

Recurrent Neural Network Grammers

豊田工業大学 知能数理研究室 外山洋太

導入

Recurrent Neural Network Grammers (RNNGs)

- ▶ 著者 : Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, Noah A. Smith.
- ▶ 所属 : Carnegie Mellon University, Pompeu Fabra University, University of Washington
- ▶ 学会 : NAACL 2016

導入

Recurrent Neural Network Grammars (RNNGs)

- ▶ Recurrent Neural Network (RNN) による文法のモデル
 - ▶ 単語や句の入れ子的・階層的構造を陽に表現
- ▶ 構文解析または文生成のアルゴリズムを用いて学習・推定
- ▶ タスク：構文解析，言語モデル
- ▶ 関連研究
 - ▶ 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
 - Recursive NN の手法を取り入れる
 - ▶ 既存の NN による手法はボトムアップ型（左隅）構文解析
 - 生成に適したトッパダウン型のアルゴリズム

提案手法

RNNG の形式的な定義

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

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

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)	$S \mid (X$	B	$n + 1$
S	$x \mid B$	n	SHIFT	$S \mid x$	B	n
$S \mid (X \mid \tau_1 \mid \dots \mid \tau_\ell$	B	n	REDUCE	$S \mid (X \tau_1 \dots \tau_\ell)$	B	$n - 1$

Input: *The hungry cat meows .*

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

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)	$S \mid (X$	T	$n + 1$
S	T	n	GEN(x)	$S \mid x$	$T \mid x$	n
$S \mid (X \mid \tau_1 \mid \dots \mid \tau_\ell$	T	n	REDUCE	$S \mid (X \tau_1 \dots \tau_\ell)$	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

- ▶ 最大化: $p(X, Y; \Theta)$
- ▶ 単語列 (x) と構文木 (y) の結合確率

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

$$p(a_t | a_{<t}) = \frac{\exp(r_{a_t}^T u_t + b_{a_t})}{\sum_{a' \in A_G(T_t, S_t, n_t)} \exp(r_{a'}^T u_t + b_{a'})}$$

$$\begin{cases} a(x, y) : \text{単語列 } x \text{ と構文木 } y \text{ に対応する行動の列} \\ u_t : \text{アルゴリズムの状態を表す埋め込み} \\ r_a : \text{生成器の各行動の埋め込み (パラメータ)} \\ b_a : \text{生成器の各行動のバイアス (パラメータ)} \end{cases}$$

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

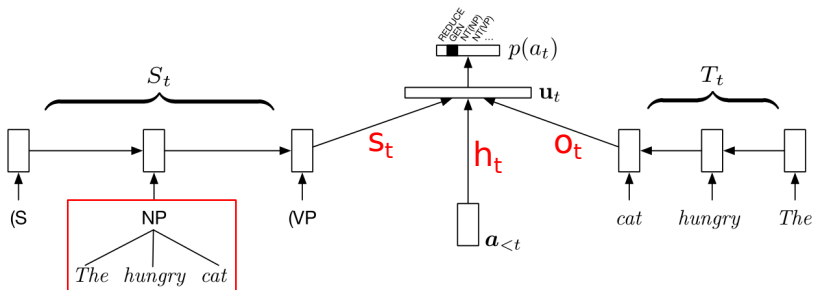
$$u_t = \tanh(W[o_t; s_t; h_t] + c)$$

$$\left\{ \begin{array}{l} o_t : \text{出力バッファの状態を表す埋め込み} \\ s_t : \text{スタックの状態を表す埋め込み} \\ h_t : \text{行動履歴を表す埋め込み} \\ W, c : \text{パラメータ} \end{array} \right.$$

提案手法

生成モデル

- ▶ スタック・出力バッファ・行動履歴内の要素の埋め込みをエンコード

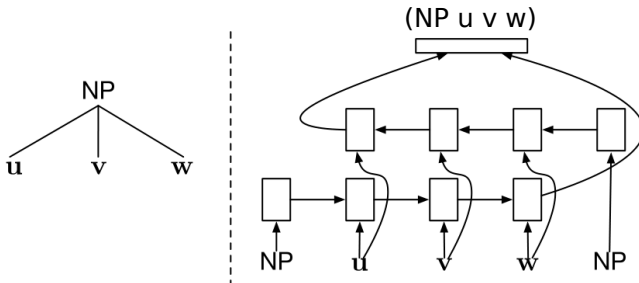


- ▶ スタック内の非終端記号の埋め込みは？

提案手法

生成モデル

- ▶ Syntactic Composition Function
 - ▶ REDUCE 時に要素の埋め込みからその非終端記号の埋め込みを生成



提案手法

識別モデル

- ▶ 最大化: $p(Y|X; \Theta)$
- ▶ 生成モデルの出力バッファ T を入力バッファ B に置き換え
- ▶ 単語列 x が与えられているため,
先ほどの $p(x, y)$ を $p(y|x)$ として学習

提案手法

生成モデルにおける 重点サンプリング

- ▶ 提案分布 $q(y|x)$: 識別モデルを利用
- ▶ 重要度重み : $w(x, y) = p(x, y)/q(y|x)$
- ▶ y を q からサンプリングしモンテカルロ法で計算
- ▶ 文生成

$$\begin{aligned} p(x) &= \sum_{y \in \mathcal{Y}(x)} p(x, y) \\ &= \sum_{y \in \mathcal{Y}(x)} q(y|x) w(x, y) \\ &= E_{q(x|y)} w(x, y) \end{aligned}$$

- ▶ 構文解析

$$\hat{y} = \operatorname{argmax}_{y \sim q(y|x)} p(x, y)$$

実験

実験設定

- ▶ データセット
 - ▶ Penn Treebank (英語)
 - ▶ Penn Chinese Treebank (中国語)
- ▶ タスク
 - ▶ 構文解析 (生成・識別モデル)
 - ▶ 言語モデル (生成モデル)

実験

Penn Treebank での F 値

Table 2: Parsing results on PTB §23 (D=discriminative, G=generative, S=semisupervised).

Model	type	F ₁
Henderson (2004)	D	89.4
Socher et al. (2013a)	D	90.4
Zhu et al. (2013)	D	90.4
Vinyals et al. (2015) – WSJ only	D	90.5
Petrov and Klein (2007)	G	90.1
Bod (2003)	G	90.7
Shindo et al. (2012) – single	G	91.1
Shindo et al. (2012) – ensemble	G	92.4
Zhu et al. (2013)	S	91.3
McClosky et al. (2006)	S	92.1
Vinyals et al. (2015) – single	S	92.5
Vinyals et al. (2015) – ensemble	S	92.8
Discriminative, $q(\mathbf{y} \mid \mathbf{x})$	D	89.8
Generative, $\hat{p}(\mathbf{y} \mid \mathbf{x})$	G	92.4

実験

Penn Chinese Treebank での F 値

Table 3: Parsing results on CTB 5.1.

Model	type	F ₁
Zhu et al. (2013)	D	82.6
Wang et al. (2015)	D	83.2
Huang and Harper (2009)	D	84.2
Charniak (2000)	G	80.8
Bikel (2004)	G	80.6
Petrov and Klein (2007)	G	83.3
Zhu et al. (2013)	S	85.6
Wang and Xue (2014)	S	86.3
Wang et al. (2015)	S	86.6
Discriminative, $q(\mathbf{y} \mid \mathbf{x})$	D	80.7
Generative, $\hat{p}(\mathbf{y} \mid \mathbf{x})$	G	82.7

実験

言語モデルの perplexity

Table 4: Language model perplexity results.

Model	test ppl (PTB)	test ppl (CTB)
IKN 5-gram	169.3	255.2
LSTM LM	113.4	207.3
RNNG	102.4	171.9

まとめ

- ▶ RNN による文法のモデルを提案
- ▶ 遷移ベースの構文解析・文生成アルゴリズムで学習
- ▶ 生成モデルと識別モデルの 2 種類
- ▶ 特徴量の設計や Treebank データの変換が要らない

感想

- ▶ 構文解析はよく知らなかったので勉強になった
- ▶ 文法を NN でモデル化しようという発想が面白いと思った