拡張固有表現タガーの作成とその問題点の考察

新納浩幸
茨城大学工学部情報工学科

関根聡
ニューヨーク大学

1 はじめに

一般に情報抽出で用いられる固有表現の種類は人名，地名，数量表現などの 10 種類程度である．しかし情報抽出の適用分野の広がりや，質問応答システムに入力される幅広い質問に対処するためには，大規模な固有表現のタイプを設定しておく必要がある．このため，ニューヨーク大学の関根は汎用的に使える固有表現のセットを意図して，200 種類の固有表現を拡張固有表現として設計した [4][3]．我々はこの拡張固有表現のタガーを作成したので，ここに報告する．

固有表現のタガーは帰納学習の手法を用いて作成できるが，拡張固有表現のように多数の固有表現のタイプを設定した場合に，従来の手法がそのまま利用できるかどうかは明らかではない．教師なし学習や能動学習の手法を用いることも提案されてはいるが [5]，それら手法も標準的な帰納学習の手法を核に持つ必要がある．そのため，まず標準的な帰納学習の手法を用いて，拡張固有表現のタガーを作成することにした．ここでは START/END 法 [2] に SVM を組み合わせた手法 [1] を用いる．

単純に実装した場合，処理速度に大きな問題があることが判明したので，この点もあわせて報告する．またこの処理速度の問題への対処として，抽出処理を 2 段階にする手法とビタビアルゴリズムの枝刈りを行う手法を試みた．

実験では，新聞記事約 1ヶ月分のタグ付きデータ [3] を用いた．それらの 8 割を訓練データ，2 割をテストデータとした．START/END 法 + SVM で得られた F-値は 79.6 % であり，タスクの特徴を考慮すると比較的高い値であった．また処理速度の改善については，提案したどちらの手法もある程度の効果を示したが，精度とのトレードオフの関係であり，完全な解決には至っていない．処理速度の問題への対処，更に高いパフォーマンスの実現及び訓練データの作成が今後の課題である．

2 拡張固有表現とそのタガー

拡張固有表現とは関根が設計した 200 種類の固有表現を指す [3]．階層構造を形成しており，その一覧は以下で公開されている．

`http://nlp.cs.nyu.edu/ene/`

我々はこの拡張固有表現のタガーを作成した．利用した手法は START/END 法 [2] である．この手法は，例えば，人名を抽出する場合，以下のような 4 つのクラスを用意する．

`HS`: 人名単語列の始まりの単語
`HM`: 人名単語列の中間の単語
`HE`: 人名単語列の最後の単語
`HI`: 人名単語列が `1` 単語

他の固有表現も同時に抽出したければ，同様に 4 つのクラスを追加すればよい．最後に，抽出対象の単語列とは無関係という NONE というクラスを用意する．そして入力テキストの各単語に上記で設定したクラスになる確率を与える．クラスの並びにはある制約 (例えば HM の直後には HS は現れない) があるので，その制約を満たしたクラス列で生起確率が最大になるような列を求めれば，固有表現の抽出が行なえる．生起確率が最大になる列を求める部分には，一般に動的計画法のビタビアルゴリズムが用いられる．また入力テキストの各単語に設定した各クラスになる確率を求めるために，帰納学習の手法を用いる．ここで用いる帰納学習手法は任意である．近年，SVM が高いパフォーマンスを実現しているので，ここでも SVM を利用することにした．

拡張固有表現の抽出に START/END 法を用いた場合には，801 ($= 4 \times 200 + 1$) 種類のクラスが必要となる．SVM は各単語に対して，この 801 個の各クラスになる確率を与えなくてはならない．これは多値の SVM として実現できる．SVM は二値分類器なので，多値分類を行うために，one-vs-rest 法を用いる．one-vs-rest 法は，$\{C_1, C_2, \cdots, C_n\}$ への多値分類の問題を，C_k かそれ以外かという，n 個の二値分類に分解する手法である．クラス C_k に対する値は，そのクラスに対する SVM によって求まる．ここでは 801 種類のクラスが用意されているので，801 個の SVM を学習する必要がある．

3　標準手法の問題とその解決案

3.1　処理速度の問題

SVM の識別精度は高いが，学習時間や実行時の識別時間に問題がある．

学習時間に関しては問題にならないタスクもある．学習するのが一度だけでよいのなら，長い時間がかかろうとも一度学習してしまえばよい．固有表現抽出のタスクもこの類である．しかし実行時の識別時間の問題は深刻である．特に，ここでのタスクの場合，801 個の SVM を稼働させなくてはならず，START/END 法の前半部分には多大な処理時間が必要となる．

START/END 法では確率を付与する部分の学習手法を任意に設定できるので，識別時間が問題にならない学習手法を使うという選択もあり得る．また SVM の識別時間の短縮する手法も存在する [1]．しかし，この場合でも拡張固有表現抽出のタスクでは，処理時間が問題となる．なぜなら START/END 法の後半部分では，生起確率が最大になるようなクラス列を求めるが，この処理はクラス数が多大であると，膨大な時間を要する．一般に，この部分の処理にはビタビアルゴリズムが用いられる．このタスクの場合，単語数を m，クラス数を n としたとき，生起確率が最大になるようなクラス列を求めるビタビアルゴリズムの処理時間は $O(mn^2)$ であり，クラス数の 2 乗のオーダーである．MUC で設定された 7 種類の固有表現ではクラス数は $29\ (= 4 \times 7 + 1)$ であるが，ここでのタスクのクラス数は 801 なので，ビタビアルゴリズムの処理時間は約 763 倍 $(= 801^2/29^2)$ である．例えば，1 秒で処理できていた処理も，13 分弱かかることになる．

3.2　2 段階処理

上記の処理速度の問題の解決のために，拡張固有表現抽出を 2 段階の処理で行う手法を提案する．

まず 200 種類の固有表現をより小さな n グループに分類する．k 番目のグループ g_k には，c_k 個の固有表現が含まれているとする．その n グループの各グループを新たな固有表現と見立てて，通常の固有表現抽出を行う．この処理結果として，ある単語列が k 番目の固有表現として抽出される．これが第 1 段階目の処理である．2 段階目の処理では，その単語列を拡張固有表現抽出で定義されている固有表現に識別する．この場合，クラス数が c_k 個の分類問題を解くことになる．

また第 1 段階目の処理と第 2 段階目の処理に学習手法を利用する場合，同一の訓練データが使えるという利点もある．

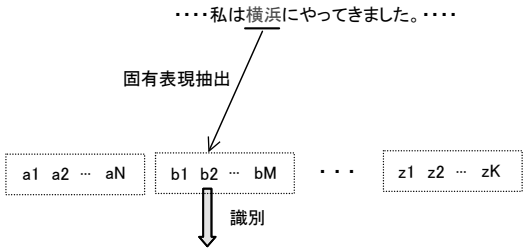

図 1: 2 段階処理

第 1 段階目の処理は通常の固有表現抽出であるので，処理時間は通常のシステムと同程度である．また第 2 段階目の処理は，識別の処理時間が短い学習手法を用いれば，高速に行える．このため，処理時間の観点では，本手法はオリジナルの処理時間を大きく改善できることが期待できる．

3.3　ビタビアルゴリズムの枝刈り

単語の並びが $w_1, w_2, \cdots, w_m$ であり，w_k に対して，各クラス i の確率 p_i^k $(i = 1 \sim 801)$ が付与されているとする．この場合，最大の生起確率になるクラス列を求めるビタビアルゴリズムは図 2 示される．ただし S_i^k は単語 w_k のクラスが i のときに，単語 w_k までの最適なパスのスコアを意味する．

```
for(k = 1; k <= m; k++) {
    for(i = 1; i <= 801; i++) {
      h = arg max_j { S_j^{k-1} + log(p_i^k) };   ⇐ (＊)
      S_i^k = S_h^{k-1} + log(p_i^k);
      /* 単語 w_k がクラス i のとき、
         その直前の単語のクラスは h と記憶 */
      memory_parent(k,i,h);
    }
}
```

図 2: ビタビアルゴリズム

上記アルゴリズムの * の max の引数部分の計算は，全てのクラス j について行われる．提案する手法では，この部分で，S_j^{k-1} の値の大きなクラス j の値 a 個と，NONE を表すクラスの $a + 1$ 個に限定する．ビタビアルゴリズムは最適パスの近似を得る手法だが，本手法はその近似を更に粗く取ったことに相当する．

4 実験

4.1 標準手法の精度

我々は新聞記事約 1ヶ月分に拡張固有表現のタグ付けを行ったデータを有している．このデータの約 8 割を訓練データ，約 2 割をテストデータとして実験を行う．訓練データの文の数は 35,000，テストデータの文の数は 8275 であった．

訓練データ中の各文を JUMAN で形態素解析し，各単語を学習用の事例（属性ベクトルとそのクラスのペア）に変換する．本論文で注目した属性は以下の 10 種類である．

```
e1:  対象単語の表記
e2:  対象単語の品詞
e3:  対象単語の字種
e4:  対象単語の直前単語の表記
e5:  対象単語の直前単語の品詞
e6:  対象単語の直前単語の字種
e7:  対象単語の直後単語の表記
e8:  対象単語の直後単語の品詞
e9:  対象単語の直後単語の字種
e10: 電話番号に関するアドホックなフラグ
```

品詞は JUMAN の解析結果の品詞の細分類を利用している．字種は以下の 9 種類である．

```
Kuten    "、"
Maru     "。"
Ten      "・"
Num-kan  漢字の数字，一二... 九十百千万億兆
Num      アルファベット数字，０１２... ９
Hira     ひらがな
Kana     カタカナ
Kan      漢字
Sym      その他
```

SVM の学習及び実行には SVM-Light[1] を利用した．

前述した START/END 法と SVM を組み合わせて，テストデータから固有表現抽出を行った．結果，再現率 77.5%，適合率 81.7%，F-値 79.6% を得た．

4.2 改良手法の精度

拡張固有表現は意味的な階層構造も設計されている．ここではその構造をほぼ利用して，200 種類の固有表現を 17 種類に分類した．

また第 1 段階目の通常の固有表現抽出では START/END 法 + SVM を用いた．第 2 段階目の具体的固有表現への識別処理では処理速度の観点から決定リストを用いた．

次に START/END 法 + SVM にビタビアルゴリズムの枝刈りの改良を行う手法を，テストデータに適用し，精度を調べた．その結果を表 1 に示す．ただしここでは前述した a の値を 8 に設定している．また表 1 には参考値として，SVM の代わりに決定リストを利用した手法も含めている．表 1 では，“SE-SVM” が START/END 法+SVM を，“2 stages” が 2 段階処理の手法を，“SE-DL” が START/END 法+決定リストを表している．また “○○+改 V” は○○の手法に，ここで提案したビタビアルゴリズムの枝刈り手法を使った手法を表している．

表 1: 改良手法の精度 (%)

手法	再現率	適合率	F-値
SE+SVM	77.5	81.7	79.6
2 stages	78.5	74.5	76.4
SE+DL	65.3	74.9	69.8
SE+SVM+改 V	74.5	80.2	77.2
SE+DL+改 V	63.1	73.1	67.7

2 段階処理は，オリジナルの手法よりも F-値が悪かった．ただし START/END 法 に決定リストを利用した手法と比べると， F-値は高く，SVM を利用した効果が現れている．またオリジナルの手法にビタビアルゴリズムの枝刈りを導入したものは，当然，F-値が低下したが，2% 程度であった．

4.3 改良手法の処理速度

次にテストデータとは別の実際の新聞記事から拡張固有表現抽出を行い，処理速度の測定を行った．実験では適当な政治に関する新聞記事から 50 文，100 文，200 文を取り出して，3 セットの文書を作成し，UNIX の `time` コマンドのユーザータイムにより処理時間を測った[2]．結果を表 2 に示す．

表 2: 改良手法の処理時間 (秒)

手法	50 文	100 文	200 文
SE+SVM	1374.66	3441.36	7157.92
2 stages	584.99	1443.69	2975.91
SE+DL	328.95	851.66	1809.63
SE+SVM+改 V	1090.81	2696.42	5584.09
SE+DL+改 V	35.32	85.45	176.93

2 段階処理では START/END 法+SVM の処理時間を約 6 割削減している．またビタビアルゴリズムの枝刈り手法では，START/END 法+SVM の処理時間を約 2 割削減し，START/END 法+決定リストの処理時間を約 9 割削減している．

2 段階処理も，ビタビアルゴリズムの枝刈り手法も，どちらも処理速度を改善していることがわかる．

[1] `http://svmlight.joachims.org/`

[2] P4 2.4G，メモリ 2G の Linux マシンで測定した．

5 考察

2 段階処理の手法が通常の標準的手法よりも精度が高くなることがあるのかどうかは微妙な問題である．2 段階処理では図 3 のような場合に優位性が出る．

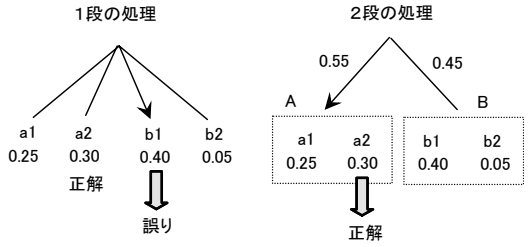

図 3: 2 段階処理の優位例

図 3 では，1 度の処理でクラスを識別すると b1 クラス（誤り）になる．しかし最初にグループ A かグループ B のクラスに識別した後に，クラスを識別すると a2 クラス（正解）に識別される．

図 3 のようなケースが実際におこるかどうかははっきりしないが，第 1 段階の処理で用いるグループ分けが 1 つの鍵になると考えている．ここでは拡張固有表現の意味的な階層関係を利用して，グループ分けを行ったが，この処理に適したようにグループ分けを調整できる可能性もある．例えば，拡張固有表現の定義では「PHONE_NUMBER」は，「LOCATION」の下位になっている．しかし「PHONE_NUMBER」は数文字の並びであるので，処理的には「NUMBER」のグループに入った方がよいかもしれない．

2 段階処理の手法の精度改善の方法として，第 1 段目の処理の再現率と適合率の調整も考えられる．2 段階処理の手法では，第 1 段目の処理で未抽出となる固有表現を抽出することはできず，再現率は第 1 段目の処理で上限が決まる．そのため，第 1 段目の処理では再現率が高くなるように抽出し，第 2 段目の処理では「固有表現ではない」という識別結果も許容することにして，適合率の向上を目指せば結果的には高い F-値が得られることも期待できる．現在は（第 1 段目の処理の）F-値を最大にするように第 1 段目の処理を行っている．上記の点を考慮して，いくつかの実験を行ったが，現在得ているスコア以上の結果は得られなかった．第 2 段目の処理では固有表現の種類を識別する処理であり，一般の固有表現抽出のタスクとは異なる素性が有効になる．そのような素性の導入が精度向上の鍵だと考える．ただし，そのような素性を最初から SVM に組み込んだ場合，標準手法でも精度が向上し，2 段階処理の手法が改善とならない可能性もある．

最後に拡張固有表現抽出のタスクについての考察を述べる．今回の実験では予想以上の結果が得られているが，それはこのタスクが容易であることを意味しない．今回の実験ではテストデータ中の固有表現のタイプにはこだわらず，単純にその数だけを考慮している．200 種類の固有表現全てがテストデータ中に存在するわけではない．実際には 168 種類であった．またその分布も隔たっている．例えば，今回のテストデータ中の固有表現は，「PERSON」「POSITION_TITLE」「DATE」だけで全体のほぼ 3 割を占める．上位 40 個の固有表現（拡張固有表現の 2 割）だけで，全体の 83%を占めている．200 種類の固有表現の中には，頻度は低いが，抽出が困難なものも多く含まれていると思われる．ある対象の分野では低い精度しか得られない状況も十分考えられる．

また訓練データの作成も大きな問題である．今回，我々は比較的大きな訓練データを使えたが，公開されてはいないし，自前で作るににはコストが高すぎる．このため教師なし学習で対処することが提案されてはいるが，このような大量のクラスをもつタスクで，従来の教師なし学習が有効かどうかは疑問である．訓練データの作成については今後の課題である．

6 おわりに

本論文では我々が作成した拡張固有表現のタガーについて報告した．利用した手法は START/END 法 + SVM という標準的な手法であり，我々のテストデータに対して F-値 79.6% という値を出した．これは比較的よい値であるが，処理速度の問題が生じる．

このため固有表現抽出を 2 段階にする手法とビタビアルゴリズムの枝刈りを行う手法を提案した．どちらも速度的にはある程度改善されるが，精度とのトレードオフの関係になっており，完全な解決には至っていない．

処理速度への対処，更に高いパフォーマンスの実現及び訓練データの作成が今後の課題となる．

参考文献

[1] Hideki Isozaki and Hideto Kazawa. Efficient Supprot Vector Classifiers for Named Entity Recognition. In *COLING-2002*, 2002.

[2] Satoshi Sekine, Ralph Grishman, and Hiroyuki Shinnou. A Decision Tree Method for Finding and Classifying Names in Japanese Texts. In *Proceedings of the 6th Workshop on Very Large Corpora*, 1998.

[3] Satoshi Sekine and Chikashi Nobata. Definition, dictionaries and tagger for Extended Named Entity Hierarchy. In *4th international conference on Language resources and evaluation (LREC-2004)*, pp. 1977–1980, 2004.

[4] Satoshi Sekine, Kiyoshi Sudo, and Chikashi Nobata. Extended Named Entity Hierarchy. In *3rd international conference on Language resources and evaluation (LREC-2002)*, pp. 1818–1824, 2002.

[5] 関根聡. 固有表現から専門用語. 第 10 回年次大会 (NLP2004)「固有表現と専門用語」ワークショップ, 2004.