Parsing and Machine learning

http://www-rohan.sdsu.edu/~gawron/aisem

Neural transition-based parsed

Jean Mark Gawron

San Diego State University, Department of Linguistics

2010-08-19

Overview

Dyer et al.s 2015 LSTM stack parser

Features

Dyer et al. (2015), Ballesteros et al. (2017)

1. Three aspects of parser state represented as "LSTM-stacks"
   1. Buffer
   2. Action history
   3. Stack
2. Recursive approach to modeling parser states; Recurrent Neural Network (RNN).

What's recursive about it?

$$\mathbf{h}_t = \sigma(\mathbf{W}_x x_t + \mathbf{W}_x h_{t-1} + \mathbf{d})$$

$$\mathbf{W}_x \in \mathbb{R}^{d_{out} \times d_{in}};\ \mathbf{W}_h \in \mathbb{R}^{d_{out} \times d_{out}};\ \mathbf{d}, \mathbf{h}_t \in \mathbb{R}^{d_{out}};\ \mathbf{x}_t \in \mathbb{R}^{d_{in}}$$

1. Parameters: $\mathbf{W}_x$, $\mathbf{W}_h$, $\mathbf{d}$
2. $\mathbf{h}_t$ is a "hidden" state that is typically transformed to create an output that is measured by a "loss" function.
3. The gradient (slope vector) for this loss is used to provide a "direction" for updating our learning parameters (backpropagation).

Vanishing gradient problem

Fundamental problem in sequence modeling

The gradient for the loss tends to "vanish" (go to zero) [or sometimes blow up] as "we trace it back to earlier iterates in a long sequence." (Ballesteros et al. 2017:324).

LSTMs (Long Short-Term Memory) to the rescue (**?**): Introduces the idea of a **memory cell** (actually a fairly complex little subnetwork), which learns what information to remember and what informatiomn to forget about long sequences.

$$\mathbf{h}_t = f(\mathbf{W}_x x_t + \mathbf{W}_x h_{t-1} + \mathbf{W}_{ic} c_{t-1} + \mathbf{d})$$

Stacks in parser represented as LSTM

Three "Stacks" represented by LSTMs

1. Stack of partially constructed dependency trees
2. The word buffer
3. Action history

Parsing at time t

Output is a parser action

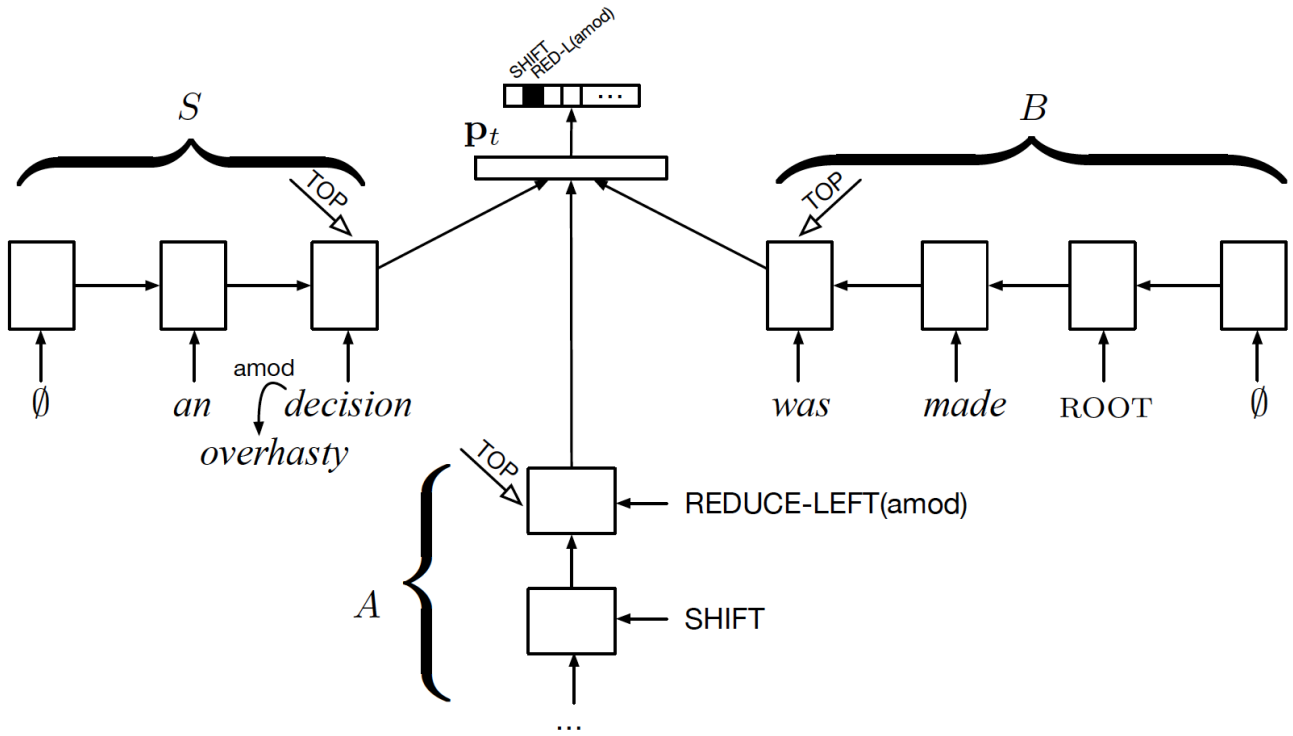

Figure 2
Parser state computation encountered while parsing the sentence *an overhasty decision was made*. Here S designates the stack of partially constructed dependency subtrees and its LSTM encoding; B is the buffer of words remaining to be processed and its LSTM encoding; and A is the stack representing the history of actions taken by the parser. These are linearly transformed, passed through a rectified linear unit nonlinearity to produce the parser state embedding $\mathbf{p}_t$. An affine transformation of this embedding is passed to a softmax layer to give a distribution over parsing decisions that can be taken.

Bibliography

Ballesteros, Miguel, Chris Dyer, Yoav Goldberg, and Noah A Smith. 2017.
Greedy transition-based dependency parsing with stack lstms.
Computational Linguistics 43(2):311–347.

Dyer, Chris, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015.
Transition-based dependency parsing with stack long short-term memory.
In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 334–343, Beijing, China, July. Association for Computational Linguistics.