

Multilingual Dependency Analysis with a Two-Stage Discriminative Parser



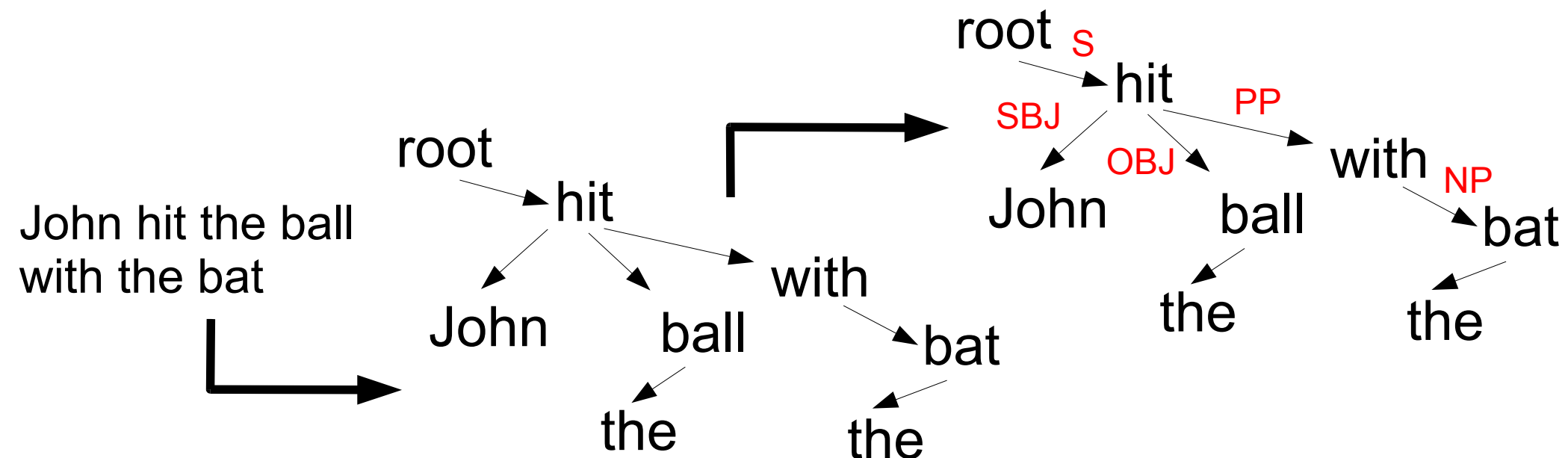
R. McDonald and K. Lerman and F. Pereira
Dept. of Computer and Information Science
University of Pennsylvania

Conference on Natural Language Learning 2006
Shared Task on Dependency Parsing
June 9th, 2006
Brooklyn, New York



Labeled Dependency Parsing

- **Two-stage:** unlabeled parsing + labeling
 - Features can be over entire dependency graph
 - Quick to train and test (no multiplicative label factor)
 - Error propagation



Discriminative Learning

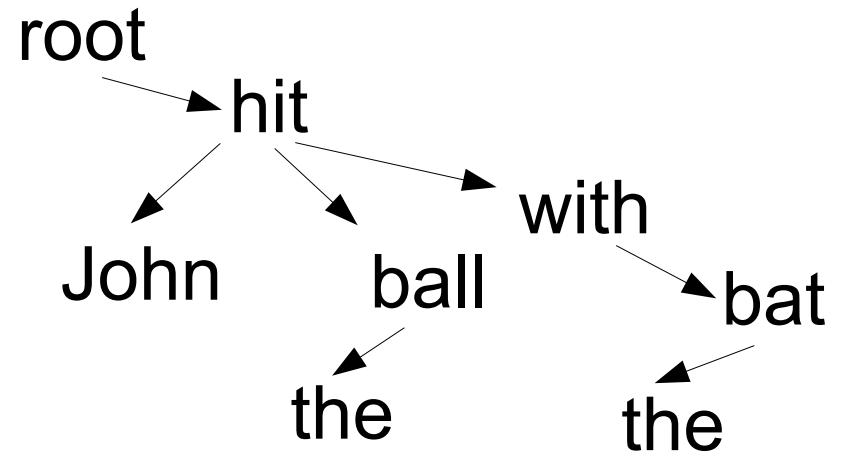
- All models are linear score classifiers
 - i.e., in $\text{score}(\dots) = \mathbf{w} \bullet \mathbf{f}(\dots)$
 - $\mathbf{f}(\dots)$ is a feature representation (defined by us)
 - \mathbf{w} is a corresponding weight vector
- Need to learn the weight vector \mathbf{w}
- Margin Infused Relaxed Algorithm (MIRA)
 - Online large-margin learner (Crammer et al. '03, '06)
 - Used in dependency parsing and sequence analysis (McDonald et al. '05 and '06)
 - Requires only inference and QP solver
 - Quick to train and highly accurate



STAGE 1

Unlabeled Parsing

John hit the ball with the bat



Maximum Spanning Tree Parsing

(McDonald, Pereira, Ribarov and Hajic '05)

- Let $\mathbf{x} = x_1 \dots x_n$ be a sentence
- Let \mathbf{y} be a dependency tree
- Let $(i,j) \in \mathbf{y}$ indicate an edge from x_i to x_j
- Let $\text{score}(\mathbf{x}, \mathbf{y})$ be the score of tree \mathbf{y} for \mathbf{x}
- Factor dependency tree score by edges

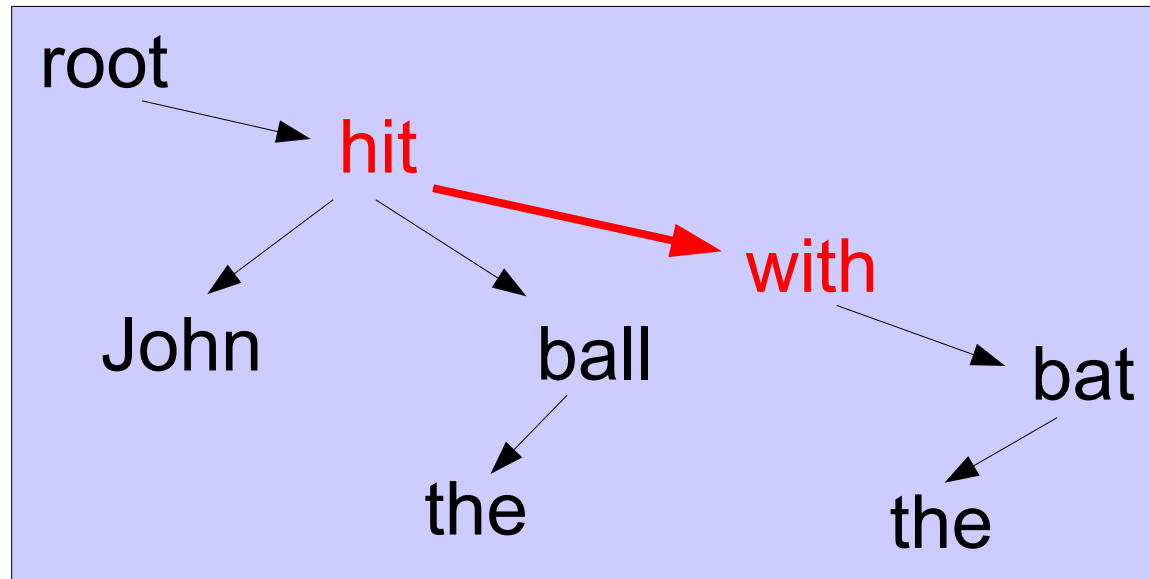
$$\text{score}(\mathbf{x}, \mathbf{y}) = \sum_{(i,j) \in \mathbf{y}} \text{score}(i,j)$$

- First-order: scores are relative to a single edge



Dependency Parsing: First-Order Tree Factorization

- For example:



$$\begin{aligned} \text{score}(\mathbf{x}, \mathbf{y}) = & \text{score}(\text{root}, \text{hit}) + \text{score}(\text{hit}, \text{John}) \\ & + \text{score}(\text{hit}, \text{ball}) + \text{score}(\text{hit}, \text{with}) \\ & + \text{score}(\text{ball}, \text{the}) + \text{score}(\text{with}, \text{bat}) \\ & + \text{score}(\text{bat}, \text{the}) \end{aligned}$$

Dependency Parsing: First-Order Tree Factorization

- Define the score of an edge as:

$$\text{score}(i,j) = \mathbf{w} \cdot \mathbf{f}(i,j)$$

$$\text{score}(\mathbf{x}, \mathbf{y}) = \mathbf{w} \cdot \sum_{(i,j) \in \mathbf{y}} \mathbf{f}(i,j)$$

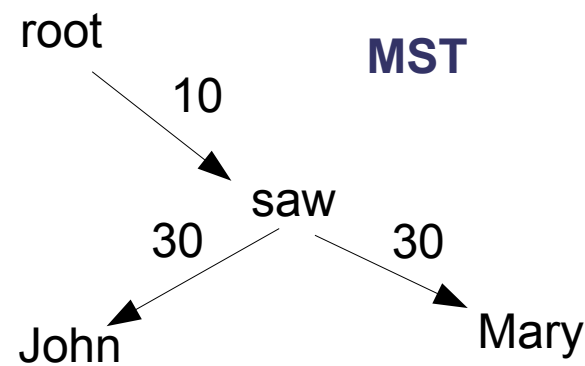
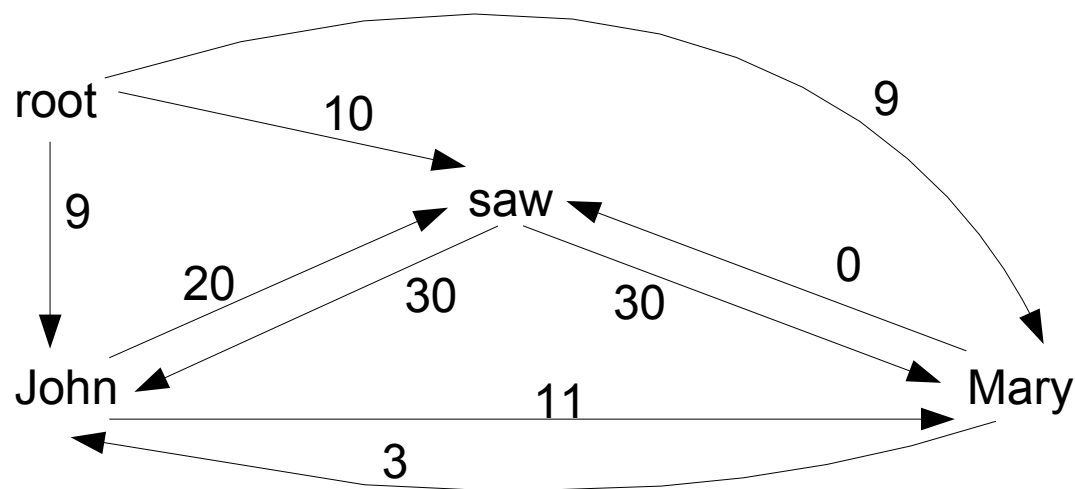
- Question: Given input \mathbf{x} can we find

$$\mathbf{y} = \arg \max_{\mathbf{y}} \text{score}(\mathbf{x}, \mathbf{y}) \quad \textit{Inference}$$

- Assuming we have defined $\mathbf{f}(i,j)$ (later)
- Also assuming we have learned \mathbf{w}
- Edge based factorization sounds familiar ...

Dependency Parsing as Maximum Spanning Trees (MST)

- Example $x = \textit{John saw Mary}$



- Finding the best (projective) dependency tree is equivalent to finding the (projective) MST.

Dependency Parsing as MSTs

- Projective algorithm: **Eisner '96**
 - Bottom-up chart parsing (dynamic programming)
 - Inference is $O(n^3)$
- Non-projective algorithm: **Chu-Liu-Edmonds**
 - Greedy recursive algorithm
 - Inference
 - Simple implementation $O(n^3)$
 - $O(n^2)$ implementation possible (Tarjan '77)

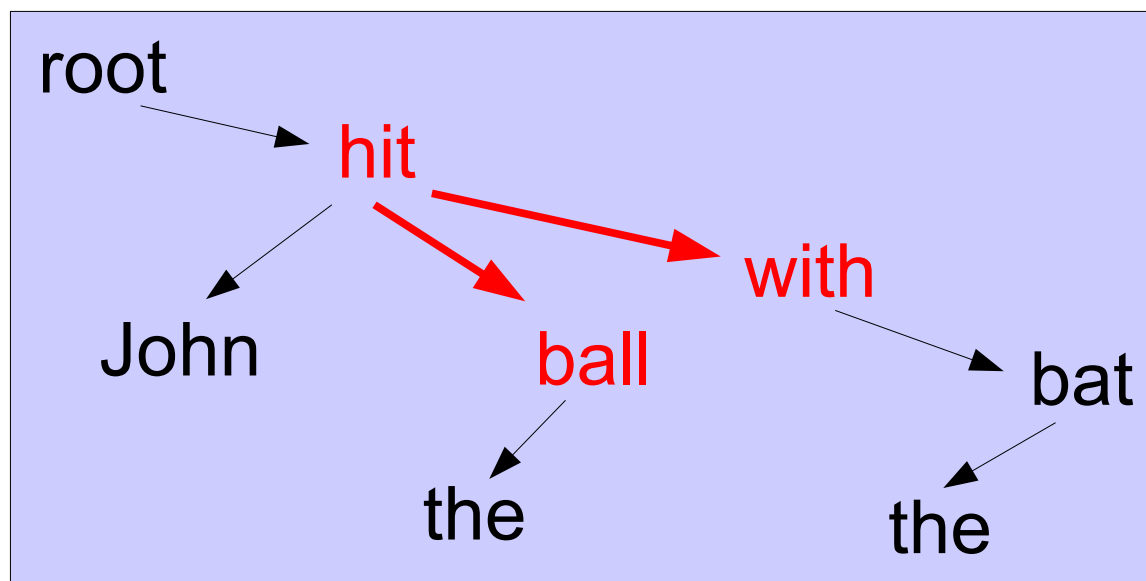


Second-order MST Parsing

Can we model scores over pairs of edges?

e.g. $\text{score}(\text{hit}, \text{ball}, \text{with})$

$$\text{score}(\mathbf{x}, \mathbf{y}) = \mathbf{w} \cdot \sum_{(i,k,j) \in \mathbf{y}} \mathbf{f}(i,k,j)$$



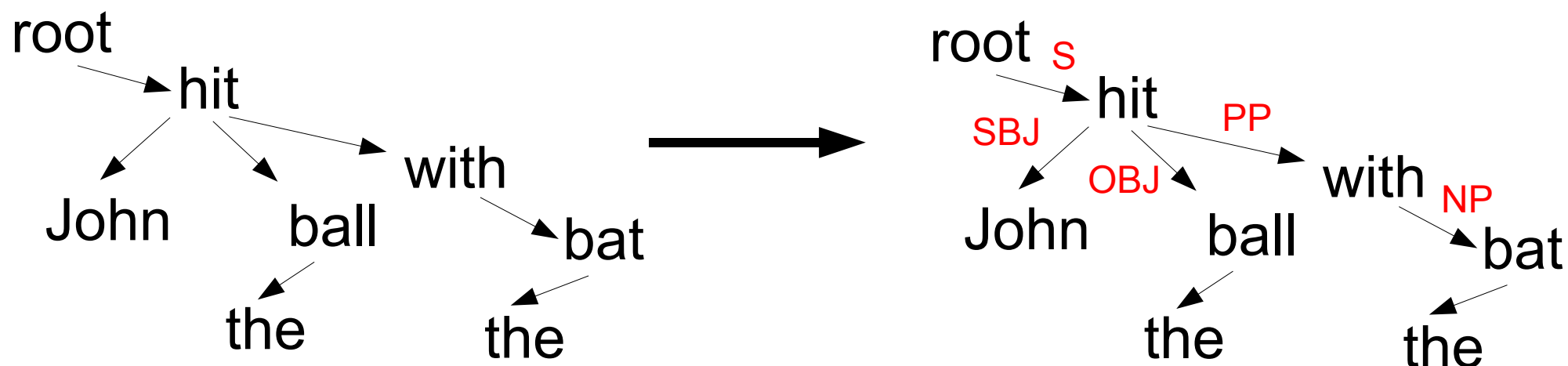
- Inference in projective case is still tractable!!
- However, non-projective case is NP-hard
 - Can use simple approximations (similar to Foth et al. '00)
 - See McDonald and Pereira '06 for details

Feature Set

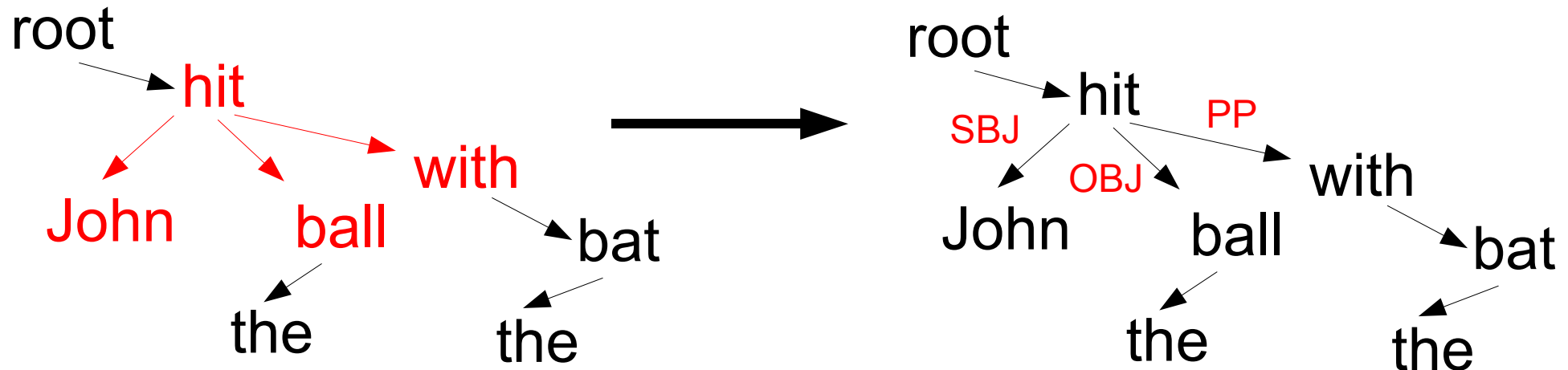
- First-Order features, $\mathbf{f}(i,j)$
 - Word, POS and morphological identities for x_i and x_j
 - POS of x_i and x_j and POS of words in-between
 - POS of x_i and x_j and POS of context words
 - Conjoined with direction of attachment & distance
- Second Order features, $\mathbf{f}(i,k,j)$
 - POS of x_i and x_k and x_j
 - POS of x_k and x_j
 - Word identities of x_k and x_j

STAGE 2

Edge Label Classification



Edge Label Classification



- Consider adjacent edges $\mathbf{e} = e_1, \dots, e_m$
 - Let $l = l_1, \dots, l_m$ be a labeling for \mathbf{e}
 - Inference: $l = \arg \max_l \text{score}(l, \mathbf{e}, \mathbf{x}, \mathbf{y}) = \mathbf{w} \bullet \mathbf{f}(l, \mathbf{e}, \mathbf{x}, \mathbf{y})$
- Label edges using standard sequence taggers
 - First-order Markov factorization plus Viterbi
- Models correlations between adjacent edges (SBJ vs. OBJ)

Edge Label Features (sample)

- **Edge Features:**

- Word/POS/morphological feature identity of the head and the dependent.
- Attachment direction.

- **Sibling Features:**

- Word/POS/morphological feature identity of the modifier's nearest siblings
- Do any of the modifier's siblings share its POS?

- **Context Features:**

- POS tag of each intervening word between head and modifier.
- Do any of the words between the head and the modifier have a different head?

- **Non-local:**

- How many children does the modifier have?
- What morphological features do the grandhead and the modifier have identical values?



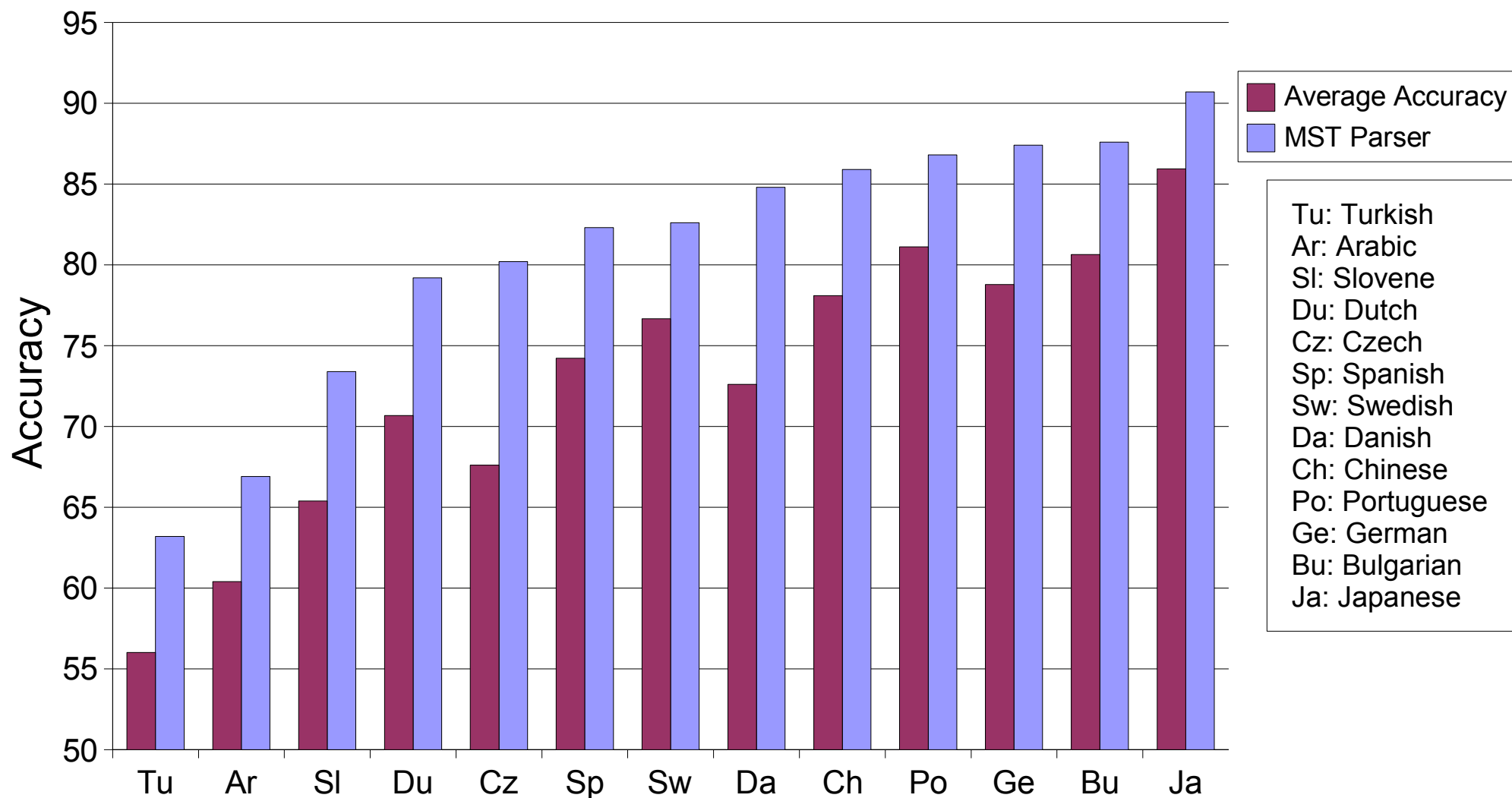
ON TO THE ...

Experiments



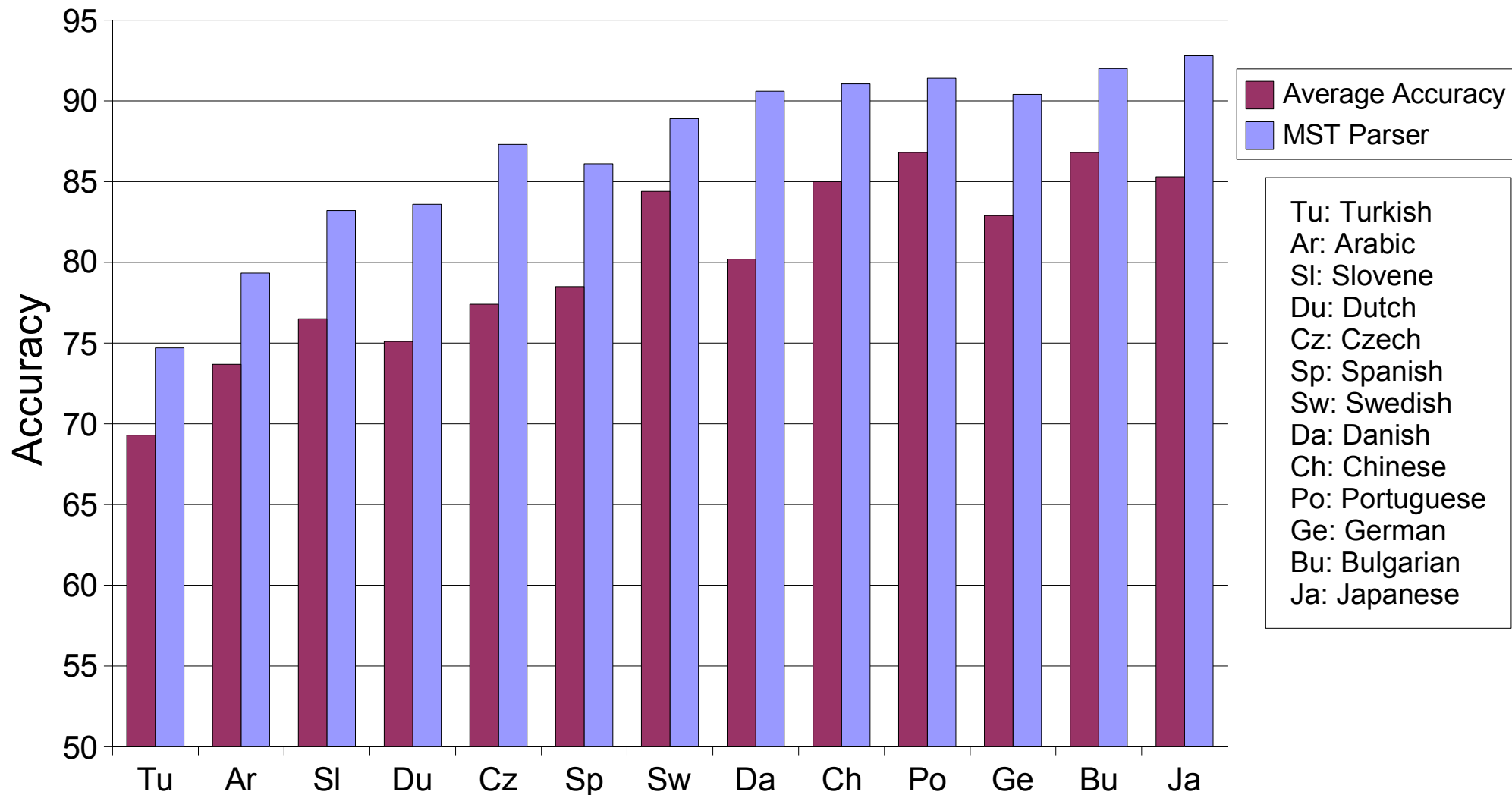
Experimental Results

Labeled Dependency Accuracy



Experimental Results

Unlabeled Dependency Accuracy

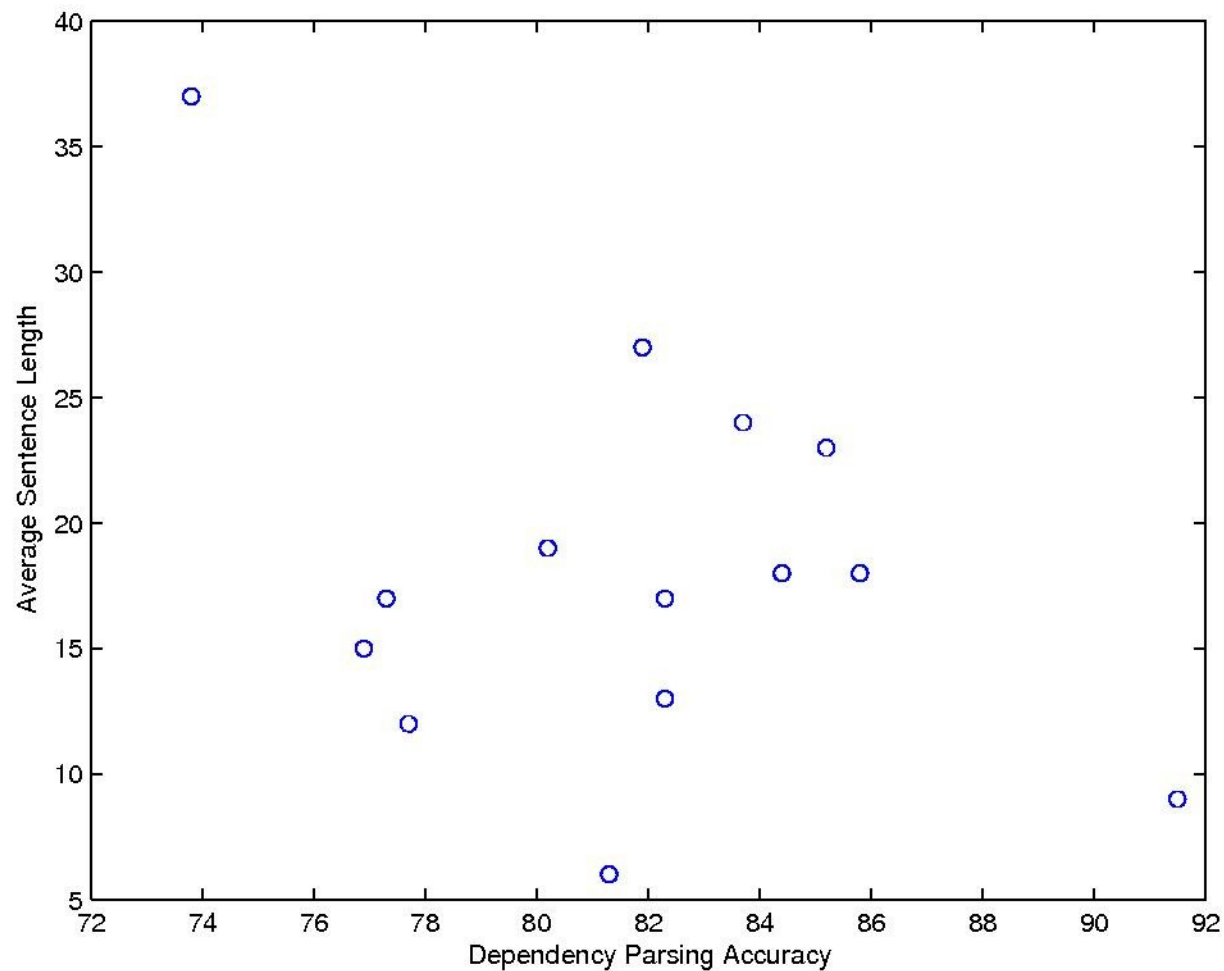


Performance Variability

- Turkish: **63/74%** vs. Japanese: **90/92%**
- What makes one language harder to parse than another?
 - Average sentence length
 - Unique tokens in data set (data set homogeneity)
 - Unseen test set tokens (i.i.d. assumptions, sparsity)
- Other properties harder to measure
 - Quality of annotations, head rules, data source, ...
- Plotted properties versus parsing accuracy
 - Used equal training set size for all languages

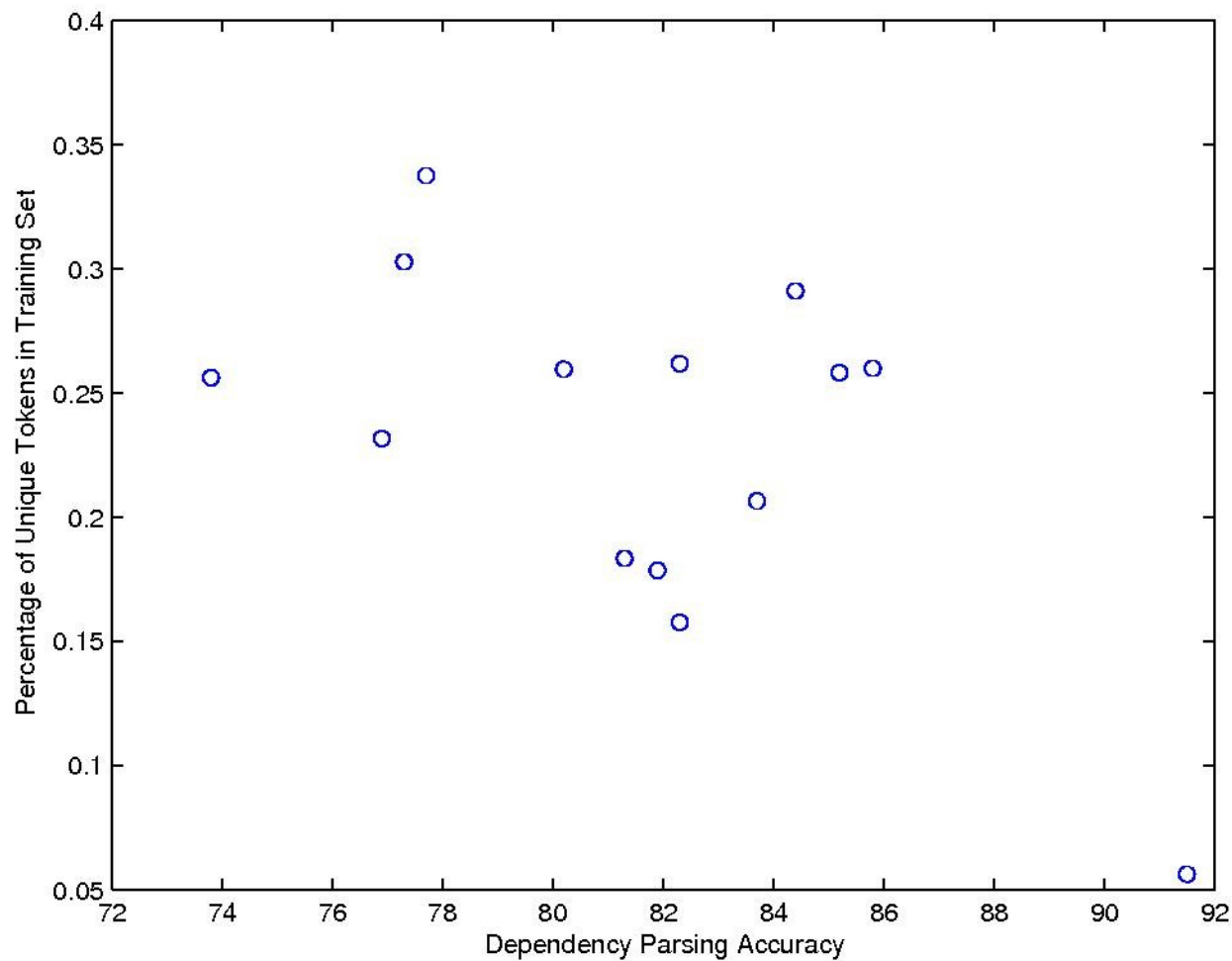


Performance Variability



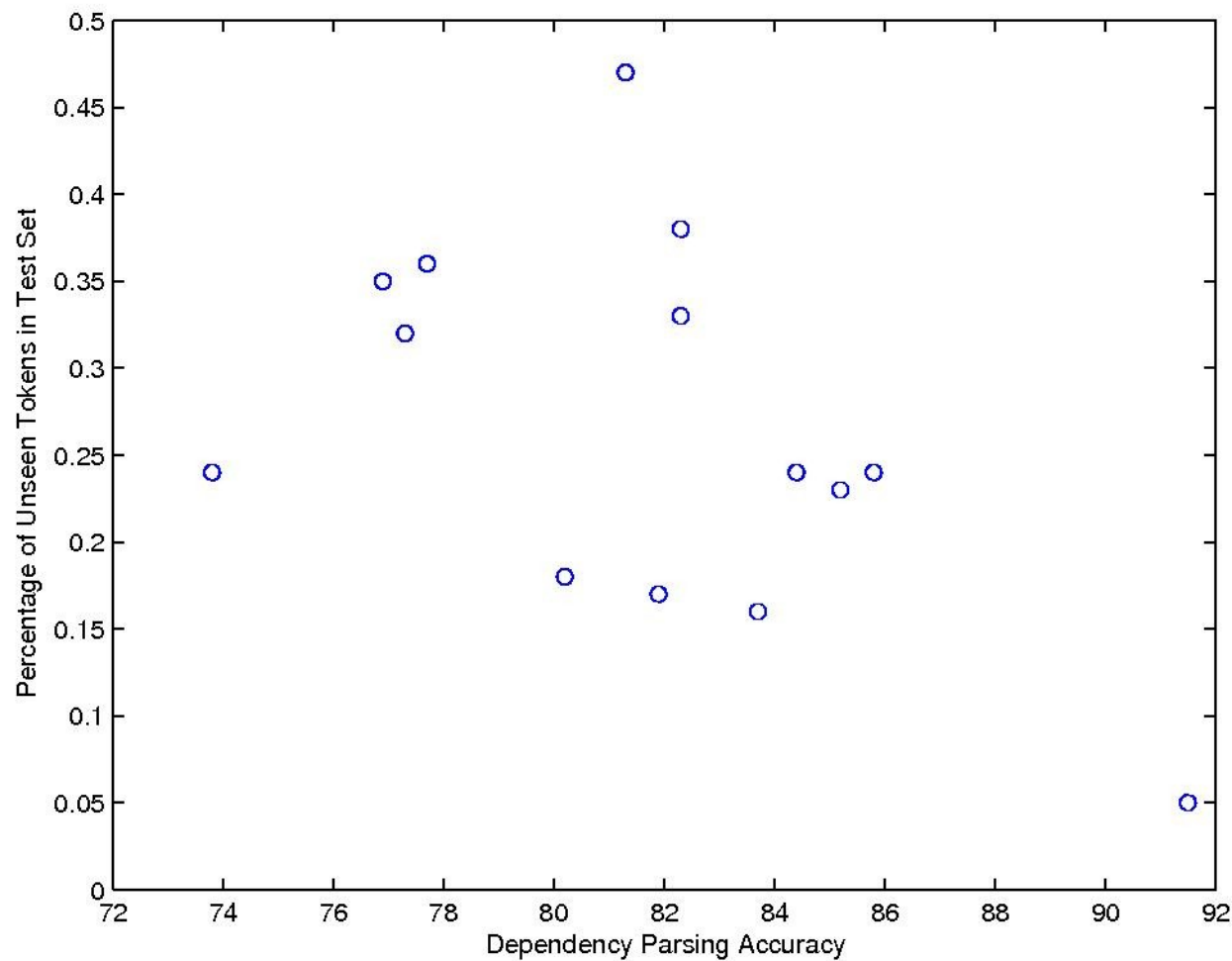
correlation: 0.36

Performance Variability



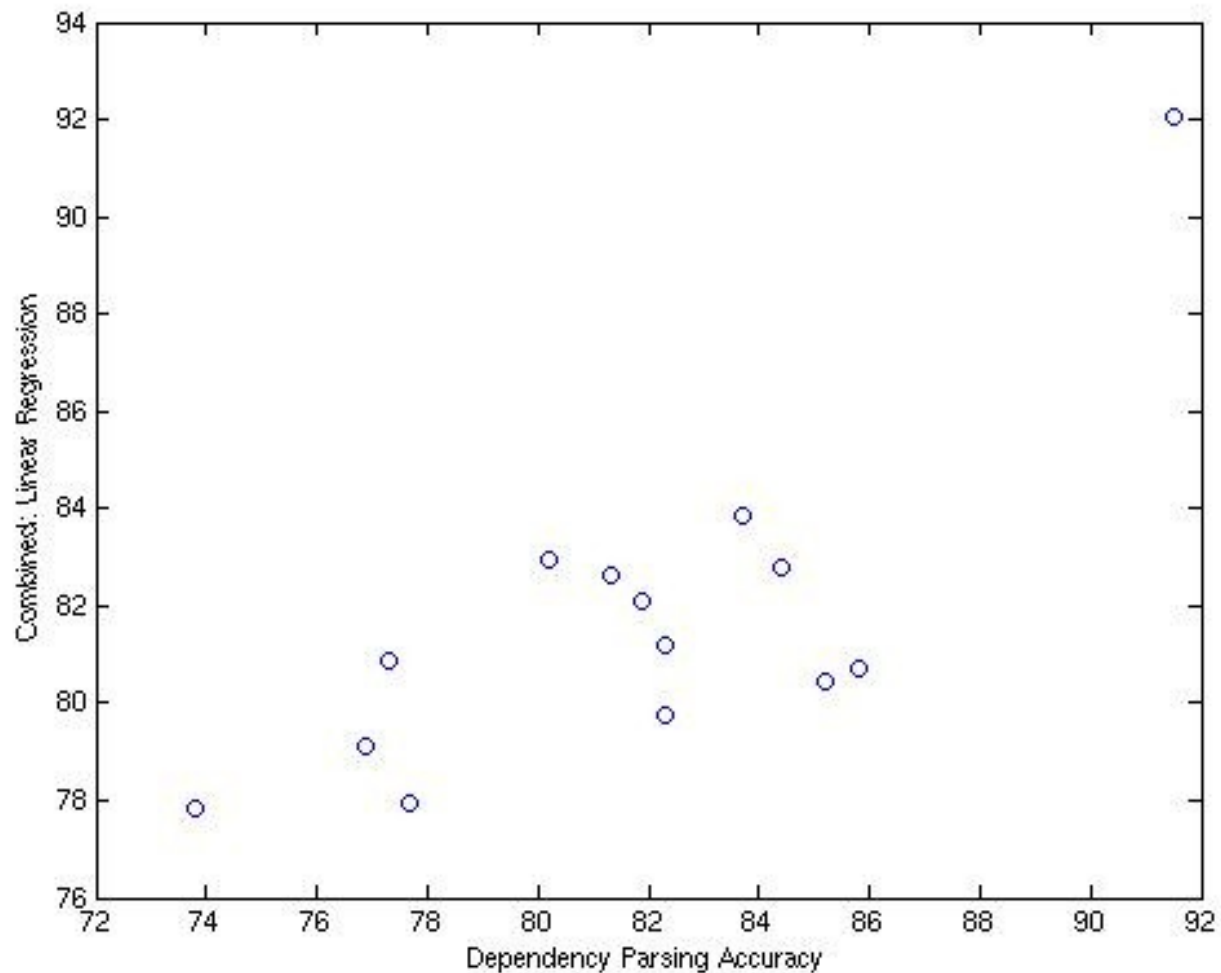
correlation: 0.56

Performance Variability



correlation: 0.52

Performance Variability



correlation: 0.85

Summary

- MST Parsing performs well on most languages
- Can approximately correlate parsing with properties of the data/languages
 - Conclusion: Parser is language general?
- Extending the model
 - Using lemma's versus inflected forms to alleviate sparsity
 - Morphology features for highly inflected languages seems to help significantly
 - Developing new language specific features an area of future work



Thanks

- CoNLL shared-task organizers for running a great program
- Joakim Nivre, Mark Liberman, Nikhil Dinesh for useful conversations
- Work supported by NSF ITR 0205456, 0205448 and 0428193



Comparison with Greedy Parsing

	Head Accuracy	Root Accuracy (F)	Sentence Accuracy
McDonald et al.	80.83	90.6	37.5
Nivre et al.	80.75	85.7	39.3

- Nivre et al: Greedy search
 - Early mistakes propagate (culminating at root)
 - Good decisions early increase accuracy
- McDonald et al: Exhaustive (MST+Viterbi)
 - Mistakes do not propagate
 - Cannot take advantage all previous decisions



Dependency Parsing as MSTs

- Consider sentence $\mathbf{x} = x_1 \dots x_n$

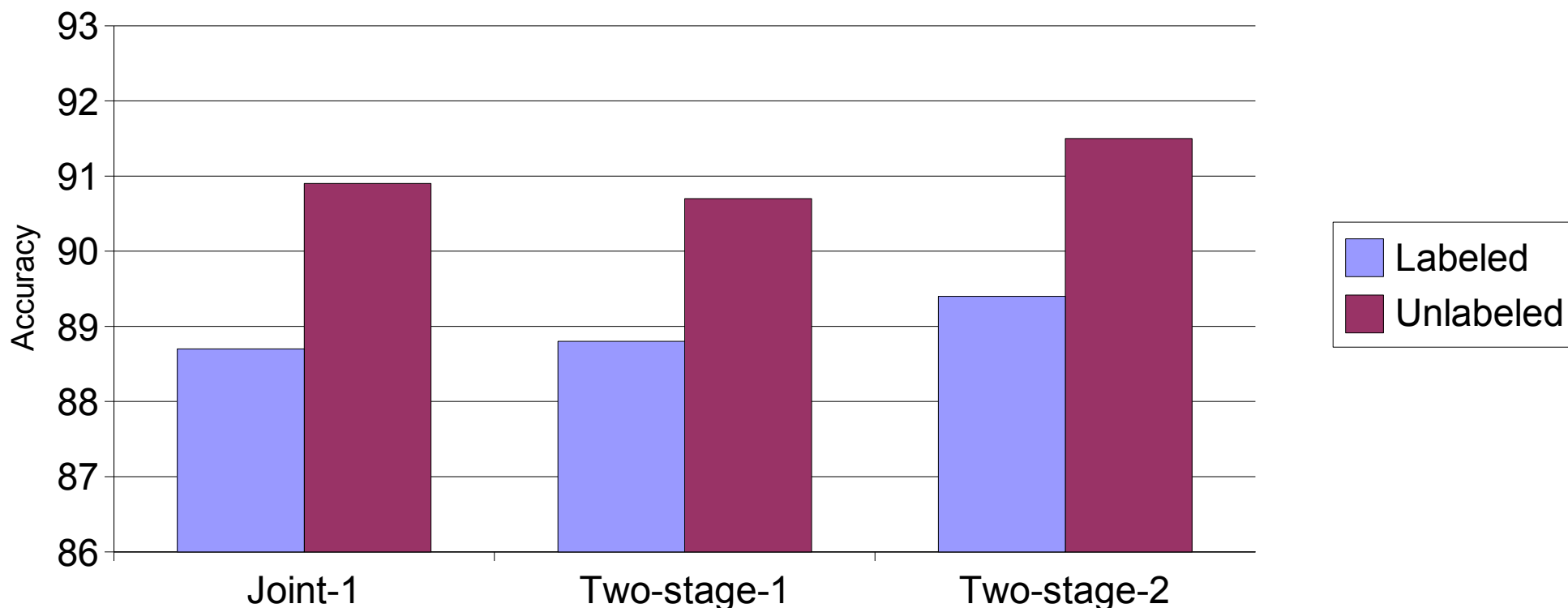
- Define $G_x = (V_x, E_x)$ as

$$V_x = \{ x_0 = \text{root}, x_1, \dots, x_n \}$$

$$E_x = \{ (i, j) \mid x_i \neq x_j, x_i \in V_x, x_j \in V_x - \{\text{root}\} \}$$

- Thus, G_x is the graph where
 - All words and the dummy root are nodes
 - There is an directed edge between all words
 - There is an edge from the root to all words

Experiments: Labeled Parsing



- Joint-1: Joint labeling + edge based factorization
- Two-stage-1: Two-stage labeling + edge based factorization
- Two-stage-2: Two-stage labeling + pairwise edge based factorization

Learning to Score Trees and Labels: The Margin Infused Relaxed Algorithm (MIRA)

- For each training instance $(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})$
 - Find current k best outputs: $k\text{-best-outputs}(\mathbf{x}^{(t)})$
 - Create constraints using these k outputs
 - Like Perceptron with aggressive margin constraints
 - Small # of constraints for each QP

$$\mathbf{w} \leftarrow \arg \min_{\mathbf{w}^*} \|\mathbf{w}^* - \mathbf{w}\|$$

$$\text{s.t. } \text{score}(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) - \text{score}(\mathbf{x}^{(t)}, \mathbf{y}) \geq L(\mathbf{y}^{(t)}, \mathbf{y})$$

$$\forall \mathbf{y} \in k\text{-best-outputs}(\mathbf{x}^{(t)})$$

MIRA

Crammer et al. 2006, McDonald et al. 2005