Web 上の資源から構築した複数の固有表現辞書を用いた日本語固有表現認識

風間 淳一 (kazama@jaist.ac.jp)　鳥澤 健太郎 (torisawa@jaist.ac.jp)
北陸先端科学技術大学院大学 情報科学研究科

1 概要

本研究では，Web 上の資源から構築した大規模な固有表現辞書の日本語固有表現認識における効果を検証する．具体的には，Wikipedia から抽出した大規模な辞書と，Web 文書中での動詞と名詞の係り受け関係を大規模にクラスタリングすることにより構築した辞書について，それを CRF を用いた固有表現認識器の素性として用いた場合の効果を比較検証する．IREX の CRL データを用いた実験により，Wikipeida による辞書，クラスタリングによる辞書ともに，認識精度を向上させることが分かった．また，これら二種類の辞書を同時に使用することにより，精度がさらに向上することも分かった．

2 はじめに

固有表現認識では，固有表現辞書が精度向上のために重要であるということが知られている．しかし，固有表現は非常に数が多く，人手による固有表現辞書の構築・維持にはコストがかかる．そのため，これを大量の文書から自動獲得する手法が従来から研究されてきた (11; 16; 4; 14; 15; 10)．

固有表現辞書を利用する従来の研究の多くは，辞書には，(東京証券取引所, ORGANIZATION) のように，文字列が固有表現クラスとペアになって格納されているものと仮定している[*1]．しかし，一貫性のあるラベルが出力されるのであれば，固有表現クラスでなくても認識器の素性として有用であると考えられる．このように見方を変えると，例えば，自動獲得された上位下位関係 (5) や，名詞クラス (17) などは，固有表現辞書の有望な候補となる．実際，我々は，Wikipedia の記事の定義文から抽出した上位語を返すような辞書を用いることで，英語の固有表現認識の精度が向上することを示した (7)．

本研究の目的は，日本語の固有表現認識を対象として，(i) 上で挙げた二つ目の可能性である名詞のクラスを辞書として用いることの有効性を示す, (ii) 英語で効果の確認された Wikipedia による辞書の有効性を日本語に対しても示す，(iii) これら二つの異なる辞書を同時に用いることの有効性を示すことである．

固有表現は非常に種類が多いため，精度向上のためには，辞書の規模も重要になる．Web 上の資源に注目するのはそのためである．例えば，Wikipedia からは，英語で 200 万項目以上, 日本語でも 55 万項目程度の辞書が簡単に構築できる．

本研究では，名詞のクラスタリングには，鳥澤 (17) の提案した方法を用いる．これは，動詞と名詞の係り受け関係を隠れクラスをもつ確率モデルで表し，それを EM アルゴリズムにより学習するものである．従来から，単語クラスをタガーの素性として用いる研究はあるが (6; 9)．これらは，隣接する単語の情報のみを用いて学習をする HMM や (3) のような方法を用いている．しかし，固有表現辞書を構築するには，単語だけでなく，複数の語からなる複合名詞も同時にクラスタリングする必要がある．鳥澤の方法は，係り受け関係の項として複合語も問題なく扱えるため都合がよい．また，係り受け関係により距離の離れた動詞との関係を考慮できるため，意味的によりきれいなクラスが学習できることも期待できる．

この EM による方法も含めて，クラスタリングは一般的に計算コストが大きい．我々の目標は，大量の Web 文書から係り受けデータを大量に集めて，大規模な名詞クラスを作成することであるが，鳥澤の方法では，数十万語規模のクラスタリングになると，　大量のメモリが必要になるため，一般的な計算機での実行は困難になる．そこで，本研究では，鳥澤のアルゴリズムを MPI を用いて並列化し，複数の計算機で分散して実行できるようにした．これにより，複合語を含む 50 万語の名詞を，クラス数最大 3,000 でクラスタリングすることが，実用的なメモリと計算時間で可能になった．

IREX(13) の CRL 固有表現データを用いた実験により，日本語の場合でも，Wikipedia 辞書により精度が向上すること (5 分割交差検定による F 値で 87.62% から 88.14%)，また，クラスタリングによる辞書によっても精度が向上すること (87.62% から 88.45%) が分かった．また，これら二つの辞書を同時に用いることで，精度がさらに向上し，最終的には F 値で 88.93% を達成した．

論文の構成は以下の通りである．まず，節 3 で辞書の獲得手法について説明する．節 3.1 では，鳥澤の手法とその並列化について説明し，節 3.3 では，Wikipedia からの獲得手法を説明する．節 4 では，獲得した固有表現辞書を，固有表現認識器の素性として用いる方法について説明する．節 5 で実験結果を報告し，節 6 で今後の課題などについて述べる．

3 辞書の獲得

3.1 係り受け関係のクラスタリングによる獲得

鳥澤の方法 (17) では，以下のような，名詞と動詞の係り受けの関係に関する隠れクラスモデルを仮定する．

$$p(v, r, n) = \sum_c p(\langle v, r\rangle|c)p(n|c)p(c). \quad (1)$$

ここで, $v \in \mathcal{V}$ は動詞, $n \in \mathcal{N}$ は v と係り受け関係 r にある名詞, $c \in \mathcal{C}$ は隠れクラスである．名詞 n は，複合語も含む．以下，本論文では，名詞は複合語も含むとして説明を行なう．関係 r は, 名詞につづく助詞で表す．例えば,「ベルギービールを飲む」という文からは, $(v, r, n) =$ (飲む, を, ベルギービール) という関係が得られる．注意点としては,「れる」などの助動詞は名詞 n を大きく変化させるので，動詞 v の一部としてつなげて用いる．また,「N_1 の N_2」という名詞と名詞の関係も，$v = N_2, r =$ の, $n = N_1$ と考えることで，クラスタリングに用いる．これは，このような「の」表現の N_2 によっても，N_1 の性質がよく表わされるからである．

L 個の学習データ $\{(vt_i, n_i, f_i)\}_{i=1}^{L}$ を考える．ここでは，説明を簡単にするため，$vt \equiv \langle v, r\rangle \in \mathcal{VT}$ とした．f_i は，係り受け関係 (vt_i, n_i) がコーパスで現れた回数である．EM アルゴリズムによるクラスタリングでは，この学習データの対数尤度を最大にするような $p(vt|c)$, $p(n|c)$, $p(c)$ を，E ステップと M ステップの反復により求める（実際の反復式は，文献 (17) を参照）

本研究では，学習に必要な係り受け関係を収集するのに，約 7,600 万の Web 文書を用いた．ここから, KNP(21) を用いる

[*1] 同一の文字列に複数のクラスが対応することもあり得るが，本研究では辞書はただ一つの可能性のみ出力すると仮定する．

ことによって，約 3.8 億個の (vt_i, n_i, f_i) の形の学習データを得た．

学習により得られた確率パラメータを用いて辞書を構築することができる．方法はいくつか考えれるが，本研究では，n に対して， $c' = \operatorname{argmax}_c p(c|n) = \operatorname{argmax}_c p(n|c)p(c)$ で定まるクラス c' の番号をラベルとして返す辞書を構築した．

3.2 クラスタリングの並列化

上のクラスタリングでは，パラメータ $p(vt|c)$, $p(n|c)$, $p(c)$ を保持するために $O(|\mathcal{VT}||\mathcal{C}| + |\mathcal{N}||\mathcal{C}| + |\mathcal{C}|)$ のメモリ領域が必要となる[*2]．また，時間計算量は，I を反復回数として $O(L \times |\mathcal{C}| \times I)$ である．これを大規模に行なおうとする場合，メモリ領域がまず問題になる．例えば，浮動小数点数が 8 byte を消費すると仮定すると，$|\mathcal{N}| = 500,000$, $|\mathcal{VT}| = 500,000$, $|\mathcal{C}| = 3,000$ の時，パラメータを保持するのに約 42GB を必要とする．また，約 3.8 億個の学習データはメモリ上では約 4GB 消費する．メモリのコストが下がった現在でも，46GB のメモリを搭載した単一のマシンを利用することはそれほど容易ではない．そこで本研究では，上の設定よりさらに大規模なクラスタリングまで視野に入れて，アルゴリズムを各ノードが 8GB 程度のメモリを積む安価な PC クラスタ上で実行できるように並列化することにした．

まず，学習データはディスク上に置くことにした．各ノードの 8GB のうち 4GB を消費してしまうのを避けるためである．これは，今後さらに大規模な学習データを用いるためにも必要である．学習データへのアクセスは連続的であるため，バッファリングをすれば速度的にそれほど問題にはならない[*3]．

次に，パラメータを保持する行列を分散させるため，パラメータ行列をクラス方向で分割した．つまり，各ノード l はクラス集合の部分集合 $\mathcal{C}_l$ に関する値のみを保持する（P 個の計算ノードに分散させる場合，$\mathcal{C}_l$ は $|\mathcal{C}|/P$ 程度のサイズになる）．この様なパラメータの分割により，メモリ量に関しては線形にスケールするようになる（P 台で P 倍のサイズの問題を解くことが可能）．メモリ量に関する線形性を実現するためには，現時点ではこれが唯一の選択肢だと思われるが，この方法には，反復の最も内側のループを複雑化し，速度を低下させるというトレードオフもある．しかし，本研究では，まずメモリ量の問題を解決し，大規模なクラスタリングを実現することを優先した[*4]．

鳥澤のアルゴリズムの最も内側のループでは，学習データ (vt_i, n_i) が与えられたとき，各々のクラス c に対して，

$$p(c|vt_i, n_i) = p(vt_i|c)p(n_i|c)p(c)/Z \qquad (2)$$

を計算する必要がある．ただし，$Z = \sum_c p(vt_i|c)p(n_i|c)p(c)$ は，正規化項である．この Z は，全てのクラスについて，$p(vt_i|c)p(n_i|c)p(c)$ の値が分からないと計算できない．従って，並列化したループは，図 1 のようなものになる．ここでは，並列化のために MPI を使用した．まず，各ノード l は自分の受け持ちのクラス $c_l \in \mathcal{C}_l$ について $p(vt_i|c_l)p(n_i|c_l)p(c_l)$ を計算し，それらの和を $localZ$ にセットする．次に，MPI の集合通信関数である MPI_Allreduce を用いて，各ノードの $localZ$ の和を計算し，それを各ノードの Z にセットする．最後に，各ノードは，Z の値を使い受け持ちのクラスに対する正しい $p(c|vt_i, n_i)$ の値を計算する．

並列化の基本的なアイデアは上記の通りであるが，MPI_Allreduce には通信にかかるオーバーヘッドがあるので，内側のループで毎回呼ぶのはコストが大きい．そこで，実際には，学習データを B 個ずつまとめ，B 個毎に一回だけ通信を行なうようにする．MPI_Allreduce は，配列の各要素に対して操作を行なうこともできるので呼び出し自体は一回にすることができる．これにより，毎回呼ぶのに比べて大幅に効率化することができる．実験では，$B = 4,096$ を用いた．

この並列化により，Opteron 2.6 GHz, 8GB メモリ × 8 ノード，ギガビットイーサネット，という環境で，最大 $|\mathcal{N}| = 500,000$, $|\mathcal{VT}| = 500,000$, $|\mathcal{C}| = 3,000$, $I = 150$ という設定でのクラスタリングを約 1 週間で行なうことができた．我々の知る限り，このクラスタリングをこのように大規模に行なった研究は他にはない．図 2 は得られた名詞クラスの例である．

Algorithm 3.1: Compute $p(c_l|vt_i, n_i)$

$localZ = 0, Z = 0$
for $c_l \in \mathcal{C}_l$ **do**
$\begin{cases} d = p(vt_i|c)p(n_i|c)p(c) \\ p(c_l|vt_i, n_i) = d \\ localZ \ += \ d \end{cases}$
MPI_Allreduce($localZ$, Z, 1, MPI_DOUBLE, MPI_SUM, MPI_COMM_WORLD)
for $c_l \in \mathcal{C}_l$ **do** $p(c_l|vt_i, n_i) \ /= \ Z$

図 1　並列化した最も内側のループ．各ノードがこのコードを並列に実行する．

クラス 791	クラス 2760
ウィンダム	マリンスタジアム
カムリ	大阪ドーム
ディアマンテ	ナゴド
オデッセイ	福岡ドーム
インスパイア	大阪球場
スイフト	ハマスタ

図 2　固有表現を含む（きれいな）クラスの例．名詞は $p(c|n)$ の値に従ってソートされている．右のクラスには複合語の固有表現が集まっている．また「ナゴド」などの興味深い省略形も含まれている．

文字	に	ソ	ニ	ー	が	開	発	···
マッチのみ	O	B	I	I	O	O	O	···
マッチ + ラベル	O	B-会社	I-会社	I-会社	O	O	O	···

図 3　辞書素性の例（文字ベースのタガーの場合）

3.3 Wikipedia からの獲得

我々が，(7) で提案した方法用いた．これは，Wikipedia の記事の第一文から，その項目の上位語に対応する語を抽出するというものである．辞書の定義文から上位語を獲得することは古くから行なわれており (18)，(7) はそれを Wikipedia に応用したものといえる．本研究では，これを日本語用に修正して用いた．(7) では，英語を対象として，第一文の最初の be 動詞の後の最初の名詞句の最後の名詞を上位語として獲得していたが，これを，文の最初の助詞「は」の後の「最後」の名詞を抽出するように変えた（ただし，英語の時と同様に「一種」や「ひとつ」「こと」などは無視する）．これにより，2007 年 4 月 10 日時点の日本語版 Wikipedia から，550,832 個の辞書項目を獲得した．このうち，482,599 項目では何らかの上位語を抽出することに成功した[*5]．

4 辞書の素性としての利用方法

本研究では，辞書の情報は図 3 のように，マッチの情報，あるいは，マッチ＋ラベルの情報を IOB2 タグでエンコードし

[*2] プログラム中では，パラーメータを保持する行列がそれぞれ二つずつ必要になる．

[*3] ただし，NFS などは使わず，各ノードがローカルディスクにコピーを持つようにする．

[*4] つまり，並列化に関して，速度に関する台数効果は考えていない．

[*5] 第一文が抽出できない場合や，曖昧性解消ページのように第一文にきれいな定義文がない場合には上位語の抽出に失敗する．

表 1　各固有表現辞書の基本的な統計.

辞書	# entries	# matches	# e-matches	# optimal	pre.	rec.	opt. rec.	# labels	# used
wikip (m)	550,054	225,607	6,804	7,602	3.02%	48.4%	54.1%	N/A	N/A
wikip (l)	550,054	189,029	5,441	6,064	2.88%	38.7%	43.1%	12,786	1,708
c100k-2k	99,671	193,897	6,822	8,233	3.52%	48.5%	58.6%	2,000	1,910
c300k-2k	295,695	178,220	7,377	9,436	4.14%	52.5%	67.1%	2,000	1,973
c300k-1k	↑	↑	↑	↑	↑	↑	↑	1,000	982
c300k-3k	↑	↑	↑	↑	↑	↑	↑	3,000	2,848
c300k-4k	↑	↑	↑	↑	↑	↑	↑	4,000	3,681
c500k-2k	497,101	174,482	7,470	9,798	4.28%	53.1%	69.7%	2,000	1,951
c500k-3k	↑	↑	↑	↑	↑	↑	↑	3,000	2,854

て使用する．ただし，複数のマッチの可能性があるときには，最左最長一致を優先する．また，Wikipedia の場合，上位語の抽出の失敗によりラベルがない場合にはタグは出力しない[*6]．これらのタグは，単語などの他の素性と同じようにタガーの中で素性として用いることができる．

5　実験

実験データとしては，IREX (13) の CRL 固有表現データを用いた．このデータでは，1,174 個の新聞記事に 8 種類の固有表現タグが付与されている．これを変換し，18,677 個の固有表現を含む 11,892 文を得た[*7]．さらに，これを学習セット (9,000 文)，開発セット（1,446 文），評価セット（1,446 文）に分割して実験に使用した．

本研究では，(1) と同じように文字ベースのタグ付けを行なうことにした．タガーとしては Conditional Random Field (CRF)(8) を用い，タグの形式は IOB2 を採用した．ベースラインモデルの素性は，日本語固有表現認識の研究でこれまで使用されてきた標準的な，文字，文字タイプ，形態素素性，文節に関係する素性 (19) を用いた．これらの素性を最大で ±2 の位置まで見て，CRF のノード素性，エッジ素性を生成した[*8]．素性は，実験を通して，学習データ中で 2 回以上出現する表層部分[*9]を持つ素性のみを使用した．

CRF の学習には，CRF++ (ver. 0.44) を用いた[*10]．過学習を防ぐため，Gaussian prior を用い，その分散 σ^2 は開発データでチューニングした．このためには，各手法で，$\sigma^2 = \{0.64, 1.28, 2.56, 5.12, \ldots, 163.84, 327.68\}$ の 10 点を調べた．

5.1　各辞書の効果

まず，これまで述べてきたクラスタリングによる辞書と Wikipedia による辞書の素性としての効果を検証・比較した．クラスタリング辞書に関しては，様々な語彙数，クラス数の辞書を作成しその効果を見た．クラスタリングの反復の回数は，いずれも，150 回とした．

表 1 は，これらの辞書に関する基本的な統計である．「wikip (m) 」はマッチのみの場合の Wikipedia 辞書，「wikip (l)」は上位語のラベルまで用いる場合である．「cXk-Yk」は，$|\mathcal{N}| = |\mathcal{VT}| = X \times 1,000$，$|\mathcal{C}| = Y \times 1,000$ という設定で学習した名詞クラスによる辞書を表す．「# entries」は辞書の項目数である[*11]．「# matches」は学習セットに対して出力されたマッチの数である．「# e-matches」は，そのうち正解の固有表現の分割と一致したものの数である．「# optimal」は，正解の固有表現に最大限マッチさせた場合のマッチ数の上限である．「pre.」はマッチの精度 (= # e-matches/# matches)，「rec.」はマッチのカバレッジ (= # e-matches/# NEs．ただし，# NEs = 14,056) である．「opt.」は，カバレッジの上限値 (= # optimal/# NEs) である．「# labels」は，辞書が返すラベルの種類の数，「# used」は，その内，学習セットに対して実際に出力されたラベルの種類の数である．

表から，これらの辞書には，40% − 50% 程度のカバレッジがあることが分かる．また，クラスタリング辞書のほうが，カバレッジが大きい傾向にある．Wikipedia 辞書に関しては，英語の場合 (7) は，wikip(m) のカバレッジは 70.6%, wikip(l) で 49.6% であったので，日本語版は英語版に比べてカバレッジが劣る．しかし，それでも精度向上の効果があることを次の実験で示す．また，マッチの精度は，いずれの辞書でも非常に低い．これは，辞書にひらがな 1 文字のみの項目などが含まれているためである[*12]．しかし，このような大量の誤マッチは精度にはほとんど影響しない[*13]．これは，CRF などの機械学習の力を示していると言えよう．また，「rec.」と「opt. rec.」を比べると，現在の最左最長優先のマッチ方法が最適ではないことも分かる．例えば，「c500k-2k」の数値を見ると，約 16% のマッチが失われていることが分かる．

表 2 は，これらの辞書素性をベースラインに加えた場合の性能である．辞書素性を加える場合には，マッチのみの素性とマッチ＋ラベルの素性を同時に加えている．まず，Wikipedia 素性は日本語の場合でも効果があることが分かる．向上幅は評価セットに対する F 値で 1.08 ポイントである．また，テストした全てのクラスタリング辞書も精度を向上させた．最大の向上は，「c300k-3k」のときで，1.55 ポイントだった．これは，Wikipedia 辞書の効果よりも大きい．「c300k-Yk」の結果を見比べると，クラスタ数は大きければ良いということではなく，あるところにピークがあることが分かる．ここでは，$|\mathcal{C}| = 3,000$ が最も良かった．また，「cXk-2k」の結果から，辞書のサイズは大きければ大きいほど効果が大きいという傾向がありそうである[*14]．

5.2　組み合わせの効果

表 3 は，Wikipedia 辞書素性とクラスタリング辞書素性を両方加えた場合の性能を示している．多くの場合で，組み合せることで性能が向上しているが，いくつかのケースでは開発セットでの精度は低下している．従って，この段階では組み合わせの一般的な有効性は断定できないが，効果はありそうである．実験を通しての最高性能は，Wikipedia 辞書とク

[*6] 現在の実装では，最左最長一致でのマッチを先に行ない，そのあとタグを出力するか決定する．
[*7] タグが曖昧な場合に付与されている OPTIONAL タグは無視した．
[*8] いくつかの素性では，連続する位置を繋げた bi-gram 素性も用いる
[*9] 素性のタグの部分を抜いたもの．
[*10] 歴史的な経緯から，forward-backward の部分で HMM で用いられるのと同様の scaling を実装し，最適化ルーチンを TAO (2) の LMVM モジュールに変更してある．
[*11] 辞書を TRIE 構造に変換するときに，ゴミや重複を除いているので元データよりわずかに少なくなっている．
[*12] Wikipedia には「あ、アは、日本語の音節のひとつであり...」のような記事がある．
[*13] 逆に，1 文字の項目を辞書から取り除くと精度がわずかに低下する．
[*14] 300k-3k と 500k-3k を比べると，この推測とは矛盾するが，これはクラスタリングの初期値のランダム性に起因した偶然かも知れない．この点はさらに調査が必要である．

表 2　各固有表現辞書の効果

モデル	F 値 (開発)	F 値 (評価)	best σ^2
baseline	87.23	87.42	20.48
+wikip	**87.60**	**88.50**	2.56
+c300k-1k	88.74	87.98	40.96
+c300k-2k	88.75	88.01	163.84
+c300k-3k	**89.12**	**88.97**	20.48
+c300k-4k	88.99	88.40	327.68
+c100k-2k	88.15	88.06	20.48
+c500k-2k	88.80	88.12	40.96
+c500k-3k	88.75	88.03	20.48

表 3　組み合わせの効果. *は精度が向上したことを示す.

モデル	F 値 (開発)	F 値 (評価)	best σ^2
+wikip+c300k-1k	88.65	*89.32	0.64
+wikip+c300k-2k	*89.22	*89.13	10.24
+wikip+c300k-3k	88.69	*89.62	40.96
+wikip+c300k-4k	88.67	*89.19	40.96
+wikip+c500k-2k	***89.26**	***89.19**	2.56
+wikip+c500k-3k	*88.80	*88.60	10.24

表 4　過去の研究との比較（5 分割交差検定）

手法	F 値
Asahara&Matsumoto (2003) (1)	87.21
中野&平井 (2004)(19)	89.03
山田 (2007) (20)	88.33
Sasano&Kurohashi (2008) (12)	89.40
本研究 (ベースライン)	87.62
本研究 (+wikip)	88.14
本研究 (+c300k-3k)	88.45
本研究 (+c500k-2k)	88.41
本研究 (+wikip+c300k-3k)	**88.93**
本研究 (+wikip+c500k-2k)	88.71

ラスタリング辞書「c500k-2k」を組み合せた時の 89.26%(開発）/89.19%(評価）であった．これは，評価セットでの F 値が，ベースラインから 1.77 ポイント向上したことを意味する．

5.3　既存手法との比較

既存研究では，CRL 固有表現データでの 5 分割交差検定の精度を示していることが多い．そこで，本研究でも，学習データを文書単位で連続する 5 個の集合に分け，いくつかのモデルについて，5 分割交差検定を行なった．σ^2 は前の実験で見つかった最良の値をそれぞれ用いた．表 4 は，この結果を最近の既存研究の結果と共に示したものである．まず，交差検定でも，前の実験と同様に辞書の有効性が確認された[*15]．また，提案手法（+wikip+c300k-3k）は，高い精度を報告している既存研究と同等の高い精度を達成している．最近発表された，Sasano&Kurohashi (12) の性能が最も良いが，これは本研究では用いていない様々な構造的素性を用いたものであり，今回提案したような固有表現辞書と同時に用いることで，さらに性能を向上できる可能性もある．

6　今後の課題とまとめ

本研究では，Wikipedia から抽出した固有表現辞書と，Web 文書中での動詞と名詞の係り受け関係をクラスタリングすることにより構築した固有表現辞書が，CRF を用いた固有表現認識器の精度を向上させることを示した．また，これら二種類の辞書を同時に使用することにより，精度がさらに向上することも分かった．今後の課題としては，より洗練されたマッチの方法によりカバレッジを上げること，本研究では扱わなかった同一の文字列に対して複数のラベルの可能性があるという曖昧性の問題を取り扱うことなどが考えられる．また，クラスタリングについては，クラスタリング結果を人が見たときの感覚的な良さと精度向上が必ずしも一致しないといった現象も観察されているので，より詳しく調査をしていきたい．加えて，高い精度を示している他の研究の成果を取り入れることも行なっていきたい．

[*15] 向上幅は，多少減少しているが，これは，前の実験の開発・評価セットの特性かも知れない．あるいは，σ^2 をそれぞれの fold について最適化していないためかもしれない．

参考文献

[1] M. Asahara and Y. Motsumoto. Japanese named entity extraction with redundant morphological analysis, 2003.

[2] S. Benson, L. C. McInnes, J. Moré, T. Munson, and J. Sarich. TAO user manual (revision 1.9). Technical Report ANL/MCS-TM-242, Mathematics and Computer Science Division, Argonne National Laboratory, 2007. http://www.mcs.anl.gov/tao.

[3] P. F. Brown, V. J. Della Pietra, P. V. deSouza, J. C. Lai, and R. L. Mercer. Class-based n-gram models of natural language. *Computational Linguistics*, Vol. 18, No. 4, 1992.

[4] O. Etzioni, M. Cafarella, D. Downey, A. M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Unsupervised named-entity extraction from the Web – an experimental study. *Artificial Intelligence Journal*, 2005.

[5] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora, 1992.

[6] J. Kazama, Y. Miyao, and J. Tsujii. A maximum entropy tagger with unsupervised hidden Markov models. In *NLPRS 2001*, 2001.

[7] J. Kazama and K. Torisawa. Exploiting Wikipedia as external knolwdge for named entity recognitioin. In *EMNLP-CoNLL 2007*, 2007.

[8] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML 2001*, pp. 282–289, 2001.

[9] S. Miller, J. Guinness, and A. Zamanian. Name tagging with word clusters and discriminative training. In *HLT-NAACL04*, 2004.

[10] D. Nadeau, Peter D. Turney, and Stan Matwin. Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In *19th Canadian Conference on Artificial Intelligence*, 2006.

[11] E. Riloff and R. Jones. Learning dictionaries for information extraction by multi-level bootstrapping. In *16th National Conference on Artificial Intelligence (AAAI-99)*, 1999.

[12] R. Sasano and S. Kurohashi. Japanese named entity recognition using structural natural language processing. In *IJCNLP 2008*, 2008.

[13] S. Sekine and H. Isahara. IREX: IR and IE evaluation project in Japanese. In *IREX 2000*, 2000.

[14] K. Shinzato, S. Sekine, N. Yoshinaga, and K. Torisawa. Constructing dictionaries for named entity recognition on specific domains from the Web. In *Web Content Mining with Human Language Technologies Workshop on the 5th International Semantic Web*, 2006.

[15] P. P. Talukdar, T. Brants, M. Liberman, and F. Pereira. A context pattern induction method for named entity extraction. In *CoNLL 2006*, 2006.

[16] M. Thelen and E. Riloff. A bootstrapping method for learning semantic lexicons using extraction pattern context. In *EMNLP 2002*, 2002.

[17] K. Torisawa. An unsupervised method for canoniclization of Japanese postpositions. In *NLPRS 2001*, 2001.

[18] H. Tsurumaru, K. Takeshita, K. Iami, T. Yanagawa, and S. Yoshida. An approach to thesaurus construction from Japanese language dictionary. In *IPSJ SIG Notes Natural Language vol.83-16, (in Japanese)*, 1991.

[19] 中野桂吾, 平井有三. 日本語固有表現抽出における文節情報の利用. 情報処理学会論文誌, 2004.

[20] 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 2007-NL-179, 2007.

[21] 黒橋禎夫, 河原大輔. 日本語構文解析システム knp version 2.0 使用説明書, 2005.