チャンクの分解・結合に基づく拡張固有表現抽出手法

岩倉友哉† 高村大也†† 奥村学††
† 株式会社富士通研究所 †† 東京工業大学精密工学研究所
iwakura.tomoya@jp.fujitsu.com {takamura, oku}@pi.titech.ac.jp

1 はじめに

固有表現抽出とは，テキストから，地名や人名，日付や時間といった固有名詞や数値表現などを抽出する技術である．従来，固有表現抽出では，10クラス程度の固有表現 [6] が抽出対象であったが，最近では，情報抽出分野や質問応答システムにおける様々なパタンに対応するために，約200クラスを含む拡張固有表現も提案され [11]，拡張固有表現抽出のためのコーパス整備も行なわれている [16].

固有表現抽出においては，教師あり学習手法が数多く適用されている．以前は，単語単位で判別する分類器を組合わせた手法 [14] が多く用いられていたが，最近では，Semi-Markov モデルに基づく手法 [2, 10]，構造学習手法 [7, 4] などが適用され，高い精度が報告されている．

しかし，Semi-Markov モデルに基づく学習や構造学習を約200クラスを対象とする拡張固有表現抽出に適用する場合，計算コストが問題になると予想される．Semi-Markov モデルに基づく手法では広域な文脈情報を利用するために，入力単語列から単語のチャンクで構成されるラティスを生成し判別を行なう．そのため，固有表現クラス数（K）に加えて，文中の単語数（N），チャンクを構成する単語数の上限値（L）が関係するため，計算量が $O(KLN)$ となる．

また，精度改善を行なうために連接する単語の固有表現タグ情報を考慮する場合は，firs order Markov モデルの構築に構造学習手法を利用することが考えられる．しかし，計算量が $O(K^2N)$ であるため，固有表現のクラス数の増加が計算量に大きく影響する．

その他にも，N-best 出力を利用する方法も提案されている [3, 5]．これらの手法では，Semi-Markov モデルに基づく手法と同様，広域な文脈情報の利用が可能となるが，N-best 生成のための解析に加え，生成した複数の候補から最終結果の選択を実行するため，計算時間はさらに問題になると考えられる．

本論文では，単語チャンク列に対する固有表現抽出手法を提案する．チャンクを単位とした固有表現抽出は，単語チャンク数が単語数 N 以下であることから，計算量 $O(KN)$ にて抽出が可能である．また，Semi-Markov モデルに基づく手法と同様に，チャンクの先頭，チャンクの最後，チャンク全体の単語といった，チャンクから得られる素性が利用可能となり，拡張固有表現のような詳細な固有表現クラス判別において有益であると考えられる．しかし，チャンクは必ずしも固有表現の単位とは一致しないという問題がある．そこで，チャンクを分解・結合する手続きを利用した固有表現抽出方法を提案する．

2 提案手法

本手法では，入力の単語列から単語チャンク列を認識し，SHIFT，POP，JOIN，REDUCE という手続きを用いて，単語チャンク列から固有表現を抽出する．これらの手続きを用いることから，本手法を，SHIFT-POP-JOIN-REDUCE（SPJR）法と呼ぶ．

2.1 初期単語チャンク列の判別

まず，単語チャンクを判別するための固有表現チャンカーの作成方法を説明する．本稿の固有表現チャンカーは，固有表現となる単語チャンクあるいは固有表現以外の単語を判別する．

固有表現チャンカーは，学習用の固有表現タグ付きデータを利用して作成する．ここでは，次を例に固有表現チャンカーの作成方法を説明する．

- [佐藤 太郎]$_{PER}$ [は]$_{O}$ [東京]$_{LOC}$ [出身]$_{O}$

以降の説明では，空白を単語の区切りとし，“[“と“]”の間をチャンクとする．“]” の後の PER と LOC は固有表現クラス名であり，O は固有表現以外の単語という意味で用いる．まず，この学習データを，固有表現の箇所を BNE というタグに置換した次のようなデータに変換する．

- [佐藤 太郎]$_{BNE}$ [は]$_{O}$ [東京]$_{BNE}$ [出身]$_{O}$

続いて，変換後の学習データを用いて，単語チャンクを判別する固有表現チャンカーを作成する．固有表現以外と判別された単語は一単語で一つのチャンクとして扱う．

2.2 チャンクに対する手続き

単語チャンク列から固有表現を抽出するための手続きを説明する．処理はチャンク列の先頭から末尾の方向に実行する．以降，$C = \langle C_1, ..., C_{|C|} \rangle$ を $|C|$ 個のチャンクから構成されるチャンク列，C_i $(1 \leq i \leq |C|)$ を i 番目のチャンクとする．

- REDUCE: 現在のチャンクの固有表現クラスを決定する．REDUCE が実行されると，次のチャンクの処理を開始する．
- POP: 二つ以上の単語から構成されるチャンクから最後の単語を取り出し，その取り出した単語を新しいチャンクとする．チャンク C_i に POP 適用後は，$i+1$ 番目の位置に新しいチャンクが作成される．そのため，まず，i 番目のチャンクの右側にある $i+1$ 番目から $|C|$ 番目のチャンクをそれぞれ一つ右側に移動させる．続いて，C_i から最後の単語 cew_i を取り出し，C_{i+1} とする．POP 実行後はチャンク数が増加する．[1]

[1] POP においては一つ例外を用意する．POP が連続して実行された場合は，元のチャンク情報を可能な限り保持することを目的に，連続して取り出されたそれらの単語は一つのチャンクとして保持する．以降の例では，紙面の都合上，この例外を用いない

- SHIFT: 二つ以上の単語から構成されるチャンクの最初の単語を取り出し，その取り出した単語を新しいチャンクとする．SHIFT を C_i に対して実行する際には，まず，C_i の最初の単語 cbw_i を取り出す．続いて，i 番目から $|C|$ 番目のチャンクをそれぞれ一つ右側に移動させる．この時点で，cbw_i が削除された C_i は C_{i+1} に移動している．最後に cbw_i を C_i とする．SHIFT 実行後はチャンク数が増加する．
- JOIN: 二つの隣接するチャンクを結合し新たなチャンクとする．JOIN を C_i と C_{i+1} に適用する場合，まず，C_i と C_{i+1} を結合し，その結果を C_i とする．続いて，$i+2$ 番目から $|C|$ 番目のチャンクをそれぞれ左に移動させる．JOIN 実行後はチャンク数が減少する．

2.3 固有表現抽出器の学習

固有表現チャンカーを作成後，チャンク列から固有表現を抽出する固有表現抽出器を作成する．本手法では，各手続きをラベルとした学習事例を生成し，教師あり学習手法を用いて手続き選択のためのモデルを構築する．

以降の説明では，T_1,... T_N を N 個の学習データとする．T_i= $\langle T_{i,1}$,... $T_{i,|T_i|}\rangle$ $(1 \leq i \leq N)$ を i 番目の学習データとし，$T_{i,j}$ $(1 \leq j \leq |T_i|)$ を T_i の j 番目のチャンク，$l(T_{i,j})$ を $T_{i,j}$ の固有表現のクラスとする．また，$T_{i,j}$ が固有表現以外である場合は O を返すとする．

学習時の手続きの選択順番はいくつか考えられる．本論文では，固有表現は複数の単語で構成される可能性があるのに対し，固有表現以外となる単語は一単語で構成されることに着目し，固有表現以外となる単語を最後や先頭に含む場合に，POP と SHIFT を優先的に実行する形で，学習事例の生成を行なう．次は，T_i から学習事例を生成する場合の説明である．

- T_i 中のチャンク列を構成する単語列から固有表現チャンカーを用いて初期チャンク列 C を認識する．現在のチャンク位置を $j=1$ とする．
- $j \leq |C|$ の間，以下を実行
  ・（条件 1）C_j と T_j が同一：$l(T_j)$ の REDUCE 事例を生成し，次のチャンクに移動．$(j++)$
  ・（条件 2）C_j の最後の単語が固有表現以外：POP の事例を生成し，POP を実行．C 中のチャンク数が増加．$(|C|++)$
  ・（条件 3）C_j の先頭の単語が固有表現以外：SHIFT の事例を生成し，SHIFT を実行．C 中のチャンク数が増加．$(|C|++)$
  ・（条件 4）C_j に二種類以上の固有表現の構成要素が含まれている：POP の事例を生成し，POP を実行．C 中のチャンク数が増加．$(|C|++)$
  ・（条件 5）(条件 1) から (条件 4) を満たさない：この場合は一つの固有表現を構成する単語が複数のチャンクに存在しているので，JOIN の事例を生成し，JOIN を実行．C 中のチャンク数は減少．$(|C|--)$

N 個の学習データに対して学習事例を生成した後，教師あり学習手法を用いて，手続きを選択するためのモデルを構築する．

次に学習事例の生成例を説明する．次の学習事例 T_i が与えられたとする．

- T_i=[元]$_O$ [A 商事]$_{ORG}$ [の]$_O$ [佐藤]$_{PER}$

まず，学習データ中のチャンク列を構成する単語列から，固有表現チャンカーを用いて，次のような単語チャンク列を得たとする．

- C=[元 A] [商事 の 佐藤]

C 中の下線箇所が現在の対象のチャンクである．

続いて，学習事例の生成を開始する．まず，C_1=[元 A] と $T_{i,1}$=[元] を比較する．ここでは，C_1 と $T_{i,1}$ が一致せず，先頭の単語「元」が固有表現以外であるので（条件 3）となり，SHIFT の事例を生成し，C_1 に対し，SHIFT を実行する．結果，C は次のようになる．

- C=[元] [A] [商事 の 佐藤]

続いての比較では，新たな C_1 と $T_{i,1}$ は一致するので，(条件 1）となり，REDUCE=O というチャンクのラベルを O と決定する REDUCE の事例を生成し，次のチャンクに移動する．

- C=[元] [A] [商事 の 佐藤]

次に，C_2=[A] と $T_{i,2}$=[A 商事] を比較する．ここでは，(条件 1）から（条件 4）にあてはまらず，「A」と「商事」という ORG を構成する二単語が二つのチャンクに別々に存在している状態である．よって，(条件 5）となり，JOIN の事例を生成し，C_2 と C_3 に対し JOIN を実行し，次の結果を得る．

- C=[元] [A 商事 の 佐藤]

続いて，新たな C_2 と $T_{i,2}$ を比較する．C_2 は [A 商事] と [佐藤] の二種類の固有表現を含むので，(条件 4）となり，POP の事例を生成後に，POP を実行し，次の結果を得る．

- C=[元] [A 商事 の] [佐藤]

再度，新たな C_2 と $T_{i,2}$ を比較する．C_2 が $T_{i,2}$ と一致せず，C_2 の最後の単語が固有表現以外の O であるので，(条件 2）となり，POP の事例を生成し，C_2 に対し，POP を実行し，次の結果を得る．

- C=[元] [A 商事] [の] [佐藤]

続いての比較では，C_2 と $T_{i,2}$ が同一であるので，(条件 1）となり，REDUCE=ORG の学習事例を生成し，次のチャンク C_3 と $T_{i,3}$ に移動する．

残りは，C_3 と $T_{i,3}$ が同一で，C_4 と $T_{i,4}$ も同一であるので，それぞれ（条件 1）となり，C_3 に対しては REDUCE=O の学習事例を，C_4 に対しては REDUCE=PER の学習事例を生成し終了する．

2.4 固有表現抽出

抽出時は，まず，固有表現チャンカーを用いて，入力からチャンク列を認識する．続いて，各チャンクに対して適用する手続きを学習したモデルを基に

場合で説明する．

決定し，その手続きを適用する[2]．全てのチャンクの処理を終えたら，チャンク列を各チャンクの固有表現クラスとともに返す．次に抽出例を示す．次の単語列が与えられたとする．
- 鈴木 君 は 京都 出身
学習時と同様，まず，固有表現チャンカーを用いて，単語チャンク列を認識する．次がその結果とする．
- C=[鈴木 君] [は] [京都] [出身]
続いて抽出を開始する．まず，C_1=[鈴木 君] に対する手続きを学習したモデルを基に決定する．ここで，C_1 に対しての手続きとして POP が選択されたとし，C_1 に対して POP を実行する．この結果，[鈴木 君] の最後の単語が新規のチャンクとなるので，C は次のようになる．
- C=[鈴木] [君] [は] [京都] [出身]
続いて，POP 実行後の C_1=[鈴木] に対する手続きの選択を行なう．ここで，REDUCE=PER が選択されたとすると，C_1 の固有表現のクラスを PER と決定し，次のチャンク C_2 に移動する．
- C=[鈴木] [君] [は] [京都] [出身]
次に，C_2 の手続きを選択し，REDUCE=O が選択されたとすると，C_2 の固有表現のクラスを O として，次のチャンク C_3 に移動する．このように残りのチャンクに対しても処理を行なう．

3 実験

3.1 実験データ

毎日新聞 2005 年の約 8,500 記事に対して，191 種類の拡張固有表現がタグ付けされた拡張固有表現コーパス [16] を用いた．本実験ではこのコーパスを次のように分割した[3]．

- 学習データ：2005 年 1 月から 10 月までの記事を利用する．205,876 の固有表現を含む．
- 開発データ：2005 年 11 月の記事を利用する．合計 15,405 の固有表現を含む．パラメータチューニングに利用した．
- 評価データ：2005 年 12 月の記事を利用する．合計 19,056 の固有表現を含む．

3.2 比較対象

本実験では，次のアルゴリズムを比較対象とした．詳細は参考文献を参照願いたい．

- Structured Perceptron (SP) [4]: 単語列に対しタグ付けを行なうための perceptron に基づく構造学習手法である．本実験では，SP のための固有表現タグは IOB1 法で表現する [8] [4].
- Semi-Markov Perceptron (SM) [2]: 単語チャンクのラティスを生成しその上で抽出する．全ての単語チャンクのパタンを展開するのが理想であるが，学習時のメモリ使用量の関係上，単語チャンクの最大長を 10 と制限した[5]．
- Recognition and Classificatio 法 (RC) [1]: 単語チャンク列を認識してから，各チャンクの固有表現のクラスを判別する．本手法との違いは，チャンクの分解や結合は行なわない点にある．
- Shift-Reduce 法 (SR) [15]: 単語列を入力とし，Shift 手続きにて固有表現となる単語チャンクを認識し，Reduce 手続きにて固有表現クラスを判別するという方法で抽出を行なう[6]．

RC，SR，SPJR の学習には，multiclass perceptron [9] を用いた．また，パラメータ推定には，averaged perceptron [4] を用いた．学習の繰り返し回数は 50 回とした．

固有表現チャンカーが必要となる RC と SPJR の学習は次のように行なう．まず，学習データを五分割する．続いて，分割したデータの 4/5 を選択し，2.1 節にあるように固有表現チャンカーを作成する．その後，作成した固有表現チャンカーで，残り 1/5 の学習データの初期チャンク列を判別し，固有表現抽出用の学習データとする．全ての分割結果に対し処理が終わった後に，その結果を使って学習を行なう．抽出用の固有表現チャンカーは，全ての学習データから作成する．本実験では，予備実験の結果，比較対象の中で，学習時間および抽出時間も高速であった Shift-Reduce 法による固有表現抽出手法 [15] を固有表現チャンカーの作成に利用した．

3.3 素性

表 1 に本実験で用いた素性を載せる．素性は ChaSen にて得られる単語と品詞を基にした[7]．

SP の素性は，現在対象の k 番目の単語とその前後二単語の表層文字列と品詞，k 番目の単語のタグ t_k と k 番目と $k-1$ 番目のタグの組合せ t_k, t_{k-1} から生成する．

チャンクを用いる SM，RC，SR，SPJR の素性は，現在対象の j 番目のチャンク内の単語，そのチャンクの先頭に位置する単語の前二単語，そのチャンクの最後に位置する単語の後ろ二単語およびチャンクの固有表現クラス t_j から生成する．

3.4 実験結果

表 2 に実験結果を載せる．本実験では，本提案手法が他の手法より高い F 値を示した．この結果から，抽出の初期からのチャンクから得られる素性の利用

[2] 抽出時には，無限の繰り返しを避けるために，POP あるいは SHIFT の直後の JOIN の実行，JOIN の実行後に複数回 POP が実行され元のチャンクに戻らないようにするためのチェックを行なっている．

[3] IGNORED というタグに囲まれている個所は除外した．

[4] 実験にあたり，IOB1，IOB2，IOE1，IOE2 [13] と Start/End (SE) [14] という五種類のチャンク表現法を比較し，タグの種類数が少ない IOB1 を用いた．タグ数は学習時間にも関係し，予備実験では，202 タグを含む IOB1 法による学習は，最もタグ数が多かった 730 タグを含む SE 法による学習と比較し 2.4 倍高速であった．また，高速化のために，固有表現タグの組合せは，学習データ中に出現した組合せしか利用しないようにした．

[5] 今回，Intel(R) Xeon(R) CPU X5680 @ 3.33GHz と 72GB メモリを搭載した計算機を利用したが，SM を長さ制限なしで動作させたところ，搭載されているメモリを全て使いきってしまい動作しなかった．また，文献 [2] には，複数解を用いたモデル更新方法が示されているが，本実験では，学習時間の関係上，最もスコアの高い解だけを利用した．

[6] この手法では，単語の一部が固有表現となる場合に対応する方法も含むが，今回は，他の手法がその機能を持たないため，単語列上での Shift と Reduce による処理に限定して評価した．

[7] ChaSen-2.4.2 を利用した．辞書には Ipadic-2.7.0 を用いた．連続する数字やアルファベットは連結した．

表 2: 実験結果． F-measure（F 値）, Recall（RE）, Precision（PR）の意味．評価データでの精度測定は，開発データ上で最も高い F 値を示した繰り返し回数を採用．MEM. は学習時のメモリ使用量，TRAIN. は学習時間（単位は時間），PROC. は開発データの処理の時間（単位は秒）．太字は最も良い値．提案手法（SPJR）とその他の手法の結果の差を，文献 [12] のように，McNemar 検定を用いて比較したところ，開発データ，評価データの両方で ($p \langle 0.01$) という結果となった．日本語では，固有表現と単語の境界が一致しないという問題が起きるため，今回は，文字単位でラベル付け結果を比較した．

手法	開発データ：F 値 (RE, PR)	評価データ：F 値 (RE, PR)	MEM.	TRAIN.	PROC.
SP	78.95 (**75.53**, 82.68)	80.62 (**77.36**, 84.18)	2.0GB	85.21	374.03
SM	60.74 (63.06, 58.60)	72.68 (71.43, 73.98)	22.5GB	58.39	349.62
SR	78.38 (75.41, 81.60)	79.66 (76.92, 82.62)	0.79GB	**0.08**	**77.50**
RC	77.95 (68.85, **89.81**)	79.83 (71.28, **90.69**)	**0.64GB**	0.51	95.86
提案手法	**79.21** (75.37, 83.45)	**80.86** (77.21, 84.86)	0.68GB	0.53	109.33

表 1: 実験に利用した素性． SP の素性では，k は単語の位置を示し，w_k は k 番目の単語，p_k は k 番目の単語の品詞である．TT_k は k 番目の単語のタグ t_k と t_k, t_{k-1} の両方が入る．チャンク利用時は，現在のチャンクの先頭の単語の位置を bp，最後の単語の位置を ep とする．ip はチャンク内部の単語を意味 ($bp < ip < ep$). t_j が j 番目のチャンクの固有表現クラス．

チャンク利用なし (SP)
$[TT_k, w_k], [TT_k, w_{k-1}], [TT_k, w_{k-2}], [TT_k, w_{k+1}],$ $[TT_k, p_k], [TT_k, p_{k-1}], [TT_k, p_{k-2}], [TT_k, p_{k+1}],$ $[TT_k, p_{k+2}], [TT_k, p_{k-2}, p_{k-1}], [TT_k, p_{k+1}, p_{k+2}],$ $[TT_k, p_{k-2}, p_{k-1}, p_{k+}], [TT_k, p_k, p_{k+1}, p_{k+2}]$
チャンク利用あり (SM, RC, SR, SPJR)
$[t_j, w_{bp}], [t_j, w_{ep}], [t_j, p_{bp}],$ $[t_j, p_{ep}], [t_j, w_{ip}], [t_j, p_{ip}]$ $[t_j, w_{bp-1}], [t_j, p_{bp-1}], [t_j, w_{bp-2}], [t_j, p_{bp-2}]$ $[t_j, w_{ep+1}], [t_j, p_{ep+1}], [t_j, w_{ep+2}], [t_j, p_{ep+2}]$ $[t_j, w_{bp}, w_{ep}], [t_j, p_{bp}, p_{ep}], [t_j, p_{bp-2}, p_{bp-1}],$ $[t_j, p_{ep+1}, p_{ep+2}]$ $[t_j, p_{bp-2}, p_{bp-1}, p_{bp}],$ $[t_j, p_{ep}, p_{ep+1}, p_{ep+2}]$

と，チャンクの分解・結合の手続きの利用が，F 値改善に貢献したことがわかる．

学習・抽出速度に関しては，本手法では，固有表現チャンカーの学習時間および，固有表現チャンカーによる初期チャンク列の判別時間が必要となるため，SR より若干遅い．また，RC との比較でも，本手法はチャンクの分解・結合処理を行なうため，若干遅い．しかし，SP との比較では，抽出で 3.4 倍，学習は 50 回の繰り返しの時間で約 160 倍高速であった．SM との比較では，抽出で 3.2 倍，学習は 50 回の繰り返しの時間で約 110 倍高速であった．これらの結果から，本手法は，大幅に計算時間を増大させることなく，高い F 値を得られたことがわかる．

4 まとめ

本論文では，チャンクの分解と結合に基づく固有表現抽出手法を提案し，拡張固有表現抽出タスクで評価を行なった．実験結果から，perceptron に基づく構造学習手法や Semi-Markov モデルを用いた固有表現抽出手法と比較し，高い精度を保持しつつ，高速な学習および抽出を実現できることを確認した．今後の課題としては，固有表現クラス数と速度の関係，固有表現チャンカーの振る舞いと精度の関係などの観点からの評価が必要である．

参 考 文 献

[1] Xavier Carreras, Lluís Màrques, and Lluís Padró. Named entity extraction using adaboost. In *Proc. of CoNLL'02*, pp. 167–170, 2002.

[2] William W. Cohen and Sunita Sarawagi. Exploiting dictionaries in named entity extraction: combining semi-markov extraction processes and data integration methods. In *Proc. of KDD'04*, pp. 89–98, 2004.

[3] Michael Collins. Discriminative reranking for natural language parsing. In *Proc. of ICML'00*, pp. 175–182, 2000.

[4] Michael Collins. Discriminative training methods for Hidden Markov Models: theory and experiments with perceptron algorithms. In *Proc. of EMNLP'02*, pp. 1–8, 2002.

[5] Liang Huang. Forest reranking: Discriminative parsing with non-local features. In *Proc. of ACL'08*, pp. 586–594, 2008.

[6] IREX 実行委員会（編）. IREX ワークショップ予稿集. 1999.

[7] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields Probabilistic models for segmenting and labeling sequence data. In *ICML'01*, pp. 282–289, 2001.

[8] Lance Ramshaw and Mitch Marcus. Text chunking using transformation-based learning. In *Proc. of VLC'95*, pp. 82–94, 1995.

[9] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. Vol. 65, No. 6, pp. 386–408, 1958.

[10] Sunita Sarawagi and William W. Cohen. Semi-markov conditional random field for information extraction. In *Proc. of NIPS'04*, 2004.

[11] Satoshi Sekine, Kiyoshi Sudo, and Chikashi Nobata. Extended named entity hierarchy. In *Proc. of LREC'02*, 2002.

[12] Fei Sha and Fernando Pereira. Shallow parsing with conditional random fields In *Proc. of NAACL HLT'03*, pp. 134–141, 2003.

[13] Erik Tjong Kim Sang and Jorn Veenstra. Representing text chunks. In *Proc. of EACL'99*, pp. 173–179, 1999.

[14] Kiyotaka Uchimoto, Qing Ma, Masaki Murata, Hiromi Ozaku, Masao Utiyama, and Hitoshi Isahara. Named entity extraction based on a maximum entropy model and transformation rules. In *Proc. of ACL'00*, pp. 326–335, 2000.

[15] 山田寛康. Shift-reduce 法に基づく日本語固有表現抽出. 情報処理学会研究報告（自然言語処理研究会）, Vol. 2007-NL-179, No. 47, pp. 13–18, 2007.

[16] 橋本泰一, 乾孝司, 村上浩司. 拡張固有表現タグ付きコーパスの構築. 情報処理学会研究報告（自然言語処理研究会）, Vol. 2008-NL-188, No. 113, pp. 113–120, 2008.