# Transforming Dependencies into Phrase Structures

Lingpeng Kong  Alexander M. Rush  Noah Smith

Carnegie Mellon  HARVARD School of Engineering and Applied Sciences

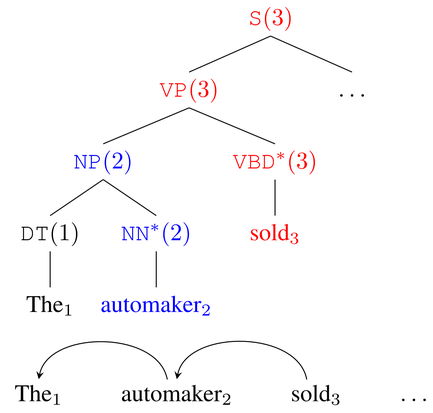
## Contributions

• **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.

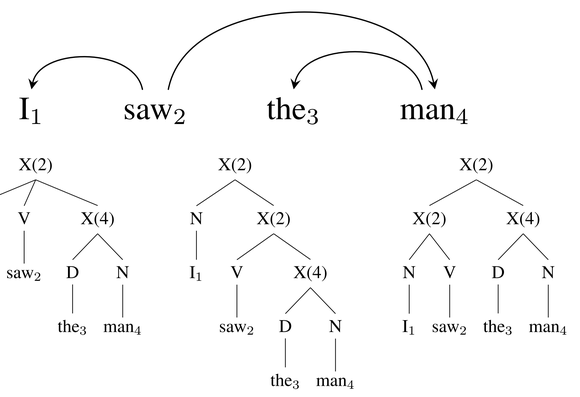
• An **linear observable time** algorithm for **transforming dependency parse trees** into **phrase-structure parse trees**.

## Transformation

Phrase-structures → Dependencies




Phrase-structures ← Dependencies




• Various **Head-rules** [Collins, 2003; De Marneffe and Manning, 2008; Yamada and Matsumoto, 2003; Johansson and Nugues, 2007].

• Transformation is **deterministic**.

• 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**.

## Algorithm

**Rules:**

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

$$\frac{(\langle i, m_{\Leftarrow} - 1\rangle, h, \beta_1) \quad (\langle m_{\Leftarrow}, m_{\Rightarrow}\rangle, m, \beta_2)}{(\langle i, m_{\Rightarrow}\rangle, h, A)}$$

For all $h, m \in \mathcal{L}(h)$, rule $A \rightarrow \beta_1 \ \beta_2^*$, and $j \in \{m'_{\Rightarrow} : m' \in \mathcal{R}(h)\} \cup \{h\}$,

$$\frac{(\langle m_{\Leftarrow}, m_{\Rightarrow}\rangle, m, \beta_1) \quad (\langle m_{\Rightarrow} + 1, j\rangle, h, \beta_2)}{(\langle m_{\Leftarrow}, j\rangle, h, A)}$$

**Premise:**

$$(\langle i, i\rangle, i, A) \quad \forall i \in \{1 \dots n\}, A \in \mathcal{N}$$

**Goal:**

$$(\langle 1, n\rangle, m, r) \text{ for any } m \in \mathcal{R}(0)$$

• Note: Binarization of the phrase strcutre tree should be done **with respected to the head**, so that each use of a rule implies a dependency arc.

## Learning

$$\min_{\theta} \sum_{i=1}^{D} \ell(x^i, d^i, y^i, \theta) + \lambda ||\theta||_1$$

$$\ell(x, d, y, \theta) = -s(y) + \max_{y' \in \mathcal{Y}(x,d)} (s(y') + \Delta(y, y'))$$

For a production $\frac{(\langle i,k\rangle, m, \beta_1) \quad (\langle k+1,j\rangle, h, \beta_2)}{(\langle i,j\rangle, h, A)}$

| Nonterm Features |
|---|
| $(A, \beta_1)$ $(A, \beta_1, \text{tag}(m))$ |
| $(A, \beta_2)$ $(A, \beta_2, \text{tag}(h))$ |

| Span Features |
|---|
| $(\text{rule}, x_i)$ $(\text{rule}, x_{i-1})$ |
| $(\text{rule}, x_j)$ $(\text{rule}, x_{j+1})$ |
| $(\text{rule}, x_k)$ $(\text{rule}, x_{k+1})$ |
| $(\text{rule}, \text{bin}(j-i))$ |

| Rule Features |
|---|
| $(\text{rule})$ |
| $(\text{rule}, x_h, \text{tag}(m))$ |
| $(\text{rule}, \text{tag}(h), x_m)$ |
| $(\text{rule}, \text{tag}(h), \text{tag}(m))$ |
| $(\text{rule}, x_h)$ |
| $(\text{rule}, \text{tag}(h))$ |
| $(\text{rule}, x_m)$ |
| $(\text{rule}, \text{tag}(m))$ |

### 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

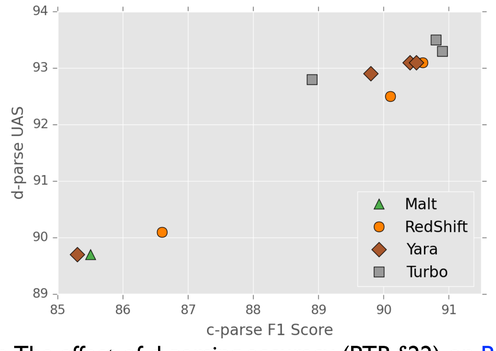
• optimized using AdaGrad of [Duchi et al., 2011]

### Features

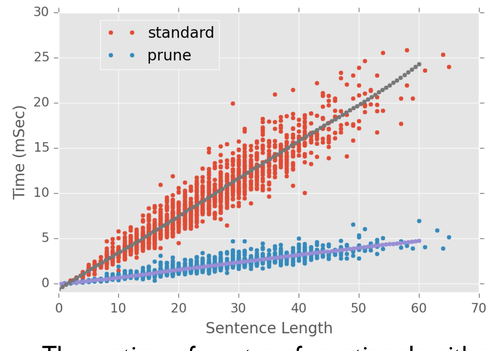
• arc-factored features [McDonald (2006)]

• 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 transformating algorithm

| Model | PTB 23 F1 | PTB 23 Sent./s. |
|---|---|---|
| Charniak (2000) | 89.5 | - |
| Stanford PCFG (2003) | 85.5 | 5.3 |
| Petrov (2007) | 90.1 | 8.6 |
| Zhu (2013) | 90.3 | 39.0 |
| Carreras (008) | 91.1 | - |
| CJ Reranking (2005) | 91.5 | 4.3 |
| Stanford RNN (2013) | 90.0 | 2.8 |
| PAD | 90.6 | 34.3 |
| PAD (Pruned) | 90.5 | 58.6 |

| Model | CTB 5 F1 |
|---|---|
| Charniak (2000) | 80.8 |
| Bikel (2004) | 80.6 |
| Petrov (2007) | 83.3 |
| Zhu (2013) | 83.2 |
| PAD | 82.4 |

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