直列化された係り受け構造によるニューラルネット文生成

渡辺 有祐　　橋本 和真　　鶴岡 慶雅
ソニー株式会社
東京大学大学院工学系研究科
YusukeB.Watanabe@jp.sony.com
{hassy, tsuruoka}@logos.t.u-tokyo.ac.jp

1 はじめに

近年、ニューラルネットワークによる言語生成のモデルが、機械翻訳 [1] や、画像のキャプション付け [2] などの応用において成功を収めている。これらの手法では、Long Short Term Memory (LSTM) のような系列データを学習する能力の高いニューラルネットワークを用いて文のパターンを学習させ、翻訳元の文や、キャプションを付けたい画像のコンテクストの元で、単語の系列を生成させている。

このように、文を純粋に単語の系列として扱うアプローチは簡明ではあるものの、文の持つ構文構造を無視していることになる。その結果、複雑な修飾関係をもつ文の生成が難しかったり、生成された文の修飾関係が曖昧になったりするという問題が起こりうる。

この問題に対する 1 つのアプローチとして、文を単語列として扱うのではなく、係り受け構文構造付きで生成するというアプローチが提案されている。しかし、論文 [3] で提案されたモデルは、Feed Forward Neural Network をベースにしたものであり、LSTM のような系列パターンを学習する能力が不十分であった。

本稿では、文の係り受け構造を系列データに変換して学習する方法を提案し、係り受け構造付きで文生成が行えることを示す。また、文生成時に係り受け構造をコントロールすることにより、様々な構文構造の文を生成できることを示す。

2 ニューラル言語モデル

言語モデルとは、文の単語の系列と見た時に、そこまでの単語の系列から次の単語を予測する確率モデルである。これを用いると、順に次の単語を予測していくことで、文を生成することができる。一番単純な言語モデルは n-gram と呼ばれるものである。これは $n-1$ 個の単語の系列から次に来る単語の頻度のカウントを確率として使うモデルである。このモデルは n が小さいと文の構造を十分にできず、逆に n を大きくすると計算量的に扱うのが困難になり学習に必要なデータ量が非常に多くなってしまうなどの問題がある。

これに対し、Recurrent Neural Network (RNN) を用いたモデルでは、RNN 内部の固定長ベクトルが過去の単語系列の情報を表現することによって、計算量を抑えつつ、文の長いパターンを考慮することが可能である。実際、言語モデルとしての性能も優れていることが知られている [5]。事前に文脈長 n を決める必要がないことも RNN の長所である。

RNN のなかでも Long Short Term Memory (LSTM) と呼ばれるモデルは、選択的に入出力を行うためのゲート構造を備えた、特殊な RNN である。正確な定義は、例えば文献 [8] がわかりやすい。LSTM は単純な RNN に比べて、学習が行い易いことが知られている [6]。本稿ではこの結果に従い、系列学習には LSTM を用いる。

3 依存文法

依存文法とは、単語の依存関係 (係り受け関係) を木構造によって表現する、構文解析の方法の 1 つである。ここで、木とは閉路のないグラフのことである。各単語は木の頂点に対応し、ただ一つの主辞 (head) と複数の子 (dependant) を持つ。任意の単語から主辞をたどって到達できる特別な頂点は、ROOT と呼ばれる。(この頂点は例外的に文の単語と対応しない。)

図 1 は、“A black dog runs through a field.” という文の、係り受け関係を木構造として図示したものである。文の動詞 ‘runs’ が ROOT の子になっていることが見て取れる。矢印に付けられた nsubj, prep 等のラ

ベルは、主辞と子の関係を表している [1]。

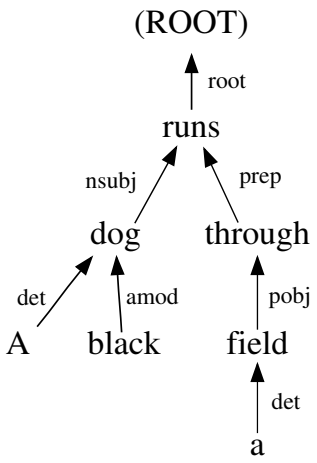

図 1: 依存文法における係り受け関係の例

依存文法には、いくつかの変種が存在する。今回はスタンフォード大学の自然言語処理グループが提案している Stanford Dependencies を用いた [7]。[2]

4 提案モデル

前節で説明したとおり、係り受け構造は木によって表現することができる。ただし、係り受け構造には語順の情報が全くないので、この木構造から文を復元することはできない。しかし、係り受け構造が非交差的 (projective) であるという仮定 [3] のもとでは、

- 子の間の語順
- 子と主辞の語順

の情報がわかっていればもとの文が復元できる。例えば、図 1 の例だと、'dog' は 'A' と 'black' という二つの子を持つが、'A' と 'black' は 'dog' よりも前で、'A' は 'black' よりも前であることがわかっていれば、この部分は "A black dog ..." となることが決定される。同様に、'dog' は 'runs' よりも前等の語順情報を集めると、"A black dog runs through a field" という文が復元される。

木構造と、上述の語順情報を何らかの方法で一列のデータに変換する (直列化する) ことができれば、これを LSTM によって学習し、生成するのは容易である。本稿では、深さ優先探索の順序を用いて直列化する方法 (Depth First Serialization; DFS) を提案する。

4.1 Depth First Serialization

DFS 法では、木の ROOT から出発し、語順の前の単語から深さ優先で探索していく。全ての頂点を探索した後、最後には ROOT に戻ってくる。木の頂点間の移動を表す <LD>, <LU>, <RD>, <RU> という特別な記号を導入する。ここで、<LD>は LeftDown で、主辞から語順が前の子に移動することを表す。<LU>は LeftUp で、子から語順が後ろの主辞に移動することを表す。<RD>, <RU> (RightDown, RightUp) も同様である。

具体例で説明しよう。図 1 の例だと、DFS によって得られる系列は、'runs', <LD>, 'dog', <LD>, 'A', 'black', <LU>, <LU>, <RD>, 'through', <RD>, 'field', <LD>, 'a', <LU>, <RU>, <RU> となる。まず ROOT から出発し頂点 'runs' に至る [4]。'dog' は ROOT よりも語順が前の子なので、記号<LD>で移動する。'A', 'black' は共に 'dog' よりも語順が前の単語なので、この順にまとめて探索する。その後の<LU>, <LU>によって 'runs' まで戻る。(以下省略)

このように DFS を使うと、文を係り受け構造付きでシリアライズすることができる。得られる系列長は、単語数を k とすると高々$3k-2$ である [5]。こうして得られた系列を使って LSTM を学習させ、生成を行うと係り受け構造付きで文生成を行うことができる。

この手法の 1 つのメリットは、文の生成順序が係り受け構文木の ROOT からになることである。通常、構文木の根に近い部分ほど文の根幹を成す単語が配置され、根から遠い部分には修飾的、付加的な単語が配置される。よって、根から順に生成するほうが、文生成のモデルとして自然であると考えられる。

4.2 関連研究

係り受け文法ではなく、句構造文法を深さ優先探索を用いて直列化することは、論文 [4] でも行われている。一方、論文 [3] では、transition parser の操作系列を用いるという全く別の手法で係り受け構造を直列化している。この場合、単純な言語モデルと同様に、文の先頭の単語から順に生成される。

[1] 本研究ではこれらのラベルは使わないが、これらを使うような拡張も容易に考えられる。

[2] 係り受け解析を実行するプログラムとしては、Stanford Parser 3.4.1 を用いた。Ver. 3.5.2 以降では、デフォルトで Universal Dependencies が用いられるように変更されたので注意が必要である。

[3] 英語、日本語を含む大半の言語では、文の係り受け構造はほぼ常に非交差的であることが知られている。

[4] ROOT は常にただ一つの子を持つので、この部分は DFS の系列から省略する。

[5] ROOT を除くと、木の頂点数は k で、辺の本数は $k-1$ である。1 つの辺は up, down で最大 2 回通る。

- a black dog running through a grassy field.
- A black dog running towards the camera.
- A dog running fast with ears flying.
- A black dog runs through a field.
- Black dog running in swampy grass.

表 1: Flickr30K データセットにおける画像とキャプションの例

5 実験

今回、Flickr30K データセット [9] を用いて文生成モデルを学習させた。このデータセットでは 3 万枚の画像に対して、それぞれ 5 つのキャプション文が付けられている。このデータセットのうち 9 割を学習に、残りの 1 割を評価に用いた。表 1 に画像とそのキャプションの例を挙げた。

実験では、画像の特徴量からキャプションを生成するようにモデルを学習させた。モデルとしては、キャプション文をそのまま学習させた LSTM (Vanilla-LSTM) と、キャプション文を構文解析して DFS 法で直列化した系列を学習させた DFS-LSTM の 2 種類を実験した。

Vanilla-LSTM によってキャプションを学習させる場合、まず画像の特徴量を LSTM に入力し、そこから単語を生成するように学習した。ここで、LSTM は二段に重ね、特徴量は GoogLeNet から取り出した 1024 次元のベクトルを用いた。LSTM のセルの次元数は 300 とした。子の方法は、論文 [10] で述べられている方法とほぼ同じである。

DFS-LSTM ではまず、キャプション文を全て Stanford Parser によって係り受け解析し、それを DFS 法でシリアライズしたものを学習データとして用いた。それ以外の設定は Vanilla-LSTM と同様である。

5.1 文生成の結果

どちらのモデルの場合でも、画像特徴量を入力として、beam search によってキャプション文を生成した。DFS-LSTM の場合、生成された系列の<LD>と<LU>の個数が異なるなど、“おかしな” 系列が生成された場合、正しく文に変換できないことが懸念されるかもしれない。しかし、数千文を生成した範囲ではそのような問題は 1 回も発生しなかった。

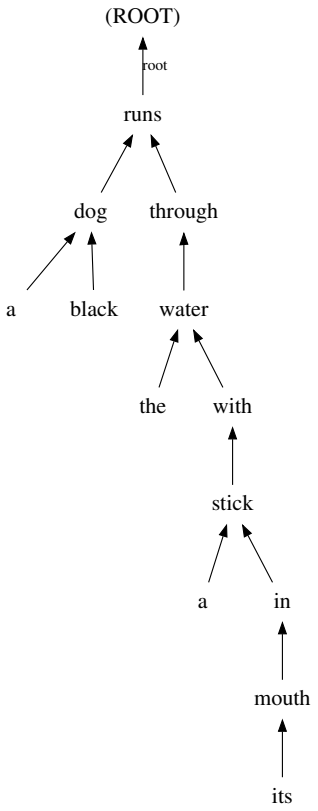

図 2: DFS-LSTM によって得られた文の係り受け構造の例

今回、表 1 にある画像は評価データであり、学習には用いなかった。これに対して、DFS-LSTM でキャプションを生成させると、‘runs’, <LD>, ‘dog’, <LD>, ‘a’, ‘black’, <LU>, <LU>, <RD>, ‘through’, <RD>, ‘water’, <LD>, ‘the’, <LU>, <RD>, ‘with’, <RD>, ‘stick’, <LD>, ‘a’, <LU>, <RD>, ‘in’, <RD>, ‘mouth’, <LD>, ‘its’, <LU>, <RU>, <RU>, <RU>, <RU>, <RU>, <RU>, <eos> が得られた。これは、英文としては “a black dog runs through the water with a stick in its mouth” であり、係り受け構造は、図 2 の通りである。これは、Stanford Parser でこの英文を係り受け解析した時の結果に一致している。この意味で、DFS-LSTM は構文構造を上手く学習できていると言える。

一方、表 1 の画像特徴量から Vanilla-LSTM モデルで文生成すると、“a black and white dog is running through the grass” という文が得られた。beam search で得られた上位 5 文はいずれも “xxx dog is running xxx” という形であった。

著者が目視で観察した範囲では、いずれのモデルでも、生成された文はほとんどすべて英文として自然なものであった。得られたキャプションの画像に対

する適切さは、Vanilla-LSTM モデルの方が高い傾向が見られた。これは、Vanilla-LSTM は単語の出現確率について最適化しているのに対し、DFS-LSTM では<LD>, <LU>, <RD>, <RU>に関しても最適化していることを考えると自然なことと考えられる。

5.2 構文構造を指定した文生成

DFS-LSTM モデルでは、ROOT の子から文を生成する。このことを使うと、構文的に異なった文を上手く生成することができる。

例として再び表 1 の画像を考える。名詞句の文では、名詞が ROOT の子になる。例えば、beam search を 'dog' から始めると、"a black and white dog running through the water with a stick in its mouth" のように、名詞句の形の文が得られる。同様に、'running', 'is' から beam search を行うと、

- "a black and white dog is running through the snow"
- "a black dog is in the water with a stick in its mouth"

のように、異なる構文のキャプションを得ることができる。現状では、DFS-LSTM のキャプション付モデルとしての性能が不十分であるため画像と生成文の内容にやや乖離があるが、性能が向上すれば、同一の内容を様々な構文で言い換えることができると期待できる。

6 おわりに

本稿では、係り受け構造つきで文生成を行うモデルを提案し、実際に生成が可能であることを確認した。このモデルは、複雑な係り受け構造を持つ文をより効率的にモデル化していることが期待されるが、その定量的な評価については今後の課題である。

DFS-LSTM モデルのもうひとつのメリットは、単語が係り受け構造と共に生成されることにある。本稿では、生成系列の先頭を指定することにより生成文の構文構造を変化させたが、より明示的に構文構造を指定して文生成することも考えられる。また、生成された文の形容詞句がどの名詞句を指しているのかについて曖昧性がないことを利用した知識抽出や、質問応答も今後の研究課題である。

参考文献

[1] D. Bahdanau, K. Cho and Y. Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, ICLR 2015

[2] K. Xu et al, Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, ICML 2015

[3] J. Buys and P. Blunsom, Generative Incremental Dependency Parsing with Neural Networks, ACL 2015

[4] O. Vinyals, T. Koo, and G. Hinton, Grammar as a Foreign Language, NIPS 2015

[5] T. Mikolov et al, Recurrent neural network based language model, INTERSPEECH, 2010

[6] K. Greff, LSTM: A Search Space Odyssey, arXiv preprint arXiv:1503.04069

[8] W. Zaremba, I. Sutskever, Learning to Execute, arXiv preprint arXiv:1410.4615

[7] M. Marneffe and C. Manning, Stanford Dependencies manual, 2008.

[9] P. Young, et al. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions, ACL 2014

[10] O. Vinyals, et al. Show and tell: A neural image caption generator, arXiv preprint arXiv:1411.4555