Vol.2017-NL-231 No.11
Vol.2017-SLP-116 No.11
2017/5/15

係り受け構造とスーパータグの同時予測による A* CCG 解析

吉川 将司[†1,a)]　能地 宏[†1,b)]　松本 裕治[†1,c)]

概要：われわれは吉川ら [26] において、CCG 木の確率を終端のカテゴリ列とその係り受け構造の確率の積とする確率モデルを提案した．これにより、木の導出を明示的にモデル化でき、また木の確率は事前に計算可能な要素の積に分解できることから高速な A*探索による構文解析ができる．この手法では、CCG 木の係り受け構造をルールで取り出すが、吉川ら [26] で用いている変換ルールは学習データ作成が困難など問題点が多い．本研究では、より単純な変換ルールを採用することでこの問題に対処する．また、半教師あり学習の Tri-training により、ラベルなしデータを利用してさらに精度向上が可能か実験を行った．単純な変換により英語、日本語 CCG 解析で精度の改善が見られ、日本語では最高精度を得た．英語の解析では Tri-training を加えることでさらに向上し最高精度を示した．

1. はじめに

語彙化文法に基づいた構文解析では、スーパータグを単語に付与することは *almost parsing* [2] と呼ばれるように、それがもつ豊かな統語的な情報から、文構造の大部分を決定することができる．組み合わせ範疇文法 (Combinatory Categorial Grammar; CCG [19], [25]) における構文解析で、近年この特徴を巧みに用いた手法が提案された [14], [15]．その手法では、長さ N の文 $\boldsymbol{x}$ に対する CCG 木 $\boldsymbol{y}$ の確率は各スーパータグ (カテゴリ) c_i の確率の積で表される (locally factored モデル):

$$P(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i\in[1,N]} P_{tag}(c_i|\boldsymbol{x}). \quad (1)$$

この確率モデルは、木の導出を含めない単純な定式化になっており、最適な木を効率的な A*探索 (2 節参照) によって求めることが可能である．

スーパータグを付与することで文構造の大部分は決定するが、これによっては解消されない曖昧性も存在し、図 1 のような例では、同じカテゴリ列から複数の木が導出できてしまう．このような状況に対する Lewis ら [14], [15] によるアプローチは、決定的なルールで対処するというものであり、例えば Lewis ら [14] では、“attach low” という経験則によるルールを適用し、図の例では (b) を解析結果として採用する．英語の解析においてはこのルールが経験的にうまく行くことがわかっているが、この単純なルールは明らかに常に正しい結果を導くとは限らない．例えば、“*a house in Paris with a garden*” という句を考えれば、これに対する正しい構造は (a) のはずである．

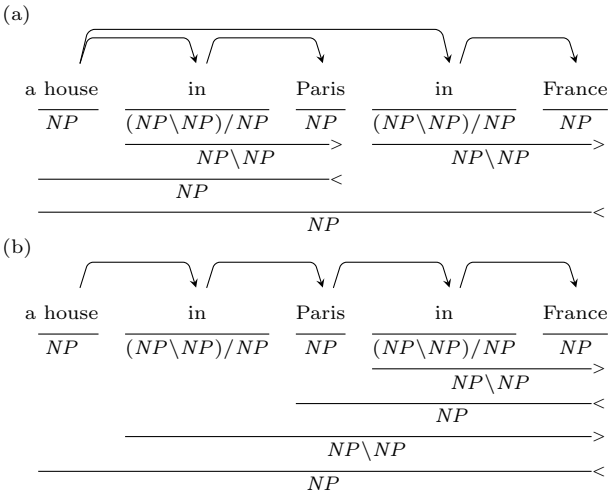


図 1: モデル (式 (1)) において等確率になってしまう CCG 木の例．提案手法においては、係り受け関係を明示的にモデル化することでこのような曖昧性に対処する．

本研究では、図 1 で示すような単語間の係り受け関係を明示的にモデル化することで、locally factored モデルで生じるこのような曖昧性に対処する．提案モデルは、文構造を考慮できる一方で、探索前に計算可能な要素の積に分解できるため、従来手法同様に高速な A*探索を行うことが

†1 現在，奈良先端科学技術大学院大学
Presently with Nara Institute of Science and Technology
a) yoshikawa.masashi.yh8@is.naist.jp
b) noji@is.naist.jp
c) matsu@is.naist.jp

© 2017 Information Processing Society of Japan

できる．ここで鍵となるのは、係り受け解析は各単語に対する係り受けの親を当てる問題であるから、式 (1) と同様に、独立に予測する問題と考えることができる．

スーパータグを超えた文構造をモデル化することの有効性は、式 (1) に再帰型ニューラルネットに基づくスコアを加えた Lee ら [12] によって示されている．しかしながら、彼らの手法では新たなノードを探索するたびに再帰型ニューラルネットでスコアを計算するため、従来手法の効率の良さを失ってしまう．提案法では代わりに係り受けを用いて文構造をモデル化する．

本論文の主要な貢献は、吉川ら [26] で提案した CCG 解析手法を様々に拡張し実験を加えたことである．吉川ら [26] で用いている CCG から係り受け構造への変換ルールは、学習データ作成が困難であるなど問題点が多い．本研究では、より単純な変換ルールを採用することによりこの問題に対処する (4 節)．また、Lewis ら [14] の Tri-training を拡張し、ラベルなしデータからスーパータグと係り受けの両方について有益な教師信号を抽出可能であるか実験を行った (5 節)．英語 CCGBank での実験で、より単純な変換ルールと Tri-training を用いることでラベルあり、なし F1 においてそれぞれ 88.8%、94.0%でこのデータセットにおける最高精度を示した．

英語に加え、日本語 CCG 解析の実験も行った．日本語では、連体修飾節や副詞節がもつカテゴリ (S/S) の右に複数の動詞がある場合に、S/S がどれを修飾するかかなり曖昧になってしまう問題があり、既存手法の単純なルールでは解析精度が悪くなってしまうことがわかった (6 節)．提案手法は、Lewis ら [14] の手法を日本語に適用した場合に対し、文節係り受け正答率において 10.0%の大差をつけて優れた精度を示した．

2. 関連研究

提案法 (3 節) では、A*探索による CCG 解析 (2.1 節) を、双方向 LSTM (bidirectional LSTM) を用いた係り受け解析手法 (2.2 節) を用いて拡張する．以下、それぞれの既存研究について紹介する．

2.1 Supertag factored モデルによる A* CCG 解析

CCG では、カテゴリが文構造を決定する多くの情報をもつため、文の各単語にそれらを付与すること (*supertagging*) が文の統語構造の大部分を決定する．Lewis ら [15] は、この特徴をうまく利用した解析手法を提案した．CCG 木 $\boldsymbol{y}$ をカテゴリ列 $\langle c_1, \ldots, c_N \rangle$ と、その導出の組とする．文 $\boldsymbol{x}$ に対し、可能な CCG 木の集合 $Y(\boldsymbol{x})$ からモデル (式 1) のもとで最も確率の高い木 $\boldsymbol{y}$ を求める:

$$\hat{\boldsymbol{y}} = \arg\max_{\boldsymbol{y} \in Y(\boldsymbol{x})} \sum_{i \in [1,N]} \log P_{tag}(c_i | \boldsymbol{x}).$$

このモデルでは、CCG 木の確率は各単語のカテゴリのみに依存し、導出についてはカテゴリ列が決まればほとんど一意であると考える．木の確率は各語にカテゴリを付与する確率の積であるから、このモデルを *supertag factored* モデルと呼ぶ．

このモデルにおける厳密解は、チャートを用いた手法である A*構文解析 [10] によって効率よく求めることができる．A*解析では、以下のチャートについての 2 つのスコアを用いる:

$$b(C_{i,j}) = \sum_{c_k \in \boldsymbol{c}_{i,j}} \log P_{tag}(c_k | \boldsymbol{x}),$$

$$a(C_{i,j}) = \sum_{k \in [1,N] \setminus [i,j]} \max_{c_k} \log P_{tag}(c_k | \boldsymbol{x}),$$

ここで、$C_{i,j}$ はエッジと呼ばれるチャートに挿入するデータ構造であり、C をルートにもつ区間 $[i, j]$ の部分木を表す．また、チャートはエッジに対し最もスコアの高い部分木 ($C_{i,j}$ に対するビタビ木) を対応させるデータ構造であり、$\boldsymbol{c}_{i,j}$ はそのようなビタビ木におけるカテゴリ列である．よって、b はビタビ内側確率を表し、また a はビタビ外側確率の近似 (上限) である．

A*構文解析は CKY 法の拡張であり、優先度付きキューを用いて探索するエッジの順序を決定する．各時刻において、優先度 $b(e) + a(e)$ が最も高いエッジ e をキューから取り出しチャートに挿入する．そして今挿入されたエッジとチャート内の他のエッジから新たなエッジを構築可能であればそれらをキューに挿入する．このアルゴリズムはエッジ $C_{1,N}$ をポップしたときに終了し、対応する木が最適な解となる．

b と a は、各単語に対する unigram な CCG カテゴリの分布から求まり、探索前に計算できることから、このモデルにおける A*探索は効率良く行うことができる．ヒューリスティック値 a は、真のビタビ外側確率の上限を与えることから許容的 (admissible) である．また、内側確率 b は探索において非増加であることから (monotonicity)、$C_{1,N}$ がポップされたとき、対応する木が最適であることが保証される．$C_{i,j}$ について、区間の外側について one-best なカテゴリ列 ($\arg\max_{c_k} \log P_{tag}(c_k | \boldsymbol{x})$) が $C_{i,j}$ と木をなすならば $a(C_{i,j})$ は真の外側確率と一致する．

2.1.1 スコア付けモデル

P_{tag} を求めるために Lewis ら [15] では固定幅ウィンドウから求まる素性を用いた対数線形モデルを用いているが、後の Lewis ら [14] では双方向 LSTM で拡張したモデルを提案した．双方向 LSTM を用いることで文全体を考慮し、長距離依存な現象を考慮した素性を計算できる．本研究においても双方向 LSTM を採用し、カテゴリを予測すると同時に各単語の係り受けをモデル化する．

2.1.2 導出の曖昧性について

A*探索では、優先度 $b + a$ が最も高いエッジから探索を

© 2017 Information Processing Society of Japan

行うが、図 1 の例のように、同じカテゴリ列 (よって同じ優先度) から複数の木が導出される場合がある. Lewis ら [15] では決定的なルールで対処し、エッジ (CCG 部分木) を変換ルールにより係り受け木に変換し (4 節参照)、長距離依存を伴うものを優先する. 後の Lewis ら [14] では、新たなルール ("attach low" ルール) を提案しているが、これらのルールによる対処法は明らかに常に正しい木を選ぶわけではない. 我々は単語間の係り受け関係を明示的にモデル化することでこの問題を解決する.

2.2 Bi-LSTM 係り受け解析

単語間の係り受け関係をモデル化するために、われわれは、近年のグラフによる係り受け解析手法を利用する. この手法では、双方向 LSTM による素性ベクトルを用いてそれぞれの係り受けエッジのスコアを直接的に計算する. このモデルは一次の関係しか考慮しないが、双方向 LSTM による文全体を考慮した素性を使うことで、PTB での係り受け解析で最高精度を示すなどかなり強力である. この手法は、文の各単語に対する係り受けの親についてのユニグラムな確率分布を求めるという点で、上述の CCG カテゴリの予測モデルと類似している.

以下で示すように、この手法の特徴を利用することによって、我々のモデルは、局所的な要素の積に分解可能となり効率的な A*探索を行うことができる. スコア計算には、Dozat ら [6] により提案された双線形変換の拡張である *biaffine* 層を用いる.

3. 提案手法

3.1 係り受け構造とスーパータグの同時予測による A* CCG 解析

文 $\boldsymbol{x} = \langle x_i, \ldots, x_N \rangle$ に対する CCG 木 $\boldsymbol{y}$ を、CCG カテゴリ列 $\boldsymbol{c} = \langle c_1, \ldots, c_N \rangle$、係り受け構造 $\boldsymbol{h} = \langle h_1, \ldots, h_N \rangle$ と導出の 3 つ組として定義する. ここでそれぞれ h_i は語 x_i の係り受け構造における親のインデックスである. 提案する CCG 木の確率モデルは以下である:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \prod_{i\in[1,N]} P_{tag}(c_i|\boldsymbol{x}) \prod_{i\in[1,N]} P_{dep}(h_i|\boldsymbol{x}). \quad (2)$$

新たに加えられた項 P_{dep} は、語 x_i の親についての unigram な分布である.

提案モデルのもとでも A*探索によって効率よく厳密解を求めることができる. このモデルでの探索問題は以下のように定式化される:

$$\hat{\boldsymbol{y}} = \arg\max_{\boldsymbol{y}\in Y(\boldsymbol{x})} \left(\sum_{i\in[1,N]} \log P_{tag}(c_i|\boldsymbol{x}) + \sum_{i\in[1,N]} \log P_{dep}(h_i|\boldsymbol{x}) \right).$$

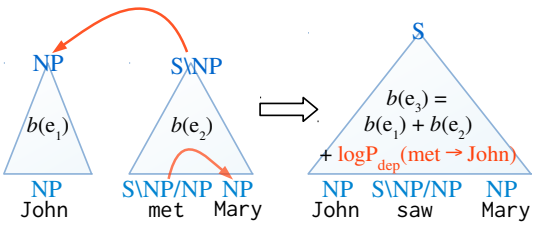


図 2: 提案モデルにおける e_3 のビタビ内側スコアは、e_1、e_2 の内側スコアと、e_2 の親から e_1 の親へ向かう係り受けエッジのスコアの足し算である (係り受けエッジの向きは子のカテゴリによって決まる).

また、内側確率によるスコアは、

$$b(C_{i,j}) = \sum_{c_k\in \boldsymbol{c}_{i,j}} \log P_{tag}(c_k|\boldsymbol{x}) + \sum_{k\in[i,j]\setminus\{root(\boldsymbol{h}^C_{i,j})\}} \log P_{dep}(h_k|\boldsymbol{x}), \quad (3)$$

となる. ここで、$\boldsymbol{h}^C_{i,j}$ は、ビタビ部分木 $C_{i,j}$ の係り受け構造であり、$root(\boldsymbol{h})$ は係り受け構造のルートの語のインデックスを返す関数とする. 部分木のルートの語については、その親は区間 $[i,j]$ 内で解決されていないため、それに関するスコアは含めない. これにより、$b(C_{1,N})$ と真のモデルのスコア (式 (2) の対数) の間で乖離が生じてしまうが、$C_{1,N}$ がポップされたときに、特別な単項規則 ($P_{dep}(ROOT|x_{root(\boldsymbol{h}^C_{1,N})})$ を足す) を適用することで調節を行う.

式 (3) の係り受けに関するスコアは、チャートの展開時に簡単に求めることができる. 今、エッジ $A_{i,k}$ がポップされ、チャート内のエッジ $B_{k,j}$ と結合し $C_{i,j}$ を導出するとする. 注目すべき点として、このとき $root(\boldsymbol{h}^A_{i,k})$ と $root(\boldsymbol{h}^B_{k,j})$ の間の係り受けエッジが唯 1 つ、組 (A,B) と組み合わせ規則の関数として決定するから (節 4 参照)、例えば、その結果 $B_{k,j}$ が親になるとすれば、$l = root(\boldsymbol{h}^A_{i,k})$、$m = root(\boldsymbol{h}^B_{k,j})$ $(l < m)$ として、$b(C_{i,j}) = b(A_{i,k}) + b(B_{k,j}) + \log P_{dep}(h_l = m|\boldsymbol{x})$ と計算できる.

提案モデルにおけるビタビ外側スコアの上限は、

$$a(C_{i,j}) = \sum_{k\in[1,N]\setminus[i,j]} \max_{c_k} \log P_{tag}(c_k|\boldsymbol{x}) + \sum_{k\in L} \max_{h_k} \log P_{dep}(h_k|\boldsymbol{x}).$$

ここで、$L = [1,N] \setminus [k'|k' \in [i,j], root(\boldsymbol{h}^C_{i,j}) \neq k']$. $root(\boldsymbol{h}^C_{i,j})$ についてはその親がまだ見つかっていないため、「外側」に属すと考える. すべての外側の語について、それぞれ独立に P_{dep} の max をとった値を外側スコアの上限にする. これは必ずしも適格な木をなさない. この手法では、解析開始時に、すべてのスーパータグ C とすべての語 x_i について以下のスコアで優先度付きキューを初期化する: $a(C_{i,i}) = \sum_{k\in I\setminus\{i\}} \max \log P_{tag}(c_k|\boldsymbol{x}) +$

© 2017 Information Processing Society of Japan

$\sum_{k \in I} \max \log P_{dep}(h_k|\boldsymbol{x})$. ここで留意すべきは、係り受けに関する項はすべての語に共通である.

3.2 ニューラルネットワークの構造

CCG スーパータグ付け [14] と、係り受け解析 [6] の既存研究に倣い、長距離を伴う言語現象を捉えるため双方向 LSTM を用いて P_{tag} と P_{dep} を計算する. 提案のネットワーク構造は、P_{tag} と P_{dep} の予測のために共通の双方向 LSTM を用いる (図 3 参照).

まず、双方向 LSTM で入力の各単語 x_i の素性ベクトル $\boldsymbol{r}_i$ を求める. LSTM への入力は単語の埋め込みベクトルであるが、それについては 6 節で詳述する. Lewis ら [14] に倣い、各文の先頭と最後を表す単語トークン (とその埋め込みベクトル) を加える. そして、LSTM による素性ベクトルに *biaffine* 変換 [6] を適用して P_{dep} を求める:

$$\begin{aligned} \boldsymbol{g}_i^{dep} &= MLP_{child}^{dep}(\boldsymbol{r}_i), \\ \boldsymbol{g}_{h_i}^{dep} &= MLP_{head}^{dep}(\boldsymbol{r}_{h_i}), \\ P_{dep}(h_i|\boldsymbol{x}) & \\ &\propto \exp((\boldsymbol{g}_i^{dep})^{\mathrm{T}} W_{dep} \boldsymbol{g}_{h_i}^{dep} + \boldsymbol{w}_{dep} \boldsymbol{g}_{h_i}^{dep}), \end{aligned} \tag{4}$$

ここで MLP は多層パーセプトロンである. LSTM による CCG スーパータグ付けの既存研究 [14] では、素性ベクトルに対して単純な多層パーセプトロンを適用することで P_{tag} を求めていたが、本研究ではそれに加え最も確率の高い係り受けの親 ($h_i = \arg\max_{h'_i} P_{dep}(h'_i|\boldsymbol{x})$) を利用し、$\boldsymbol{r}_i$ と $\boldsymbol{r}_{h_i}$ に双線形変換を適用する [*1]:

$$\begin{aligned} \boldsymbol{g}_i^{tag} &= MLP_{child}^{tag}(\boldsymbol{r}_i), \\ \boldsymbol{g}_{h_i}^{tag} &= MLP_{head}^{tag}(\boldsymbol{r}_{h_i}), \\ \boldsymbol{\ell} &= (\boldsymbol{g}_i^{tag})^{\mathrm{T}} \boldsymbol{U}_{tag} \boldsymbol{g}_{h_i}^{tag} + W_{tag} \begin{bmatrix} \boldsymbol{g}_i^{tag} \\ \boldsymbol{g}_{h_i}^{tag} \end{bmatrix} + \boldsymbol{b}_{tag}, \\ P_{tag}(c_i|\boldsymbol{x}) &\propto \exp(\ell_c), \end{aligned} \tag{5}$$

ここで、$\boldsymbol{U}_{tag}$ は三階のテンソルである. Lewis らの手法と同様にこれらの値は探索を行う前に求まるため, 提案手法でも A*探索は効率良く行うことができる.

4. 係り受け構造への変換

我々の手法では、CCG 木から係り受け木への変換ルールを定義する必要がある. その理由として、1) A*探索時に、解析途中の CCG 木から係り受けエッジを得るためであり、また 2) 係り受け予測モデル P_{dep} の学習データを作成するためである.

以下では、Lewis ら [15] による既存手法で定義された変換ルール (LewisRule) に加え、さらに 2 つの代替となる変換ルールを提案する. また、6 章においてこれらの変換ルールの影響を検証する. 図 4 に変換ルールの具体的な例を示す.

[*1] これは Dozat ら [6] の研究で係り受けラベルの予測の定式化をもとにしたものである.

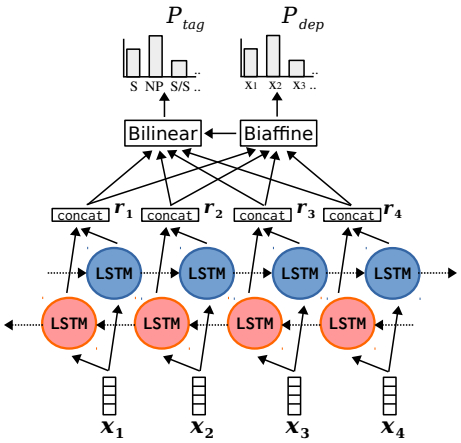


図 3: 提案モデルにおけるニューラルネットワークの構造. 各単語 x_i をまず、双方向 LSTM で素性ベクトル $\boldsymbol{r}_i$ に変換し、*biaffine* 変換 (式 4) と双線形変換 (式 5) により係り受け (P_{dep}) と CCG カテゴリ (P_{tag}) それぞれの分布を得る.

4.1 LewisRule

この変換ルールでは、順方向 (一般化) 関数適用、関数合成において、組み合わせ規則の左側の項を係り受け関係の親と定義する. 例外として、左の項が X/X もしくは $X/(X\backslash Y)$ にマッチする場合は、右の項を親とする. 例えば、図 4a の *"Black Monday"* においては *Monday* が *Black* の親となる.

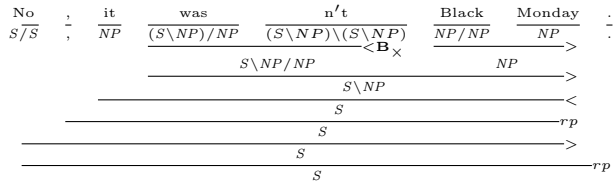
同様に、逆方向の規則については、右の項が $X\backslash X$ もしくは $X\backslash(X/Y)$ である場合を除いて、右の項を係り受けの親とする. その他の規則について、*RemovePunctuation* (*rp*) では、句読点でない項を親とし、また *Conjunction* (Φ) では常に右の項を親とする [*2].

4.2 HeadFinal

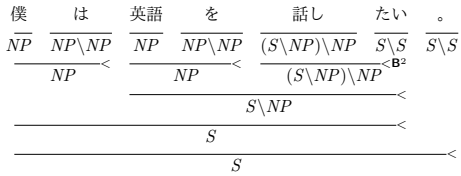
日本語は、主要部終端型 (head-final) の言語であるという特徴があり、京大コーパス [27] などの主要なコーパスにおいても係り受けエッジはすべて右から左にかかっている. この特徴は日本語係り受け解析で利用されているが [11], [21]、本研究でもこれらに倣い、より単純な HeadFinal ルールを提案する. この変換ルールにおいては、任意の句について常に右の子が親となるように定義する. 例えば、図 4b の CCG 木に対して、主要部終端型の係り受け木 (図 4e) を得る.

[*2] 日本語に対して LewisRule を適用する場合、CCG カテゴリから係り受け関係を決定するとき、カテゴリの素性値は除いて行う. 「食べ た」の助動詞「た」の CCG カテゴリは一般に $S_{f_1}\backslash S_{f_2}$ で $f_1 \neq f_2, S_{f_1} \neq S_{f_2}$ であるため、修飾的 $X\backslash X$ ではないが、素性値を取り除いて $S\backslash S$ で考えるため修飾的とみなされる.

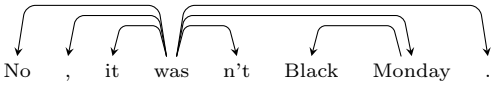
© 2017 Information Processing Society of Japan



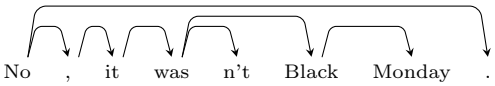
(a) 英語の CCG 木の例



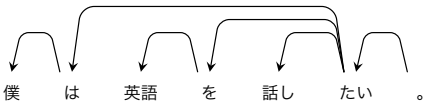
(b) 日本語の CCG 木の例



(c) LewisRule ルール



(d) HeadFirst ルール



(e) HeadFinal ルール

図 4: 4 節の変換ルールを英語と日本語の CCG 木に適用した例.

4.3 HeadFirst

英語の場合も、日本語の HeadFinal ルールと同様に単純な変換ルールの適用を試みる. 英語は日本語と異なり、主要部先導型 (head-initial) の言語であると考えられるため、日本語の場合と逆に、句の左の項を常に区全体の親であると定義する (図 4d).

これにより得られる係り受け構造は直感的でなく奇妙であるが、いくらかの点において、LewisRule に対して有利である. まず、上述のように LewisRule では、CCGBank に付与された係り受け構造に基づいてモデルが学習されるため、予測した P_{dep} と P_{tag} の間で衝突 (不一致) が発生する場合がある. 例えば、ときに真のビタビ木が係り受けに関して低いスコアを付与されてしまうことがあり、解析が遅くなってしまうか、または精度の悪化にもつながってしまう. 一方で単純な HeadFinal の係り受けではこのような衝突は発生しない. また、係り受けエッジの向きが常に一方向であるから、係り受けの親の予測がより簡単になるという利点がある. このことは、既存の係り受け解析器を用いた場合でも正しいことが経験的にわかっており (5 節参照)、また CCG 解析の実験でもこの単純な変換ルールはより高い精度を導くことが示された (6 節参照).

5. Tri-training

Tri-training とは、半教師あり学習の一種であり、ラベルなしデータを 2 つの構文解析器で解析し、それらの出力で一致する部分のみを取り出すことで別の構文解析器のための大量かつ高品質な教師データを作成する手法である.

この手法は、係り受け解析 [24] と CCG スーパータグ付け [14] の両方において有効であることが示されており、我々の手法においてもこれらの既存手法を組み合わせることで、ラベルなしデータから有益な教師信号を抽出できることが期待できる.

CCG スーパータグ付け [14] の実験に用いられたデータは著者らによって公開されているため [*3]、我々はこのデータにさらに係り受け構造を付与することで提案法に適したデータを作成する.

CCGBank のセクション 02-21 から 4 章の変換ルールに従って抽出した係り受け構造を学習データとし、Tri-training のための係り受け解析器の学習を行う. 2 つの解析器が解析誤りについて異なった傾向を示すほど、出力結果が一致している部分についてはより信頼できると考えられるため、解析器には shift-reduce 法による `lstm-parser` [7] と、グラフに基づく手法である `RBGParser` [13] を採用した.

開発データ (セクション 00) において、LewisRule に基づく係り受け構造では、`lstm-parser` はラベルなし係り受け正答率において、92.5%、`RBGParser` は 93.8%であり、一方で HeadFirst ルールでは、それぞれ 94.6%、94.9%とより高い正答率を示した. それぞれの係り受け構造において、約 100 万 7 千文において 2 つ解析器の予測が一致し、これらを学習データとして用いる.

この Tri-training データを用いて提案モデルを学習する際には、Lewis ら [14] に倣い、CCGBank の学習データ (Section02-21) を 15 部に複製し、このデータに加える. また、Tri-training データのサンプルから得られる損失の値に 0.4 を掛けることで、このデータからの影響を CCGBank のものから比較的小さく設定する.

6. 実験

英語、日本語の CCGBank で実験を行った.

6.1 英語 CCG 解析の実験設定

英語 CCGBank の実験では、標準のデータ分割に従い、セクション 02-21 を学習、セクション 00 を開発、セクショ

[*3] https://github.com/uwnlp/taggerflow

© 2017 Information Processing Society of Japan

手法	ラベルあり	ラベルなし
CCGbank		
LewisRule (係り受けなし)	85.8	91.7
LewisRule	86.0	92.5
HeadFirst (係り受けなし)	85.6	91.6
HeadFirst	**86.6**	**92.8**
Tri-training		
LewisRule	86.9	93.0
HeadFirst	**87.6**	**93.3**

表 1: 英語 CCGBank の開発セットにおける結果 (F1).「係り受けなし」は、解析時に係り受けに関するスコアを除いた場合の結果を表す.

手法	ラベルあり	ラベルなし	# NF 違反
CCGbank			
LewisRule (係り受けなし)	85.8	91.7	2732
LewisRule	85.4	92.2	283
HeadFirst (係り受けなし)	85.6	91.6	2773
HeadFirst	**86.8**	**93.0**	**89**
Tri-training			
LewisRule	86.7	92.8	253
HeadFirst	**87.7**	**93.5**	**66**

表 2: 標準形制約なしでの英語 CCGBank の開発セットにおける結果 (F1). # NF 違反は、解析結果において NF 制約を満たさない導出の数である.

ン 23 を評価データとして用いる. 評価は既存研究に従い、C&C パーザに付属する generate プログラムから得られる依存関係によるラベルあり、なしの F1 を示す.

提案モデルについて、Lewis ら [15] と同様に枝刈りを行い、各単語に確率の高い 50 カテゴリのみを許し、可変のビーム幅 $\beta = 0.00001$ を採用する. また、辞書を用いて、頻出語についてはそれに付与可能なカテゴリのみ許容する [*4]. また、明示的に述べない場合を除いて、Lewis らと同じ標準形の制約 [8], [9] を採用する.

単語の埋め込みベクトルとして、GloVe[*5] [18] で初期化した 100 次元の単語ベクトルと、乱数で初期化した接頭辞、接尾辞ベクトル (長さ=1 ∼ 4) を連結したものを用いる. すべての接辞について、学習データに 2 回以下しか現れなかったものは “UNK” に変換する.

モデルの他の設定については、双方向 LSTM は順、逆方向それぞれ 4 層 300 次元、MLP については全て 1 層 100 次元で、非線形関数として ELU [4] を用い、最適化は Adam、$\beta_1 = 0.9, \beta_2 = 0.9$、L2 正則化 ($1e^{-6}$)、また Dozat ら [6] 同様、学習率を $2e^{-3}$ から 2,500 イテレーションごとに 0.75 の比率で減衰させる.

6.2 日本語 CCG 解析の実験設定

日本語の実験においても、標準の日本語 CCGBank [22] の学習/開発/評価の分割に従う. ベースラインとして Noji ら [17] による自然言語処理ツール jigg に実装されている shift-reduce 法による CCG パーザと Lewis らの supertag factored モデルを日本語に適用したものを用いる.

[*4] 辞書については、Lewis らによって公開されているモデルに付属するものを利用した.
[*5] http://nlp.stanford.edu/projects/glove/

手法	ラベルあり	ラベルなし
CCGbank		
C&C パーザ [3]	85.5	91.7
LSTM タガーによる拡張 [23]	**88.3**	-
EasySRL [14]	87.2	-
EasySRL_reimpl	86.8	92.3
HeadFirst (NF 制約なし, 提案)	87.7	**93.4**
Tri-training		
EasySRL [14]	88.0	92.9
neuralccg [12]	88.7	93.7
HeadFirst (NF 制約なし, 提案)	**88.8**	**94.0**

表 3: 英語 CCGBank の評価セットにおける結果 (F1).

日本語では、単語の埋め込みベクトルとして日本語 Wikipedia エンティティベクトル [*6] で初期化した 200 次元のベクトルに、乱数で初期化した 50 次元の文字ベクトルから畳み込み演算 [5] により 100 次元に変換したベクトルを連結したものを用いる. 日本語においては、接辞はあまり有益な素性とならないためこの方法を採用する. 学習データに 2 回以下しか現れなかった文字については “UNK” に変換する. 双方向 LSTM、MLP、最適化については英語の実験と同じ設定を用いた.

日本語の実験での問題に評価方法がある. 日本語 CCGBank は英語と異なったフォーマットで記述されており、また現在のところ英語のような (意味を考慮した) 標準的な評価スクリプトも存在しない. このため、以下では解析結果の CCG 木から変換された文節係り受け木を用いて、解析器の性能を評価する. CCG 木に対して、まず CaboCha[*7] を用いて文節単位に区切り、CCG に基いて図 4b のような単語レベルでの係り受けを習得してから文節境界をまたぐ係り受け関係を抽出する. 例えば、図 4e の文は “僕 は | 英語 を | 話し たい” のように文節単位に区切ることができ、これから (僕 は) ← (話し たい) と (英語 を) ← (話し たい) の係り受け関係を抽出する. この処理を正解と予測結果の CCG 木に対して行い、ラベルなし係り受け正答率を計算する. この方法は不完全ではあるが、曖昧な修飾関係の誤った予測など重要な解析誤りを検出でき、解析精度の良い近似となっている.

6.3 英語の実験結果

6.3.1 係り受けの項の影響

まず、提案法で追加された係り受けのスコアの影響を検証する. 表 1 に開発データにおける実験結果を示す. ここで、「係り受けなし」では、(ニューラルネットワークの構造は固定しながら) A*探索時に係り受けに関するス

[*6] http://www.cl.ecei.tohoku.ac.jp/∼m-suzuki/jawiki_vector/
[*7] http://taku910.github.io/cabocha/

© 2017 Information Processing Society of Japan

	EasySRL_reimpl	neuralccg	Ours
タグ付け	24.8	21.7	16.6
*A** 探索	185.2	16.7	114.6
全体	21.9	9.33	14.5

表 4: 解析速度についての実験結果. それぞれの数字は 1 秒あたり何文処理したかを示す.neuralccg と EasySRL の再実装を比較対象とする.

コアは捨て、代わりに “attach low”(1 節参照) ルールで構造の曖昧性に対処したときの結果である (i.e., 2.1 節の supertag-factored モデル). LewisRule と HeadFirst のどちらにおいても、係り受けの項を加えた場合のほうが精度が向上していることが確かめられる.

6.3.2 係り受けの変換ルールの影響

驚くべきことに、より単純な HeadFirst ルールによる係り受け構造を用いた場合のほうが、言語学的に動機づけられた LewisRule ルールによるものよりも常に精度が高いことがわかった. Tri-training による精度の向上はどちらも同様に高く (約 1.0 ポイント)、これにより提案モデルにおいてもラベルなしデータから恩恵を受けられることが確かめられた.

6.3.3 標準形制約の除外について

HeadFirst ルールの利点の 1 つは、係り受けの向きが常に右であることから、予測が簡単であることであった (5 節参照). 別の観点からすれば、係り受けの向きを固定するということは、(HeadFirst での) 係り受け構造に対応する句構造は唯一つになるという特徴がある. CCGBank の木の構造は基本的には標準形 (Normal Form; NF) に従っていると仮定すれば、HeadFirst ルールを学習したモデルは、解析結果に対し自動的に標準形を強制できるのではないか、と考えられる. 標準形の制約なしで解析を行った実験結果を表 2 にまとめる. 実験の結果、上の仮説は正しいことが明らかになった. HeadFirst ルールにおいて標準形制約を違反した回数は LewisRule ルールの場合と比べて遥かに少ない (89 対 283). また、興味深いことに、制約なしの場合のほうが解析精度が僅かに高く (CCGBank において 86.8 対 86.6)、このことから標準形制約は HeadFirst における探索の妨げになる場合があると推測される.

6.3.4 評価データにおける結果

評価データ (セクション 23) での解析結果を表 3 に示す. 表では、開発データで最も性能の良い HeadFirst ルールによる係り受けを用いて、標準形制約なしの解析器を既存手法と比較する. CCGBank のみを用いた実験で、提案法は LSTM タガーを用いた C&C パーザ [23] を除いたすべてのベースラインに対し性能を上回り、EasySRL に対し 0.5、また筆者らによる再実装 (EasySRL_reimpl) に対し 0.9 の差で高いスコアを示した. Tri-training の実験では、提案法はラベルありで 88.8%、ラベルなしで 94.0%というさらに高い性能を示し、現在の最高精度の neuralccg [12] をそれぞれ 0.1、0.3 ポイント差で上回った. この数値は英語の CCG 解析で報告されている最も高い精度である.

手法	カテゴリ	文節係り受け
shift-reduce [17]	93.0	87.5
Supertag factored モデル	93.7	81.5
LewisRule (提案)	93.8	90.8
HeadFinal (提案)	**94.1**	**91.5**

表 5: 日本語 CCGBank の評価データにおける結果.

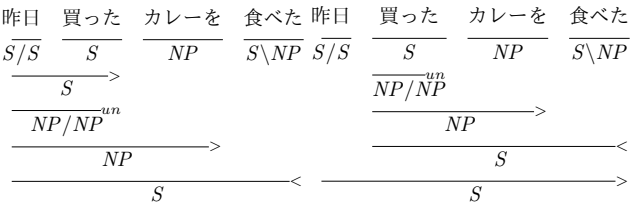


図 5: スーパータグが決まっても構造が曖昧な日本語の木の例.

6.3.5 解析速度

解析速度について、提案法と既存手法の neuralccg と EasySRL_reimpl を比較した [*8]. 実験結果を表 4 に示す. 全体的な解析速度においては、提案法は neuralccg より早く、EasySRL_reimpl よりも遅いという結果になった. 詳細を見てみると、提案法はスーパータグの付与において neuralccg と EasySRL_reimpl の両方に対しより時間がかかっている一方で, A*探索においては neuralccg と比較して 7 倍高速に文を処理することができている. スーパータグ付与において遅れを取っている原因は、双方向 LSTM の層数 (4 対 2) や、双線形変換を用いているなど、より複雑なネットワークを採用しているからであると考えられる. 留意すべき点としては、多くの実装上の違いがあげられる (提案法: C++による A*解析器と Chainer [20] によるスーパータガー、neuralccg: Java による A*解析器と C++ TensorFlow [1] によるスーパータガー、C++ DyNet [16] による再帰モデル)[*9].

6.4 日本語の実験結果

日本語の CCG 解析の結果を表 5 に示す. Supertag factored モデル [14] を単純に日本語を適用することは有効ではなく、最も低い 81.5%の係り受け正答率となった. 提案手法を用いることで精度の改善が見られ、特に HeadFinal による係り受け関係を用いることで、shift-reduce 法による解析器をカテゴリの予測において 1.1 ポイント、係り受

[*8] 解析速度の比較は、4 スレッド 2.0GHz CPU の PC 上で行った.
[*9] supertag foctored モデルの再実装 (EasySRL_reimpl) は、Lewis ら [14] によって示されている解析速度に対し劣っている. そのため、それを基に実装された提案法の解析器は、解析速度において改善の余地があると考えられる.

© 2017 Information Processing Society of Japan

け正答率で 4.0 ポイント上回った.

supertag factored モデルの日本語へ適用した場合に性能が下がってしまう原因として、日本語では、英語と比べて、スーパータグを付与しても文の構造について曖昧性がより多い、という事実が考えられる (図 5). この問題は、特に連体修飾節や副詞節 (ともにカテゴリ S/S) が関わる構文において特に見られ、この結果は、言語によっては (少なくとも日本語では) 係り受けを明示的にモデル化することが重要であることを示している.

7. おわりに

本研究では、CCG 木の確率を終端のカテゴリ列の確率と係り受け構造の確率の積とする新しい CCG 解析モデルを提案した. 係り受け構造を明示的にモデル化することで、修飾関係の曖昧性などに対して決定的なルールを用いずに柔軟に対処可能であり、また、確率モデルは探索前に計算可能な局所的な要素の積に分解でき、高速な探索を可能とする. 提案法は英語、日本語における実験で最高精度を示した.

参考文献

[1] Abadi, M. et al.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems (2015).
[2] Bangalore, S. and Joshi, A. K.: Supertagging: An Approach to Almost Parsing, *Computational linguistics*, Vol. 25, No. 2, pp. 237–265 (1999).
[3] Clark, S. and R. Curran, J.: Wide-Coverage Efficient Statistical Parsing with CCG and Log-Linear Models, *Computational Linguistics, Volume 33, Number 4, December 2007* (2007).
[4] Clevert, D., Unterthiner, T. and Hochreiter, S.: Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs), *CoRR*, Vol. abs/1511.07289 (2015).
[5] dos Santos, C. N. and Zadrozny, B.: Learning Character-level Representations for Part-of-Speech Tagging, ICML (2014).
[6] Dozat, T. and Manning, C. D.: Deep Biaffine Attention for Neural Dependency Parsing, *CoRR*, Vol. abs/1611.01734 (2016).
[7] Dyer, C., Ballesteros, M., Ling, W., Matthews, A. and Smith, A. N.: Transition-Based Dependency Parsing with Stack Long Short-Term Memory, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Association for Computational Linguistics, pp. 334–343 (2015).
[8] Eisner, J.: Efficient Normal-Form Parsing for Combinatory Categorial Grammar, *34th Annual Meeting of the Association for Computational Linguistics* (1996).
[9] Hockenmaier, J. and Bisk, Y.: Normal-form parsing for Combinatory Categorial Grammars with generalized composition and type-raising, *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, Coling 2010 Organizing Committee, pp. 465–473 (2010).
[10] Klein, D. and D. Manning, C.: A* Parsing: Fast Exact Viterbi Parse Selection, *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics* (2003).
[11] Kudo, T. and Matsumoto, Y.: Japanese Dependency Analysis using Cascaded Chunking, *Proceedings of the 6th Conference on Natural Language Learning, CoNLL 2002, Held in cooperation with COLING 2002, Taipei, Taiwan, 2002* (2002).
[12] Lee, K., Lewis, M. and Zettlemoyer, L.: Global Neural CCG Parsing with Optimality Guarantees, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, pp. 2366–2376 (2016).
[13] Lei, T., Xin, Y., Zhang, Y., Barzilay, R. and Jaakkola, T.: Low-Rank Tensors for Scoring Dependency Structures, *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, pp. 1381–1391 (2014).
[14] Lewis, M., Lee, K. and Zettlemoyer, L.: LSTM CCG Parsing, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, pp. 221–231 (2016).
[15] Lewis, M. and Steedman, M.: A* CCG Parsing with a Supertag-factored Model, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, pp. 990–1000 (online), DOI: 10.3115/v1/D14-1107 (2014).
[16] Neubig, G. et al.: DyNet: The Dynamic Neural Network Toolkit, *arXiv preprint arXiv:1701.03980* (2017).
[17] Noji, H. and Miyao, Y.: Jigg: A Framework for an Easy Natural Language Processing Pipeline, *Proceedings of ACL-2016 System Demonstrations*, Association for Computational Linguistics, pp. 103–108 (online), DOI: 10.18653/v1/P16-4018 (2016).
[18] Pennington, J., Socher, R. and Manning, C. D.: GloVe: Global Vectors for Word Representation, *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543 (2014).
[19] Steedman, M.: *The Syntactic Process*, The MIT Press (2000).
[20] Tokui, S., Oono, K., Hido, S. and Clayton, J.: Chainer: a Next-Generation Open Source Framework for Deep Learning, *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS)* (2015).
[21] Uchimoto, K., Sekine, S. and Isahara, H.: Japanese Dependency Structure Analysis Based on Maximum Entropy Models, *Ninth Conference of the European Chapter of the Association for Computational Linguistics* (1999).
[22] Uematsu, S., Matsuzaki, T., Hanaoka, H., Miyao, Y. and Mima, H.: Integrating Multiple Dependency Corpora for Inducing Wide-coverage Japanese CCG Resources, *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Sofia, Bulgaria, Association for Computational Linguistics, pp. 1042–1051 (2013).
[23] Vaswani, A., Bisk, Y., Sagae, K. and Musa, R.: Supertagging With LSTMs, *Proceedings of the 2016 Con-*

© 2017 Information Processing Society of Japan

ference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, pp. 232–237 (online), DOI: 10.18653/v1/N16-1027 (2016).

[24] Weiss, D., Alberti, C., Collins, M. and Petrov, S.: Structured Training for Neural Network Transition-Based Parsing, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Association for Computational Linguistics, pp. 323–333 (2015).

[25] 戸次大介：「日本語文法の形式理論－活用体系・統語構造・意味合成－」，くろしお出版 日本語研究叢書２４ (2010).

[26] 吉川将司，能地　宏，松本裕治：係り受け構造との同時予測による A* CCG 解析，言語処理学会第 23 回年次大会発表論文集 (2017).

[27] 黒橋禎夫，長尾　眞：京都大学テキストコーパス・プロジェクト，言語処理学会第 3 回年次大会発表論文集 (1997).

© 2017 Information Processing Society of Japan