コネクショニスト日本語複文生成システム

福田大輔　本木実　嶋津好生
九州産業大学

1. はじめに

我々は，脳における言語理解，産出の認知モデルとして，ニューラルネットワークを用いたコネクショニスト日本語解析・生成システムについて研究を行っており，その目的は，人工神経回路網を使った日本語理解・産出システムの構築，及びコネクショニズム・サブシンボリックパラダイムに基づく内語過程のシミュレーションである．

ある1つのシンボルのコンテクストがその外周辺にある他のシンボルを使って表現されることをシンボリックパラダイム（局所モデル）という．一方，ある1つのシンボルのコンテクストが，そのシンボルをサブシンボル表現に変換することで，そのシンボルの内側に表現されることをサブシンボリックパラダイム（分散化モデル）という．

言語的表現の意味を理解する過程や，意味に適合する言語表現の産出過程を内語と呼んでいる．内語は具体的な発声を伴わず，個体の内部において進行している言語過程である．一方，外語は発声を伴い他人に向けられた言語過程である．

本システムにおいては，解析部，生成部ともに，形容詞，及び形容動詞の連用形を含む複文の解析・生成が可能なシステムモデルとして提案されている．これらはニューロマルチネットシステムによって実現され，形態素の概念や機能を反映してそれぞれの形態素を表すアセンブリパターンが学習によって獲得される．

解析部，すなわち日本語複文の理解過程については，このモデルから，形容詞・形容動詞の連用形についての処理を行う部分を除いたシステムで実験を行い，ある程度の見通しを得ている[2]．

今回は，生成部すなわち日本語複文の産出過程モデルを実現し，学習データや実行データを作成し，実際にシミュレーションを行った．このシステムは，提案されたモデルを実現する準備段階として，前述した解析部と同レベルの機能を備えたものである．

本論文では，その生成部のネットワークアーキテクチャ，日本語複文生成過程のシミュレーション，及びその結果について述べる．

最終的には，これら解析部と生成部を統合した，有益な日本語解析・生成システムの実現を目指す．

2. コネクショニスト日本語複文生成システム

図1にコネクショニスト日本語複文生成システムを示す．ネットワークアーキテクチャは，いくつかの階層型ネットワークをさらにネットワーク結合したものであり，必要な所にシンプルリカレントネットワークを組み込んでいる．入力は，脳内言語表現を表した格構造パイル，出力はかな綴り表現である．

また図2に，実際にシミュレーションで用いた日本語複文の例文「私は子供達が遊ぶ公園で女を殴る男を見た。」と，これを構成する3つの単文，及びこの複文によって作られる深層格構造の集まり（格構造パイル）を示す．

格構造パイルにおいて，2つの格構造間で同じ名詞句の同一性を表象する為に，名詞句のユニットアセンブリにいくつかユニットを追加し，同じ名詞句のその部分が呼応して，同一のパターンで活性化するようにしている．このユニットを交差ユニットと呼ぶ．これは，生成過程における，格

構造の入れ替え，及び格構造間における同一名詞句の消去などの役割を果たす．格構造中に，同一の名詞的シンボルが存在し，違うトークンを表している場合，同一のアセンブリパターンに異なる交差ユニットが付加されることで区別される．

図 1 において，日本語複文生成システムは，5つの階層型ネットによって構成される．左からスペラ(speller)，整列スタック(line up stack)，バインダ(binder)，単文ジェネレータ(clause generator)，スタック(stack)である．

単文ジェネレータは，Jordan ネットワークで構成され，単文の構成要素を時系列的に出力する．まず，統括成分が出力され，続けて補足成分が重要なものから優先して出力される．1 回出力されると，そのあと出力層が 2 回の左シフト出力を終えるまで，次の出力は制御される．また，出力すべき成分が無くなったとき，最後に自立語出力「＊」を出力する．バインダ入力「＊」の制御の下で，埋め込みレベルが直上位の単文生成の状態が復元される．すなわち，単文ジェネレータのプラン層へその格構造を，またスタックからポップして状態層を復元する．このとき，埋め込みレベル直上位の格構造が存在しない場合，複文生成が終了したことを示す自立語出力「！」を出力する．

バインダは，Jordan ネットワークで構成され，単文ジェネレータの自立語出力を監視しながら，格構造パイルと，複文生成システムの全てのモジュール相互のコミュニケーションを制御する．複数の格構造をひとつの複文で表現するには，単文ジェネレータがある格構造を単文に変換している間，出力されている名詞を監視していて，他の格構造の中に同じ名詞を含むものを見つけたら，直ちにプラン層をその格構造で置き換え，下位文の生成にかかる．そのときの状態層活性値ベクトルは，スタックにプッシュ保存した後クリアする．単文ジェネレータの自立語出力に動詞が出力されたら，プラン層を上位の格構造に置換し戻して，かつスタックをポップして状態層を復元し，上位文生成を再開する．

スタックは，RAAM (Recursive Auto-Associative Memory) ネットワークで構成され，コンピュータのメモリのスタックと同様の働きをする．単文ジェネレータが下位文の生成に切り換えるとき，状態層の活性値ベクトルをスタックにプッシュして保存し，単文ジェネレータが上位文の生成を再開するとき，スタックからポップして状態層を復元する．スタックのプッシュ動作とは，中間層の活性値ベクトルを入力層の一部としてフィードバックし，新しい入力ベクトルを受け入れることである．ポップ動作とは，出力層の一部の活性値ベクトルを中間層にフィードバックし，その後の出力を取り出すことである．

整列スタックは，RAAM ネットワークで構成され，単文ジェネレータからプッシュされたものを受け入れ，文として正しく綴られるべき順番に積み重ねていくものである．

綴り生成は整列スタックとスペラによって行われる．整列スタックからポップされた各単語は，スペラのプラン層に左シフト入力される．バインダが綴り生成をトリガーするが，あとは整列スタックとスペラがバインダの制御から離れ，自律的に動作する．

スペラは，Jordan ネットワークで構成される．ネットワークの入力には，単語の概念ベクトルを入力し，ネットワークの出力には，単語のかな文字綴りを 1 文字単位に逐次出力するように学習する．また，スペラは 2 アセンブリ入力であり，左アセンブリ入力単語のみの綴りを文字の時系列出力として生成する．左アセンブリ単語が用言のとき，右アセンブリに入力された後続の単語を考慮して，用言の活用形を学習する．各単語の綴りの最後はブランクであり，このブランク出力が，状態層のクリアと整列スタックのポップ，そしてプラン層の左シフト入力を制御する．

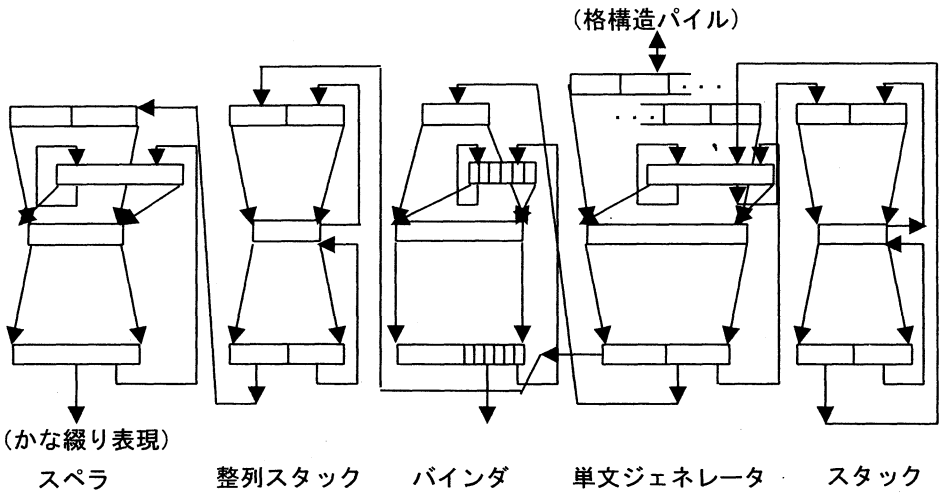

図１　コネクショニスト日本語複文生成システム

（連体修飾節複文）
私は子供達が遊ぶ公園で女を殴る男を見た。

（単文）
子供達が公園で遊ぶ。
公園で男が女を殴る。
私は男を見た。

	子供達		公園	遊ぶ	
	男	女	公園	殴る	
私	私	男			見る
主題	主格	対格	位格	現在	過去

図２　連体修飾節複文の格構造パイル（交差ユニット付）

3. 実験

今回の実験では，図2に示した例文1文を用いて実験を行った．学習方式は，同期学習方式，すなわち，個々の学習文に対し，全てのモジュールを同期して学習させる方式を用いた．また，学習アルゴリズムには，誤差逆伝播学習則を用い，学習係数η=0.1とした。

計10000回の学習を行ったところ，結果として，およそ3000回の学習でスペラの学習が収束し，例文における47のかな文字綴りを正しく生成することができた．また，学習回数3700回から4500回までの間，綴り生成にばらつきのある不安定さが見られたが，その後は，学習終了まで安定した動作を見せた．

単文ジェネレータ，バインダ，スタック，整列スタック，スペラの，各モジュールのrms誤差であるが，スペラ学習収束時，すなわち，学習回数3000回の時点での最小値はそれぞれ，0.221，0.113，0.029，0.071，0.020であり，学習終了時，すなわち，10000回の時点での最小値はそれぞれ，0.059，0.032，0.039，0.068，0.004であった．スペラを除く各モジュールのこれらの値には，学習を通して，ばらつきが見られた．

また，今回は，単文ジェネレータ，バインダ，整列スタックのrms誤差が下がらなかった為，実行データを用いたシステムの実行を行うことができなかった．

4. 考察

結果を見てもわかる通り，単文ジェネレータ，バインダ，整列スタックの学習が収束していない．学習係数を0.5にしての実験も行ってみたが，最終的に記録した各モジュールの最小rms誤差には，さほど変化は見られなかった．

本実験では，単文ジェネレータ，及びバインダの状態層を逐次クリアするという操作を行っていない．この為，これらのモジュールに，期待された機能を判別できなくしている．この操作を取り入れることで，誤差低下を望める．

また，整列スタックの学習が収束しないのはRAAMの共通した傾向であるが，単文ジェネレータ，及びバインダの誤差を小さくすることにより、誤差低下を見込めるのではないかと考えている．さらに，整列スタックの中間層を2層増やすことによって入力と出力の写像関係がより近くなり，誤差を小さくすることができると考えている．

5. おわりに

今後まず，学習データや実行データを増やして実験を行う必要がある．構文上の再帰的構造の生成については，この問題をアーキテクチャでクリアしている．しかしそのことを検証するために，埋め込みレベルがより深い実行データを用意すべきである．

次に，個別学習方式での学習，すなわち個々のモジュールを別々に学習させて後で統合する学習方式でも，学習実験を行い，その結果を評価したい．解析部においては，同期学習方式と個別学習方式の両方式で学習実験を行っている．その正答率は，個別学習方式の方が低いという結果を得ている．

次に，本システムに新たなモジュールを追加し，既に提案されている，形容詞，及び形容動詞の連用形を含む複文の生成を可能とするシステムの，ネットワークアーキテクチャの実現であり，これは，解析部も同様である．これにより，より広い範囲での複文の生成が可能になると考える．

参考文献

[1]嶋津，“コネクショニスト自然言語処理の方法，”九州産業大学工学部研究報告，第37号，pp.115～122，2000.

[2]嶋津，“ニューロマルチネットシステムによる日本語修飾-被修飾関係複文の解析と生成，”電子情報通信学会技術研究報告，信学技法Vol.101，No.615，pp.25～32，2002.