自然言語処理　132－1
（1999. 7. 22）

有限状態変換器の誤り駆動型学習を用いた固有表現抽出

颯々野 学　塚本 浩司

富士通研究所

川崎市中原区上小田中 4–1–1 〒 211-8588

{sassano,tukamoto}@flab.fujitsu.co.jp

あらまし

本論文では、Eric Brill が提案した変換に基づく誤り駆動型学習を日本語の固有表現抽出に適用する方法について述べる。形態素解析と学習で獲得した有限状態変換器 (FST) を使って固有表現の抽出を行なうシステムを作成し、IREX (Information Retrieval and Extraction Exercise) の named entity task の formal run (総合ドメイン) に対して実験を行なった。約 10,000 文の CRL 固有表現データから 1428 個の FST を学習し、F-measure 71.28 を得た。人手作成の FST の性能には及ばないものの、IREX NE に参加するシステムの半数よりもいい結果である。また、過学習が起きないことも確認した。

キーワード　情報抽出, 固有表現抽出, 誤り駆動型学習, 有限状態変換器

Named Entity Extraction Using Error-Driven Learning of Finite-State Transducers

Manabu Sassano and Koji Tsukamoto

Fujitsu Laboratories, Ltd.

4–1–1, Kamikodanaka, Nakahara-ku, Kawaski 211-8588, Japan

Abstract

This papaer describes a method of extracting named entities from Japanese text based on Eric Brill's transformation-based error-driven learning. We developed an extraction system which uses a morphological analyzer and machine-learned finite-state transducers (FSTs), and performed an experiment against the formal run (general topics) of the IREX (Information Retrieval and Extraction Exercise) NE (named entity task). Our system learned 1,428 FSTs from the CRL NE data containing about 10,000 sentences and achieved an overall named entities F-measure of 71.28. The score was lower than that of the hand-crafted FSTs. However, the machine-learned FSTs outperformed the half of the systems participating in the IREX NE. Also, we didn't encounter overfitting in the learning process.

key words　information extraction, named entity task, error-driven learning, finite-state transducer

1 はじめに

近年、ネットワークの普及によりさらに電子化文書の利用が増大している。そうした大量の電子化文書の中から有益な情報を取り出す技術に期待が高まっている。それらの技術の一つに情報抽出がある。

情報抽出の最も基本的な要素技術の課題設定として固有表現抽出がある。MUC (Message Understanding Conference) の NE (Named Entity task) や IREX (Information Retrieval and Extraction Exercise) の NE でも多数の団体が参加し、盛んに研究されている。特に日本語の固有表現抽出は MET (Multilingual Entity Task) や IREX NE で議論されている (Maiorano 1996; Merchant and Okurowski 1996; IREX Home Page)。

固有表現抽出の分野で重要な研究課題の一つは、機械学習の有効性の検証である。実際の情報抽出のシステムでは、人手で作成した規則を利用することが多い。これはコストもかかるし、抽出すべき情報の変化や、文書の分野の変化に対して可搬性が小さい。これら既存システムの弱点を軽減するために機械学習に期待がかかっているのである。

本研究では、情報抽出での固有表現抽出を題材に、機械学習がどの程度有効かを探ることを目指す。今回の報告では、有限状態変換器 (以下では Finite-State Transducer の略である FST を使う) の誤り駆動型学習 (error-driven learning) を日本語の固有表現抽出に適用した結果を述べる。

筆者らが FST の誤り駆動型学習に注目した理由は大きく二つある。一つは、従来の日本語の固有表現抽出に機械学習を利用した研究では、決定木を利用したものはある (大石他 1998; Sekine 1998) が、FST の学習[1]を利用した研究は筆者らが知る限りないからである。もう一つは、固有表現抽出が形態素解析と同じ枠組みでも捉えることができる (2.3 節参照) ことから、英語の形態素解析で有効性が報告されている誤り駆動型学習 (Brill 1995) が、日本語固有表現抽出に対してどの程度有効か検証したいからである。

以下では、我々が採用した有限状態変換器の誤り駆動学習の枠組を説明し、それを使って固有表現抽出を行なった実験結果について報告する。なお、本論文で述べるシステムは IREX NE 参加システムではないことを予め断っておく。

[1](Brill 1995) の品詞タグ付けにおいて、transformation (書き換え規則) は FST とみなせる (Roche and Schabes 1995)。

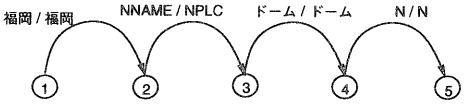

図 1: FST の例

2 有限状態変換器の誤り駆動型学習を用いた固有表現抽出

2.1 固有表現抽出

本論文で対象とした固有表現は、IREX NE で対象としているものである。詳細は (IREX 実行委員会 1999) を参照されたい。IREX NE では次のようなタグを定義している。

ORGANIZATION 組織名、政府組織名

PERSON 人名

LOCATION 地名

ARTIFACT 固有物名[2]

DATE 日付表現

TIME 時間表現

MONEY 金額表現

PERCENT 割合表現

最終的には、抽出対象となるテキストに SGML のタグを付ける。例えば「日本銀行」に対しては、「`<ORGANIZATION>`日本銀行`</ORGANIZATION>`」のように付ける。

2.2 有限状態変換器

有限状態変換器 (FST, Finite-State Transducer) の形式的な定義やその詳細は (Roche and Schabes 1997) に譲る。本論文で述べる範囲では、ある入力シンボル列を受け取りつつ状態遷移を行ない、その遷移と同時に別のシンボルを出力するものと考えて差し支えない。図 1 に例を示す。これは「福岡 `NNAME` ドーム `N`」が入力されたら「福岡 `NPLC` ドーム `N`」を出力する FST を表している (各タグの意味は付録 A を参照のこと)。

以下では我々の FST ツールでの個々の FST の書き方を簡単に紹介する。各エッジを一つの S 式で表現し、そのリストで一つの FST を表現する。エッジの入力シンボルと出力シンボルが同じときは、(福岡) のように書き、違うときは入力シンボルと出力シンボルを / で区切って (`NNAME / NPLC`) のように書く。

[2](IREX 実行委員会 1999) によれば、固有物名とは「人間の活動によって作られた具体物、抽象物を含む物の固有の名前」である。例えば、商品名や作品名、法律名、条約名などが含まれる。

```
((福岡)
 (NNAME / NPLC)
 (ドーム)
 (N) )
```

先の FST では、タグの付け替えしかできないが、ϵ (empty string) 出力と ϵ 入力に対応することで区切り位置の変更も可能になる。次の FST は「日米 NPLC」が入力されたら「日 NPLC 米 NPLC」を出力する。/ の右側が空のとき ϵ 出力を表し、左側が空のとき ϵ 入力を表す。

```
((日米 / )
 (NPLC / )
 ( / 日)
 ( / NPLC)
 ( / 米)
 ( / NPLC) )
```

更に、特殊なシンボル + をその前後のシンボルを連結するオペレータだと解釈するようにシステムを拡張している。次の FST は「神奈川県 NPLC 川崎市 NPLC」が入力されたら「神奈川県川崎市 NPLC」を出力する。

```
((神奈川県)
 (NPLC / +)
 (川崎市)
 (NPLC) )
```

また、任意のシンボルとマッチする ? も導入している。次の FST は、(任意の片仮名未知語) に続けて「州 SUF_NPLC」が入力されると、一つにまとめて地名にする。例えば「アルバータ UNDEF_K 州 SUF_NPLC」が入力されると「アルバータ州 NPLC」を出力する。

```
((?)
 (UNDEF_K / +)
 (州)
 (SUF_NPLC / NPLC) )
```

2.3 形態素解析と有限状態変換器を使った固有表現抽出

我々は日本語形態素解析と FST とを組み合わせて固有表現抽出を行なう (図 2)。形態素解析システムは品詞のタグを付けるが、これを FST によって 2.1 節のタグに一対一に対応するタグ (NE タグと呼ぶことにする。付録 B 参照) に書き換える。そして、それを SGML 形式の出力に変更し、通常の品詞タグを削除することで最終的な結果を得る。このような構成にすると、固有表現抽出をタグ付けの問題に帰着させることができる。形態素解析のときと異なるのは、タグの正解の判定の基準のみである。

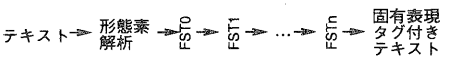

図 2: 形態素解析と FST による固有表現抽出

例えば、次のような FST を使うと、「神奈川県 NPLC 川崎市 NPLC」という形態素解析の結果を「神奈川県川崎市 NE_LCT」と変換する。これを SGML の形にすることで IREX NE で要求されている形式の「<LOCATION>神奈川県川崎市</LOCATION>」を得る。

```
((?)
 (NPLC / NE_LCT) )

((?)
 (NE_LCT / +)
 (?)
 (NE_LCT) )
```

2.4 誤り駆動型学習

Brill が提案している変換に基づく誤り駆動型学習 (transformation-based error-driven learning (Brill 1995)) を少し一般化すると、誤り駆動型学習の基本的な枠組は次のようになるだろう。

Step 0 まずベースラインとなるシステム (M_0) でコーパスを処理する。

Step 1 次に、システムを改善するための選択肢 (m_i) を M_n に追加して、コーパスを処理する。処理の結果を正解と比較して、ある基準を満たす (通常は最もエラーが減少する) かどうか調べる。複数の m_i の中から基準を満たすものを選択し、$M_{n+1} = M_n + m_i, n = n + 1$ として Step 1 を繰り返す。

事前に決められた条件 (例えば、最低何個以上エラーが減少しなければならないなど) を満たしたら学習は終了する。

Brill の POS tagger の場合、ベースラインとなるシステム (Brill の論文の initial-state annotator に相当) は、正解コーパスから各単語について最も頻度が多い品詞を求めておき、そのタグ付けを行なう。また、Step 1 でのシステムを改善する選択肢として transformation すなわち FST を用いている。学習した FST は順序付きリストとして獲得されていく。学習後の実行は、ベースラインシステムでタグ付けした後、その結果に対して順序付きリストを適用する (図 3)。

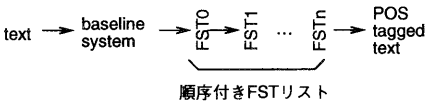

図 3: Brill タガー

2.5 固有表現抽出のための有限状態変換器の誤り駆動型学習

ベースラインシステム

我々のシステムでは、ベースラインシステムとして、形態素解析システム Breakfast (颯々野他 1997) と、品詞タグの NKOYU、NPLC、NNAME をそれぞれ NE_ORG、NE_LCT、NE_PRS に変換する FST を用いた。

正解コーパスの作成

固有表現抽出のための正解コーパスでは、固有表現の部分にしかタグが付いていない。そこで、固有表現以外の部分には通常の品詞タグを付け、それを学習時に使う正解コーパスとした。固有表現のタグが付いているテキストと形態素解析済みのテキストをつき合わせることで作成した。

例えば、固有表現にタグが付いているテキスト「<ORGANIZATION> 神奈川県警 </ORGANIZATION> が銀行 …」と形態素解析済みのテキスト「神奈川 NPLC 県警 N が JS_KAKU 銀行 N …」から「神奈川県警 NE_ORG が JS_KAKU 銀行 N …」を作る。

形態素の区切り位置の変更が必要になる場合は、元のタグの情報を含む適当なタグを与えた。例えば、「<LOCATION> 日本 </LOCATION> 人」と「日本人 NN」からは、「日本 NE_LCT 人 QNE_LCT_NN」を作った。

選択肢となる候補 FST の生成方法

選択肢となる候補 FST の生成には二つの段階を経る。まず第一段階として、学習中のシステムの現在の出力と正解コーパスとを比較し、その違いを吸収するような FST を生成させる[3]。同時に前後の 2 形態素まで含めて生成させる。すなわち、形態素列 $w_{-2}w_{-1}W_0W_1\cdots W_nw_1w_2$ を $w_{-2}w_{-1}W'_0W'_1\cdots W'_mw_1w_2$ に書き換える FST を生成させる (ここでの表記方法は (久光, 丹羽 1998) をベースにした)。ここで w と W、W' は、表記と品詞タグを持つ形態素である。$W_0W_1\cdots W_n$ は誤り部分、$W'_0W'_1\cdots W'_m$ は正解部分、$w_{-2}w_{-1}$ と w_1w_2 はそれぞれ前後の 2 形態素である。

[3](Tashiro et al. 1994) にもあるように、簡単なアルゴリズムによってこのような FST を生成することができる。

例えば「東京 NE_LCT 、 TTN 名古屋 NE_PRS、 TTN 大阪 NE_LCT」と「東京 NE_LCT 、 TTN 名古屋 NE_LCT、 TTN 大阪 NE_LCT」からは次のような FST が生成される。

```
((東京)
 (NE_LCT)
 (、)
 (TTN)
 (名古屋)
 (NE_PRS / NE_LCT)
 (、)
 (TTN)
 (大阪)
 (NE_LCT) )
```

第二段階として、第一段階で生成した FST それぞれに対し、次の一般化を行ない、それら全てを候補 FST とする。

1. w_{-2} を削る。
2. $w_{-2}w_{-1}$ を削る。
3. w_2 を削る。
4. w_1w_2 を削る。
5. $w_{-2}w_2$ を削る。
6. $w_{-2}w_{-1}w_2$ を削る。
7. $w_{-2}w_{-1}w_1w_2$ を削る。
8. $w_{-2}w_1w_2$ を削る。
9. なにもしない。
10. 1.–6. 8. 9. のそれぞれについて、$W_0W_1\cdots W_n$ の部分に、以下のタグがあればその表記部分をスロット (?) にする。
    - 品詞タグ (N NN NSH NNUM UNDEF UNDEF_K UNDEF_AN NKOYU NPLC)
    - 全ての NE タグ (NE_ORG NE_PRS NE_LCT NE_ART NE_DT NE_TM NE_MNY NE_PRC)

    ただし、一つの FST の中でスロットを持つのは一種類のタグに限定する[4]。

先に示した FST の例に 4. と 10. を適用すると、次の FST が生成される。「東京、」の直後に現れる人名は全て地名に変換されることを意味する。

[4]((浜田 /) (NE_PRS /) (町 /) (N /) (/ 浜田町) (/ NE_LCT)) から ((?) (NE_PRS / +) (?) (N / NE_LCT)) を作ったりしない。

```
((東京)
 (NE_LCT)
 (、)
 (TTN)
 (?)
 (NE_PRS / NE_LCT) )
```

FST の評価

IREX NE に従い、FST の評価には、次式のように適合率 (precision) と再現率 (recall) を使って定義される F-measure を使った。

$$\text{F-measure} = 2 * \text{適合率} * \text{再現率} / (\text{適合率} + \text{再現率})$$

適合率は、(システムの出力のうちの正解数)/(システムの全出力数) で定義され、再現率は、(システムの出力のうちの正解数)/(全正解数) で定義される。

FST 選択の方法

Brill のタガーのように、最も精度が上がる (固有表現抽出では F-measure が上がる) FST を単純に探すと、学習には非常に時間がかかる。タグだけでなく、個別の表記を持つものまで試す必要があるし、前後の形態素の異なりも非常に多く、候補 FST の数が膨大になるからである。そこで、今回は以下のような方法で簡単化し、学習の高速化を計った。

1. まず、候補となる FST それぞれにヒューリスティックなコストを付け、その値で順序を付ける。
2. そのコストが高い順に FST を適用して、F-measure を計測する。
3. 前回の学習のサイクルまでの F-measure の値を越えたら、その FST を採用し、順序付きリストに追加する。次の学習サイクルに移る。

ヒューリスティックなコスト (h) を次のように定義した。

- その FST によって正解になる箇所の数 n
- $W_0W_1\cdots W_n$ の中の NE タグ数 i
- $W'_0W'_1\cdots W'_m$ の中の NE タグ数 o
- 考慮に入れるコンテキストの形態素 (w) の数 c
- スロットの数 s

とし、$h = n * (i + o)$ とした。ただし、n には FST を適用することで増える誤りの箇所の数は差し引かれていない。i は FST の適用によって減少する誤りタグの数、o は FST の適用によって増加する正解タグの数を表している。順序付けの際、h が同点の場合には、c が少ないものを上位にし、それでも同じ順位の場合は、s が少ないものを上位にする。

3 実験と考察

3.1 実験システムと実験条件

2 節で説明した枠組みに従って実験システムを作成した[5]。形態素解析システムには Breakfast (見出し数約 139,000。内 **NKOYU** 30,000、**NNAME** 10,000、**NPLC** 7,000) を使い、後段の FST を処理するツールと、候補となる FST の生成するツールを C++ で作成した。誤り駆動の学習器のメインは Perl で記述した。F-measure の計測には、**NEscorer.perl** を改造したものを用いた。

実験には、トレーニングセットとして CRL 固有表現データ (毎日新聞 95 年 1 月 1 日から 10 日までの全記事、約 1 万文に対して固有表現にタグ付けしたもの) を利用した。テストセットとして、IREX NE formal run の総合ドメインの課題 (毎日新聞 99 年 4 月 14 日から 5 月 14 日までの 72 記事。正解ファイルバージョン 1 **general01.idx**) を使った。

なお、**NEscorer.perl**、CRL 固有表現データは、IREX 実行委員会が配布している **IREX990215.tar.gz** に含まれているものである。**general01.idx** は IREX メーリングリストで公開された正解データである。

3.2 実験結果

実際の実験では、処理時間の都合から 1,428 個の FST を獲得したところで学習を打ち切った。獲得した FST の上位 10 個を図 4 に示す。上位には表記を問わないものが多く現れている。スロット (?) を持つ FST の数の推移を図 5 に示す。候補 FST の順序付けのときにスロットの数も考慮に入れたので、スロットの数が同じものが連続して学習される。そのため、段のついたグラフになっている。

表 1 に formal run に対する実験結果を示す。獲得した FST の数と F-measure の関係を図 6 に示す。なお、FST を一つも学習していない段階での F-measure は、トレーニングセット、formal run でそれぞれ 43.14、39.01 であった。

[5] ここで述べるシステムは IREX NE 参加システムではないので注意されたい。我々の実際の参加システムでは、辞書が強化されているほか、人手で作成した FST を基本に、学習で獲得した FST も利用している。共起辞書や人名の照応解析なども行なっている。

```
;; ? + ? NE_PRS <- ? NE_PRS ? NE_PRS
((?)
 (NE_PRS / +)
 (?)
 (NE_PRS) )
;; ? + 年 NE_DT <- ? NNUM 年 SUF_NNUM
((?)
 (NNUM / +)
 (年)
 (SUF_NNUM / NE_DT) )
;; ? + ? NE_LCT <- ? NE_LCT ? NE_LCT
((?)
 (NE_LCT / +)
 (?)
 (NE_LCT) )
;; ? + % NE_PRC <- ? NNUM % SUF_NNUM
((?)
 (NNUM / +)
 (%)
 (SUF_NNUM / NE_PRC) )
;; ? + 日 NE_DT <- ? NNUM 日 SUF_NNUM
((?)
 (NNUM / +)
 (日)
 (SUF_NNUM / NE_DT) )
;; ? + 円 NE_MNY <- ? NNUM 円 SUF_NNUM
((?)
 (NNUM / +)
 (円)
 (SUF_NNUM / NE_MNY) )
;; ? + 軍 NE_ORG <- ? NE_LCT 軍 SUF_NPLC
((?)
 (NE_LCT / +)
 (軍)
 (SUF_NPLC / NE_ORG) )
;; 六日 NE_DT <- 六日 NTIME
((六日)
 (NTIME / NE_DT) )
;; ? + ? NE_PRS <- ? NE_PRS ? NE_PRS
((?)
 (NE_PRS / +)
 (?)
 (NE_PRS) )
;; 今年 NE_DT <- 今年 NTIME
((今年)
 (NTIME / NE_DT) )
```

図 4: 獲得 FST 上位 10 個

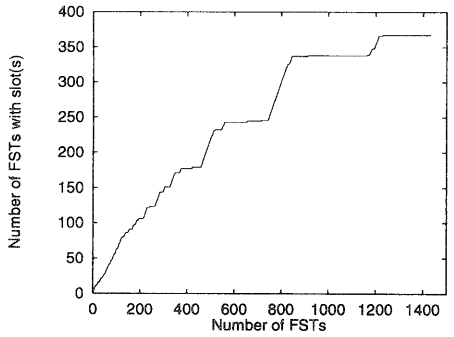

図 5: スロットを持つ FST の数の推移

	Formal Run	
Entity	Recall	Precision
Organization	61.18	78.55
Person	67.89	66.76
Location	73.32	76.44
Artifact	8.16	36.36
Date	76.81	79.40
Time	89.66	83.87
Money	86.67	86.67
Percent	52.38	50.00
Overall	68.14	74.72
F-measure	71.28	

表 1: Formal Run データに対する実験結果

	356 FSTs		548 FSTs	
Entity	Rec.	Prec.	Rec.	Prec.
Organization	53.47	74.82	62.98	80.33
Person	72.11	71.51	72.68	71.07
Location	75.48	74.58	77.64	77.46
Artifact	10.20	83.33	10.20	83.33
Date	93.48	93.14	94.93	93.91
Time	96.55	100.00	96.55	100.00
Money	100.00	100.00	100.00	100.00
Percent	100.00	95.45	100.00	95.45
Overall	71.62	79.06	74.91	81.00
F-measure	75.16		77.83	

表 2: Formal Run に対する人手作成 FST の結果

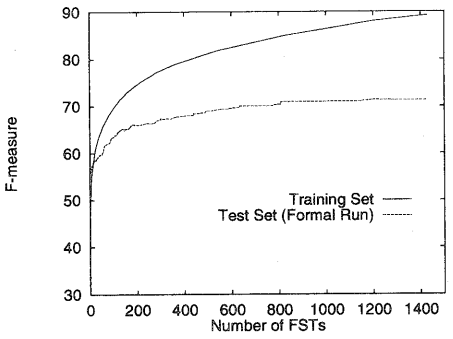

図 6: 獲得 FST の数と F-measure

3.3 考察

過学習 (overfitting)

図 6 を見ると、トレーニングセットに対しては徐々に上がり続けるが、テストセットに対してもほぼ単調に上がり続ける。800 個を学習した辺りから殆んど一定か、かすかに上昇しつつある。多くの学習機構では過学習が問題になるが、今回実験した範囲では過学習は見られなかった。これは、Ramshaw らが変換に基づく誤り駆動型学習を使ったギリシャ語の品詞タグ付けにおいて、過学習が起きないと報告していたこととも一致する (Ramshaw and Marcus 1996)。

人手作成の FST との比較

機械による学習と平行して、人手で書いた FST でどの程度の精度が得られるかも調べた。筆者らの一人がトレーニングセットだけを見ながら、15 日ほどかけて手で FST を書いた。356 個の FST を作成した。更に継続して作業を行ない、合計 35 日ほどで 548 個の FST を作成した[6]。表 2 に formal run に対する結果を示す。

人手で作成した FST のほうが機械が獲得した FST よりも全体的に成績がよい。しかし、356 個の FST のほうと比べると、機械が獲得した FST のほうが organization に関して再現率、適合率ともに上回っている。location でも適合率は上回っている。

4 関連研究

IREX NE には 14 団体 (15 システム) が参加している。同一の課題に対してさまざまな方式のシステムが取り組んでいるため、本論文の我々の結果と比較することができる。1999 年 9 月の IREX ワークショップが開催されれば徐々に明らかになるだろう。1999 年 5 月 28 日現在、15 システムの公開されている結果 (総合ドメイン) によれば、我々の結果は 8 位に相当する (IREX Home Page)。辞書などさまざまな要素が関係するため一概に優劣は言えないが、悪くない結果である。

Vilain らは、FST の誤り駆動型学習を英語の固有表現抽出に適用している (Vilain and Day 1996)。MUC-6 NE に対して組織名、人名、地名に適用し、F-measure 85.2 を得ている。日本語と英語の違いもあり、我々の結果との比較は難しい。Vilain らの場合も人手で作成したもののほうが精度が高い。

佐々木は、形態素解析と有限状態変換器を利用した日本語の固有表現抽出を行なうツールを報告している (佐々木 1999)。基本的な枠組みは我々と同じだと言えるが、機械学習は使われていない。

Kameyama のシステムも同様の構成である (Kameyama 1996)。形態素の区切り位置の変更は行なっていない。

固有表現抽出ではなく、日本語の形態素解析 (特に後処理) に FST を使っているものとして、Matsukawa らの研究や久光らの研究がある (Matsukawa et al. 1993; 久光, 丹羽 1998)。Matsukawa らは、基本的には形態素列をまとめることのみ行なっている。久光らは Brill の手法を簡易化し、FST の順序付きリストは生成していない。

5 おわりに

本論文では、Brill が提案した FST の誤り駆動型学習を日本語の固有表現抽出に適用する方法を示した。IREX NE formal run (総合ドメイン) に対し実験を行ない、F-measure 71.28 を得た。性能は人手で作成したものに及ばないものの、過学習を起こしにくいという優れた性質が確認された。

今回の報告では、他の手法と比べたときの優位性までは示せなかった。IREX ワークショップ後に再度実験し、さまざまな手法との比較を行ないたい。また、今回示した手法でも候補 FST の改良は考えられる。文字種の情報を利用しても良いだろう。ただ、候補 FST の種類を単純に増やすと学習に時間がかかるため、学習器の高速化も同時に考えていく予定である。

A Breakfast の品詞タグ (一部)

`JS_KAKU` 格助詞

`N` 普通名詞

[6]但し、人手で書いた FST には表記のパターンマッチには正規表現を記述しているため、機械が学習している FST の個数とは同列に比べられない。

NKOYU (人名や地名以外の) 固有名詞

NN 人名詞 (英語で he や she で受けるような名詞)

NNAME 人名

NNUM 数詞

NPLC 地名

NSH サ変名詞

NTIME 時間名詞

SUF_NNUM 数詞接尾語

SUF_NPLC 地名接尾語

UNDEF 未知語

UNDEF_AN 英数字未知語

UNDEF_K 片仮名未知語

B NE タグ

NE_ORG Organization (組織名、政府組織名)

NE_PRS Person (人名)

NE_LCT Location (地名)

NE_ART Artifact (固有物名)

NE_DT Date (日付表現)

NE_TM Time (時間表現)

NE_MNY Money (金額表現)

NE_PRC Percent (割合表現)

参考文献

Brill, Eric (1995). "Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging," *Computational Linguistics*, **21**(4), 543–565.

Roche, Emmanuel and Schabes, Yves (1995). "Deterministic Part-of-Speech Tagging with Finite-State Transducers," *Computational Linguistics*, **21**(2), 227–253.

Roche, Emmanuel and Schabes, Yves (1997). "Finite-State Language Processing," MIT Press.

久光 徹, 丹羽 芳樹 (1998). 書き換え規則と文脈情報を用いた形態素解析後処理, 情報処理学会自然言語処理研究会, NL-126-8, 55–62.

IREX Home Page, `http://cs.nyu.edu/cs/projects/proteus/irex/`.

IREX 実行委員会 (1999). 固有表現抽出課題 (version 990214).

Kameyama, Megumi (1996). "MET Name Recognition with Japanese FASTUS," In *Proc. of TIPSTER PROGRAM PHASE II*, 471–474.

Maiorano, Steven (1996). "Multilingual Entity Task (MET): Japanese Results," In *Proc. of TIPSTER PROGRAM PHASE II*, 449–451.

Matsukawa, Tomoyoshi; Miller, Scott; and Weishedel Ralph (1993). "Example-Based Correction of Word Segmentation and Part-of-Speech Labelling, " In *Proc. of Human Language Technology 1993*, 227–232.

Merchant, Roberta and Okurowski, Mary Ellen (1996). "The Multilingual Entity Task (MET) Overview," In *Proc. of TIPSTER PROGRAM PHASE II*, 445–447.

大石 巧, 黒橋 禎夫, 長尾 眞 (1998). コーパス中の特徴と文法的意味的情報を統合的に用いた新聞記事中の固有名詞認識, 言語処理学会第 4 回年次大会発表論文集, 43–46.

Ramshaw, Lance A. and Marcus, Mitchell P. (1996). "Exploring the Nature of Transformation-Based Learning," In Klavans, Judith L. and Resnik, Philip (Eds.), *The Balancing Act*, MIT Press, 135–156.

佐々木 裕 (1999). トランスデューサによる日本語固有表現抽出, 言語処理学会第 5 回年次大会発表論文集, 108–111.

颯々野 学, 斎藤 由香梨, 松井 くにお (1997). アプリケーションのための日本語形態素解析システム, 言語処理学会第 3 回年次大会発表論文集, 441–444.

Sekine, Satoshi (1998). "NYU: Description of the Japanese NE System Used for MET-2," *MUC-7*.

Tashiro, Toshihisa; Uratani, Noriyoshi; and Morimoto, Tsuyoshi (1994). "Restructuring Tagged Corpora with Morpheme Adjustment Rules," In *Proc. of COLING-94*, 569–573.

Vilain, Marc and Day, David (1996). "Finite-State Phrase Parsing by Rule Sequences," In *Proc. of COLING-96*, 274–279.