The 32nd Annual Conference of the Japanese Society for Artificial Intelligence, 2018

修飾節付与による複文のニューラル生成

Divide and Generate: Neural Generation of Complex Sentences

尾形朋哉 *1 Tomoya Ogata　小町守 *1 Mamoru Komachi　高谷智哉 *2 Tomoya Takatani

*1首都大学東京 Tokyo Metropolitan Univercity　*2トヨタ自動車株式会社 TOYOTA MOTOR CORPORATION

In recent years, research on chat dialogue systems has been attracted much attention. Typical chat dialogue system selects and outputs an appropriate response from a dialogue database for the input user utterance. However, it is not possible to output an appropriate response if the coverage of the database is limited. Therefore it is necessary to augment the database beforehand. In this research, in order to amplify various kinds of responses in the database, we propose a task to generate a complex sentence from a simple sentence. We first divide a complex sentence into a main clause and a subordinate clause to learn a generator model of modifiers, and then use the model to generate a modifier clause to create a complex sentence from a simple sentence. We present an automatic evaluation metric to estimate the quality of models.

1. はじめに

近年，コンピュータが人と会話を行うための雑談対話システムの研究が行われている．雑談対話システムには，入力されたユーザ発話に対して，データベースから適切な応答文を選択して出力する手法が広く使われている [Ji 14]．この手法では，データベース内に十分な応答文がない場合，ユーザの発話に対して適切な応答文を出力できないことがある．したがって，ユーザのあらゆる発話に対応するためには，事前に多様な応答文を生成し，データベースを拡張する必要がある．

応答文を生成するための研究の 1 つとして，テンプレートとして用意された応答ルールに単語を当てはめることで応答文を生成する手法がある [Stent 04]．この手法は人手で応答ルールを作成するため，文法的に違和感のない応答文を生成できるが，人手で作成できる応答ルールには限界があり生成できる文の種類は限られてしまう．

一方，ニューラルネットを用いて対話行為という構造化されたデータから応答文を生成する研究がある [Wen 15][Dušek 16]．これらの研究では入力の制約を満たす応答文の生成に成功しているが，対話行為がアノテーションされたコーパスが大量に必要になる．しかし，日本語において対話行為がアノテーションされた利用可能な大規模コーパスは存在しない．

そこで，本研究では単文に対して修飾節を付与することで，複文を生成することを目指す．複文の生成を人手で行う場合，修飾節を挿入する位置や主節に対してふさわしい修飾節を考えなくてはならず，大量に複文を生成するための手間が大きい．しかし，本研究のように自動的に複文を生成することで，生成された文が自然かどうかを人手で判断するだけでよく，複文の生成にかかる人的コストを減らすことができる．本研究はドメインごとに一定数利用可能な単文を入力として，それらを元にした複文を生成することでデータベースの拡充を目指す．

本研究の入力と出力は表 1 に示す通りである．本研究の入力は修飾可能な名詞を含む文であり，出力は入力された文に修飾節を加えた文となる．修飾節は入力の文より長くならず，出力の長さは入力の高々2 倍である．

連絡先: 尾形朋哉，首都大学東京，ogata-tomoya1@ed.tmu.ac.jp

表 1: 本研究における入力と出力の応答文の例

	入力	出力
応答文 1	車に乗りました	彼に借りた車に乗りました
応答文 2	方法を探しています	この先に進む方法を探しています

本研究の主な貢献は以下の通りである．

- 入力の応答文に対して修飾節を拡張した応答文を生成するタスクを提案した．
- 複文を含む応答文データベースから自動的に修飾節拡張のためのコーパスを生成し，ニューラルネットワークによって修飾節拡張生成モデルを学習する手法を提案した．
- 修飾節拡張に対する評価尺度を検討し，全てを End-to-end で行うベースラインと比較して，修飾節拡張をパイプラインで行う手法が流暢性と多様性の両方で改善することを示した．

2. 関連研究

文書要約や文圧縮が入力の文書や文から部分的な情報を生成する研究であるのに対して，部分的な情報から文を生成する研究がある [Wen 15][Dušek 16]．Wen らや Dušek らの研究は，入力として与えられた対話行為の制約を満たすような自然言語文を生成する．これらの研究において出力に含むべき内容は全て入力の対話行為に与えられているが，本研究では修飾節を指定せず，学習データに基づいた修飾節を生成している．

また，入力として情報を与えずに学習データに基づいた文生成を行う研究がある [Bowman 16]．この研究は事前にデータから学習した確率分布に従い文を生成することができ，学習データに基づき修飾節が自動で生成される点で本研究と類似しているが，本研究は入力を制約として修飾節を生成している．すなわち，本研究は生成される文を入力側に与える情報で操作することができるので，特定のドメインに含まれる文や特定のキーワードを含んだ文を用いることができる．

さらに，複文を生成することを目的とした研究がある [緒方 16]．緒方らは 2 つの単文が与えられたとき，従属節とする単文をルールベースで変形し，主節の指定された位置に挿

表 2: 修飾節の構成文節として抽出する文節の例

文節の条件	例
動詞を含む	飲む，連れてきた
名詞述語を含む	中枢である

入することで複文を生成する．緒方らの研究では従属節と主節に当たる単文を入力として与えなくてはならないが，本研究では学習データに基づいて主節にふさわしい従属節を自動で予測し，修飾節を生成する．また，緒方らの研究では入力の 2 つの文の他に，従属節の変形や挿入位置に関しても入力として与える必要がある．本研究ではルールで検出した位置に対し，学習データに基づいて自動で修飾節の生成を行う．

3. 修飾節付与による応答文生成

本研究では既存の応答文データベースに存在する単文の応答文を入力として，修飾節を挿入して複文の応答文を生成することを目指す．

本研究の生成モデルとして Attention 機構付きの Encoder-Decoder を用いる [Bahdanau 15]．Encoder-Decoder は任意長の入力列から任意長の出力列を出力するように重みを学習するモデルである．Attention 機構により，単語を生成する際に対応する入力側の情報を考慮することができる．本研究において入力列は文，出力列は入力の文に修飾節を付与した文からなるパラレルコーパスを用いて Encoder-Decoder を学習する．

3.1 節において修飾節付与を行うためのコーパスの作成方法を説明し，修飾節の拡張を行うモデルを 3.2 節で提案する．さらに，生成された文の評価尺度を 3.3 節にて説明する．

3.1 コーパス

修飾節拡張を学習するためには単文または複文に修飾節がアノテートされた複文からなる修飾節拡張パラレルコーパスが必要であるが，それを人手で作成するのは困難である．したがって，本研究では複文からなる生コーパスから修飾節を削除することで，疑似的に修飾節拡張パラレルコーパスを作成した．本研究の実際の入力は単文を想定しているが，単文を収集するためにはルールにより複文や重文を除く必要があるため，十分な量コーパスが集まらない可能性がある．そのため，本研究では学習データは単文に限定せず収集し，修飾節拡張の学習に用いた．本研究では以下のルールを用いて修飾節を抽出した．なお，係り受け解析には CaboCha を用い，形態素解析には MeCab+IPAdic を利用した．

1. 文を係り受け解析する．
2. 品詞細分類が “一般”または “固有名詞”の名詞を含む文節に係る文節を抽出する．
3. 抽出した文節において，表 2 に該当する文節を修飾節の構成文節として抽出する．なお，複数存在する場合はランダムに 1 つ選択する．
4. 修飾節の構成文節に含まれる文節に係る文節を抽出し，修飾節の構成文節に追加する．
5. 手順 4 を繰り返し行い，抽出されたすべての文節を 1 つの修飾節とする．

本研究では，単文と複文からなるコーパスでモデルの学習をし，テスト時には単文のみのコーパスに対して修飾節付与を行なうように，モデルの学習と実際の文生成で異なるコーパスを使える．なお，モデルの学習を行うコーパスと実際に文生成に使うコーパスのドメインがかけ離れている場合，不自然な修飾節が挿入されることがあるので学習コーパスのドメインには気をつける必要がある．

Algorithm 1 挿入位置の検出

```
chunks ← 文を係り受け解析
名詞 index リスト : noun_idx ← []
検出フラグ : d ← FALSE
for i = 0 to |chunks| − 1 do
  if chunks[i] に品詞が名詞の単語を含む then
    append i to noun_idx
    if chunks[i] に係る動詞または助動詞が存在しない then
      chunks[i] の名詞の前後に特殊記号を挿入
      d ← TRUE
      break
    end if
  end if
end for
if d ≠ TRUE then
  i ← min(noun_idx)
  chunks[i] 内の名詞の前後に特殊記号を挿入
end if
return chunks
```

表 3: パイプラインモデルにおける修飾節生成モデルの入力と出力の応答文

	入力	出力
応答文 1	<ins> 車 </ins> に乗りました	彼に借りた車に乗りました
応答文 2	<ins> 方法 </ins> を探しています	この先に進む方法を探しています

3.2 パイプラインモデル

本研究のベースラインは Encoder-Decoder を用いて，入力の文に対して修飾節を付与した文を End-to-end で生成するモデル (End-to-end モデル) である．End-to-end モデルは，入力された文に対して修飾節を挿入する位置の検出と，修飾節の生成を同時に行なっているため，問題を複雑にしている可能性がある．本研究では，挿入位置の検出と修飾節の生成を別々に行うことで，より頑健に修飾節を生成するパイプラインモデルを提案する．

パイプラインモデルは挿入位置の検出をルールベースにより行い，修飾節の生成を Encoder-Decoder を用いて行う．まず，挿入位置の検出のルールを Algorithm 1 に示す．

修飾節の生成の学習に使うコーパスは，表 3 で示すように，3.1 節で作成したコーパスの入力に対して，1 つの単語の前後に特殊記号を挿入したものとなる．このコーパスを用いて Encoder-Decoder を学習することで，入力側で修飾位置を指定した単語に対して，修飾節を生成することができる．

3.3 評価尺度

本研究は入力の単文に対して修飾節を付与することで複文を生成する．この時，生成される修飾節は修飾先の単語にふさわしい単語であれば何でもよいため特定の正解は存在しない．したがって，生成される文を評価するために BLEU などの参照訳を利用する評価尺度を用いるのは適切ではない．しかし，生成される文としては流暢であるものが望まれるため，テストデータにおいて作成した N-gram 言語モデルによる Perplexity

で流暢性を評価する．

また，修飾節を付与した後の文は，元の文より情報量が増えていると考えられる．ここでの情報量の増加は単語のタイプ数の増加に依存するものなので，文中の情報量を評価するための尺度として文内の単語タイプ数を用いることを提案する．本タスクの目的は，流暢性をできるだけ損ねることなく，多様性を向上させることである．

4. 単文に対する修飾節付与

4.1 実験設定

コーパス．会話文コーパスの作成のために，小説投稿サイトである「小説家になろう」に投稿された小説の文章データを利用する．この文章データにおいて「」に囲まれた部分を会話として抽出し，会話文コーパスを作成する．このとき，会話は複数の文を含む可能性があるので，文を正規化したあと，“。”や“?”などの記号を文の区切りとみなして文を作成している．ここで作成された会話文コーパスに対して，3.1 節で説明した手法を用いて修飾節を除いた文と元の文のパラレルコーパスを作成した．テストデータと開発データに関しては，入力として単文を想定しているので，3.1 節のルールに加え，修飾節を 1 つ抽出した後の文が動詞または助動詞を 1 つのみ含む文のみを抽出することで作成した．なお，拡張される修飾節が極端に短い場合や長い場合には，修飾節にほとんど意味がないことや，修飾節が長すぎるために不自然な文になることがあるので，修飾節の文節数が 2 以上のものと修飾節の文字数が元の文よりも短い文のみを利用している．さらに，生成される修飾節が特定の修飾節に偏るのを防ぐため，学習データにおいて拡張される修飾節が同じであるものを取り除いた．ここで作成したコーパスは学習データが 95,234 文で，テストデータと開発データはそれぞれ 1,000 文である．

また，「小説家になろう」から作成されたコーパスによって学習したモデルが，外部ドメインの文に対して正しく修飾節付与を行えるかを実験する．外部ドメインとして使用するデータは対話破綻チャレンジの雑談対話コーパス [東中 14] から単文のみを抽出したものを用いる．このコーパスはシステムとユーザの発話からなる対話コーパスで，本研究ではユーザの発話を抽出して修飾節の拡張を行う．

モデル．単文を入力した時に，適切な位置に適切な修飾節を挿入できるかを End-to-end モデルとパイプラインモデルでそれぞれ実験する．また，パイプラインモデルにおいて探索幅 10 でビームサーチをした時にどのような出力が生成されるかを実験する．ニューラルネットワークのハイパーパラメータは，語彙サイズを 10,000，埋め込み層を 512，隠れ層を 512，バッチサイズを 128 として実験を行なった．単語ベクトルの初期値は学習データで学習した word2vec，最適化アルゴリズムは Adagrad で学習率は 0.01 を用いた．モデル選択はエポックを 20 まで回し，各 dev セットで BLEU が最大になるエポック数を選択した．

評価．それぞれのモデルに対する自動評価は生成された文を N-gram (N=4) 言語モデルによる Perplexity と単語のタイプ数の平均により評価する．なお，N-gram 言語モデルは modified Kneser-Ney 法によるディスカウンティングと補間を用いている．また，人手での評価としてそれぞれのシステムからランダムにサンプルした文のうち，どちらの方が自然な文かを相対評価した．この時，それぞれのシステムの出力文の文としての自然さが同程度である場合も考慮し，優劣がつく文数が 100 文になるまでサンプルした．

表 4: Perplexity と単語タイプ数による評価

	Perplexity	単語タイプ数
End-to-end モデル	54.9	13.85
パイプラインモデル	**46.9**	**14.44**

4.2 実験結果

定量的評価．Perplexity と単語タイプ数による評価の結果を表 4 に示す．パイプラインモデルの方が End-to-end モデルよりも Perplexity が低くなっており，より流暢性の高い出力が出せていると言える．また，単語タイプ数はパイプラインモデルの方が大きく，より情報量の多い文を出せていると言える．

ランダムにサンプルした 210 文について人手で出力文のみを見て流暢性を相対評価した結果，End-to-end モデルが良い : パイプラインモデルが良い : 同程度＝ 32 : 68 : 110 となり，パイプラインモデルの方が流暢性の高い文を生成できていた．

出力例．End-to-end モデルとパイプラインモデルのそれぞれの出力を表 5 に示す．なお，パイプラインモデルの下線部は挿入された修飾節を表している．表 5 の 1 行目のように End-to-end モデルは修飾節内に修飾先の単語を出力してしまうことがあった．一方で，パイプラインモデルは修飾先の単語と同じものを修飾節内に出すことはなかったが，文の主語と重複する単語を出力することがあった．また，表 5 の 2 行目の元の文における “大手”が End-to-end モデルでは似た意味の “大型”に変わることで，不自然な日本語になってしまうことがある．さらに，表 5 の 3 行目や 5 行目のように End-to-end モデルは出力が，おかしな単語が出力され，文法的におかしく，意味の分からない文になる事例が多く見られた．パイプラインモデルは文法的におかしいものはほとんど出力されないが，表 5 の 4 行目や 5 行目の例のように，不自然な位置に修飾節を挿入したり，修飾先に対して適切でない修飾節を生成してしまうことがある．

最後に，外部ドメインの文に対して，パイプラインモデルが実際に生成した文を表 7 に示す．表 7 の 1 行目のように，修飾先の単語がモデルの語彙に含まれるものに関しては比較的正しい修飾節が生成できている．一方，修飾先の単語が未知語であるものに対しては，表 7 の 2 行目のように正しい修飾節が生成できた例もあるが，表 7 の 3 行目のように，正しい修飾節を挿入できないものが多く見られた．特に，未知語に対しては “俺が作った”のようにモデルが生成しやすい修飾節を挿入する例が多く見られた．

5. 考察

パイプラインモデル．End-to-end モデルとパイプラインモデルの両方に共通して，あらゆる入力文に対して汎用的な修飾節が付与されやすいという問題があった．この問題は，モデルを選択するための評価尺度と関係している．本研究では BLEU が最大のモデルを選択して用いたが，修飾節に関しては正解は 1 つに定まらないため，BLEU が最大のモデルが良い修飾節を生成できるとは限らない．また，エポック数が増えるにつれて，生成される修飾節は多様性を増す傾向にあるので，エポック数の高いモデルを何らかの尺度で選択することができれば，多様性のある修飾節を挿入することが可能である．本研究では開発データにおける Loss や Perplexity が上昇し続けたため，BLEU に基づくモデル選択をしたが，N-gram 言語モデルの Perplexy や単語のタイプ数に基づきモデルを選択することも考えられる．

表 5: End-to-end モデルとパイプラインモデルの出力の比較

End-to-end モデル	パイプラインモデル
流石 , この 国 の 英雄 と 呼ば れ た 英雄 と いう べき です	流石 , この 国 を 救っ て くれ た 英雄 と いう べき です な
大型 柄 , 私 が 見 た 分量 ござい ます	この 国 を 守る 大手 柄 , おめでとう ござい ます
ケイ ぅ , 何 か を する コース で ね	俺 が 持っ て き た 手作り チョコ って 名目 で ね
ルシエル 様 は この 街 に ある 飛行船 を 見 られ ませ ん でし た か	私 の よう な ルシエル 様 は 飛行船 を 見 られ ませ ん でし た か
商業 ランク なんて , 勝手 に ある 幻々 する だけ だ よ	俺 が 連れ て き た 商業 施設 なんて , 苛々 する だけ だ よ

表 6: ビームサーチによるパイプラインモデルの上位 5 件の出力

入力	犯人 は , リフレイア という 娘 だ	俺 も できる だけ 早く , 術 を 完成 さ せる
beam 1	それ を 知っ て いる 犯人 は , リフレイア という 娘 だ	俺 も できる だけ 早く , 魔力 を 使う 術 を 完成 さ せる
beam 2	私 達 を 襲っ て き た 犯人 は , リフレイア という 娘 だ	俺 も できる だけ 早く , 俺 たち を 倒す 術 を 完成 さ せる
beam 3	私 達 を 倒し た 犯人 は , リフレイア という 娘 だ	俺 も できる だけ 早く , 俺 たち を 守る べき 術 を 完成 さ せる
beam 4	私 達 を 含め た 犯人 は , リフレイア という 娘 だ	俺 も できる だけ 早く , それ を 使う 術 を 完成 さ せる
beam 5	私 達 を 襲っ た 犯人 は , リフレイア という 娘 だ	俺 も できる だけ 早く , 俺 たち を 守る 術 を 完成 さ せる

表 7: 外部ドメインに対するパイプラインモデルの出力（<unk:>は入力側の未知語を表す）

入力	パイプラインモデル
海 は <unk:うきうき> し ま す ね	この 街 に ある 海 は うきうき し ま す ね
<unk:野菜> <unk:ジュース> だけ で す	俺 が 作っ た 野菜 ジュース だけ で す
<unk:カエル> が 好き な ん で す か	俺 が 作っ た カエル が 好き な ん で す か

End-to-end モデルでは似た意味の単語が代わりに出力されたり，おかしな単語が出力されることで不自然な文が生成されることがあった．これらは，End-to-end モデルが修飾節の挿入位置を予測するために隠れ層に保存する，挿入位置に関する情報が単語選択に影響したと考えられる．

パイプラインモデルにおいて不自然な位置に修飾節が挿入される問題に対しては，ルールではなく，RNN を用いて修飾節の挿入位置の検出を行うことで，柔軟な修飾位置の検出ができ，より自然な文が生成されると考えられる．

表 6 にパイプラインモデルにおけるビーム幅 10 のビームサーチにおける上位 5 文の結果を示す．それぞれ流暢性が高く，意味合いの異なる文を生成することができている．したがって，文内で同じ単語を複数回出力してしまう問題に対しては，Decoder でビームサーチをすることで複数候補文を保持し，同じ単語が出現するものに対してペナルティをかけてリランキングすることで，重複が発生する出力を避けることができると考えられる．

修飾先の単語が未知語となる場合に適切な修飾節が生成できない問題に対しては，単語のカテゴリを元にして，物や人物，場所などカテゴリに応じた未知語を表す記号を用意することで，カテゴリに対応した無難な修飾節を生成できる可能性がある．

評価方法. 今回，文の情報量の評価尺度として，文中の単語タイプ数を用いた．この評価尺度ではより多くの種類の単語を使った文が良い評価になるが，文長に対する制限を加えていないため，より長い文が有利になりやすいという問題点がある．また，本研究では助詞のように単体ではほとんど意味のない単語も，名詞や動詞などの文において重要度の高い単語と同列に扱っているため，助詞などで冗長になっている文の方が良い評価になってしまう可能性がある．したがって，名詞や動詞など文にとって重要度の高い品詞のタイプ数を情報量の尺度として用いることも考えられる．

6. おわりに

本研究では入力文に修飾節を付与し，複文を生成する手法を提案した．この手法は修飾節を含む文さえあればよいので，任意のドメインでコーパスを作成し，入力の文に対して修飾節を付与した文を得ることができる．今回の実験で，入力側の単語が未知語になった時にふさわしい修飾節を生成するのが難しいという問題が明らかになった．今後は，入力側での未知語を表す記号を複数用意するなど，入力側の未知語の問題に対して実験を行いたい．

参考文献

[Bahdanau 15] Bahdanau, D., Cho, K., and Bengio, Y.: Neural Machine Translation by Jointly Learning to Align and Translate, in *ICLR* (2015)

[Bowman 16] Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A., Józefowicz, R., and Bengio, S.: Generating Sentences from a Continuous Space., in *CoNLL* (2016)

[Dušek 16] Dušek, O. and Jurcicek, F.: Sequence-to-Sequence Generation for Spoken Dialogue via Deep Syntax Trees and Strings, in *ACL* (2016)

[Ji 14] Ji, Z., Lu, Z., and Li, H.: An Information Retrieval Approach to Short Text Conversation, *arXiv* (2014)

[Stent 04] Stent, A., Prasad, R., and Walker, M.: Trainable Sentence Planning for Complex Information Presentation in Spoken Dialog Systems, in *ACL* (2004)

[Wen 15] Wen, T.-H., Gasic, M., Mrksic, N., Su, hao P., Vandyke, D., and Young, t. J.: Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems, in *EMNLP* (2015)

[緒方 16] 緒方 健人, 佐藤 理史, 松崎 拓也：日本語文生成器 Haori における複文合成, 言語処理学会 第 22 回年次大会 (2016)

[東中 14] 東中 竜一郎, 船越 孝太郎 : Project Next NLP 対話タスクにおける雑談対話データの収集と対話破綻アノテーション, *SIG-SLUD*, Vol. B4, No. 02, pp. 45–50 (2014)