PaperDaily -- 2017.09.20

Dependency Parsing as Head Selection

Xingxing Zhang, Jianpeng Cheng and **Mirella Lapata**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB
{x.zhang,jianpeng.cheng}@ed.ac.uk, mlap@inf.ed.ac.uk

Abstract

- They formalize dependency parsing as the problem of independently selecting the head of each word in a sentence.
- DENSE (as shorthand for Dependency Neural Selection) model produces a distribution over possible heads for each word using features obtained from a bidirectional recurrent neural network.
- Without enforcing structural constraints during training, non-tree outputs can be adjusted with a maximum spanning tree algorithm.
- Their parsers are on par with the state of the art on four languages (English, Chi-nese, Czech, and German).

Introduction

In this work, they propose a simple neural network-based model which learns to select the head for each word in a sentence without enforcing tree structured output. DENSE model employs bidirectional recurrent neural networks to learn feature representations for words in a sentence. These features are subsequently used to predict the head of each word.

Although there is nothing inherent in the model to enforce tree-structured output, when tested on an English dataset, it is able to generate trees for 95% of the sentences, 87% of which are projective. The remaining non-tree (or non-projective) outputs are post-processed with the Chu-Liu-Edmond (or Eisner) algorithm.

Related Work

- Graph-based Parsing
  - Chu-Liu-Edmonds algorithm
  - Eisner algorithm
- Transition-based Parsing
  - Non-local training methods (greedy error propagation)
- Neural Network-based Features
  - feed forward neural network
  - stack long short-term memory net-works
  - bidirectional LSTMs

Dependency Parsing as Head Selection

Input: `a sentence of length N`

Outputs: `N <head, dependent> arcs`

`unlabeled dependencies` $\longrightarrow$ `labeled setting`

Word Representation

$$\mathbf{h}_i^F, \mathbf{c}_i^F = \mathbf{LSTM}^F(\mathbf{x}_i, \mathbf{h}_{i-1}^F, \mathbf{c}_{i-1}^F)$$

$$\mathbf{h}_i^B, \mathbf{c}_i^B = \mathbf{LSTM}^B(\mathbf{x}_i, \mathbf{h}_{i+1}^B, \mathbf{c}_{i+1}^B)$$

$$\mathbf{x}_i = [\mathbf{W}_e \cdot e(w_i); \mathbf{W}_t \cdot e(t_i)]$$

$$\mathbf{a}_i = [\mathbf{h}_i^F; \mathbf{h}_i^B] \quad i \in [0, N]$$

Head Selection

Given a sentence $S = (w_0, w_1, \cdots, w_N)$, we aim to find for each word $w_i \in \{w_1, w_2, \cdots w_n\}$ the most probable head $w_j \in \{w_0, w_1, \cdots w_N\}$

$$P_{head}(w_j|w_i, S) = \frac{\exp(g(\mathbf{a}_j, \mathbf{a}_i))}{\sum_{k=0}^{N} \exp(g(\mathbf{a}_k, \mathbf{a}_i))}$$

$$g(\mathbf{a}_j, \mathbf{a}_i) = \mathbf{v}_a^\top \cdot \tanh(\mathbf{U}_a \cdot \mathbf{a}_j + \mathbf{W}_a \cdot \mathbf{a}_i)$$

$$J(\theta) = -\frac{1}{|\mathcal{T}|} \sum_{S \in \mathcal{T}} \sum_{i=1}^{N_S} \log P_{head}(h(w_i)|w_i, S)$$

$$w_j = \underset{w_j : j \in [0, N_S]}{\arg\max} \; P_{head}(w_j|w_i, S)$$

predict arc label <w_j, w_i> : $[a_i; a_j; x_i; x_j]$

Maximum Spanning Tree Algorithms

`Chu-Liu-Edmonds algorithm` $\rightarrow$ `non-projective trees`

`Eisner (1996) algorithm` $\rightarrow$ `projective trees`

We view a sentence $S = (w_0 = ROOT, w_1, \cdots, w_N)$ as a graph $G_S =<V_S, E_S>$

$$
\begin{aligned}
V_S &= \{w_0 = \text{ROOT}, w_1, \ldots, w_N\} \\
E_S &= \{\langle i, j\rangle : i \neq j, \langle i, j\rangle \in [0, N] \times [1, N]\} \\
s(i, j) &= P_{head}(w_i | w_j, S) \quad \langle i, j\rangle \in E_S
\end{aligned}
$$

Non-projective Parsing

Only run the Chu-Liu-Edmonds algorithm through graphs with cycles

Projective Parsing

The time complexity of the Eisner algorithm is $O(N^3)$, while checking if a tree is projective can be done reasonably faster, with a $O(N \log N)$ algorithm.

Experiments

Datasets

`English Penn Treebank (PTB)` `Chinese Treebank 5.1 (CTB)`

`CoNLL 2006 multilingual dependency parsing shared task`

Parser	Dev UAS	Dev LAS	Test UAS	Test LAS
Z&N11	—	—	86.00	84.40
Z&M14	—	—	**87.96**	**86.34**
C&M14	84.00	82.40	83.90	82.40
Dyer15	87.20	**85.90**	87.20	85.70
K&G16 *graph*	—	—	86.60	85.10
K&G16 *trans*	—	—	87.60	86.10
DENSE-Pei	82.50	80.74	82.38	80.55
DENSE-Pei+E	83.40	81.63	83.46	81.65
DENSE	87.27	85.73	87.63	85.94
DENSE+E	**87.35**	85.85	87.84	86.15

Table 3: Results on Chinese dataset (CTB). +E: we post-process non-projective outputs with the Eisner algorithm.

Parser	PTB Dev	PTB Test	CTB Dev	CTB Test
C&M14	43.35	40.93	32.75	32.20
Dyer15	51.94	50.70	**39.72**	**37.23**
DENSE	51.24	49.34	34.74	33.66
DENSE+E	**52.47**	**50.79**	36.49	35.13

Table 4: UEM results on PTB and CTB.

Experiments

Parser	Czech		German	
	UAS	LAS	UAS	LAS
MST-1st	86.18	—	89.54	—
MST-2nd	87.30	—	90.14	—
Turbo-1st	87.66	—	90.52	—
Turbo-3rd	90.32	—	**92.41**	—
RBG-1st	87.90	—	90.24	—
RBG-3rd	**90.50**	—	91.97	—
DENSE-Pei	86.00	77.92	89.42	86.48
DENSE-Pei+CLE	86.52	78.42	89.52	86.58
DENSE	89.60	81.70	92.15	89.58
DENSE+CLE	89.68	81.72	92.19	89.60

Table 5: Non-projective results on the CoNLL 2006 dataset. +CLE: we post-process non-tree outputs with the Chu-Liu-Edmonds algorithm.

Experiments

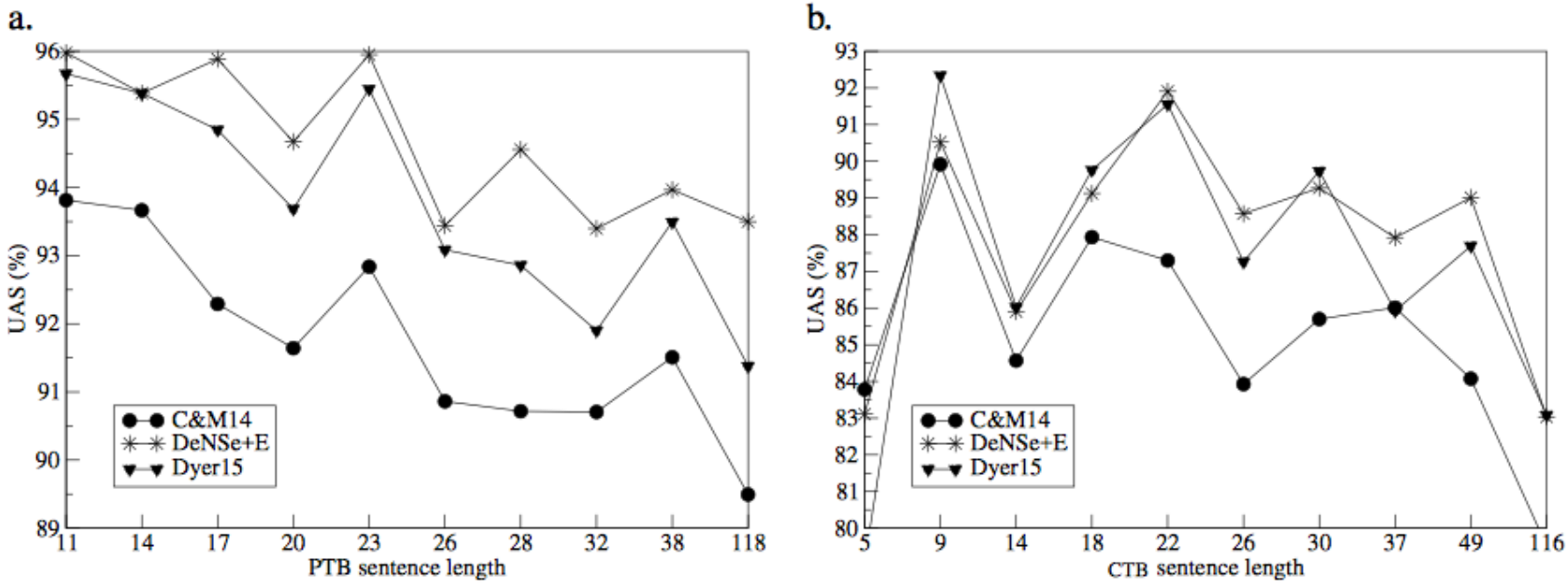

Figure 2: UAS against sentence length on PTB and CTB (development set). Sentences are sorted by length in ascending order and divided equally into 10 bins. The horizontal axis is the length of the last sentence in each bin.

Experiments

Dataset	#Sent	Before MST		After MST	
		Tree	Proj	Tree	Proj
PTB	1,700	95.1	86.6	100.0	100.0
CTB	803	87.0	73.1	100.0	100.0
Czech	374	87.7	65.5	100.0	72.7
German	367	96.7	67.3	100.0	68.1

Table 6: Percentage of trees and projective trees on the development set before and after DENSE uses a MST algorithm. On PTB and CTB, we use the Eisner algorithm and on Czech and German, we use the Chu-Liu-Edmonds algorithm.