

# Labeled Pseudo-Projective Dependency Parsing with Support Vector Machines

Joakim Nivre<sup>1</sup>   Johan Hall<sup>1</sup>   Jens Nilsson<sup>1</sup>  
Gülşen Eryiğit<sup>2</sup>   Svetoslav Marinov<sup>3</sup>

<sup>1</sup>Växjö University, Sweden

<sup>2</sup>Istanbul Technical University, Turkey

<sup>3</sup>University of Skövde, Sweden

# Introduction

- ▶ General approach:
  - ▶ Deterministic algorithm for building labeled projective dependency graphs in linear time [Nivre 2006a]
  - ▶ History-based feature models for predicting the next parser action at nondeterministic choice points [Black et al. 1992]
  - ▶ Support vector machines for mapping histories to parser actions [Kudo and Matsumoto 2002]
  - ▶ Graph transformations for recovering non-projective structures in post-processing [Nivre and Nilsson 2005]
- ▶ Implementation:
  - ▶ MaltParser 0.4  
<http://www.vxu.se/msi/users/nivre/research/MaltParser.html>

# Shift-Reduce Type Algorithms

- ▶ Data structures:
  - ▶ Stack  $[\dots, w_i]_S$  of partially processed tokens
  - ▶ Queue  $[w_j, \dots]_Q$  of remaining input tokens
- ▶ Parsing actions built from atomic actions:
  - ▶ Adding arcs ( $w_i \rightarrow w_j, w_i \leftarrow w_j$ )
  - ▶ Stack and queue operations
- ▶ Several algorithms:
  - ▶ Strictly head-final [Kudo and Matsumoto 2002]
  - ▶ Mixed headedness (arc standard) [Yamada and Matsumoto 2003]
  - ▶ Mixed headedness (arc eager) [Nivre 2003]
  - ▶ Labeled dependencies [Nivre et al. 2004]

# Our Algorithm

- ▶ Initialization:

$$[w_0]_s \quad [w_1, \dots, w_n]_Q$$

- ▶ Termination:

$$[\dots]_s \quad []_Q$$

- ▶ Four parsing actions:

$$\text{Shift} \quad \frac{[\dots]_s \quad [w_i, \dots]_Q}{[\dots, w_i]_s \quad [\dots]_Q}$$

$$\text{Reduce} \quad \frac{[\dots, w_i]_s \quad [\dots]_Q \quad \exists w_j : w_j \rightarrow w_i}{[\dots]_s \quad [\dots]_Q}$$

$$\text{Left-Arc}_r \quad \frac{[\dots, w_i]_s \quad [w_j, \dots]_Q \quad i > 0 \wedge \neg \exists w_k : w_k \rightarrow w_i}{[\dots]_s \quad [w_j, \dots]_Q \quad w_i \xleftarrow{r} w_j}$$

$$\text{Right-Arc}_r \quad \frac{[\dots, w_i]_s \quad [w_j, \dots]_Q \quad \neg \exists w_k : w_k \rightarrow w_j}{[\dots, w_i, w_j]_s \quad [\dots]_Q \quad w_i \xrightarrow{r} w_j}$$

# Our Algorithm

- ▶ Given an **oracle**  $o$  that correctly predicts the next parsing action  $o(S, Q, E)$  ( $E =$  arc relation), parsing is deterministic:

Parse( $w_1, \dots, w_n$ )

1  $S \leftarrow [w_0]$

2  $Q \leftarrow [w_1, \dots, w_n]$

3  $E \leftarrow \emptyset$

4 **while**  $Q$  is not empty

5     **do**  $o(S, Q, E)$

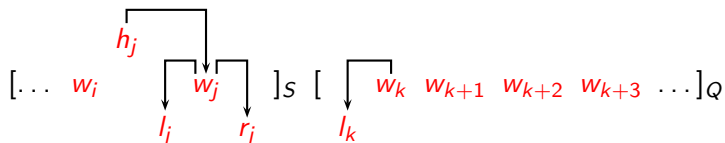
6 **return**  $G = (\{w_0, w_1, \dots, w_n\}, E)$

- ▶ The oracle can be bypassed in special cases:
  - ▶ With a unique  $r_0$  for arcs  $w_0 \xrightarrow{r_0} w_i$ , always Shift if  $S = [w_0]$ .
  - ▶ With head-final intra-word dependencies labeled  $r_{deriv}$ , always Left-Arc <sub>$r_{deriv}$</sub>  if  $S = [\dots, w_i]$  with non-head intra-word token  $w_j$ .

# History-Based Parsing

- ▶ Learning problem:
  - ▶ Approximate oracle  $o(S, Q, E)$  by classifier  $c(S, Q, E)$
- ▶ History-based feature model:
  - ▶ Parse history  $(S, Q, E)$  represented by feature vector  $\Phi = (\phi_1, \dots, \phi_p)$
  - ▶ Features  $\phi_i$  defined by
    - ▶ address function  $a_i(S, Q, E) = w_j$ ,
    - ▶ attribute function,  $f_i(a_i(S, Q, E)) = \phi_i$ .
  - ▶ Attribute functions  $f_i$ :
    - ▶ Lexical: FORM, LEMMA
    - ▶ Part-of-speech: POS, CPOS
    - ▶ Morpho-syntactic: FEATS (split into atomic features)
    - ▶ Dependency: DEPREL (**labels**)

# Base Model



FORM		+	+		+	+		
LEMMA			+		+			
CPOS			+		+			
POS	+		+		+	+	+	+
FEATS			+		+			
DEPREL		+	+	+	+			

# Learning Methods

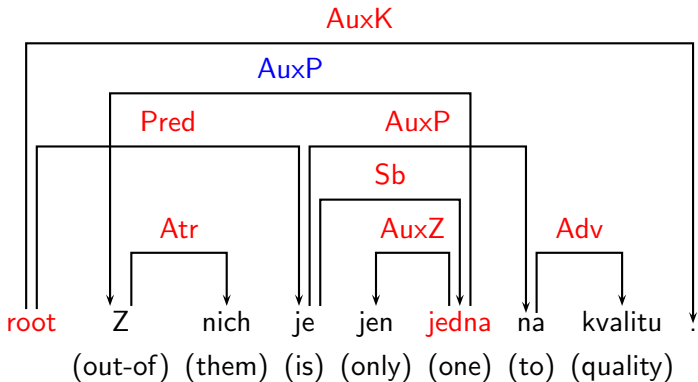
- ▶ Previous work:
  - ▶ Support vector machines (SVM)  
[Kudo and Matsumoto 2002, Yamada and Matsumoto 2003, Isozaki et al. 2004, Cheng et al. 2004]
  - ▶ Memory-based learning (MBL)  
[Nivre et al. 2004, Nivre and Scholz 2004]
  - ▶ Maximum entropy modeling (MaxEnt)  
[Cheng et al. 2005]
- ▶ Our choice:
  - ▶ Support vector machines (LIBSVM) [Chang and Lin 2001]
  - ▶ Polynomial kernel, degree 2:  $K(x_i, x_j) = (\gamma x_i^T x_j + r)^2$
  - ▶ One-versus-**one** for multi-class classification
  - ▶ Optionally dividing data sets to reduce training times

# Non-Projective Dependencies

- ▶ Two main approaches:
  - ▶ Algorithms for non-projective dependency parsing [McDonald et al. 2005, Nivre 2006b]
  - ▶ Post-processing of projective dependency graphs [Nivre and Nilsson 2005, Hall and Novák 2005, McDonald and Pereira 2006]
- ▶ Our choice:
  - ▶ Pseudo-projective parsing [Nivre and Nilsson 2005]:
    - ▶ Projectivize training data, encoding information about transformations in extended arc labels
    - ▶ Deprojectivize parser output, using top-down, breadth-first search guided by extended arc labels

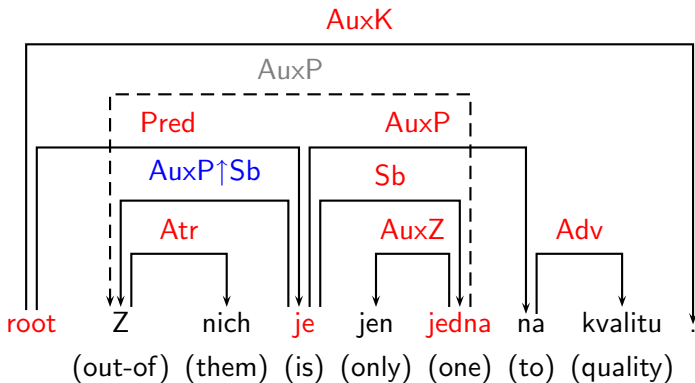
# Pseudo-Projective Parsing

- ▶ Projectivize training data:
  - ▶ Projective head nearest permissible ancestor of real head
  - ▶ Arc label extended with dependency type of real head



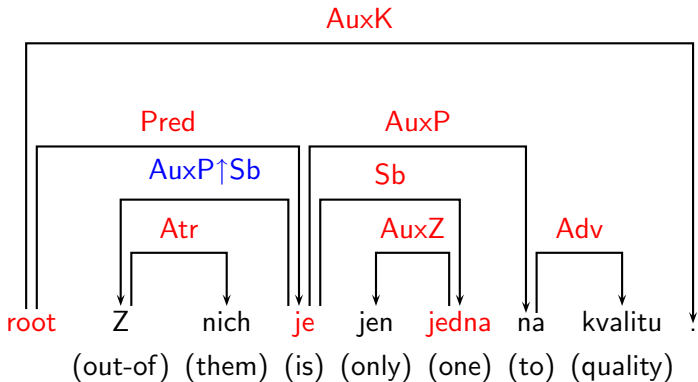
# Pseudo-Projective Parsing

- ▶ Projectivize training data:
  - ▶ Projective head nearest permissible ancestor of real head
  - ▶ Arc label extended with dependency type of real head



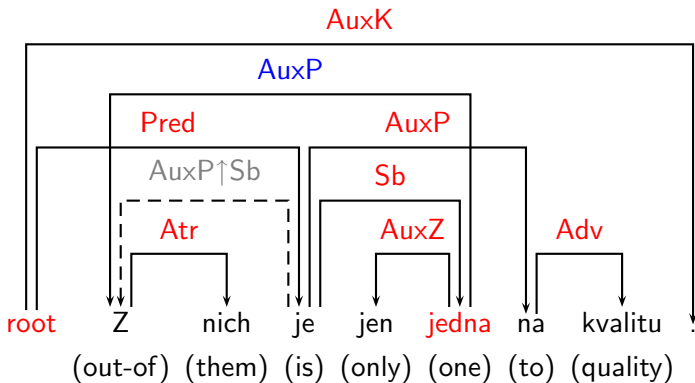
# Pseudo-Projective Parsing

- ▶ Deprojectivize parser output:
  - ▶ Top-down, breadth-first search for real head
  - ▶ Search constrained by extended arc label



# Pseudo-Projective Parsing

- ▶ Deprojectivize parser output:
  - ▶ Top-down, breadth-first search for real head
  - ▶ Search constrained by extended arc label



# Experiments

- ▶ Early experiments to decide on
  - ▶ learning algorithm (SVM vs. MBL),
  - ▶ parsing algorithm (several variants),
  - ▶ pseudo-projective parsing (several variants).
- ▶ Interleaved optimization of
  - ▶ feature model (forward and backward selection),
  - ▶ learning algorithm parameters (grid search).
- ▶ Optimization criterion: LAS

# Results

- ▶ Average (12 languages):
  - ▶ Labeled attachment score: 80.19 (2nd)
  - ▶ Unlabeled attachment score: 85.48 (2nd)
  - ▶ Label accuracy: 86.75 (1st)
- ▶ Best labeled attachment score for:
  - ▶ Japanese: 91.65
  - ▶ Portuguese: 87.60 (not significant)
  - ▶ Swedish: 84.58
  - ▶ Turkish: 65.68

# Error Analysis: Comparing Languages

- ▶ Two diagnostic metrics:
  - ▶ Root precision (RP)
  - ▶ Attachment score for distance  $> 1$  (AS)
- ▶ Language groups:
  - ▶ +RP, +AS: Bul, Chi, Swe, Dan
  - ▶ +RP, -AS: Jap, Tur
  - ▶ -RP, +AS: Cze, Dut, Ger, Por, Slo, Spa
  - ▶ -RP, -AS: Ara

# Error Analysis: Comparing Languages

- ▶ Two diagnostic metrics:
  - ▶ Root precision (RP)
  - ▶ Attachment score for distance  $> 1$  (AS)
- ▶ Language groups:
  - ▶ +RP, +AS: Bul, Chi, Swe, Dan – fixed word order?
  - ▶ +RP, -AS: Jap, Tur – head final?
  - ▶ -RP, +AS: Cze, Dut, Ger, Por, Slo, Spa – free word order?
  - ▶ -RP, -AS: Ara – head initial?

# Error Analysis: Comparing Systems

- ▶ Relative to other systems, our performance seems to be better
  - ▶ for **small** training set sizes
  - ▶ for languages with a **low** proportion of non-projective structures
  - ▶ for **labeled** (as opposed to unlabeled) accuracy

# Error Analysis: Comparing Systems

- ▶ Relative to other systems, our performance seems to be better
  - ▶ for **small** training set sizes – **resilience against data sparseness?**
  - ▶ for languages with a **low** proportion of non-projective structures – **limitations of pseudo-projective parsing?**
  - ▶ for **labeled** (as opposed to unlabeled) accuracy – **advantage of integrated labeling?**

## Error Analysis: Swedish

- ▶ High accuracy for grammatical function words and prenominal modifiers (labeled F scores):
  - ▶ Infinitive marker (**IM**) and subjunction (**UK**): 100%
  - ▶ Attribute (**AT**): >97%
  - ▶ Determiner (**DT**): >95%
- ▶ Relatively high accuracy for core arguments of the verb:
  - ▶ Subject (**SS**): >90%
  - ▶ Objects (**OO**, **IO**) and predicate complement (**SP**):  $\approx$ 85%
- ▶ Lower accuracy for post-nominal modifiers ( $\approx$ 75%) and adverbials (often below 70%)

## Error Analysis: Turkish

- ▶ Parsing based on inflectional groups (IGs) – not word forms
- ▶ Heads belonging to multiple-IG words major source of errors:
  - ▶ IG-to-IG UAS: 75.8
  - ▶ Word-to-word UAS: 82.7
  - ▶ UAS for single-IG heads: 83.7
  - ▶ UAS for multiple-IG heads: 53.2
- ▶ Accuracy for dependency types proportional to distance:
  - ▶ High accuracy for determiners and particles (1–1.4 IGs)
  - ▶ Medium accuracy for subjects, objects, adjuncts (1.8–5.2 IGs)
  - ▶ Low accuracy for distant dependencies (sentence modifiers, vocatives, appositions, etc.)

# Conclusion

- ▶ Labeled pseudo-projective dependency parsing with support vector machines gives competitive parsing accuracy for all languages involved in the shared task.
- ▶ Hypotheses for future research:
  - ▶ The learning methodology is well suited for languages with limited amounts of training data.
  - ▶ The parsing methodology is well suited for languages with a low proportion of non-projective structures.
  - ▶ Integrated labeling gives better labeled parsing accuracy.

- ▶ E. Black, F. Jelinek, J. D. Lafferty, D. M. Magerman, R. L. Mercer, and S. Roukos. 1992. Towards history-based grammars: Using richer models for probabilistic parsing. In *Proc. of the 5th DARPA Speech and Natural Language Workshop*, pages 31–37.
- ▶ C.-C. Chang and C.-J. Lin, 2001. *LIBSVM: A Library for Support Vector Machines*. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- ▶ Yuchang Cheng, Masayuki Asahara, and Yuji Matsumoto. 2004. Deterministic dependency structure analyzer for Chinese. In *Proceedings of the First International Joint Conference on Natural Language Processing (IJCNLP)*
- ▶ Yuchang Cheng, Masayuki Asahara, and Yuji Matsumoto. 2005. Machine learning-based dependency analyzer for Chinese. In *Proceedings of International Conference on Chinese Computing (ICCC)*.
- ▶ Keith Hall and Vaclav Novák. 2005. Corrective modeling for non-projective dependency parsing. In *Proceedings of the 9th International Workshop on Parsing Technologies (IWPT)*.
- ▶ Hideki Isozaki, Hideto Kazawa, and Tsutomu Hirao. 2004. A deterministic word dependency analyzer enhanced with preference learning. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, pages 275–281.
- ▶ T. Kudo and Y. Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In *Proc. of CoNLL-2002*, pages 63–69.
- ▶ Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.
- ▶ Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pages 523–530.
- ▶ J. Nivre and J. Nilsson. 2005. Pseudo-projective dependency parsing. In *Proc. of ACL-2005*, pages 99–106.
- ▶ Joakim Nivre and Mario Scholz. 2004. Deterministic dependency parsing of English text. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, pages 64–70.

- ▶ J. Nivre, J. Hall, and J. Nilsson. 2004. Memory-based dependency parsing. In *Proc. CoNLL-2004*, pages 49–56.
- ▶ J. Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proc. of IWPT-2003*, pages 149–160.
- ▶ J. Nivre. 2006a. *Inductive Dependency Parsing*. Springer.
- ▶ Joakim Nivre. 2006b. Constraints on non-projective dependency graphs. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.
- ▶ H. Yamada and Y. Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proc. of IWPT-2003*, pages 195–206.