Transforming Dependencies into Phrase Structures

Lingpeng Kong  Alexander M. Rush  Noah Smith

Contributions

• A phrase-structure parser (PAD) achieves 0.4% higher f-score on the Penn Treebank and ~7x faster than the Berkeley parser, without reranking or semisupervised training.

• An linear observable time algorithm for transforming dependency parse trees into phrase-structure parse trees.

Transformation

Phrase-structures —> Dependencies

Phrase-structures —< Dependencies

Transformation is deterministic.

Algorithm

Rules:
For all \( h, m \in \mathcal{R}(h) \), rule \( A \to \beta_1^* \beta_2 \), and \( i \in \{ i', i'' \} : m' \in \mathcal{L}(h) \cup \{ h \} \),

\[
(i, \mu_{e(m)} - 1, h, \beta_1) \quad (m_{e(m'), \mu_{e(m' + 1)}} m, \beta_2) \\
(i, m_{e(m)}, h, A)
\]

Premise:

\[
(i, i, i, A) \quad \forall i \in \{ 1 \ldots n \}, A \in \mathcal{N}
\]

Goal:

\[
(i, 1, m, r) \quad \text{for any} \ m \in \mathcal{R}(0)
\]

• Hand-written rules — Need to make various decisions. [Xia and Palmer, 2001; Xia et al., 2009; Collins et al., 1999]

• Our approach — data-driven algorithm using the structured prediction framework

• Transformation is ambiguous.

Learning

Objective

• Parameters are estimated using a structural support vector machine [Taskar et al., 2004]

• \( \Delta(y, y') \) is a hamming loss where \( y \) is an indicator for production rules firing over pairs of adjacent spans

Features

• arc-factored features [McDonald (2006)]

• span features used in the X-bar-style parser of [Hall et al. (2014)]

Experiments

• The effect of d-parsing accuracy (PTB §22) on PAD

• The runtime of our transforming algorithm

Accuracy and Speed on the English Penn Treebank (PTB) and the Chinese Penn Treebank (CTB)