Vol.2015-NL-220 No.1
2015/1/19

係り受け情報を利用した日本語形態素解析

俵 雄貴[1,a)]　東 藍[1,b)]　松本 裕治[1,c)]

概要：現在までに様々な形態素解析手法が提案されており，形態素解析の精度は高い水準に達している．その一方で既存の手法では上手く解析できない事例が報告されている．本研究では，そういった事例に対して係り受けの情報を用いることにより解決を試みる．しかし，係り受けの情報を使うためには少なくとも文が単語に区切られている必要があり，形態素解析の段階で係り受けの情報を利用することは困難である．そこで本研究では形態素解析と係り受け解析を同時に行うことにより，係り受けの情報を形態素解析に利用する．同時解析では，形態素ラティスに対して CYK アルゴリズムを適用し，形態素の並びのスコアと係り受けのスコアの２つのスコアを考慮することにより解析を行う．

1. はじめに

現在までに様々な形態素解析手法が提案されており，形態素解析の精度は高い水準に達している．その一方で既存の手法では上手く解析できない事例が報告されている．中村等 [1] は現代日本語書き言葉均衡コーパス，RWCP テキストコーパス，日本語話し言葉コーパスに対して MeCab で解析を行い，形態素解析結果の誤り傾向を分析している．それによると，特に助詞/格助詞の「で」と助動詞の「で」の間の誤りが目立ち，品詞同定誤りの 1/3 を占めていると報告している．

なぜこのような誤りが起こりやすいかは，MeCab で用いられているアルゴリズムと，助詞の「で」と助動詞の「で」が文中でどうように使われているかを合わせて考える事によって理解することができる．この２つの形態素は用法によっては隣接する前後の形態素が同一になることがある．MeCab では隣接する前後の単語をもとに系列のコストを計算し，解析を行うので，隣接する形態素が同一だと曖昧性をうまく解消できない．同じことは，隠れマルコフモデルを用いた系列ラベリング [2] や分類器を用いた点推定 [3] といったモデルをもとにした形態素解析器にも当てはまる．

助詞の「で」の直前に出現した名詞（句）は格要素として扱われるが，助動詞の「で」の前に出現した名詞は述語として扱われ，統語的な取り扱いが大きく異なってくる．このような形態素解析の誤りは述語項構造解析や機械翻訳といった高次の処理において，大きな問題になってくる．

我々はこのような系列の情報だけでは解くことが難しい問題に対して，係り受けの情報を利用することで解決を試みる．そうすることにより，隣接する単語といった狭い範囲の情報だけでなく，もっと広い情報を取り入れた解析が可能になる．しかも，単に窓幅を広げる以上の，言語学的な観点からみて意味のある語の情報を取り入れることができる．しかし，形態素解析を行う段階では文が単語に区切られておらず，係り受けの情報を利用することは難しい．そこで，我々は形態素解析と係り受け解析を同時に行うことによって，係り受けの情報を形態素解析に利用することを試みる．同時解析では形態素ラティスに対して CYK アルゴリズムを適用し，系列スコアと係り受けスコアの２つのスコアを考慮することにより解析を行う．

本稿の構成は以下の通りである．まず，2 節で本研究の関連研究を紹介する，そして，3 節で前提知識となる CYK アルゴリズムによる係り受け解析について説明し，4 節で提案手法について述べる．5 節では実験について説明し，6 節で実験結果の考察を行う．最後に今回の研究をまとめ，今後の課題について述べる．

2. 関連研究

日本語では，係り受けの情報を形態素解析に利用した研究はほとんどされていないが，そのような研究として岸本等の研究 [4] がある．岸本等の研究では，形態素解析の N ベスト解を係り受け情報を用いてリランキングするという形で係り受け情報を取り入れており，形態素解析においても係り受けの情報が重要であることを報告している．

日本語に比べて，中国語では品詞付与と構文解析（係り

[1] 奈良先端科学技術大学院大学
Nara Institute of Science and Technology
[a)] tawara.yuki.tn7@is.naist.jp
[b)] ai-a@is.naist.jp
[c)] matsu@is.naist.jp

受け解析や句構造解析）を同時に行う研究が盛んに行われている．その理由として，中国語においては品詞付与と構文解析を別々に行っても，高い精度を達成することが困難であることが挙げられる．

単語分割，品詞付与，構文解析を同時に行っている研究としては羽鳥等 [5] と Qian 等 [6] の研究がある．羽鳥等は，いくつかのアクションを加えた，文字ベースの遷移型依存構造解析アルゴリズムによって同時解析を実現している．Qian 等はトレーニング時には単語分割のモデル，品詞付与のモデル，構文解析のモデルを別々に学習し，デコード時には，3 つのモデルを組み合わせて CYK アルゴリズムの枠組みでデコードすることにより，同時解析を実現している．

同時解析の研究は英語や屈折語においても行われており，Bohnet 等 [7]，[8] は遷移型依存構造アルゴリズムをベースとした手法に様々な素性を組み込むことで，解析が難しい屈折語においても高い精度を実現している．

我々の提案する手法は Qian 等の手法に極めて近い．しかし，Qian 等は形態素解析（単語分割と品詞付与）と句構造解析の同時解析を行っているのに対して，我々は形態素解析と係り受け解析の同時解析を行っている．また，彼らは文字を基本単位として処理を行っており，非常に処理が複雑になっているが，我々は辞書引きによって構築した形態素ラティスをもとに処理を行うので，既存の CYK アルゴリズムの単純な拡張で済んでいる．

3. CYK アルゴリズムによる係り受け解析

本節では同時解析アルゴリズムを理解するために必要な CYK アルゴリズムによる係り受け解析について説明する．

最初，文中の各単語はそれ単体からなる係り受け木とみなされる．木が覆う範囲のことをスパンと呼ぶ．日本語の場合，右の単語から左の単語へ係ることはないので，木のスパンの一番右側にある単語がその係り受け木の主辞となる．CYK アルゴリズムでは，スパンが隣接している係り受け木同士を合併することで，ボトムアップに係り受け木を組み上げていく．ここで，隣り合う係り受け木を合併することは，左側の木の主辞から右側の木の主辞への係り関係があることを意味する．また，CYK アルゴリズムでは，係り受け木の良さを評価するためのスコアを用いる．そして，木を組み上げていく際には，このスコアが高い係り受け木が残るよう組み上げを行う．

この木の組み上げを効率良くかつ取りこぼしのないように行うために，CYK アルゴリズムでは，チャートテーブルと呼ばれる 2 次元のテーブルを考える．このテーブルは組み上げた木を管理するためのテーブルで，1 つ目のインデックスがスパンの幅を表し，2 つ目のインデックスがスパンの開始位置を表す．例えば，チャートテーブルを Table という変数で表すとすると，Table[4][5] には，スパンの幅が 4 で開始位置が 5 の係り受け木が含まれている．ここで，幅や位置の単位は単語でなく文字であり，スパンの開始位置とはスパン中の一番左端の文字が文中で何文字目なのかを表している．

図 1 CYK アルゴリズムによる係り受け解析の擬似コード

```
Table = initializeTable(sentence)
for w = 1 to W do
  for s = 0 to W − w do
    candidateTrees = {merge(t_l, t_r) | t_l ∈ Table[i][s], t_r ∈
    Table[w − i][s + i], 1 ≤ i < w}
    betterTrees = selectNBest(condidateTree)
    Table[w][s].push(betterTrees)
  end for
end for
return selectBest(Table[W][0])
```

CYK アルゴリズムの擬似コードを図 1 に示す．この擬似コードは形態素解析された文 $sentence$ を受け取り，その係り受け木を返す処理を表している．$initializeTable$ 関数は解析対象の文をもとにチャートテーブル $Table$ を初期化している．初期化処理では，文中の各単語をその単語のみからなる係り受け木に変換し，それぞれの木をスパンの幅や開始位置をもとにチャートテーブル中の適切な位置に配置を行う．$merge$ は 2 つの係り受け木を合併した係り受け木を返す関数である．$selectNBest$ はスコアが高い N 個の係り受け木を返す関数，$selectBest$ は一番スコアが高い係り受け木を返す関数である．

4. CYK アルゴリズムによる同時解析

本節では同時解析のアルゴリズムについて説明する．前節で説明した CYK アルゴリズムは自然な形で入力をラティスに拡張することができる [9] ．すなわち，チャートテーブルの初期化処理において，解析対象の文から展開した形態素ラティスに含まれる単語すべてをチャートテーブルに配置するように変更すればよい．そこで，我々は形態素ラティスに対して CYK アルゴリズムを適用することにより同時解析を行う．係り受け解析において CYK アルゴリズムを利用する場合，解消しなければならない曖昧性は係り受けの曖昧性である．したがって，2 つの係り受け木を合併した木のスコアは合併前の 2 つの木が係り受け関係にあるかどうかを元に設計される．しかし，入力を形態素ラティスとする同時解析において CYK アルゴリズムを利用する場合，係り受けの曖昧性に加えて系列の曖昧性も解消しなければならない．そこで我々は，2 つの係り受け木を合併した木のスコアを係り受けスコアと系列スコアの 2 つの部分スコアの線形和という形で表現する．すなわち，左の係り受け木を t_l，右の係り受け木を t_r としたとき，それらを合併した木 t のスコアを以下のように計算する．

$$Score(t) = \alpha \cdot depScore(t_l, t_r) + \beta \cdot seqScore(t_l, t_r) \quad (1)$$

ここで α，β は非負の実数値，$depScore$ は係り受けスコア，

$seqScore$ は系列スコアである．α と β を変化させることにより，係り受けスコアと系列スコアの重要度を調整することができる．また，このようにスコアを部分スコアの線形和という形で設計することにより，学習時には各部分スコアのモデルを別々に学習することが可能になる．

4.1 系列スコア

このスコアは形態素の並びの尤もらしさを評価するためのスコアである．そのようなモデルとしては，N グラムによる言語モデル，隠れマルコフモデル，条件付き確率場（CRF）といったモデルがあげられる．今回，我々は形態素解析に用いられ高い精度を達成している CRF [10] を用いた．

左の係り受け木のスパンに含まれる単語系列および品詞系列をそれぞれ，$\vec{x}_l$，$\vec{y}_l$ とする．右の係り受け木のスパンに含まれる系列についても，同じように $\vec{x}_r$，$\vec{y}_r$ と定める．そうすると，系列スコアは以下のように計算される．

$$seqScore(t_l, t_r) = \log P(\vec{y}_l, \vec{y}_r | \vec{x}_l, \vec{x}_r) \quad (2)$$

ここで，$P(\vec{y}_l, \vec{y}_r | \vec{x}_l, \vec{x}_r)$ は CRF による条件付き確率である．

4.2 係り受けスコア

このスコアは単語の係り受け関係の尤もらしさを評価するためのスコアである．このスコアは CYK による係り受け解析を行うときに用いられるスコアと全く同じスコアを利用することができる．今回，我々は素性の重み付き線形和によりこのスコアをモデル化する．すなわち左右の係り受け木 t_l と t_r をもとにした素性ベクトルを $\vec{x}$，素性の重みベクトルを $\vec{w}$ としたとき，係り受けスコアは

$$depScore(t_l, t_r) = \vec{x} \cdot \vec{w} \quad (3)$$

と計算される．素性の重みベクトル $\vec{w}$ は構造化パーセプトロン [11] によって学習を行う．

4.3 形態素ラティスの枝刈り

CYK アルゴリズムによる解析時間のオーダーは入力列の長さに対して 3 乗と，非常に高い．通常，文を展開したときの形態素ラティスに含まれる単語数はその文に含まれる単語数と比べて数倍から数十倍になるので，形態素ラティスを計算オーダーの高い CYK アルゴリズムの入力とすると，解析に膨大な時間がかかってしまう．そこで，形態素ラティスに対して CYK アルゴリズムを行う前に形態素ラティスの枝刈りを行い，予め不要なノードを削除する処理を行った．実験の章で詳しく述べるように，この処理は単に解析時間を短縮するだけでなく，初期の段階で不要な曖昧性を解消することにより，CYK アルゴリズム中で正解がビームから漏れることを防ぎ，結果的に解析精度向上にもつながった．

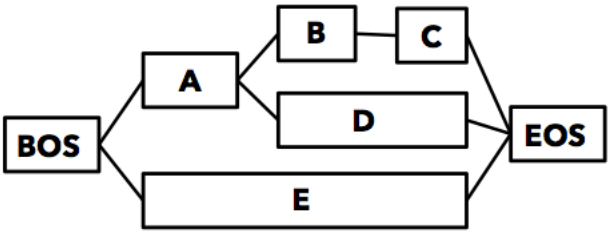

図 2 形態素ラティスの例．ノード B を削除した場合，ノード C は前部のエッジが 1 本も無くなってしまうので，ノード C も削除する

次に，形態素ラティスの枝刈りを行う方法について述べる．我々が採用した方法は，Li 等の方法 [12] とほぼ同じものである．系列のスコアをもとにして，形態素ラティス中の各ノード（単語）に対して，そのノードの周辺確率を求めることができる．この周辺確率はフォワードバックワードアルゴリズムを用いることによって，ラティス中の全ノードに対して効率的に求めることができる．そして，この周辺確率があるしきい値よりも低いノードを形態素ラティスから削除する．あるノードを削除したときに，図 2 のように，隣接するノードの前部のエッジ又は後部のエッジがひとつもなくってしまうような場合，周辺確率の値に係わらず，そのようなノードも一緒に削除する．また，あるノードを削除したときに，形態素ラティスが非連結になってしまう場合，そのノードの削除は行わない．

4.4 パーセプトロンの並列化

学習を高速化するためにパーセプトロンの並列化を行った．McDonald 等 [13] は Iterative Parameter Mixing というパーセプトロンを並列化する手法を提案しており，この手法に基づき並列化を行った．この手法ではトレーニングデータを分割し，それぞれのトレーニングデータで並列して重みベクトルの更新を行っていく．そして 1 イテレーションが終わるごとに，別々に更新した重みベクトルの平均ベルトルを求め，それを次のイテレーションにおける重みベクトルの初期値とし，また同じように並列して重みベクトルを更新していく．この手法によるパラメータ更新はトレーニングデータが線型分離可能なら収束することが保証されている．McDonald 等はこの手法を係り受け解析の学習に適用し，通常のパーセプトロンよりも学習時間が短縮されただけでなく，解析精度も向上したことを報告している．本稿の実験においてもこの手法を採用した．

5. 実験

提案手法の形態素解析性能を評価するために実験を行った．本節ではその実験について述べる．

表 1　コーパスの分割

訓練データ	開発データ	評価データ
25000 文	3000 文	5000 文

5.1　データセット

今回の実験では CoNLL 2009 の Shard Task で使用されたデータセットを用いた．このデータセットは京都大学テキストコーパスをもとに，文節係り受けから単語係り受けに変換したコーパスである．このデータセットを表 1 のように訓練データ，開発データ，評価データの 3 つに分割し実験を行った，

5.2　ベースライン

提案手法と比較する 3 つのベースラインについて説明する．

5.2.1　通常の形態素解析

このベースラインは通常の形態素解析によるものである．このベースラインには MeCab を用いた．

5.2.2　リランキング

このベースラインでは，まず CRF による形態素解析で N ベストの形態素結果を出力する．次に，それぞれの形態素解析結果に対して係り受け解析を行う．そして，形態素解析のスコアと係り受け解析のスコアの重み付き線形和が一番高い結果を最終的な解析結果とするものである．このベースラインは岸本等 [4] の研究に倣った手法である．形態素解析には MeCab を利用した．また，係り受け解析には，同時解析手法の節で説明した係り受けスコアをもとにした CYK アルゴリズムで解析を行った．

5.2.3　事後訂正

このベースラインは一旦 CRF による形態素解析で解析を行い，その解析結果に対して誤りやすい箇所に限定して訂正を行う手法である．1 節でも述べたように，形態素解析で誤りやすい箇所は非常に限定されている．今回は「で」の訂正のみを行った．事後訂正のモデルには SVM を用いた．SVM の素性として，訂正対象形態素の前 2 単語と後ろ 3 単語の正規形，品詞大分類，品詞小分類，活用型と活用形を使用した．このベースラインは中村等の研究 [1] に倣った手法である．形態素解析は MeCab で行い，SVM には LIBLINEAR[*1] を用いた．

5.3　提案手法の設定

提案手法で利用する系列スコアには MeCab で学習した生起コストと連接コストを利用した．また，今回は未知語の影響を無くすために，形態素辞書にはコーパスに出現する全ての形態素を登録した．

係り受けスコアの計算に利用した素性を表 2 に示す．Head-l は左の係り受け木の主辞を，Modif-l は左の係り受け木の修飾辞を，Head-r は右の係り受け木の主辞を，Modif-r は右の係り受け木の修飾辞をそれぞれ表している．カッコ内は用いた形態素の素性を表している．また，スラッシュは複数の素性を組み合わせたことを表している．例えば，3 番目の素性は左の主辞の活用形と右の主辞の品詞の組み合わせ素性であることを表している．

表 2　係り受けスコアに利用した素性

1	Head-l(品詞) / Head-r(品詞)
2	Head-l(正規形) / Head-r(正規形)
3	Head-l(活用形) / Head-r(品詞)
4	Head-l(品詞小分類) / Head-r(品詞)
5	Head-l(品詞小分類) / Head-r(品詞小分類)
6	Head-l(品詞) / Head-r(品詞小分類)
7	Head-l(品詞) / Head-r(正規形)
8	Head-l(品詞小分類) / Head-r(正規形)
9	Head-l(活用形) / Head-r(品詞小分類)
10	Head-l(正規形) / Head-r(品詞)
11	Head-l(正規形) / Head-r(品詞小分類)
12	Head-l(活用形) / Head-r(正規形)
13	Modif-l(品詞) / Head-l(品詞) / Head-r(品詞)
14	Modif-l(正規形) / Head-l(正規形) / Head-r(品詞)
15	Modif-l(活用形) / Head-l(品詞) / Head-r(品詞)
16	Modif-l(品詞小分類) / Head-l(品詞小分類) / Head-r(品詞小分類)
17	Modif-l(活用形) / Head-l(品詞小分類) / Head-r(品詞小分類)
18	Modif-l(品詞) / Head-l(品詞) / Modif-r(品詞) / Head-r(品詞)

係り受けスコアの学習にはパーセプトロンを使用し，4 節で述べたように Iterative Parameter Mixing による並列化を行った．学習のイテレーション回数は 20 回とした．

5.4　ハイパーパラメータの調整

今回の提案手法では，CYK アルゴリズム中のスコアの重み，形態素ラティスの枝刈りを行う時のしきい値，デコード時のビーム幅の 3 種類のハイパーパラメータが存在する．ビーム幅は 2 に固定して実験を行った．スコアの重みとしきい値に関しては，開発データを用いて個別に決定した．まず，予備実験を行い比較的低めのしきい値をひとつ決定する．そして，スコアの重みを決定する際にはしきい値をその値に固定し，スコアの重みを変化させ，最適な重みを決定する．その後，スコアの重みを先ほど求めた値に固定し，しきい値を変化させ，最適なしきい値を決定した．

またリランキングによるベースラインにおいても，N ベスト数とスコアの重みという 2 つのハイパーパラメータが存在する．N ベスト数は岸本等の研究 [4] において，$N = 5$ で十分な精度が出ていることから 5 に固定した．スコアの重みについては開発データをもとに最適な重みを決定した．

6.　結果と考察

6.1　係り受け解析の結果

係り受けスコアがどの程度正しく学習できているかをみるために，係り受けスコアを用いて CYK アルゴリズムによる係り受け解析を行ったときの結果を表 3 に示す。こ

[*1] http://www.csie.ntu.edu.tw/~cjlin/liblinear/

表 3　係り受け解析の結果

	文単位	単語単位
正解率	26.38 %	91.12 %

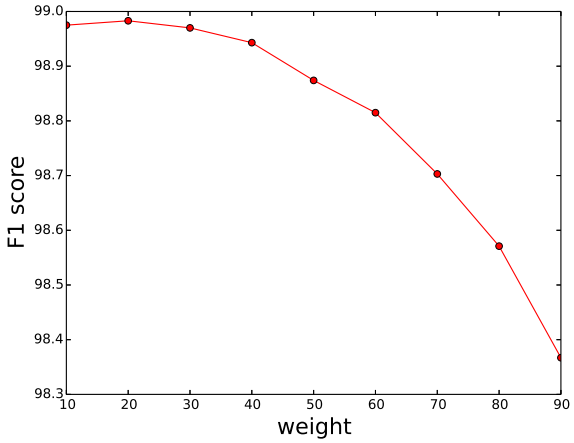

図 3　係り受けスコアの重み α を変化させていったときの品詞大分類の F1 スコア．

の結果は，CYK アルゴリズムの入力を Gold の形態素列，CYK アルゴリズム中のスコアを係り受けのスコアのみ考慮するようにしたとき，すなわち，通常の係り受け解析と同じ枠組みで解析を行ったときの結果である．

この結果から学習した係り受けスコアがある程度正しく係り受け関係をスコア化できていることがわかる．

6.2　2 つのスコアの関係

形態素ラティスの枝刈りを行うときのしきい値を $\lambda = -15000$ と固定，系列スコアの重みを $\beta = 0.1$ と固定し，係り受けスコアの重みを $\alpha = 10, 20, ..., 90$ と変化させていったときの結果を図 3 に示す．この結果から $\alpha = 10, 20$ あたりで高い精度に達していることがわかる．そこで，他のパラメータを固定したまま，係り受けスコアの重みを $\alpha = 0, 3, ..., 30$ と変化させ，さらに詳しく調べた．その結果を図 4 に示す．その結果 $\alpha = 24$ で品詞大分類の F 値において最も良い結果を示した．以上の結果から，形態素ラティスを入力として同時解析を行う場合であっても，系列のスコアが欠かせないことがわかる．また，図 4 より，系列スコアと係り受けスコアを同時に考慮することにより，より精度の高い解析を行える可能性が窺える．

6.3　形態素ラティスの枝刈りの効果

係り受けスコアの重みを上で得られた $\alpha = 24$（系列スコアは $\beta = 0.1$）に固定し，今度は枝刈りのしきい値を $\lambda = -15000, .., -1000, -500, -100, 0, 100, 300, 2000, 5000$ と変化させていったときの結果を図 5 に示す．また，開発データを用いてラティスの枝刈りの性能を調べたので，その結果を図 6 に示す．Sentence Accuracy は，どれだけ正

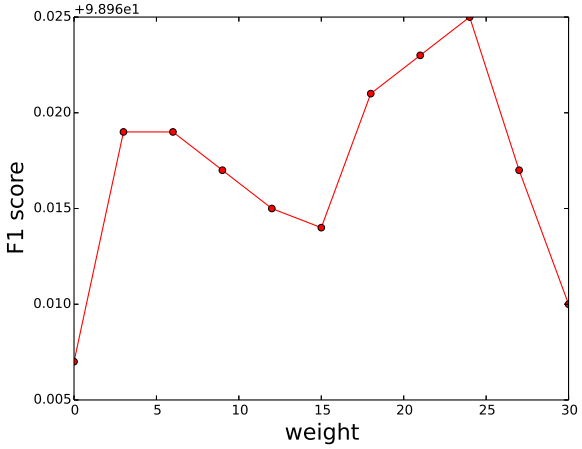

図 4　係り受けスコアの重み α を更に変化させていったときの品詞大分類の F1 スコア $\alpha = 24$ で最大となっている．

解のノードを削除せずに済んだかを示す指標で，正解ノードをひとつも削除されなかった文の割合を示す．Reduce Rate は一文あたりどの程度ノードを削除できたを示す指標で，各文ごとに以下の式で削除率を算出し，それを全ての文で平均した値である．

$$\frac{\text{元のラティスのノード数} - \text{削除したノード数}}{\text{元のラティスのノード数}} \quad (4)$$

図 5 と図 6 から，削除率を増やしていくと，ある地点を境に F 値が急上昇し，それ以上削除率を増やしても F 値はあまり下がらないことが読み取れる．これから，大幅なノードの削除を行っても，解析の精度にはほとんど影響しないということがいえる．このようなことが起こる原因として，同時解析時のビーム幅が挙げられる．同時解析においては係り受けの曖昧性と系列の曖昧性の両方に対処しなければならない．したがって，ビーム幅があまりに狭いと，解析の途中で正解がビームからこぼれてしまうおそれがある．今回はビーム幅を 2 に固定して実験を行ったが，予備実験でビーム幅を 1 に固定して解析を行った際には，形態素解析と係り受け解析の精度のどちらもビーム幅 2 のときに比べて，大きく精度を下げてしまった．このようなことから，ビーム幅の狭さが図 5 のような結果となった原因ではないかと考えられる．

また，形態素ラティスの枝刈りを行ったことによって，速度的にも大きな改善がみられた．

6.4　同時解析の結果

最後に提案手法である同時解析とベースラインの結果を表 4 に示す．ベースラインである MeCab や事後訂正の係り受け精度は，それぞれの形態素解析結果に対して通常の係り受け解析を行った結果である．同時解析における各種ハイパーパラメータは上で求めた通りである．ベースラインの 1 つであるリランキングによる手法でのスコアの重みは，開発データでグリッドサーチを行った結果，係り受け

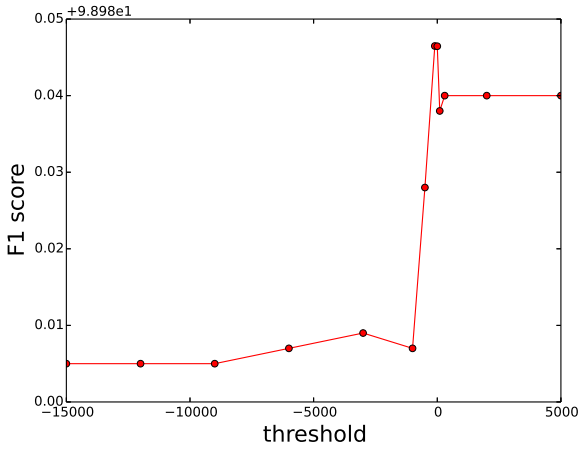

図 5　枝刈りのしきい値 λ を変化させていったときの品詞大分類の F1 スコア．λ = −100 で最大となっている．

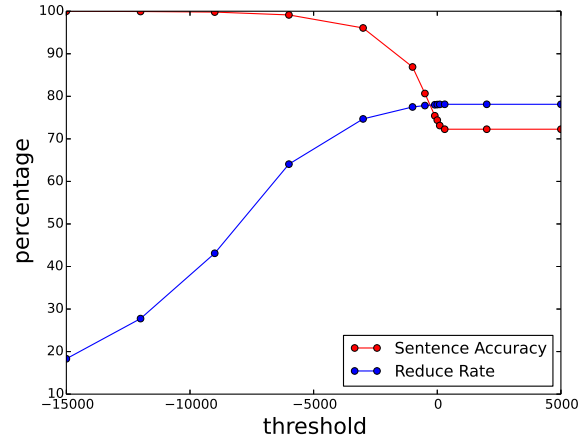

図 6　枝刈りのしきい値を変化させていったときの，品詞大分類の F1 スコア

スコアの重み α が α = 18 となった（β は β = 0.1 で固定）．

結果を見ると，数値の上では提案手法である同時解析手法はベースラインの 1 つである MeCab を形態素解析結果において上回っている．しかし，bootstrap resampling 法を用いて，分割，品詞大分類，形態素の有意差を調べたところ，有意差を確認することはできなかった．また，誤り傾向を分析するために形態素解析のエラー分析を行ったが，エラーとして確認できるのは，ほとんどがコーパス中のアノテーションの不整合が原因で解析を誤ったと考えられる事例だった．

係り受け解析の精度に注目すると，提案手法は 3 つのベースラインのいずれにも僅かながら劣っている．この原因としては，6.3 節でも述べたように，ビーム幅に比べて解くべき曖昧性が高く，正解がビームから落ちてしまった可能性が考えられる．

7. おわりに

本稿では，形態素解析に係り受けの情報を利用するという目的で，CYK アルゴリズムによる形態素と係り受けの同時解析手法を提案した．そして，提案手法の有効性を示すために CoNLL 2009 の Shard Task で使われたコーパスを用いて実験を行ったが，有意な結果を出すことは出来なかった．また，実験のエラー分析を行う過程で，使用したコーパスに多くのアノテーション揺れがみつかり、提案手法の有意差が出なかったひとつの要因なのではないかと考えている．今回，実験に使用したコーパスは新聞記事をもとに作成されたもので，通常の形態素解析でも非常に高い精度が出ている．これも有意差が確認出来なかった要因なのではないかと考えている．今後は，今回使用したのとは性質が大きく異るコーパスや，アノテーション規準が厳格に統一されたコーパスを用いて，提案手法の有用性を実証していきたいと考えている．

表 4　同時解析の結果．数値は全て F 値で形態素の列では形態素の情報が全て合っているかどうかで正解を判定している

	分割	品詞大分類	形態素	係り受け
MeCab	99.649 %	99.058 %	98.413 %	89.112 %
リランキング	99.649 %	99.083 %	98.443 %	89.181 %
事後訂正	99.649 %	99.132 %	98.488 %	89.248 %
同時解析	99.655 %	99.082 %	98.441 %	89.094 %

参考文献

[1] 中村純平，伝　康晴：形態素解析誤りの多い助詞・助動詞の再解析，言語処理学会第 14 回年次大会 発表論文集，pp. 73–76 (2008).
[2] Asahara, M. and Matsumoto, Y.: Extended Models and Tools for High-performance Part-of-speech Tagger, *Proceedings of the 18th conference on Computational linguistics-Volume 1*, Association for Computational Linguistics, pp. 21–27 (2000).
[3] 中田陽介，Neubig, G.，　森信介，河原達也：点予測による形態素解析，第 198 回自然言語処理研究会 (2010).
[4] 岸本貴之，高橋治久，堀田一弘：CRF による係り受け解析の結果を反映させた日本語形態素解析，第 189 回自然言語処理研究会 (2009).
[5] Hatori, J., Matsuzaki, T., Miyao, Y. and Tsujii, J.: Incremental Joint Approach to Word Segmentation, POS Tagging, and Dependency Parsing in Chinese, *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, Association for Computational Linguistics, pp. 1045–1053 (2012).
[6] Qian, X. and Liu, Y.: Joint Chinese Word Segmentation, POS Tagging and Parsing, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Association for Computational Linguistics, pp. 501–511 (2012).
[7] Bohnet, B. and Nivre, J.: A Transition-Based System for Joint Part-of-Speech Tagging and Labeled Non-Projective Dependency Parsing, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Association for Computational Linguistics, pp. 1455–1465 (2012).
[8] Bohnet, B., Nivre, J., Boguslavsky, I., Farkas, R., Gin-

ter, F. and Hajic, J.: Joint Morphological and Syntactic Analysis for Richly Inflected Languages, *Transactions of the Association for Computational Linguistics*, Vol. 1, pp. 415–428 (2013).

[9] Chappelier, J.-C., Rajman, M., Aragüés, R., Rozenknop, A. et al.: Lattice Parsing for Speech Recognition, *Proceedings of 6ème conférence sur le Traitement Automatique du Langage Naturel (TALN 99)*, pp. 95–104 (1999).

[10] Kudo, T., Yamamoto, K. and Matsumoto, Y.: Applying Conditional Random Fields to Japanese Morphological Analysis, *Proceedings of Conference on Empirical Methods in Natural Language Processing*, Vol. 4, pp. 230–237 (2004).

[11] Collins, M. and Roark, B.: Incremental Parsing with the Perceptron Algorithm, *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics, p. 111 (2004).

[12] Li, Z., Zhang, M., Che, W., Liu, T., Chen, W. and Li, H.: Joint Models for Chinese POS Tagging and Dependency Parsing, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, pp. 1180–1191 (2011).

[13] McDonald, R., Hall, K. and Mann, G.: Distributed Training Strategies for the Structured Perceptron, *Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics, pp. 456–464 (2010).