Recurrent Neural Network Grammers

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros,
Noah A. Smith

導入

Recurrent Neural Network Grammers (RNNGs)

- 文の確率的生成モデル
  - 単語や句の入れ子的・階層的構造を陽に表現
- 問題：構文解析，文生成
- 動機：Sequential な Recurrent Neural Networks (RNNs) は自然言語の潜在的な入れ子構造を考慮できていない
- 構成要素
  - 構文解析または文生成のアルゴリズム
  - NN による生成モデル

提案手法

RNNG の形式的な定義

$$RNNG := (N, \Sigma, \Theta)$$

$$\begin{cases} N : \text{非終端記号の有限集合} \\ \Sigma : \text{終端記号の有限集合} (N \cup \Sigma = \emptyset) \\ \Theta : \text{NN のパラメータ} \end{cases}$$

提案手法

構文解析のアルゴリズム

$$f : X \to Y$$

$$\begin{cases} x : \text{終端記号（単語）の列（入力）} \\ y : \text{構文木（出力）} \\ S : \text{スタック} \\ B : \text{入力バッファ} \end{cases}$$

- スタックの要素：終端記号，open または closed な非終端記号
- 入力バッファの要素：終端記号
- 初期状態

$$S = \emptyset$$

$$B = [T_1, \ldots, T_n]$$

提案手法

構文解析のアルゴリズム

- 遷移の制約
  - n: スタック内の open な非終端記号の数

遷移	制約
NT(X)	$B \neq \emptyset \wedge n < 100$
SHIFT	$B \neq \emptyset \wedge n \geq 1$
REDUCE	スタック内の一番上の要素が open な非終端記号でない $\wedge (n \geq 2$(1じゃないの？) $\vee B = \emptyset)$

提案手法

構文解析のアルゴリズム

Stack$_t$	**Buffer**$_t$	**Open NTs**$_t$	**Action**	**Stack**$_{t+1}$	**Buffer**$_{t+1}$	**Open NTs**$_{t+1}$
S	B	n	NT(X)	$S \mid (\mathrm{X}$	B	$n+1$
S	$x \mid B$	n	SHIFT	$S \mid x$	B	n
$S \mid (\mathrm{X} \mid \tau_1 \mid \ldots \mid \tau_\ell$	B	n	REDUCE	$S \mid (\mathrm{X}\ \tau_1\ \ldots\ \tau_\ell)$	B	$n-1$

Input: *The hungry cat meows .*

	Stack	**Buffer**	**Action**
0		*The* | *hungry* | *cat* | *meows* | .	NT(S)
1	(S	*The* | *hungry* | *cat* | *meows* | .	NT(NP)
2	(S | (NP	*The* | *hungry* | *cat* | *meows* | .	SHIFT
3	(S | (NP | *The*	*hungry* | *cat* | *meows* | .	SHIFT
4	(S | (NP | *The* | *hungry*	*cat* | *meows* | .	SHIFT
5	(S | (NP | *The* | *hungry* | *cat*	*meows* | .	REDUCE
6	(S | (NP *The hungry cat*)	*meows* | .	NT(VP)
7	(S | (NP *The hungry cat*) | (VP	*meows* | .	SHIFT
8	(S | (NP *The hungry cat*) | (VP *meows*	.	REDUCE
9	(S | (NP *The hungry cat*) | (VP *meows*)	.	SHIFT
10	(S | (NP *The hungry cat*) | (VP *meows*) | .		REDUCE
11	(S (NP *The hungry cat*) (VP *meows*) .)		

提案手法

文生成のアルゴリズム

$$f : X \to Y$$

$$\begin{cases} x : ? \\ y : \text{終端記号（単語）の列（出力）} \\ S : \text{スタック} \\ \mathbf{T:}\ \text{出力バッファ} \end{cases}$$

- スタックの要素：終端記号，open または closed な非終端記号
- 出力バッファの要素：終端記号
- 初期状態

$$S = \emptyset$$
$$T = \emptyset$$

提案手法

文生成のアルゴリズム

- 遷移の制約
  - n: スタック内の open な非終端記号の数

遷移	制約
GEN(X)	$n \geq 1$
REDUCE	スタック内の一番上の要素が open な非終端記号でない $\wedge\ n \geq 1$

提案手法

文生成のアルゴリズム

Stack$_t$	**Terms**$_t$	**Open NTs**$_t$	**Action**	**Stack**$_{t+1}$	**Terms**$_{t+1}$	**Open NTs**$_{t+1}$
S	T	n	NT(X)	$S \mid (\mathrm{X}$	T	$n+1$
S	T	n	GEN(x)	$S \mid x$	$T \mid x$	n
$S \mid (\mathrm{X} \mid \tau_1 \mid \ldots \mid \tau_\ell$	T	n	REDUCE	$S \mid (\mathrm{X}\ \tau_1\ \ldots\ \tau_\ell)$	T	$n-1$

	Stack	**Terminals**	**Action**
0			NT(S)
1	(S		NT(NP)
2	(S | (NP		GEN(*The*)
3	(S | (NP | *The*	*The*	GEN(*hungry*)
4	(S | (NP | *The* | *hungry*	*The* | *hungry*	GEN(*cat*)
5	(S | (NP | *The* | *hungry* | *cat*	*The* | *hungry* | *cat*	REDUCE
6	(S | (NP *The hungry cat*)	*The* | *hungry* | *cat*	NT(VP)
7	(S | (NP *The hungry cat*) | (VP	*The* | *hungry* | *cat*	GEN(*meows*)
8	(S | (NP *The hungry cat*) | (VP *meows*	*The* | *hungry* | *cat* | *meows*	REDUCE
9	(S | (NP *The hungry cat*) | (VP *meows*)	*The* | *hungry* | *cat* | *meows*	GEN(.)
10	(S | (NP *The hungry cat*) | (VP *meows*) | .	*The* | *hungry* | *cat* | *meows* | .	REDUCE
11	(S (NP *The hungry cat*) (VP *meows*) .)	*The* | *hungry* | *cat* | *meows* | .	

提案手法

生成モデル

- 単語列（x）と構文木（y）の結合確率

$$p(x,y) = \prod_{t=1}^{|a(x,y)|} p(a_t|a_{<t})$$

$$= \prod_{t=1}^{|a(x,y)|} \frac{\exp r_{a_t}^T u_t + b_{a_t}}{\sum_{a' \in A_G(T_t,S_t,n_t)} \exp r_{a'}^T u_t + b_{a'}}$$

$$u_t = \tanh(W[o_t; s_t; h_t] + c)$$

$$\begin{cases} u_t : \text{アルゴリズムの状態を表す埋め込み} \\ r_a : \text{各遷移の埋め込み（パラメータ）} \\ b_a : \text{各遷移のバイアス（パラメータ）} \end{cases}$$

提案手法

生成モデル

- u_t: アルゴリズムの状態を表す埋め込み

$$u_t = \tanh(W[o_t; s_t; h_t] + c)$$

$$\begin{cases} o_t : \text{出力バッファの状態を表す埋め込み} \\ s_t : \text{スタックの状態を表す埋め込み} \\ h_t : \text{遷移歴を表す埋め込み} \\ W, c : \text{パラメータ} \end{cases}$$