社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2006－NL－176（3）
2006／11／22

符号化問題として解く日本語係り受け解析

田村 晃裕[†]　　高村 大也[‡]　　奥村 学[‡]

概　要

係り受け解析を符号化・復号化問題として解く手法を提案する．従来は，2 文節間の係りやすさ，つまり係り受け木でいう親子関係になるかを基に係り受けを解析している．この従来の考えに従うと，親子関係の情報を表した符号を用いた符号化・復号化問題を解くことになる．係り受け解析を符号化・復号化問題と捉えると，符号化・復号化問題における，誤りがある程度生じても訂正できるように，符号に冗長な情報を加え，使用する符号間の距離を大きくする手法を係り受け解析に援用できる．そこで，本研究では，親子関係の情報の他に，祖先子孫関係になるかという情報を冗長な情報として符号に加えることで精度の向上をはかった．実際に本手法で係り受け解析をし，高い精度が得られたことを報告する．

Japanese Dependency Analysis as a Coding Problem

Akihiro TAMURA[†]　　Hiroya TAKAMURA[‡]　　Manabu OKUMURA[‡]

Abstract

We propose a novel method for Japanese dependency analysis. In deterministic approaches to this task, dependency trees are constructed by actions of attaching a bunsetsu chunk to one of the nodes in the trees. Therefore the task is reduced to deciding the node for the new bunsetsu chunk to be attached. We propose to encode each decision with a sequence of binary values, that is, a code. This representation of decisions enables the model to incorporate ancestor-descendant relations between nodes in addition to parent-child relations. We also propose to concatenate the code of parent-child relation and the code of ancestor-descendant relation, so that the added redundancy in codes helps errors be corrected. Experimental results show that the proposed method achieves higher accuracy in the task of Japanese dependency analysis.

1　はじめに

日本語係り受け解析は日本語処理の基本技術の一つであり，その解析結果は様々な応用技術に頻繁に使用される．そのため，係り受け解析の精度を向上させることは重要であり，これまで多数の研究がなされてきた．初期の研究では，2 文節間の係りやすさを決定するルールを人手で作成し，係り受けを解析していたが，網羅性や一貫性という点で問題が多い．そのため，近年では，係り受けの情報が付与された大規模コーパスが利用可能になったこともあり，機械学習アルゴリズムを用いた統計的手法が多く提案されている．従来の統計的手法は，文内の 2 文節間の係りやすさを学習し，その学習結果を利用して係り受けを解析するものがほとんどである．1 文内の文節の係り受け関係は，1 文節を 1 ノードに対応させ，文節 A が文節 B に係る時，ノード B をノード A の親とした，係り受け木という木で表される．この係り受け木で従来の手法を解釈すると，学習段階で係り受け木において 2 ノードが親子関係になりうるかを学習し，学習結果を用いて各ノードとの親子関係のなりやすさを判定しながらノードの位置を決定していき，係り受け木を構築していく手法がほとんどである．しかし，木構造において 2 ノードの関係は親子関係以外にもあり，その情報も利用できるのではないかと考えた．そこで，本研究では係り受け木の親子関係だけではなく，2 ノードが祖先子孫関係になりうるかという情報も利用する事で，係り受け解析の精度向上を目指す．つまり，祖先子孫関係の情報を冗長な情報として有効に利用したいと考えた．

通信における符号化・復号化問題では，冗長な情報を有効に利用して，情報が正しく伝わる精度を向上させている．そこで，祖先子孫関係の情報を有効利用するために，符号化・復号化問題に着目した．具体的には，ある情報を符号化し伝達する時，伝達による誤りがある程度生じても訂正でき正しい情報が伝わるように，冗長な情報（符号）を加えている事に着目した．これを係り受け解析の問題に対応させ，あるノードの係り受け木での位置を判定するとき，親子関係の情報だけでなく，祖先子孫関係の情報を冗長な情報として加えることで，ノードの係り先を正しく決定でき，精度が向上するのではないかと考えた．

本研究では，日本語係り受け解析を，親子関係だけで

[†]東京工業大学大学院 総合理工学研究科
Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology
aki@lr.pi.titech.ac.jp
[‡]東京工業大学 精密工学研究所
Precision and Intelligence Laboratory, Tokyo Institute of Technology
{takamura,oku}@pi.titech.ac.jp

なく祖先子孫関係の情報も表した符号を利用した，符号化・復号化問題として解く手法を提案し，その有効性を確かめる．

以降，2 節で従来の日本語係り受け解析手法を説明し，3 節では通信における符号化・復号化問題について説明する．そして，4 節で日本語係り受け解析を符号化・復号化問題として解く手法の説明をし，5 節で京大コーパスを用いた評価実験の結果を示し，考察をする．最後に 6 節で本研究のまとめと今後の課題を述べる．

2 従来の係り受け解析

本節では，近年提案されている統計的手法を紹介する．まず，最もよく用いられている考え方について説明する．あらかじめ文節にまとめられた文節列 $b_1, b_2, ..., b_m$ の集合を B，係り受けパターン列 $Dep(1), Dep(2), \ldots, Dep(m)$ の集合を D とする．ここで，$Dep(i)$ は文節 b_i の係り先文節番号を示すものとする．この時，係り受け解析は，入力として文節列の集合 B が与えられた時，その入力文節に対する条件付き確率 $P(D \mid B)$ を最大にする係り受けパターン列の集合 D を求めることと定義できる．そして，それぞれの係り受け関係は独立であると仮定し，$P(D \mid B)$ を $\prod_{i=1}^{m-1} P(Dep(i) \mid b_i)$ と捉え，これを最大にする D を求めることが係り受け解析とされている．この $P(Dep(i) \mid b_i)$ を推定する手法として，決定木を用いる手法 [4]，最大エントロピー法を用いる手法 [5]，SVM を用いる手法 [6] 等が提案されている．これらの手法は，$P(Dep(i) \mid b_i)$ を利用している事から，2 文節間の係りやすさ（係り受け木で考えると，2 ノード間の親子関係のなりやすさ）を利用し，係り受け解析をしていると考えられる．

工藤ら [7] は，ある文節が自分の直後の文節に係るかどうかを判定し，係る場合にはまとめ上げるといったチャンキングを段階的に適用し，係り受け解析をしている．この手法も，ある 2 ノード同士が係り受け木内で親子関係になるかを判定し，係り受け木を組み上げていると捉えられる．Sassano[8] は，スタックを用いて係り先を決める文節の順番を制御している．しかし，ある文節の係り先を決める際，前述した手法と同様，文節同士が係り受け関係になりうるか（2 ノードが親子関係になりうるか）を判定している．

このように従来手法は，係り先を決める（あるノードの係り受け木での位置を決める）際，各ノードと親子関係になりうるかの情報のみを基に判断しているが，提案手法は，2 ノードが親子関係になりうるかだけではなく，祖先子孫関係になりうるかも考慮する点が従来手法と異なる部分である．また，提案手法は，係り受け解析を符号化・復号化問題として解くが，この考え方を用いている従来手法はないことも強調しておく．

3 通信における符号化・復号化問題

情報を通信路を通じて伝達させる際，どのように符号化すればよいかは符号理論の分野で研究されている [9]．その中で，通信路の雑音などの障害に対して，どのように符号化すれば信頼性を失うことなく情報を正しく伝達できるか，という事は重要な観点である．その基本的な考えを提案手法で利用するので，本節で説明する．

今，A か B の 2 種類の情報を正しく送りたい状況を考える．最も単純には，送信側，受信側で，情報 A を符号 0，情報 B を符号 1 と取り決めておけば情報は伝わる．このように，情報 A や B を符号 0 や 1 に対応させることを符号化と呼ぶ．しかしこの符号化では，仮に通信路で雑音が生じ，0 や 1 が反転してしまった場合，正しく情報が伝わらない．一方，情報 A を符号 000，情報 B を符号 111 と符号化する場合を考えると，通信路で雑音が発生し，受信側の符号のある 1 ビットで誤りが生じても，符号同士の距離が近い方（符号が似ている方）が伝わったと認識することで情報が正しく伝わる．例えば，符号 000（情報 A) を送り，通信路で 1 ビット誤りが生じ受信側で 010 を受信しても，010 は 000 の方に似ているので情報 A であると解釈すれば，情報が正しく伝わる．このように，2 つの情報を 0 か 1 ではなく符号にあえて冗長性をもたせることで，誤りを訂正することが可能になる．

今までの議論だけでは，符号の長さが誤り訂正能力に関係していると考えられるかもしれないが，そうではないことを説明する．ここで，3 ビットの符号への符号化を考え，情報 A を符号 010，情報 B を符号 111 と符号化したとする．すると，受信側の符号のある 1 ビットで誤りが生じた場合，どちらの情報が伝わったかが分からない場合がある．例えば，符号 010(情報 A) を送り，1 ビット目に誤りが生じ，受信側で 110 を受信した場合，符号 010(情報 A) との距離と 111(情報 B) との距離が同じであるので，どちらの情報が伝わったのかが分からない．つまり，A を 000，B を 111 と符号化した時より，誤り訂正能力が低いといえる．このように同じ長さの符号でも誤り訂正能力が異なる．

では，何が誤り訂正能力に関わっているかを考えると，符号理論の分野で次の事が知られている．

送信されうる情報を符号化した際の符号同士の距離が離れているほど，誤り訂正能力がある． ···(1)

距離の尺度としてハミング距離で考えた場合，前者の符号化の例（A:000，B:111）では，2 符号間のハミング距離は 3 である．一方，後者の符号化の例（A:010，B:111）では，2 符号間のハミング距離は 2 であり，ハミング距離の大きい前者の方が誤り訂正能力があるといえる．ここで，ハミング距離とは，2 つの符号で異なるビットを数えたものである．例えば,「010」と「111」のハミング距離は，1 ビット目と 3 ビット目の 2 ビット異なるので 2 である．

本研究では，前述した (1) の性質を利用した手法を提案する．

4 符号化・復号化問題として解く係り受け解析

係り受け解析を符号化・復号化問題として解く手法を説明する．まず，我々も，2 節で説明したような先行研

究同様，係り受け木内の各ノードと親子関係になりうるかの情報を用いるが，親子関係の情報を用いて符号化・復号化問題として係り受け解析を行う手法 PARENT を 4.1 節で説明する．4.2 節では，親子関係ではなく祖先子孫関係になりうるかの情報を用いて，符号化・復号化問題として係り受け解析を行う手法 ANCESTOR を説明する．そして，4.3 節で提案手法である手法 PARENT-ANCESTOR を説明する．これは，3 節の (1) の性質を利用し，親子関係になりうるかだけではなく，祖先子孫関係になりうるかの情報も利用した符号化・復号化問題として係り受け解析を行う手法である．以降，係り受け解析を係り受け木の構築という立場で説明する．したがって，係り受け木における用語を使用するので，「ノード」を「文節」,「ノード A がノード B につく」を「文節 A が文節 B にかかる」と理解してほしい．

4.1 親子関係を表した符号を利用した手法 (手法 PARENT)

4.1.1 解析フェーズ

手法 PARENT で係り受け木を構築するアルゴリズムを以下に疑似コードで記述する．

1:for i=1,2,...,n do
2:　for j=1,2,...,n-1 do
3:　　code_model[j]=MODEL_PARENT($node_i$,$node_j$)
4:　end
5:　Parent[$node_i$]=$argmin_k$Dis($code[k], code\_model$)
6:end

ここで，MODEL_PARENT($node_i$,$node_j$) は $node_i$ と $node_j$ が親子関係になるかを学習モデルにより判定した結果，$code\_model$ は親子関係かどうかを判定した結果の code_model[1] から code_model[n-1] までを並べた符号，$code[k]$ は $node_k$ についた際に出力されるべき符号，Parent[$node_i$] は $node_i$ のつく先，Dis は距離関数を表す．

このアルゴリズムは，日本語の係り受けの性質である“ 係り先は自分の節より後方の節になる ”という事を利用し，文末の節から，今まで構築された木のどのノードにつくかを順々に決め，係り受け木を構築していく．文末の節から文頭の節に向かい，$node_1$，$node_2$，... という風に対応づける．この解析順序は，[5] や [6] と同じである．[5] や [6] と異なる部分は，あるノードの木での位置を決める（どのノードにつくかを決める）手順である．アルゴリズムでいうと step3 や step5 にあたる．[5] や [6] で解析の際のビーム幅が 1 の時の手順は，step3 で学習モデルで判定した結果を確率 $P(node_j \mid node_i)$ として格納する．そして，step5 で $argmax_j P(node_j \mid node_i)$ を考え，$node_i$ のつく先を決めることになる．一方，手法 PARENT では，符号化・復号化問題として，あるノードの位置を決めている．

あるノード (判定ノード)$node_m$ の木での位置を決める手順を説明する（図 1 参照）．手法 PARENT は，あるノードと親子関係になりうるかを基に決める手法であり，MODEL_PARENT を用いて，判定ノード $node_m$ と構築済みの木の各ノード $node_1$，$node_2$，...，$node_{m-1}$ が親子関係になるかを判定する．この m-1 個の判定結果の並びを，符号と捉える．図 1 でいうと,「0001」が判定結果の符号となる．次に，判定ノードが構築済みの木の各ノードについた場合を想定し，出力されるべき符号を考える．手法 PARENT の場合，判定ノードがあるノードについた場合を想定し，構築済みの木の各ノードと判定ノードとの親子関係に基づいた符号を考える．図 1 では，破線の四角で囲まれた部分がその符号の集合である．例えば，$node_1$ につく場合は，$node_5$ は $node_1$ と親子関係になり，その他のノードとは親子関係にならない．したがって「1000」が出力されるべき符号となる．そして，この出力されるべき符号の中から，判定結果の符号との距離が最も近いものを選び，選ばれた符号に対応するノードの下に判定ノードを実際につける，という手順になる．図 1 の場合,「0001」が選ばれ $node_5$ は $node_4$ の下につく．この手順の中で，“ 日本語の係り受けは一般に交差しない ”という非交差条件を考慮するため，アルゴリズムの step5 で親を決める際，非交差条件を満たさないものは親の候補からはずすという制約を用いている．

このあるノードの木での位置を決める手順を通信の問題に対応づけて説明する．送信側が，判定ノードが構築済みの木で正しい位置につくような符号を送っていると仮定する．つまり，図 1 でいうと，$node_5$ の正しい位置が $node_4$ の下である場合,「0001」という符号を送っているとする．その符号が，通信路として学習モデル MODEL_PARENT を通り，受信側に伝わると対応付けられる．そして，受信側が観測された符号（判定結果の符号）から，送る可能性のある符号（出力されるべき符号）を考慮し，送信したであろう正しい符号を推測し，どのノードにつくかという情報を得る．ここで，通信路の雑音による誤りは学習モデルの誤りに相当し，その誤りを訂正する部分は，判定結果の符号から出力されるべき符号を考慮し，送信したであろう符号を推測する部分（アルゴリズムの step5）である．したがって，誤り訂正能力は，出力されるべき符号間同士の距離の離れ具合に依存する．つまり，出力されるべき符号間の距離が離れていればいる程，3 節の (1) より，学習モデルで判定した際に誤りが生じても，訂正されやすいと考えられる．そこで，手法 PARENT で用いた符号の符号間の距離について述べておく．あるノードの親は必ず 1 つであるので，手法 PARENT で用いる符号は，その親の部分だけ 1 になりその他は 0 である．したがって，符号間のハミング距離を考えると，親の部分のビットだけが異なり，どの符号間同士の距離も 2 となる．

4.1.2 学習フェーズ

4.1.1 節で述べたように，手法 PARENT は 2 ノード同士が親子関係になりうるかを判定し，係り受けを解析するので，その判定モデル MODEL_PARENT を学習・作成する必要がある．そこで，学習事例である係り受け木から親子関係を図 2 のように学習する．つまり，あらゆるノードのペアについて，親子関係になっているペアを正例，その他を負例として学習する．

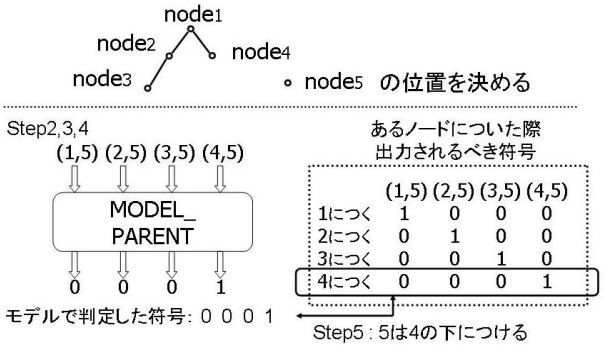

図 1: 手法 PARENT での解析例

図 2: 学習事例の作成例

4.2 祖先子孫関係を表した符号を利用した手法 (手法 ANCESTOR)

4.1 節では，親子関係になりうるかの情報を用いて符号化・復号化し係り受け解析を行ったが，祖先子孫関係になりうるかの情報を用いて符号化・復号化し係り受けを解析する手法 ANCESTOR について説明する．

4.2.1 解析フェーズ

手法 ANCESTOR で係り受け木を構築するアルゴリズムは，4.1 節で示した手法 PARENT によるものとほとんど同じであり，step3だけ変更すればよい．具体的には，MODEL_PARENT を用いるのではなく，MODEL_ANCESTOR を用いて $node_i$ と $node_j$ が祖先子孫関係になりうるかを判定するように変更する．また，アルゴリズム上では表れないが，step5 で $code[k]$ として $node_k$ についた際に出力されるべき符号を考える時に，手法 PARENT の場合は判定ノードと構築済みの各ノードが親子関係になるかを基に符号を考えたが，手法 ANCESTOR の場合は祖先子孫関係になるかを基に出力されるべき符号を考える点も大きな違いであり，注意してほしい．例えば，図 3 において，$node_5$ が $node_2$ についた時に出力されるべき符号は，$node_5$ は $node_1$，$node_2$ とそれぞれ祖先子孫関係になるので,「1100」となる．ちなみに，親子関係による符号ならば「0100」となる．手法 ANCESTOR により，あるノードの係り受け木での位置を決める手順の具体例を図 3 に示すので参照してほしい．

ここで，誤り訂正能力に影響する，手法 ANCESTOR で用いる符号間の距離について述べる．4.1.1 節で述べたように手法 PARENT で用いる符号では，どの 2 つの符号同士のハミング距離も 2 で一定であった．しかし，

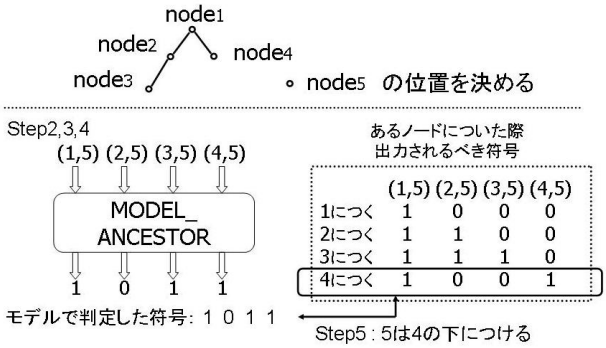

図 3: 手法 ANCESTOR での解析例

手法 ANCESTOR で用いる符号では，符号間のハミング距離は一定ではなく，2 つの符号の組み合わせによりハミング距離も異なる．例えば，図 4 で codeA と codeB のハミング距離は 3 であるが，codeA と codeC のハミング距離は 4 である．そこで，どのような特徴があるのかを，図 4 の状況を考え，判定ノードが nodeA についた際に出力されるべき符号 codeA と nodeB についた際に出力されるべき符号 codeB のハミング距離を例にとり考える．codeA は木のルートから nodeA までのパス上のノードに対応するビットが 1 になり，その他のビットは 0 である．codeB も同様である．ここで，nodeA と nodeB の 2 つの上位ノードで最も下位にあるノードを node0 とすると，ルートから node0 までのパス上のノードに対応するビット (図 4 では 1 ビット目と 2 ビット目) は codeA，codeB 共に 1 である．そして，node0 から nodeA までのパス上のノードに対応するビットは codeA では 1，codeB では 0 であり，node0 から nodeB までのパス上のノードに対応するビットは codeA では 0，codeB では 1 である．また，前述した条件にあわなかったノードに対応するビットは codeA，codeB 共に 0 である．したがって，異なるビット数は nodeA から node0 を通って nodeB までのパスの長さ (パス上の枝の数) と等しい．つまり，codeA と codeB のハミング距離は，nodeA から nodeB までの最短パスの長さと等しいことが分かる．以降,「nodeA から nodeB までの最短パスの長さ」を「nodeA と nodeB の木での距離」と定義する．

3 節の (1) より，ハミング距離が大きい符号同士ほど見分けがつきやすく，誤りが訂正されやすいはずである．つまり，図 4 において codeA が伝わるべきものの際，誤りやすさを考えると，codeC より codeB に誤りやすいはずである．なぜならば，codeA と codeC の距離が 4 で，codeA と codeB の距離 3 より大きく，誤りが訂正されやすいからである．係り受け木で考えると，正解が nodeA につく際，正解 nodeA から離れたノード程，その下に誤ってつくことはないはずである．それは，そのノードにつく符号と nodeA につく符号との距離も大きくなり，誤りが訂正されやすく間違えにくくなると考えられるからである．この理論的にいえる事を，5 節で実験結果を通して現象として確認する．

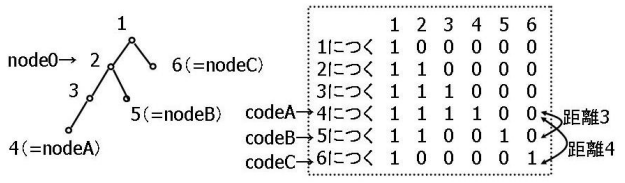

図 4: 手法 ANCESTOR で用いられる符号間の距離

4.2.2 学習フェーズ

手法 ANCESTOR は，2 ノード同士が祖先子孫関係になりうるかを判定し，係り受けを解析するので，その判定モデル MODEL_ANCESTOR を学習・作成する必要がある．そこで，図 2 のように学習事例である係り受け木から祖先子孫関係を学習する．つまり，あらゆるノードのペアについて，祖先子孫関係になっているペアを正例，その他を負例として学習する．

4.3 提案手法 PARENT-ANCESTOR

手法 PARENT では親子関係になりうるかの情報，手法 ANCESTOR では祖先子孫関係になりうるかの情報のみを用いたが，その両者の情報を用いて，符号化・復号化問題として係り受け解析を行う手法 PARENT-ANCESTOR を説明する．この手法は，3 節の (1) の「符号同士の距離が離れているほど，誤り訂正能力がある」事を利用した手法である．つまり，符号化・復号化の際に用いる符号の符号間の距離を増やし，4.1.1 節のアルゴリズムの step3 で学習モデルの判定結果に誤りが生じても，step5 でそのビット誤りを吸収し正しく係り受け木を構築できるようにする．具体的には，手法 PARENT では親子関係のみを表す符号，手法 ANCESTOR では祖先子孫関係のみを表す符号を用いたが，提案手法である PARENT-ANCESTOR では親子関係を表す符号に祖先子孫関係を表す符号を単純結合させ冗長性をもたせた符号を用いる．こうすることで，手法 PARENT や ANCESTOR で用いた符号の符号間距離よりも，手法 PARENT-ANCESTOR で用いる符号の符号間距離を大きくすることができる．

手法 PARENT-ANCESTOR で係り受け木を構築するアルゴリズムは以下の通りである．

1:for i=1,2,...,n do
2: for j=1,2,...,n-1 do
3: model_parent[j]=MODEL_PARENT($node_i$,$node_j$)
model_ancestor[j]=MODEL_ANCESTOR($node_i$,$node_j$)
4: end
5: Parent[$node_i$]=$argmin_k$Dis($code[k], code\_model$)
6:end

ここで，*code_model* は親子関係かどうかを判定した結果である model_parent[1] から model_parent[n-1] を並べ，その後ろに祖先子孫関係かどうかを判定した結果の model_ancestor[1] から model_ancestor[n-1] を並べた符号である．4.1.1 節で示した手法 PARENT のアルゴリズムとの違いは，step3 で MODEL_PARENT で親子関係になりうるかだけではなく MODEL_ANCESTOR により祖先子孫関係にもなりうるかを判定する部分である．またアルゴリズム上には表れないが，step5 で用いる *code_model* や $code[k]$ は，親子関係を表す符号と祖先子孫関係を表す符号を結合させた符号になっている事も大きな違いであり，注意してほしい．具体的に，手法 PARENT-ANCESTOR により，あるノードの係り受け木での位置を決める手順を図 5 に示すので参照してほしい．手法 PARENT-ANCESTOR では，MODEL_PARENT と MODEL_ANCESTOR を用いて各関係になりうるかを判定しているが，この学習モデルは 4.1.2 節や 4.2.2 節で述べたように学習させたものである．

手法 PARENT-ANCESTOR で用いた符号の符号間距離が，手法 PARENT や手法 ANCESTOR で用いた符号の符号間距離よりどの程度大きくなったかを述べる．この大きさが，符号を結合させることにより向上する誤り訂正能力の大きさになる．まず手法 PARENT-ANCESTOR で用いた符号と手法 ANCESTOR で用いた符号を比較する．手法 PARENT-ANCESTOR で用いた符号の方が手法 ANCESTOR で用いた符号より，親子関係をあらわす符号の符号間距離分だけ大きくなる．つまり，手法 PARENT-ANCESTOR で用いた符号の符号間距離の方が，手法 ANCESTOR で用いた符号の符号間距離より，全ての符号間で 2 だけ大きくなる (4.1.1 節参照)．同様に手法 PARENT-ANCESTOR で用いた符号と手法 PARENT で用いた符号を比較すると，手法 PARENT-ANCESTOR で用いた符号の符号間距離の方が，手法 PARENT で用いた符号の符号間距離より，祖先子孫関係を表す符号の符号間距離分だけ大きくなる．これは，2 つの符号に対応する係り受け木のノードを考えた時，その 2 ノード間の木での距離だけ大きくなる（4.2.1 節参照）．例えば，図 5 では，祖先子孫関係の符号を結合させることにより，code[1] と code[3] で $node_1$ と $node_3$ の木での距離 2，code[3] と code[4] で $node_3$ と $node_4$ の木での距離 3 だけ符号間距離が大きくなっている．実際に，2 つの符号を結合させ符号間距離を大きくすることで，誤りが訂正されやすくなり，係り受け木を正しく構築できる精度が向上することを，5 節で実験を通じて示す．また，2 つの符号を結合させることで大きくなる符号間距離に応じて，誤りの訂正されやすさが変わることも 5 節で示す．

4.4 既存の符号の利用可能性について

多クラス分類問題を，誤り訂正符号を用いた符号化問題として解く手法が提案されている [2]．この手法では，1 クラスに n ビットの符号を 1 つ対応させ，学習事例から各ビットを予測する学習モデルを n 個作成する．分類する際は，事例に対してその n 個の学習モデルを適用し，その出力から n ビットの符号を作成する．そして，それに対応するクラスに分類するという手法である．この際，用いられる符号は BCH 符号など，符号理論によって符号間の距離を十分に離し，十分な誤り訂正能力があると証明されているものである．今回

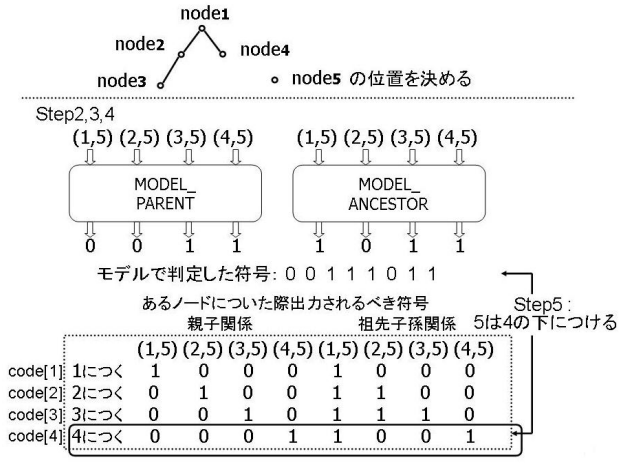

図 5: 手法 PARENT-ANCESTOR での解析例

は，親子関係を表す符号，祖先子孫関係を表す符号を用いたが，BCH 符号などの既存の符号を用いるとどうなるのか，そもそもそのような符号が本手法に適用できるかどうかを議論する．

本手法は，符号を 2 つのノード間の関係により作成しているため，各ビットの値が意味を持ち，その関係を判定する学習モデルのみで符号を生成できる．しかし，BCH 符号などの誤り訂正符号は，符号間距離をできる限り大きくするという観点で作られるため，各ビットの値は意味をもたず，各ビットごとに学習器を作成しなければならない．さらに，構築された木のノードの数により符号の数が変わり，符号も変化する．例えば，図 1 の木における係り受け解析の問題と図 4 の木における係り受け解析の問題で用いる符号は異なる．そのため，構築していくそれぞれの段階での各ビットごとに学習モデルを作成しなければならず，現実的には不可能である．

5 評価実験および考察

5.1 実験設定

実験には，[5]，[6]，[7]，[8] などの多くの先行研究にあわせ，京大コーパス (Version2.0)[3] の一部を使用した．ノード間が親子関係になりうるかを判定する MODEL_PARENT や祖先子孫関係になりうるかを判定する MODEL_ANCESTOR の学習には 1 月 1 日から 8 日分 (7,958 文) を使用し，テストには 1 月 9 日分 (1,246 文) を使用した．また，ノード間の各関係の学習・解析には SVM を用いた．そのため，係り受け解析アルゴリズムの step3 での判定結果として，ある関係かどうかを示す 2 値（0 か 1）ではなく，学習した識別関数の値が出力される．この値は，モデルで判定した際，ある関係かどうかの確信度とも捉えられるため，数字の大小も重要な情報である．そこで，SVM の出力によりある関係かどうかを判定し，code_model[j] をある関係ならば 1，そうでなければ 0 とし，*code_model* として 0，1 の 2 値で符号を表すのではなく，SVM の出力そのものを並べたものを符号として捉えた．従って，アルゴリズムの step5 で用いる符号間の距離の尺度は，ハミング距離ではなく，コサイン距離を使用した．カーネル関数には多項式カーネルを用い，次元数は [7] にあわせ 3 とした．

学習に用いる素性は [7] を参考に，静的素性と動的素性の 2 種類を用いた．静的素性とは，途中の解析結果に関わらず，入力として与えられた情報から一意に得られる素性集合である．一方，動的素性とは，解析途中に得られた係り受け関係そのものをフィードバックし得られる素性集合である．用いた静的素性を表 1 に示す．これは，[7] で用いられているものと同じである．表 1 内の主辞とは，文節内で品詞が特殊，助詞，接尾辞となるものを除き，文末に一番近い形態素，語形とは，文節内で品詞が特殊になるものを除き，文末に一番近い形態素のことを指す．また，見出し語の選択も [7] にあわせ，適当な頻度による閾値を設けず，学習データ中のすべての単語を用いている．

表 1: 使用した素性

判定／比較文節	主辞見出し，主辞品詞，主辞品詞細分類，主辞活用 主辞活用形，語形見出し，語形品詞，語形品詞再分類 語形活用，語形活用形，括弧の有無，句読点の有無 文節の位置（文頭，文末）
文節間	距離 (1,2-5,6 以上), 全ての助詞，括弧，句読点の有無

次に，動的素性について説明する．[7] では動的素性として，以下の 3 つの情報を考慮している．

(1) 着目している係り先に係る文節（A）
(2) 着目している係り先が係る文節（B）
(3) 着目している係り元に係る文節（C）

提案手法では解析手順が [7] と異なるため C の素性は考慮できない．したがって，提案手法では A，B を拡張し，再帰的につなげた素性 A’，B’ を用いている（図 6 参照）．A’，B’ の情報は，[7] で A や B の素性として考慮したものを用いた．具体的には，A’ の素性として，助詞，副詞，連体詞，接続詞については見出し語そのものを，活用形のあるものはその活用形を，その他の品詞については品詞と品詞細分類を与えた．B’ の素性として，主辞の品詞と品詞細分類を与えた．

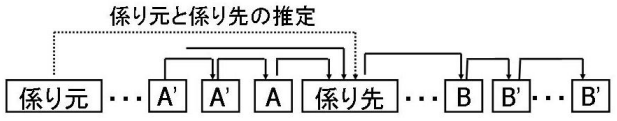

図 6: 動的素性

5.2 実験結果と考察

本節で，提案手法の有効性を確認する．3 つの手法，PARENT，ANCESTOR，PARENT-ANCESTOR を用いて，実験した結果を表 2 に示す．実験結果の評価尺度は，2 文節間の係り受け関係 (木の親子関係) が正解している割合を表す係り受け正解率と，1 文内の係り受け関係全て (構成した木) が正解している割合を表す文正解率を用いた．

表 2: 符号化・復号化問題として解いた係り受け解析結果

手法	係り受け正解率	文正解率
PARENT	88.95% (10,018/11,263)	44.87% (556/1,239)
ANCESTOR	87.64% (9,871/11,263)	43.74% (542/1,239)
PARENT -ANCESTOR	89.54% (10,085/11,263)	47.38% (587/1,239)

表 2 より，提案手法 PARENT-ANCESTOR の係り受け解析結果が他の 2 手法よりも良い事が分かる．この結果が有意な差であるかどうかを符号検定を用いて検定すると，有意水準 1%で有意差が認められた．したがって，2 つの符号を結合し冗長性を持たせ，符号間距離を大きくする事で，結果が改善されることが実験的にいえる．

次に，提案手法と先行研究の実験結果を比較する．[7] と素性をあわせ比較したものを表 3 に示す．[7] の手法は，係り受け解析ツールとして頻繁に用いられている CaboCha[1]の手法である．素性など，手法以外の実験設定があわせやすく，ツールとして一般的に用いられているので，[7] と比較した．また，同じコーパスを用いている先行研究の実験結果との比較を表 4 に示す．表 3，4 より，提案手法は先行研究の手法よりも良い精度を示していることが分かる．ただし，[8] よりは低い精度である．しかし，[8] では，今回使用しなかった素性の他に，より高級な素性を使用している．例えば,「A と B」や「〜し，〜」などの接続構造を捉えるための素性を [1] を参考に使用したり，係り先の文節の最初の形態素は係るかどうかの重要な情報になるとして素性に使用している．したがって，表 4 の結果から一概に本手法が劣っているとはいえない．[8] の手法で今回使用した素性に対応する素性を用いた結果と比較すると，本手法の方が精度がよいことも確認している．

表 3: 工藤ら 02[7] との結果の比較

素性	手法	係り受け正解率	文正解率
静的素性のみ	提案手法	88.88%	46.33%
	工藤ら [7]	88.71%	45.19%
静的素性＋動的素性 AB	提案手法	89.43%	47.94%
	工藤ら [7]	89.19%	46.64%

表 4: 先行研究との実験結果の比較

手法	係り受け正解率	文正解率
PARENT-ANCESTOR	89.54%	47.38%
関根 00[5]	87.20%	40.76%
工藤ら 00[6]	89.09%	46.17%
工藤ら 02[7]	89.29%	47.53%
Sassano 04[8]	89.56%	48.35%

今回，2 ノード間の関係として親子関係，祖先子孫関係を用いたが，これらの関係を表した符号の符号間距離を比較すると，4.1.1 節や 4.2.1 節でも述べたように特徴が異なっている．つまり，親子関係の符号の場合，どの符号同士の距離も 2 で一定であったが，祖先子孫関係の符号の場合，それぞれの符号同士で符号間距離が異なる．そこで，この違いが手法 PARENT，ANCESTOR で係り受けを解析した結果に表れるかを調べる．あるノードを構築済みの木のノードにつけるという問題で，間違えてしまったものを考える．その分布を，本来つくべきノードと間違えてつけてしまったノードとの係り受け木における距離別に，表 5 に示す．例えば，図 1 において，本来，$node_5$ は $node_4$ につくべきであるが，仮に手法 PARENT を適応した際に $node_2$ についてしまった場合，表 5 の（PARENT，木での距離が 2）のセルに含める．この表 5 より，手法 ANCESTOR の方が手法 PARENT に比べて，正解ノードと木での距離が近いノードに間違えやすく，距離が離れるほど間違えにくいといえる．これは，手法 PARENT で用いられる符号では，表 5 の木での距離に関わらず，正解の符号と間違えて選択された符号の符号間距離は 2 である（4.1.1 節参照）が，手法 ANCESTOR で用いられる符号では，表 5 の木での距離と，正解の符号と間違えて選択された符号の符号間距離が同じである（4.2.1 節参照）という特徴があるため，このような違いが生じたと考えられる．つまり，手法 ANCESTOR の場合，正解ノードと木での距離が離れれば離れる程，正解の符号とその符号との符号間距離が大きくなり，その符号に間違えにくくなるといえる．このように間違える場所の特徴が，用いる符号の特徴に影響されて，符号間の距離という観点で説明できる．

手法 PARENT-ANCESTOR では，2 種類の関係を表す符号を結合させることで誤り訂正能力をあげ，精度を向上させた．そこで，結合させる符号によりどのような違いが生じるかを調べる．まず，手法 PARENT や ANCESTOR ではノードのつく場所が間違えて判定されていたのに，符号を結合させる事により正しい位置に訂正されたものについて調べる．正解ノードとの距離別に，1 種類の関係の符号を用いた際は間違えていたものが，結合させることで正解になる訂正率を表 6 にまとめる．例えば，図 1 において，仮に手法 PARENT を適応した際，$node_5$ が $node_3$ に誤ってついていたが，手法 ANCESTOR-PARENT を適応することで本来つくべき $node_4$ につくと正しく判定された場合，表 6 の（祖先子孫関係の符号を結合，木での距離が 3）のセルに含める．表 6 より，祖先子孫関係の符号を結合させることで正解になる割合は，元々間違えていたノードと正解ノードの木での距離が大きければ大きい程高いことが分かる．一方，親子関係の符号を結合させることで正解になるノードの割合と，元々間違えていたノードが正解のノードとどの程度離れていたかということは，相関はないと考えられる．この事は，符号を結合させることにより増加する符号間距離の違いで説明できる．祖先子孫関係を表す符号を結合させると，4.3 節でも述べたように，2 つの符号に対応する係り受け木のノードを考えた時，その 2 ノード間の木での距離だけ大きくなる．つまり，間違えてついていたノードと正解ノードの木での距離が大きい程，結合させること

[1] http://chasen.org/~taku/software/cabocha/

表 5: 誤ったノードの正解ノードとの距離別分布

木での距離	1	2	3	4	5	6	7	8	9 以上
ANCESTOR	64%	24%	8.3%	2.1%	0.78%	0.64%	0.14%	0.07%	0%
PARENT	57%	27%	9.4%	3.9%	1.8%	0.88%	0.24%	0.08%	0%

各セル：ある木での距離で誤った数/誤った全ての数

表 6: 符号を結合させることで正解になる割合

木での距離	1	2	3	4	5	6	7	8	9 以上
親子関係の符号を結合	30%	20%	33%	14%	27%	11%	50%	0%	0%
（ANCESTOR に結合）	268/901	70/343	39/118	4/29	3/11	1/9	1/2	0/1	0
祖先子孫関係の符号を結合	14%	19%	22%	27%	26%	18%	67%	0%	0%
（PARENT に結合）	99/706	65/337	26/117	13/49	6/23	2/11	2/3	0/1	0

各セル：割合%(ある距離で正解になった数/ある距離で間違えていた数)

により正解の符号との符号間距離が大きくなり，誤り訂正しやすくなる．そのため，もともと正解ノードと離れて間違えていたもの程，祖先子孫関係の符号を結合させる事により正解になる割合も大きくなっている．一方，親子関係を表す符号を結合させると，全ての符号同士間で符号間距離が 2 だけ一様に増える．そのため，正解になるノードの割合と，もともと正解ノードとどの程度離れていたかは関係ないと考えられる．

今までは，元々誤っていたものが符号を結合させることで正解となったものについて議論してきたが，結合させることで正しく判定されていたものが間違えてしまうものや，間違いのままのものも当然ある．これらについて，表 7 にまとめる．表 7 より，祖先子孫関係の符号を結合させた時の方が，親子関係の符号を結合させる時より，正解ノードに近づくノードが多い．以上のことから，祖先子孫関係の符号を結合する（手法 PARENT から手法 PARENT-ANCESTOR にする）ことは，コードの特性上，正解のノードに近づける働きをするといえる．

表 7: 符号を結合させることによる変化

正解ノードとの変化	不変	遠ざかる		近づく
		正解から不正解	不正解のまま	不正解のまま
親子関係を結合	901	157	89	38
祖先子孫関係を結合	975	151	13	46

各セル：個数

6 おわりに

係り受け解析を符号化・復号化問題として解く手法を提案した．従来は，係り受け木での親子関係になるかどうかの情報を基に解析をしていたが，祖先子孫関係になるかの情報も利用する．そして，この祖先子孫関係の情報を冗長な情報として符号に含めることで，符号間の距離を大きくし誤り訂正能力をあげ，係り受け精度の向上を実現した．

今回は，祖先子孫関係を学習させる際も，親子関係（直接の係り受け関係）を学習させるのに有効な素性を用いたが，今回使用した素性の中に祖先子孫関係を学習するのに不必要な素性があるかもしれない．また，今回使用しなかったが祖先子孫関係を学習するのに有効な素性があるかもしれないので，素性の洗練は必要であると考えられる．さらに，[8] を参考にして，親子関係の学習の際の素性も洗練する必要がある．これらにより，親子関係や祖先子孫関係，それぞれの判定精度を向上できれば，提案手法の精度も更に向上すると考えられる．

また，今回の手法は，木を構築していく手法に一般化できるため，係り受け解析以外の木を構築するタスクにも適用してみたいと考えている．

参考文献

[1] Sadao Kurohashi and Makoto Nagao. “A syntactic analysis method of long Japanese sentences based on the detection of conjunctive structures”, Journal of Computational Linguistics, Vol.20, No.4, pp.507–534, 1994

[2] Thomas G. Dietterich and Ghulum Bakiri. “Solving Multiclass Learning Problems via Error-Correcting Output Codes”, Journal of Artificial Intelligence Research, Vol.2, pp.263–286, 1995

[3] 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, pp.115–118, 1997

[4] 春野雅彦，白井諭，大山芳史. “決定木を用いた日本語係り受け解析”, 情報処理学会論文誌, Vol.39，No.12, pp.3177–3186, 1998

[5] Satoshi Sekine. “Japanese dependency analysis using a deterministic finite state transducer”, Proc. of COLING-00, pp.761–767, 2000

[6] Taku Kudo and Yuji Matsumoto. “Japanese Dependency Analysis Based on Support Vector Machines”, Proc. of EMNLP/VLC 2000, pp.18–25, 2000

[7] 工藤拓，松本裕治. “チャンキングの段階適用による日本語係り受け解析”, 情報処理学会論文誌, Vol.43，No.6, pp.1834–1842, 2002

[8] Manabu Sassano. “Linear-Time Dependency Analysis for Japanese”, Proc. of COLING 2004, pp.8–14, 2004

[9] David J. C. Mackay. “Information Theory, Inference, and Learning Algorithms”, Cambridge University Press,2003