ニューラル言語モデルを用いた法令文の並列構造解析

山腰 貴大[1]　大野 誠寛[1,2]　小川 泰弘[1,2]　中村 誠[3]　外山 勝彦[1,2]
[1] 名古屋大学 大学院情報科学研究科　[2] 同 情報基盤センター　[3] 同 大学院法学研究科
yamakoshi@kl.i.is.nagoya-u.ac.jp

1　はじめに

一般の人々にとって，法令文は読みにくいものであるとされている．その原因の一つは，階層的な並列構造が多用されることであり，例えば，図 1 の法令文には，階層を成した 4 個の並列構造が含まれている．このように複雑な階層を持つ並列構造は，人間による可読性を低下させるだけでなく，機械による法令文書処理の性能を低下させる要因にもなる．そのため，法令文の読解支援 [1] や法令用語シソーラスの自動構築 [2] などの法令文書処理においては，法令文に対する高性能な並列構造解析技術が望まれる．

一方，法令文に対する並列構造解析手法は，松山ら [3] によって既に提案されている．松山らの手法（以下，従来手法）は，法令文における等位接続詞の使い分け [4] に基づいて，並列構造を決定的に同定する．しかし，十分な解析性能を達成しているとは言い難い．その原因の一つとして，並列構造の同定の手がかりに，一対一の単語アライメントに基づく句の類似度を用いていることが考えられる．

そこで，本稿では，ニューラル言語モデル（NLM）[5] を用いた法令文の並列構造解析手法を提案する．本手法は，文脈を考慮した並列句間の類似性や，並列句を互いに入れ替えたときの文の流暢性を NLM によって求め，それらに基づいて並列構造を決定的に同定する．

2　法令文に特有な並列構造

本節では，構文情報付き法令文コーパス作成のための基準 [6] に基づいて，法令文特有の並列構造を簡単に解説する．なお，本稿では，図 1 の「両院若しくは一院」における「両院」や「一院」のように，並列関係にある語句を**並列句**と呼ぶ．また，「若しくは」，「又は」のように，並列句の連接と並列の種類を明示する語句を**並列キー**と呼ぶ．

階層的並列構造：法令文において階層的並列構造を表すために，並列キー「又は」と「若しくは」，「及び」と「並びに」はそれぞれ使い分けられる [4]．「又は」と「若しくは」は選択的な並列を表す．「又は」は最上位の並列構造に対して，「若しくは」はそれ以外の並列構造に対して用いられる．図 1 の文 S_1 中に現れる並列

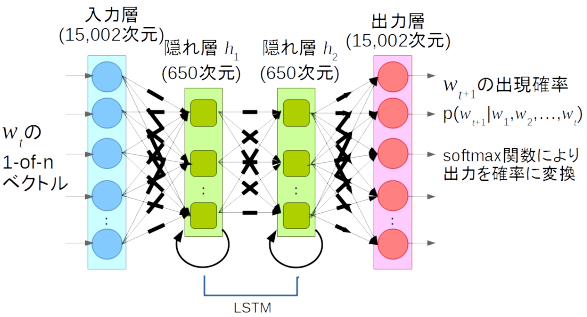


図 2: 本手法が用いる NLM

キーは，この規則に基づいて使い分けられている．一方，「及び」と「並びに」は併合的な並列を表す．「及び」は最下位の並列構造に対して，「並びに」はそれ以外の並列構造に対して用いられる．

「その他」と「その他の」による並列構造：「その他」と「その他の」は語句を例示的に列挙する場合に用いられる．これら二つは，一般の文では区別されることなく用いられるが，法令文では明確に区別される [4]．本稿では，紙面の都合により詳細な説明を割愛するが，これら二つは異なる構文を持つ．

3　ニューラル言語モデル

ニューラル言語モデル（NLM）[5] は，ニューラルネットワークを用いた言語モデルである．入力された語をベクトルに変換し，次に現れる語の確率分布を出力する．最新の NLM の多くは，再帰型ニューラルネットワーク（RNN）[7] によって構築されている．RNN は再帰的な結合を持つため，過去の情報，すなわち文脈を利用して出力値を計算できる．

本手法は，図 2 の RNN によって言語モデルを学習する．再帰的結合を持つ二つの隠れ層 h_1 と h_2 が文脈情報を保持している．

4　従来手法

松山ら [3] の手法は，図 3 に示す手順により，1 文中の全ての並列構造 par^i $(1 \leq i \leq N)$ を順に同定する．この手法では，一つの並列構造 par^i の単語列を次の式 (1) で形式化できることを前提とし，その内部

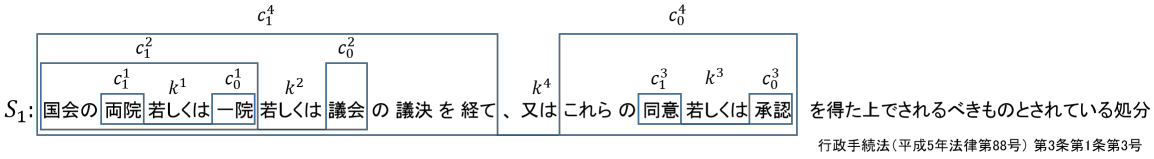


図 1: 階層的並列構造を含む法令文の例

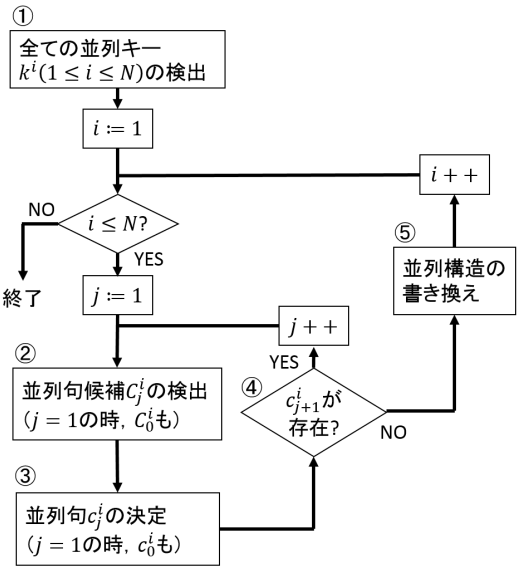


図 3: 並列構造解析処理の流れ

$$\left(\begin{array}{c}\text{及び}\\ \text{若しくは}\end{array}\right) > \left(\begin{array}{c}\text{並びに}\\ \text{又は}\end{array}\right) > \left(\begin{array}{c}\text{その他}\\ \text{かつ}\\ \text{と}\\ \text{や}\end{array}\right)$$

図 4: 従来手法における並列キーとその優先順位

の並列句 c_j^i を文末から文頭に向かって順に決定することにより，並列構造 par^i を同定する．

$$par^i = (c_j^i \cdot 「、」)^* \cdot c_1^i \cdot k^i \cdot c_0^i \qquad (1)$$

ここで，k^i は par^i を構成する並列キーを，“·” は語の連接をそれぞれ表す．図 1 における記号 c と k は，式 (1) に基づいて各並列構造の各要素を表したものである．以下の 4.1 節から 4.5 節では，図 3 の処理①から⑤をそれぞれ詳述する．

4.1 並列キーの検出

図 4 に示す並列キーを対象として，1 文中の全ての並列キー k^i $(1 \le i \le N)$ を検出する．ここで，i は，図 4 の優先順位（同順位の場合は，文頭に近い順）に基づいて並列キーを並び替えた後の順位を示す．

4.2 並列句候補の検出

並列句 c_j^i の候補の集合 C_j^i を求める．j が 0 か，1 か，2 以上かによって候補の求め方は異なるが，候補となる単語列が他の並列キーや読点，同定済みの並列構造の一部を含まないことを共通の条件とする．

C_1^i に含まれる単語列の始点は文節の最左語で，終点は必ず並列キー k^i の直前の語（読点の場合はその前の語）である．

C_0^i に含まれる単語列の始点は必ず k^i の直後の語（読点の場合はその後の語）で，終点は c_1^i の最右語と同じ品詞で文節末に最も近い語である．c_1^i の最右語が名詞のとき，シソーラスに基づく語の意味的類似性によって候補の数を制限する．ただし，同定済みの並列構造の終点は必ず候補に含める．

C_j^i $(j \ge 2)$ に含まれる単語列の始点は文節の最左語で，終点は c_{j-1}^i の始点の二つ前の語である [1]．

例えば，図 1 の S_1 において，$C_1^1 = \{$ 国会の両院, 両院 $\}$, $C_0^1 = \{$ 一院 $\}$ となる．

4.3 並列句の決定

c_1^i と c_0^i は同時に決定する．具体的には，c_1^i の候補と c_0^i の候補の全ての組み合わせから，並列句間の類似度が最も高くなるものを選択する．並列句間の類似度は，並列句候補ペアに対する一対一の単語アライメントを考え，対応関係を持つ単語の割合と，対応関係にある単語間の類似度の和により求める．単語間の類似度は，二つの単語が同じ品詞であると高くなる．また，名詞の場合は意味的類似性を考慮する．

一方，c_j^i $(j \ge 2)$ に関しては，c_j^i の各候補と決定済みの c_{j-1}^i との間で類似度を計算し，類似度が最も高くなるときの候補を c_j^i として決定する．

4.4 さらなる並列句の存在判定

c_j^i の直前の語が読点であり，かつ，その前の語と c_j^i の最右語が同じ品詞の場合，c_{j+1}^i が存在すると判定する．ただし，その品詞が名詞の場合，意味的類似性に関する制約を加える．

例えば，図 1 の S_1 において，c_1^1 を「両院」と決定した後の c_2^1 の存在判定を考える．このとき，「一院」の直前の語「の」は読点でないため，c_2^1 は存在しないと判定する．

4.5 並列構造の書き換え

並列句のどちらか一方の内部に下位の並列構造が存在すると，それらの間の類似度が低く算出される．これを避けるため，同定した並列構造 par^i の全体を c_0^i で書き換える．

図 1 の S_1 において，c_1^1 を「両院」，c_0^1 を「一院」と決定し，並列構造 par^1 の同定処理が終わった後，S_1 を「国会の一院若しくは議会の…」に書き換える．

5 提案手法

本手法は，従来手法と同じ手順（図 3）で 1 文に対する並列構造解析を進めるが，並列句の決定において NLM を用いる点に大きな特徴がある．本節では，図 3 の処理①から⑤について，従来手法と異なる点を中心に述べる．

5.1 並列キーの検出

図 5 の並列キーを対象として，入力文中の全ての並列キーを検出した後，その優先順位に基づき番号付けを行う．本手法では，並列キー「その他の」は並列キー「その他」と用法が異なるため，また，並列キー「から」は条番号などの並列において頻繁に使用されるため，これら二つの並列キーを解析の対象に加えることとした．二つの並列キー「その他の」と「から」により構成される並列構造は，それぞれ次の式 (2) と式 (3)

[1] c_{j-1}^i の始点の一つ前の語は読点であるため

(から) > (及び / 若しくは) > (並びに / 又は) > (その他 / その他の / かつ)

図 5: 本手法における並列キーとその優先順位

で形式化されるものであり，式 (1) とは異なる．そのため，これらの並列構造の同定の際に一部例外的な処理を施すが，その詳細については稿を改めて述べる．

$$par^i_{\text{その他の}} = (c^i_j \cdot \text{「、」})^* \cdot c^i_1 \cdot \text{「その他の」}, \quad (2)$$

$$par^i_{\text{から}} = c^i_1 \cdot \text{「から」} \cdot c^i_0 \cdot \text{「まで」} \quad (3)$$

なお，図 4 の並列キーのうち,「や」や「と」は実験データ中にほとんど出現しなかったため，本手法ではこれらを解析の対象から外した．

5.2 並列句候補の検出

並列句 c^i_j の候補の集合 C^i_j は，原則的に従来手法と同様に求める．ただし，本手法はシソーラスを用いないため，これに由来する制約は設けない．一方，階層的並列構造を形成する並列キーの使い分けを考慮し，各並列句候補が満たすべき制約として，次のものを追加する．

- k^i が「及び」のとき,「及び」による同定済みの並列構造を含まない．
- k^i が「又は」のとき,「又は」による同定済みの並列構造を含まない．
- k^i が「若しくは」のとき，他の制約に違反せずに含められる場合,「若しくは」による同定済みの並列構造を必ず含める．

5.3 並列句の決定

本手法は,「並列句は互いに類似し，かつ，並列句の順序を入れ替えても文の流暢性が保たれる」という仮定に基づいて並列句を決定する．具体的には，c^i_1 と c^i_0 は次の式 (4) により同時に決定し， c^i_j $(j \geq 2)$ は式 (5) により決定する．ここで，S は入力文である．

$$(c^i_1, c^i_0) = \arg\max_{(c_l, c_r) \in C^i_1 \times C^i_0} sim(S, (c_l, c_r), c_r) \times flu(S, (c_l, c_r)) \quad (4)$$

$$c^i_j = \arg\max_{c_l \in C^i_j} sim(S, (c_l, c^i_{j-1}), c^i_0) \times flu(S, (c_l, c^i_{j-1})) \quad (5)$$

ここで，$sim(S, (c_l, c_r), c_0)$ は並列句の類似性を表すスコア（**類似性スコア**）であり，$flu(S, (c_l, c_r))$ は並列句の順序を入れ替えたときの文の流暢性を表すスコア（**流暢性スコア**）である．いずれのスコアも，通常の語順の NLM（F-NLM）と，語順を逆にした NLM（B-NLM）によって計算される．

5.3.1 類似性スコア

$sim(S, (c_l, c_r), c_0)$ は次の式 (6) により求める．

$$sim(S, (c_l, c_r), c_0) = sim_f(S, (c_l, c_r), c_0) + sim_b(S, (c_l, c_r), c_0), \quad (6)$$

$$sim_f = 1 + |sim_c(vec_f(W_{fl}), vec_f(W_{fr}))|, \quad (7)$$

$$sim_b = 1 + |sim_c(vec_b(W_{bl}), vec_b(W_{br}))| \quad (8)$$

ここで，$sim_c(\boldsymbol{u}, \boldsymbol{v})$ は，二つのベクトル $\boldsymbol{u}$ と $\boldsymbol{v}$ のコサイン類似度である．式 (7) と式 (8) は，予備実験の結果に基づいて決定した．

$vec_f(W)$ は F-NLM に単語列 W を入力し終えたときの，また，$vec_b(W)$ は B-NLM に W を逆順にしたものを入力し終えたときの隠れ層 h_2 の値を表す．隠れ層の値を用いることで，文脈を考慮した類似性を捉えられることが期待できる．

式 (7) と式 (8) 中の単語列 W_{fl}, W_{fr}, W_{bl}, W_{br} は，それぞれ次の式 (9) から (12) によって生成する．

$$W_{fl} = W_f \cdot c_l, \quad (9) \qquad W_{bl} = c_l \cdot W_b, \quad (11)$$

$$W_{fr} = W_f \cdot c_r, \quad (10) \qquad W_{br} = c_r \cdot W_b \quad (12)$$

ここで，W_f は c_l の前方にある単語列，また，W_b は c_0 の後方にある単語列である．並列構造の前後にある単語列も用いることにより，より詳細に文脈を捉えられることを期待できる．

例えば，図 1 において，$sim(S_1, (\text{「両院」}, \text{「一院」}), \text{「一院」})$ を計算するとき，W_{fl} = 「国会の両院」，W_{fr} =「国会の一院」，W_{bl} =「両院若しくは議会の…」，W_{br} =「一院若しくは議会の…」となる．

5.3.2 流暢性スコア

$flu(S, (c_l, c_r))$ は式 (13) により求める．

$$flu(S, (c_l, c_r)) = flu_f(W_s) + flu_b(W_s) \quad (13)$$

ここで，W_s は，S 中の c_l と c_r を入れ替えた単語列とする．例えば，図 1 の S_1 において，$flu(S_1, (\text{「両院」}, \text{「一院」}))$ を計算するときの W_s は「国会の一院若しくは両院若しくは議会の…」となる．$flu_f(W_s)$ と $flu_b(W_s)$ は，それぞれ F-NLM と B-NLM に基づく W_s の流暢性を返す関数であり，次の式 (14) と (15) により求める．

$$flu_f(W_s) = \sqrt[|W_s|]{\prod_{t=1}^{|W_s|} P_f(w_t | w_1, w_2, \ldots, w_{t-1})}, \quad (14)$$

$$flu_b(W_s) = \sqrt[|W_s|]{\prod_{t=1}^{|W_s|} P_b(w_t | w_{|W_s|}, w_{|W_s|-1}, \ldots, w_{t+1})} \quad (15)$$

ここで，$P_f(w_t | w_1, w_2, \ldots, w_{t-1})$ は，F-NLM において，単語列 w_1, w_2, …, w_{t-1} の次に w_t が現れる確率を表す．$P_b(w_t | w_{|W_s|}, w_{|W_s|-1}, \ldots, w_{t+1})$ は，B-NLM における同様の確率を表す．文長の影響を排除するために，確率の相乗平均を求める．

5.4 さらなる並列句の存在判定

原則的に，従来手法と同様に判定する．ただし，本手法はシソーラスを用いないため，類似性に関する制約を設けない．

表 1: 実験結果

手法	精度	再現率
従来手法	36.8%(285/775)	39.7%(285/717)
本手法	65.2%(448/687)	62.5%(448/717)

5.5 並列構造の書き換え

従来手法と同様，同定した並列構造 par^i 全体を c_0^i で書き換える．

6 実験

本手法の有効性を検証するため，構文情報付きの法令文コーパス [6] に収録された法令を対象に並列構造解析を行った．

6.1 実験概要

上記の法令文コーパス [6] 中に出現する括弧内文字列は別の文として扱うこととし，並列構造 717 個を含む 592 文を解析した．解析時に使用する形態素情報と文節境界情報は，コーパス中に付与されたものを用いた．比較のため，従来手法でも解析した．

NLM 学習用コーパスは，2016 年 9 月に JLT[2] よりダウンロードした 716 法令の法令文から作成した．

NLM は，図 2 に示す RNN により構築し，単語の基本形を RNN の入力とした．形態素解析は mecab(v0.98)[3] で行い，辞書は IPA 辞書を用いた．出現頻度が高い 15,000 語と終端記号，未知語を有効語彙としたため，入力層と出力層は 15,002 次元である．

NLM の学習は，Chainer(v1.15.0)[4] を介して行った．パラメータの更新は確率的勾配降下法（学習率 1）により行い，更新時に，勾配ベクトルの L2 ノルムの最大値を 5 に設定し，ユニットを 0.5 の確率でドロップアウトさせた．エポック数は 8 とした．

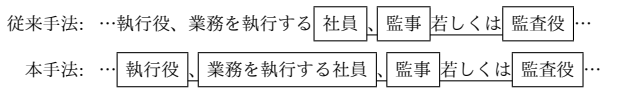
各手法の評価のために，正しく解析した並列構造の精度と再現率を求めた．並列構造における全ての並列句の範囲が正解データと完全に一致した場合，その並列構造を正しく解析したと判定した．

6.2 実験の結果と考察

表 1 に結果を示す．本手法は，従来手法と比べて，精度，再現率ともに大幅に向上した．

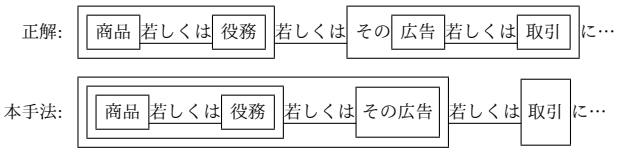
解析に成功した例を図 6 に示す．従来手法は並列句「業務を執行する社員」の決定に失敗し，さらに前方にある並列句「執行役」を探索できなかったが，本手法は並列構造を正しく解析できた．従来手法は，比較する並列句候補ペアに対して，一対一の単語アライメントを考え，対応関係を持つ単語の割合と対応関係にある単語間の類似度の和に基づいて，並列句間の類似度を構成的に計算する．そのため，並列句内の単語構成に影響を受けやすく，特に，単語数が異なる場合，並列句間の真の類似度に対して計算結果が低くなる．

[2]http://www.japaneselawtranslation.go.jp/
[3]http://taku910.github.io/mecab/
[4]http://chainer.org/



不正競争防止法（平成 5 年法律第 47 号）第 21 条第 1 項第 5 号の一部

図 6: 解析に成功した例



不正競争防止法（平成 5 年法律第 47 号）第 2 条第 1 項第 14 号の一部

図 7: 解析に失敗した例

その結果として，図 6 のように単語数が近い並列句候補を選択してしまったと考えられる．一方，本手法は，並列句とその前後の単語列を NLM によって一つのベクトルに変換し，それを用いて並列句間の類似度を直接的に計算するため，並列句の単語数に影響を受けることなく，正しく解析できたと考えられる．

次に，解析に失敗した例を図 7 に示す．図 7 の法令文中には，並列キー「若しくは」が三つ現れ，二番目の「若しくは」に対応する並列構造が最上位となる．しかし，本手法は，文頭に近い並列キーに対応する並列構造から順番に同定するため，必ず図 7 のように誤った解析をする．

7 まとめ

本稿では，NLM を用いた法令文の並列構造解析手法を提案した．法令文を対象とした実験の結果，従来手法と比べてより正確に解析できることを確認した．今後は，図 7 で示した問題に対処し，言語モデルやスコアリング関数の精緻化を行うことで，更なる性能向上を図る．

参考文献

[1] 山田大介，島津明．法令文の言語的特徴を利用した可読性向上のための表示．言語処理学会第 12 回年次大会発表論文集，pp. 196–199，2006.

[2] 萩原正人，小川泰弘，外山勝彦．グラフカーネルを用いた非分かち書き文からの漸次的語彙知識獲得．人工知能学会論文誌，Vol. 26，No.3，pp. 440–450，2011.

[3] 松山宏樹，白井清昭，島津明．法令文書を対象にした並列構造解析．言語処理学会第 18 回年次大会発表論文集，pp. 975–978，2012.

[4] 大島稔彦． 法制執務の基礎知識． 第一法規株式会社，2005.

[5] Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, Vol. 3, pp. 1137–1155, 2003.

[6] 山田将之，小川泰弘，外山勝彦． 構文情報付き法律文コーパスの設計と構築．第 14 回言語処理学会年次大会発表論文集，pp. 605–607，2008.

[7] Martin Sundermeyer, Ralf Schleuter, and Hermann Ney. LSTM Neural Networks for Language Modeling. Proc. of *INTERSPEECH 2010*, pp.194–197, 2010.