスタックである. また, α, β は文字列を表す (空文字列も含む).

4 抽出項目に対する制約

パターンマッチングの結果の曖昧性の爆発を防ぐために, ノードからノードへの遷移の段階で制約を導入する. 制約を満たしたものだけが, パターンマッチングの段階をひとつ進めて次のパターンとのマッチングを行うことができるものとする (図 5).

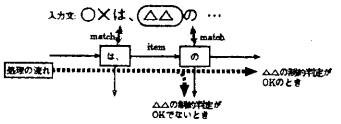

図 5: 制約

制約の役割は 2 つある. 一つは, パターンマッチによって取り出された文字列が, 情報として明らかにおかしいものを排除することであり, 抽出した解の候補数の爆発を抑制することができる. もうひとつの役割はパターンマッチングの段階で制約をかけることにより無駄なパターンマッチングの処理を行わないようにすることである.

解として正しいものを排除するような制約は作成しないように注意すべきである. テンプレート作成用データに関しては 100 % の再現率を保ったまま解の候補数を減らすことが望ましい. 制約は, 抽出項目に共通の制約と抽出項目に依存しない一般的な制約がある.

1. **抽出項目に共通の制約** 抽出項目に共通の制約では, 抽出項目の性質と関係なく, 明らかに意味のある句として成り立たないものを排除する. 例えば, ある抽出項目の情報として取り出された文字列が, その文字列中で括弧の対応がついてないものや, 読点から始まるような文字列を抽出候補から外す.

2. **抽出項目別の制約** 抽出項目に依存した制約では, 抽出項目の表層上の特徴を利用して明らかにおかしい文字列を抽出情報の候補から排除する. 抽出項目別の制約では, 価格と発売日に関しては表記がほぼ決まっており, 正規表現を用いて文字列としての妥当性を判定することができる. また, 抽出項目別の制約を相互に利用することによって, チェックを厳しくすることができる. 例えば, 販売元として抽出された文字列が価格のパターンにマッチすることはほとんどないと考えてよい. そこで, 販売元として得られた文字列が発売日や価格の表記パターンとマッチする場合は, 販売元の候補としては妥当ではないとして抽出情報の候補から外す.

5 実験と考察

本章では, 実験とその結果, および結果に関する考察について述べる.

5.1 実験対象

実験では，日本経済新聞1994年版の中の製品紹介記事2000記事を用いた．1記事中に文は平均6.8文(13668文／2000記事)あり，最大24文，最小2文の記事があった．また，1記事中の文字数は平均303.2文字(606051文字／2000記事)，最大1131文字，最小69文字であった．1文当たりの文字数は平均44.3文字(606051文字／13668文)最大212文字，最小3文字であった(文字数のカウントは，読点を含む)．全文13668文のうち，抽出を行う情報を含む文の数は，4990文であった(全文の36.51%)．

製品紹介記事2000記事のうち1900記事をテンプレート作成用データとして使用し，残り100記事を評価用として利用した．評価用記事中には605文あり，実際に情報が含まれる文は264文であった．

テンプレート作成用データは，テンプレート集合，製品種別と販売元のデータベース，および制約の作成に用いた．テンプレート作成用記事の例を以下に示す．図中のタグの役割を表1に示す．タグは，⟨*item*⟩と⟨/*item*⟩の間に囲まれた文字列が*item*という情報であるということを示す．この例では，「高砂酒造」が販売元であることをタグにより表している．

```
<number>29</number>
<date>94.12.21</date>
<headline>高砂酒造、「吟嶺大雪」道内で発売。</headline>
<body>　<C>高砂酒造</C>（旭川市、小檜山亨社長）は、<S>吟醸酒</S>「<P>吟嶺大雪</P>」を道内で発売した。酒造好適米の美山錦を原料米とし、五五％まで精白、大雪山系の伏流水で造った。芳醇（ほうじゅん）な香りに、まろやかで切れのよい味が特徴。容量は一・八リットルで、価格が<K>二千八百円</K>。吟醸酒としては多少安くしたという。八千本の限定出荷。</body>
```

抽出実験は評価用データについて行い，入力は記事単位のプレインテキストとする．評価用データについてもタグ付きテキストを作成し，システムの出力とタグ付き評価用テキストの照合により評価を行う．

抽出する項目は，「販売元」「製品種別」「製品名」「価格」「発売日」の5つとする．

5.2 情報抽出実験

実験は，テンプレート作成実験，処理速度に関する実験，抽出精度に関する実験に分けられる．

タグ	役割
number	記事番号
date	記事掲載日
headline	記事見出し
body	記事本文
C	販売元
S	製品種別
P	製品名
K	価格
D	発売日

表1: テンプレート作成用記事中のタグ

5.2.1 実験方法

テンプレート作成

以下のテンプレートの集合T1，T2，T3，T4を作成し，実験1を行う．

T1: 抽出項目の隣接1文字を残し，その他はワイルドカード化したテンプレートの集合．

T2: 助詞，活用語尾，2文字以上の文字列をパターンとして残し，これに該当しないものは1文字だけパターンとして残して，その他はワイルドカード化したテンプレートの集合．助詞，用言の活用語尾，記号列は字面で判定した．

T3: T2の方法で抽出項目に隣接する文字列を残し，さらに人手で収集した固定パターンをパターンとして残すテンプレートの集合．

T4: ワイルドカードを使用しないテンプレートの集合．

実験1: テンプレート作成用記事1900記事からテンプレート集合の作成およびTRIE構造化テンプレートを作成し，テンプレート集合中の，テンプレート数，パターン数，およびTRIE構造化テンプレートのノード数を比較する．

処理速度

処理速度に関する実験は，SparcStation 20(メモリ96MB)のもとで行った．

実験2: 従来のテンプレートファイルを用いる方式と，TRIE構造化テンプレートを用いた方式のパターンマッチングの処理速度を比較する．実験データ100記事に対してテンプレート集合T1を用い，1文にすべてのテンプレートをマッチさせるのに要する処理時間

の比較を行う．計測時間はパターンマッチング処理の開始から終了までの時間であり，テンプレート数に対する処理時間を比較する．

抽出精度

実験 3-1: 制約処理を行ったときと行わなかったときの１文に対する解候補の個数を比較する

実験 3-2: それぞれのテンプレート集合を用いて抽出処理を行った結果を求める．抽出処理では制約の処理を行い，テンプレートがマッチした個数によって解候補の優先順位付けを行って，最も優先順位の高いものを解とする．評価は，以下に示す再現率と適合率を用いて行う．

$$\text{再現率} = \frac{\text{抽出した正しい情報の個数}}{\text{正解データ中の情報の個数}}$$

$$\text{適合率} = \frac{\text{抽出した正しい情報の個数}}{\text{抽出したすべての文字列の個数}}$$

5.2.2 実験結果

テンプレート作成

実験１の結果を表2に示す．表には，テンプレート作成用記事1900記事から作成されたテンプレートの数と，TRIE構造化する前後のパターン数を示す．

	テンプレート数 (パターン数)	TRIE
T1	1688(10986)	5967
T2	1804(11470)	6321
T3	2477(16269)	9289
T4	3605(12338)	9885

表 2: 実験１の結果 (個)

処理速度

実験２の結果を図6に示す． 図6は，従来のテンプレートのマッチングと，TRIEを用いたマッチングの処理時間の平均を比較したものである．

抽出精度

実験3-1の結果を表3に，実験3-2の結果を表4，refseido1-tekigouに示す．表4，5は優先順位１位の解の再現率と適合率を表す．

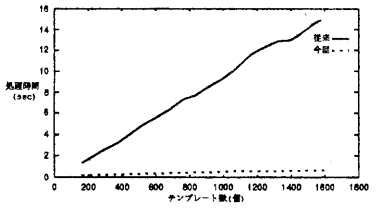

図 6: 実験２の結果

	制約なし		制約あり	
テンプレート	平均	最大	平均	最大
T1	76.59	1163	7.02	70
T2	63.62	749	6.23	58
T3	32.03	375	4.71	49
T4	4.49	55	1.20	4

表 3: 抽出された解候補の数 (個)

	販売元	製品種別	製品名	価格	発売日
T1	77.14	50.91	68.60	79.52	82.76
T2	76.19	51.82	66.94	78.30	85.06
T3	79.05	55.46	74.38	78.30	82.76
T4	0	0	9.92	52.83	56.32

表 4: 優先順位１位の解候補に対する再現率 (%)

	販売元	製品種別	製品名	価格	発売日
T1	77.89	36.13	69.75	84.85	76.60
T2	73.39	33.53	71.68	84.69	78.72
T3	76.15	31.44	76.27	88.30	82.76
T4	-	-	52.17	90.32	98.00

表 5: 優先順位１位の解候補に対する適合率 (%)

5.2.3 実験結果の考察

テンプレート作成

抽出項目の隣接1文字だけを残したテンプレート(T1)は，テンプレートとしては最も一般化が進んだものである．よって，T1のテンプレート集合よりも小さいテンプレート集合は，テンプレートの定義により作成されない．また，T4は逆に最も一般化されていないテンプレートの集合である．

1900記事から作成されたテンプレート集合についてみると，T4はT1の2倍以上の数のテンプレートが生成された．テンプレートの数は，従来の方法であればパターンマッチングの処理時間と大きく関係するため，処理時間の観点からはT1のテンプレート集合の方が望ましい．

処理速度

処理時間に関しては，TRIE構造化テンプレートを用いたパターンマッチングの方が従来の方法に比べて速い．従来の方法では，テンプレートの数の増加に比例して処理時間が増大するが，TRIE構造化テンプレートではテンプレート数の増加はそれほど処理時間に影響していない．

抽出精度

一般化されたテンプレートほど，1文にマッチングするテンプレートの数は多いが，制約を用いることにより，最も一般化したテンプレートを用いても，1文にマッチングするテンプレートの数は7個となり，解の候補を減らすことができた．

抽出精度は，テンプレート(T1)，(T2)，(T3)の優先順位1位の解候補に対する再現率，適合率をそれぞれ見ると，「製品種別」以外の抽出項目の結果は80％弱であり，今回提案した情報抽出方式は定型性を有する文章に対しては有効であると言える．また，「製品種別」を抽出する方法としては，今回の処理方式では不十分であり，別途に解決方法を検討する必要がある．

6 結論

6.1 本論文のまとめ

本研究では，テンプレートを用いた情報抽出処理におけるパターンマッチング処理の効率の向上を目指し，その結果，テンプレート集合中には類似したテンプレートが多いことから，重複するマッチングを何度も行わないことで処理効率を高めることに成功した，

また，従来の処理方式は，テンプレートと入力文のマッチングが成功した時点で終了するものであったが，今回はすべての可能なマッチングの解を求めた後に，適切な解を選択する方式について検討した，すべての可能なパターンマッチングを行うと，パターンマッチングの曖昧性が爆発するため，抽出する情報の特性を利用した字面処理による「制約」をかけることによって，曖昧性の爆発を抑えた，また，複数の解候補の中から適切な解を選択する方法についても検討した．

6.2 今後の課題

今回は，1記事中に複数製品を紹介している記事については考慮していない．この様な記事の場合，抽出項目同士の対応が難しく，今回用いた方法で情報抽出を行っても，きちんと抽出できていない．これらの記事を扱えるように，処理方式を改良できれば，特に「製品名」，「価格」，及び「発売日」の項目の抽出精度の向上が期待できる．

さらに，複数の解の中から，適切なものを選択する方法については，さらなる検討が必要である．今回は，多くのテンプレートが抽出した文字列ほど優先順位が高いものとして処理を行った．これはすべてのテンプレートは同じくらいの割合で正しく情報抽出を行うものとして処理を行ったが，実際にはテンプレートにより抽出精度は異なるはずであるので，さらなる検討が必要である．

謝辞

本論文で使用したテキストデータは，「日本経済新聞記事データ CD-ROM(1994版)」を使用した．使用を許可して下さった日本経済新聞社，および日経総合販売(株)に深く感謝致します．

参考文献

[1] 江里口 善生, 木谷 強：冨田一般化LRパーザを用いた情報抽出, 情報処理学会論文誌 Vol.38 No.1, pp.44 – 54, 1997

[2] 松尾比呂志, 木本晴夫：抽出パターンの階層的照合に基づく日本語テキストからの内容抽出法, 情報処理学会論文誌, Vol.36, No.8, pp.1838–1844, 1995

[3] 藤吉 誠：構造化テンプレートを用いた新聞記事からの製品情報抽出に関する研究, 平成8年度九州工業大学修士論文 (1997)

[4] 井出 裕二, 藤吉 誠, 永井 秀利, 中村貞吾, 野村浩郷：テンプレートを用いた新聞記事からの製品情報抽出システム, 情報処理学会研究報告 96-NL-115, pp. 83 – 90, 1996

[5] 青江　順一：トライとその応用, 情報処理 Vol.34,No.2,pp.244-251,1993