Lexicalized Tree-adjoining Grammar applied to Semantic Role Labeling

Anoop Sarkar

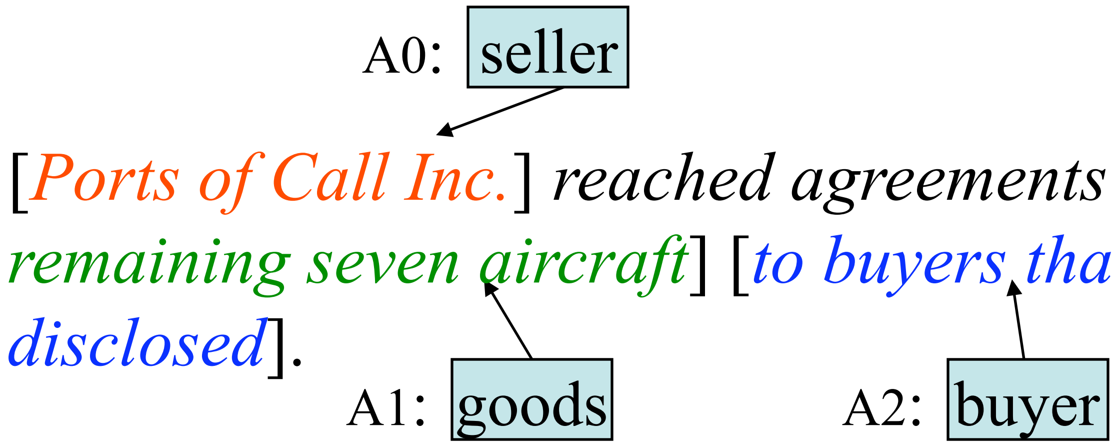
Simon Fraser University

Jun 8, 2008

(joint work with *Yudong Liu* and *Libin Shen*)

Semantic Role Labeling (SRL)

- For a given verb (predicate), SRL aims to *identify* and *label* all its arguments with semantic roles, such as Agent, Patient, and Theme.



A0: seller

[*Ports of Call Inc.*] *reached agreements to* **sell** [*its remaining seven aircraft*] [*to buyers that weren't disclosed*].

A1: goods

A2: buyer

SRL: two-phase task

- Argument identification:
  - If a portion of a sentence should be assigned a semantic role? (YES/NO)

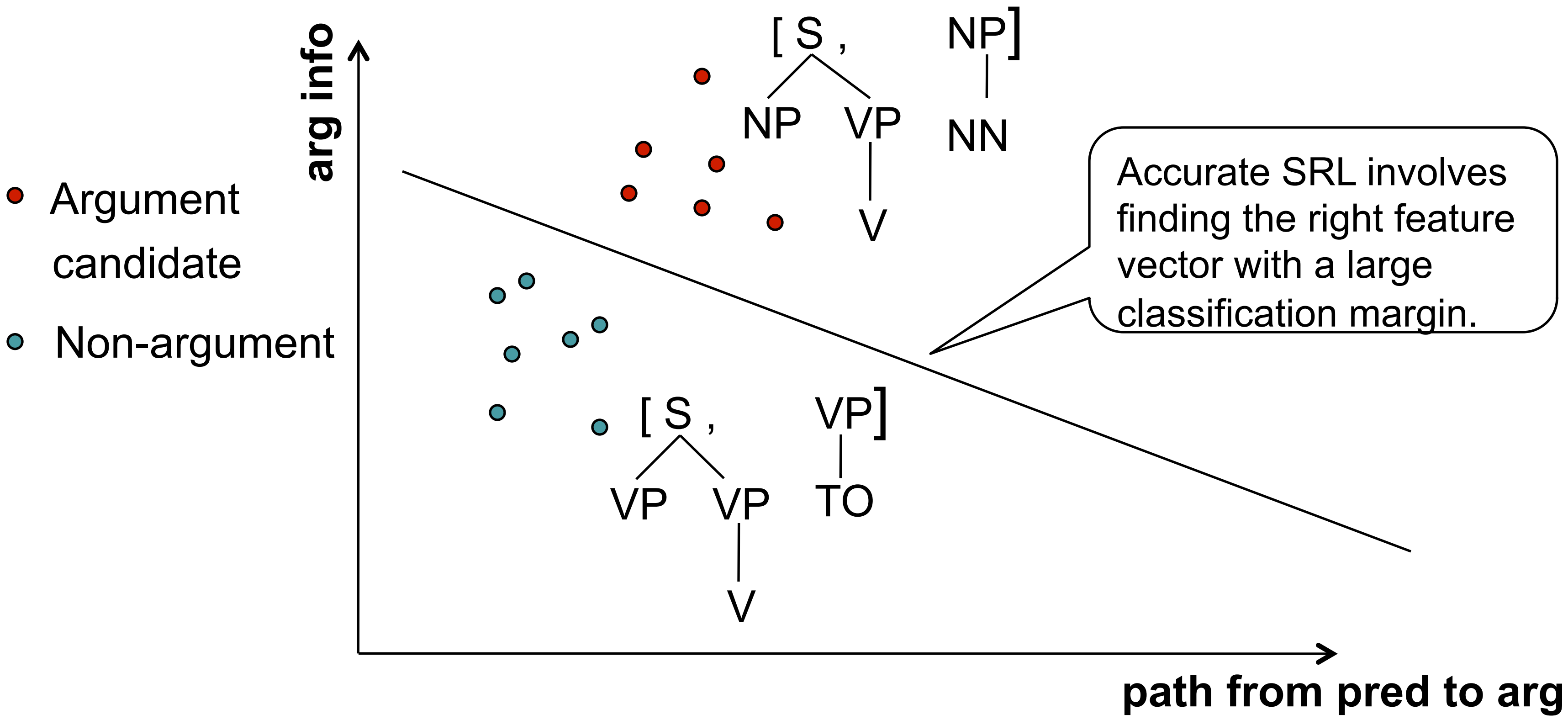
[*Ports of Call Inc.*]Y [*reached*]N [*agreements*]N [*to*]N **sell** [*its remaining seven aircraft*]Y [*to buyers that weren't disclosed*]Y.

- Argument classification:
  - If yes, what semantic role should be assigned to that portion? (Agent/Patient/Theme/…)

[*Ports of Call Inc.*]seller *reached agreements to* **sell** [*its remaining seven aircraft*]goods [*to buyers that weren't disclosed*]buyer.

Feature Selection in SRL

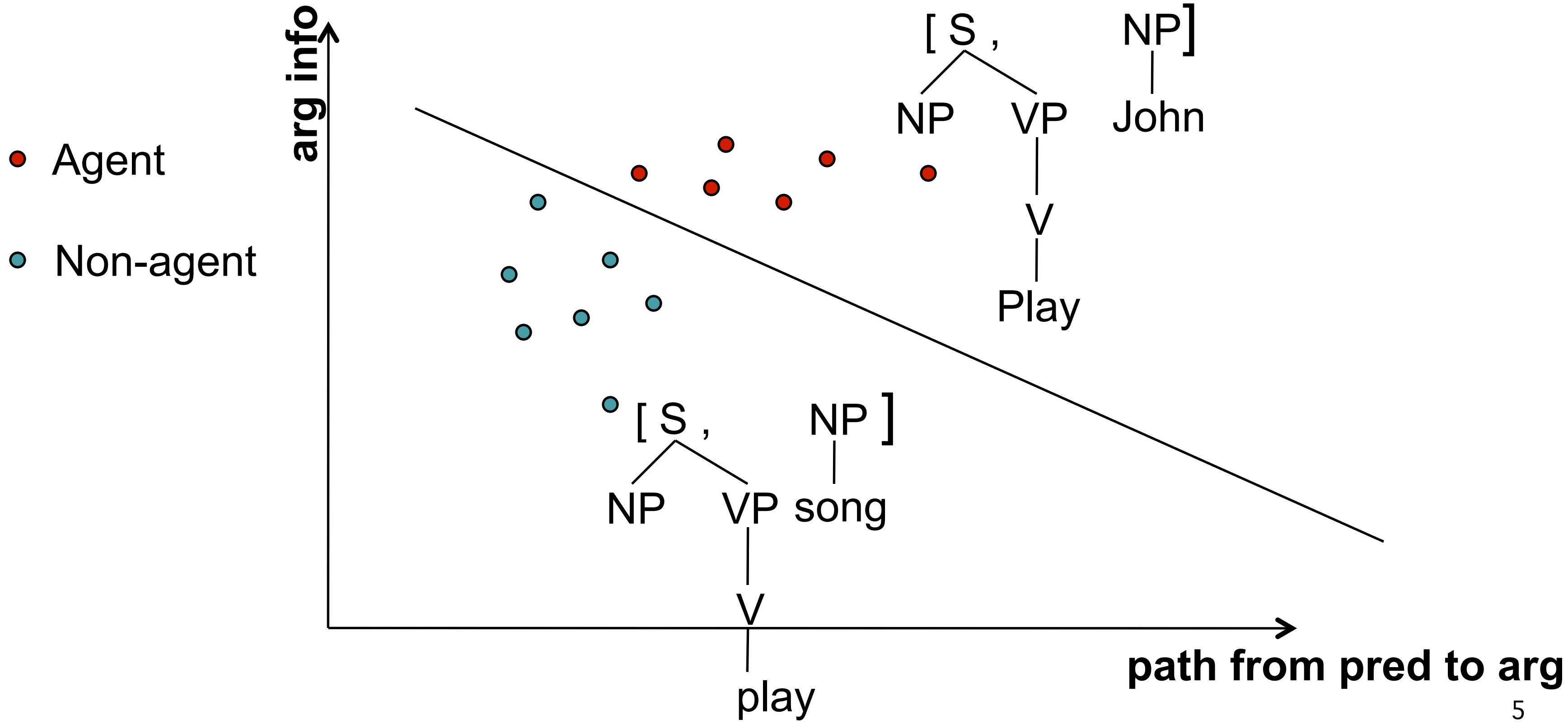
- Argument identification:



Accurate SRL involves finding the right feature vector with a large classification margin.

Feature Selection in SRL (cont’d)

- Argument classification:



Current SRL systems

- High accuracy is achieved by:

  -- Proposing new types of *features* from different *syntactic views*: token-level, sentence level…

  -- Modeling the predicate frameset by *capturing dependencies between arguments*

  -- Dealing with incorrect parser output by *using more than one parser*

– syntactic chunks (Hacioglu et al., 2004)

– syntactic parses (Gildea and Jurafsky, 2002; Gildea and Palmer, 2002; Punyakanok et al., 2005)

– CCG derivations (Gildea and Hockenmaier, 2003)

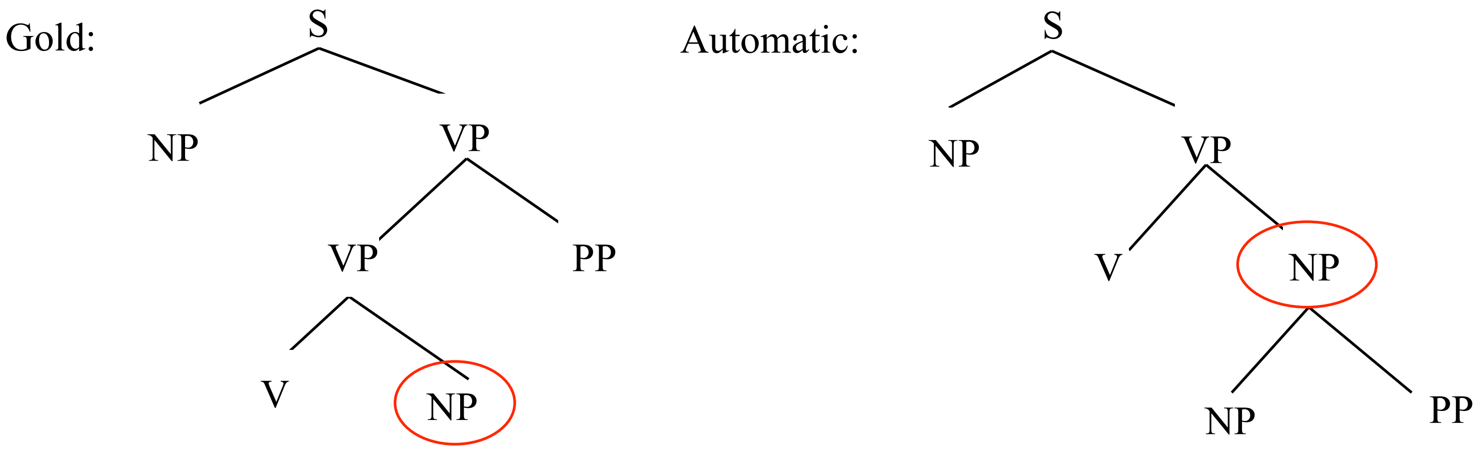
– dependency trees (Hacioglu et al., 2004)

– A0,A0 V A2 A1 (Gildea and Jurafsky, 2002; Pradhan et al., 2004; Toutanova et al., 2005; Punyakanok et al., 2005)

– (Punyakanok et al., 2005; Pradhan et al., 2005)

Current SRL systems (cont'd)

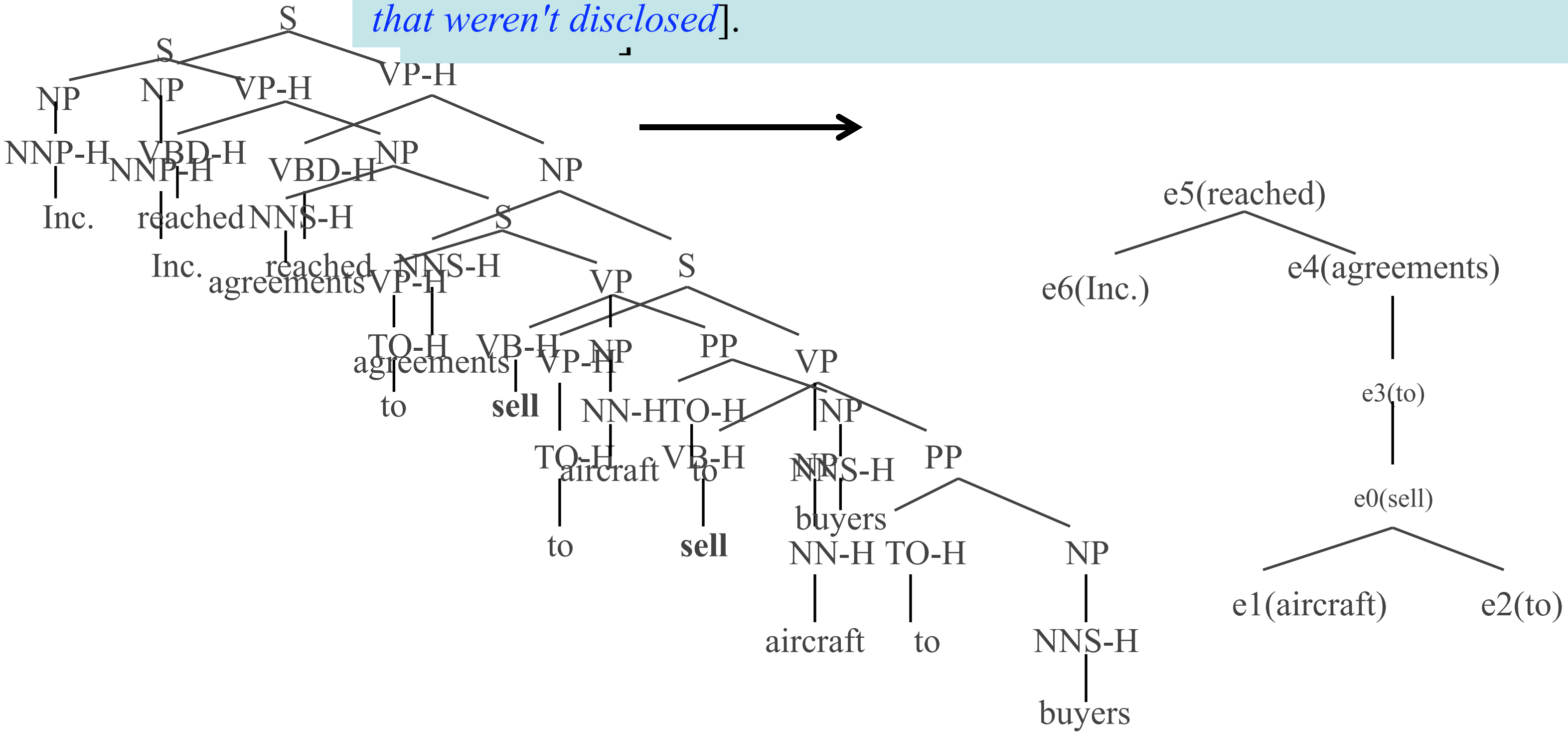
- High accuracy is achieved by:
  - Proposing new types of *features* from different *syntactic views*: token-level, sentence level…
  - Modeling the predicate frameset by *capturing dependencies between arguments*
  - Dealing with incorrect parser output by *using more than one parser* (Punyakanok et al., 2005; Pradhan et al., 2005)



Source of features for SRL

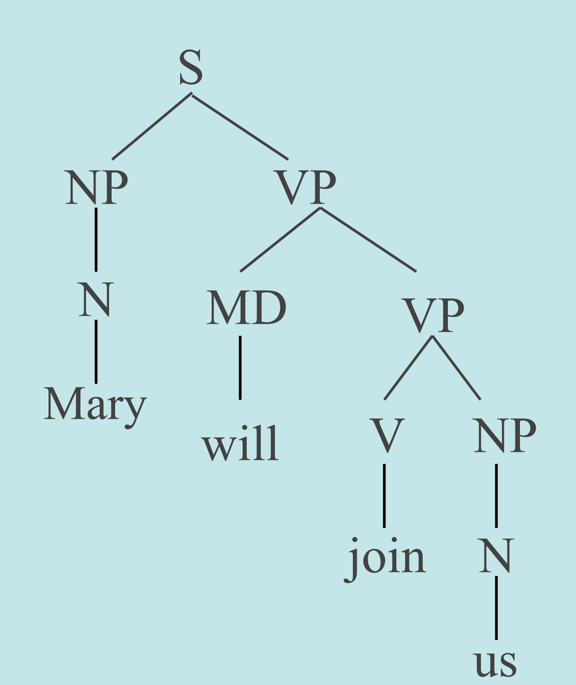
[*Ports of Call Inc.*] *reached agreements to* **sell** [*its remaining seven aircraft*] [*to buyers that weren't disclosed*].

ements to **sell** [*its*
ers that weren't

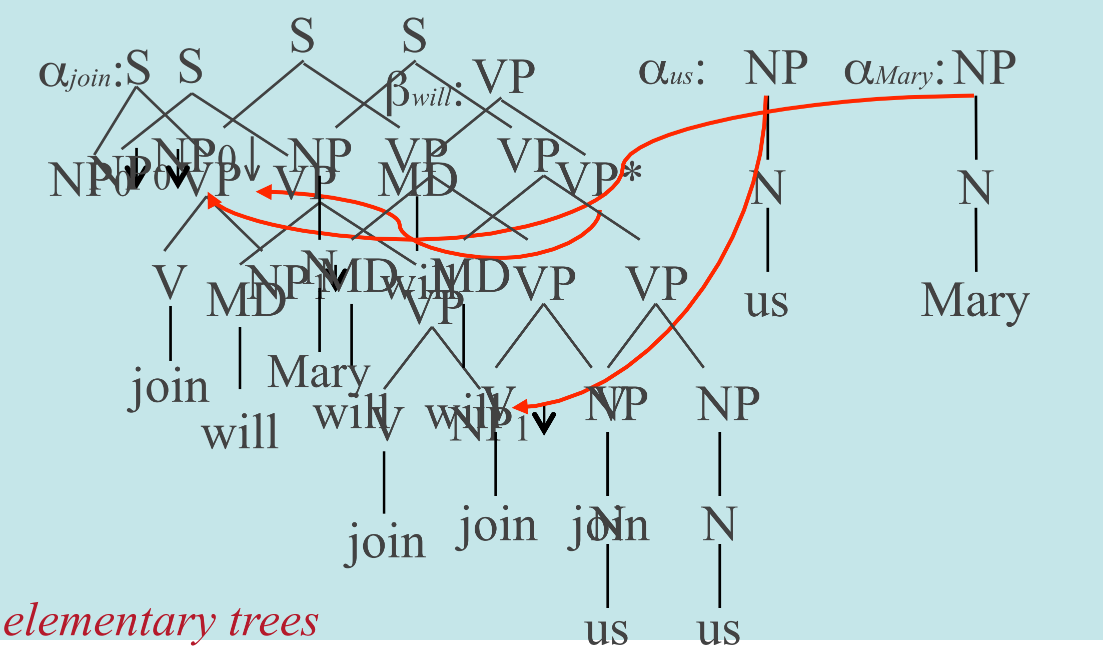


from *derived trees* to *derivation trees*

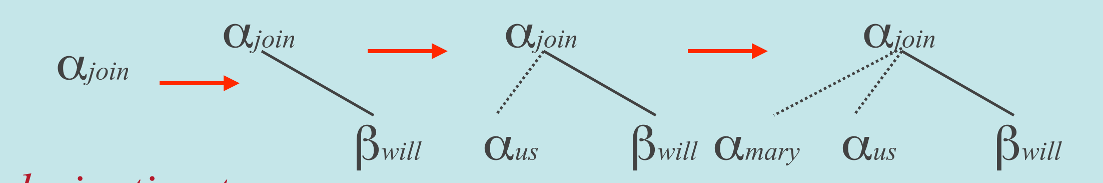
Lexicalized Tree-Adjoining Grammar (LTAG)



parse tree

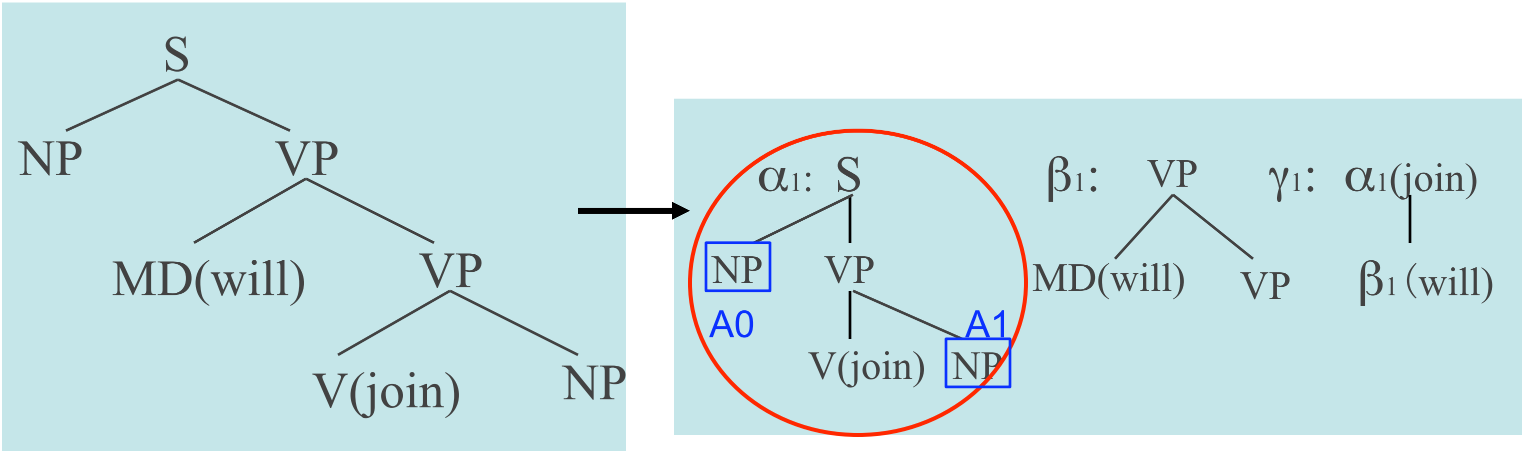


elementary trees



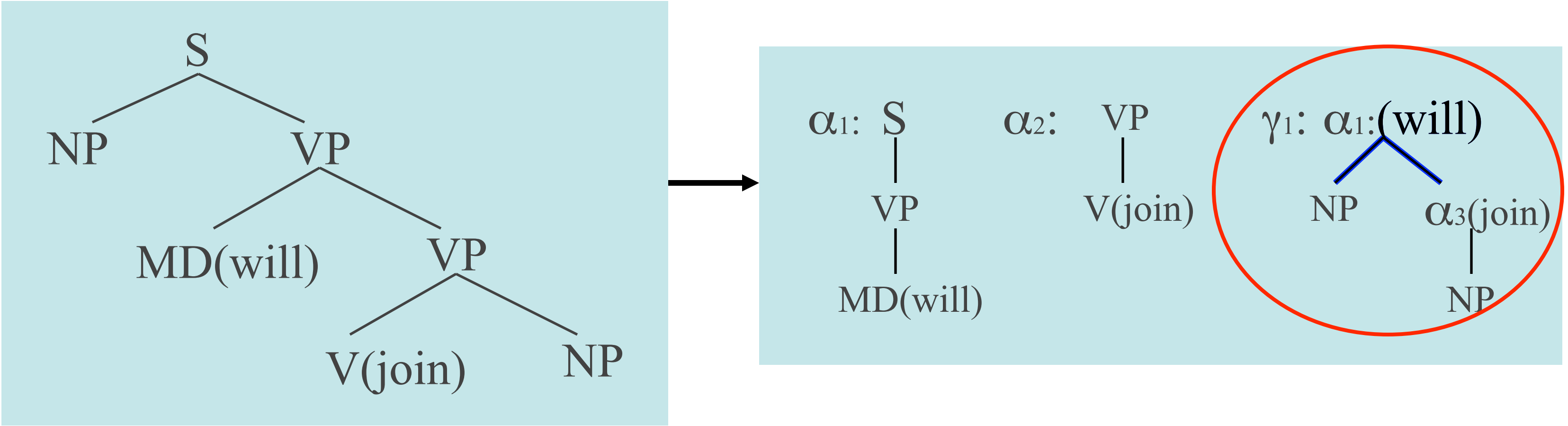
derivation tree

LTAG derivation trees for SRL (1)



- only ~87% of dependencies between predicate and argument are captured (Chen and Rambow, 2003)

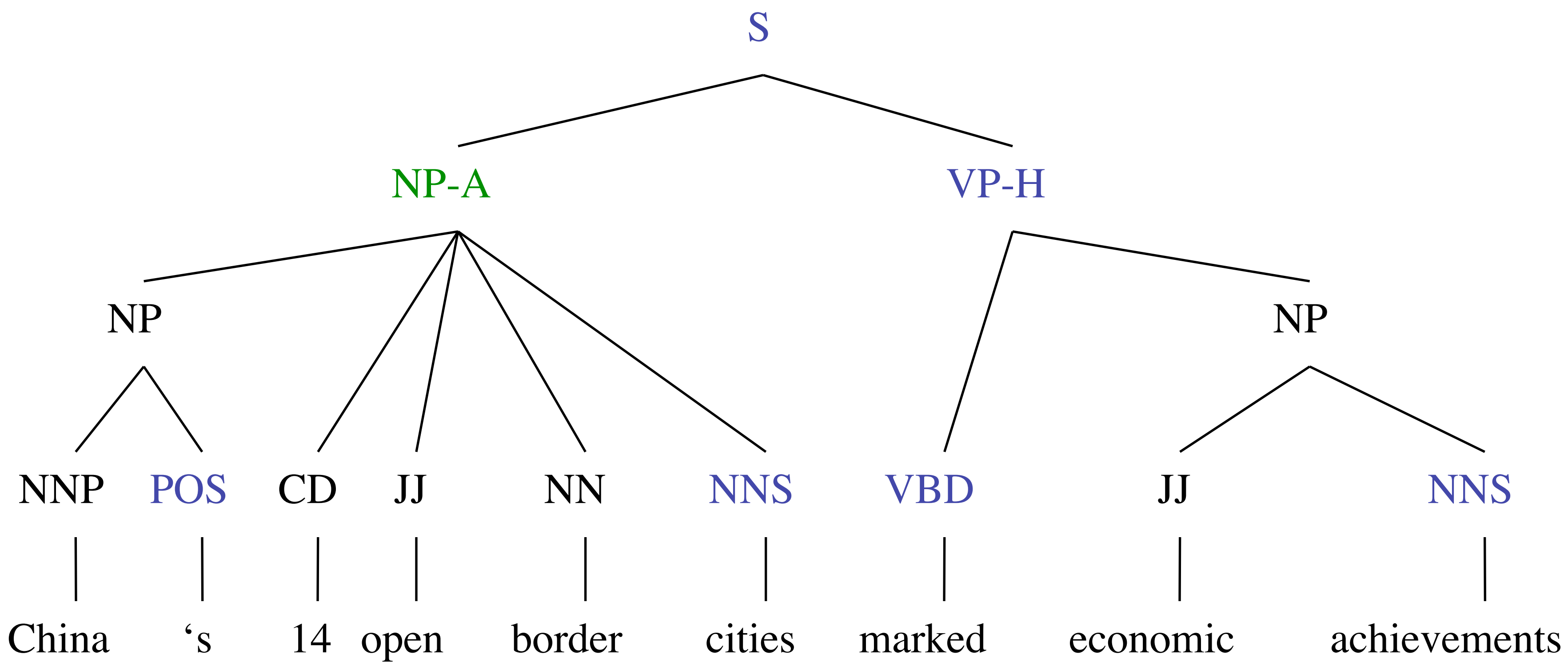
LTAG derivation trees for SRL (2)



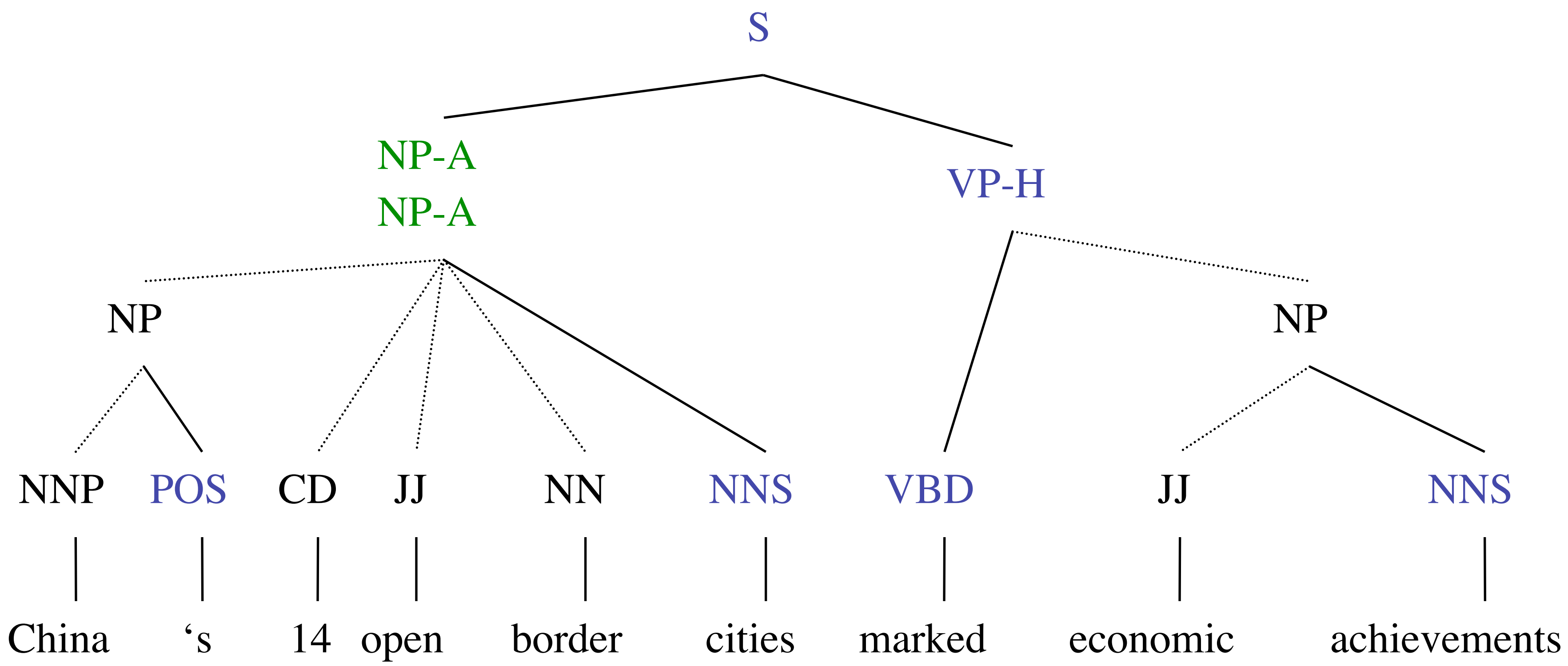
- Sister-adjunction
- In this work, paths in the derivation trees are also considered

  F-score: 82.34% → 85.27%

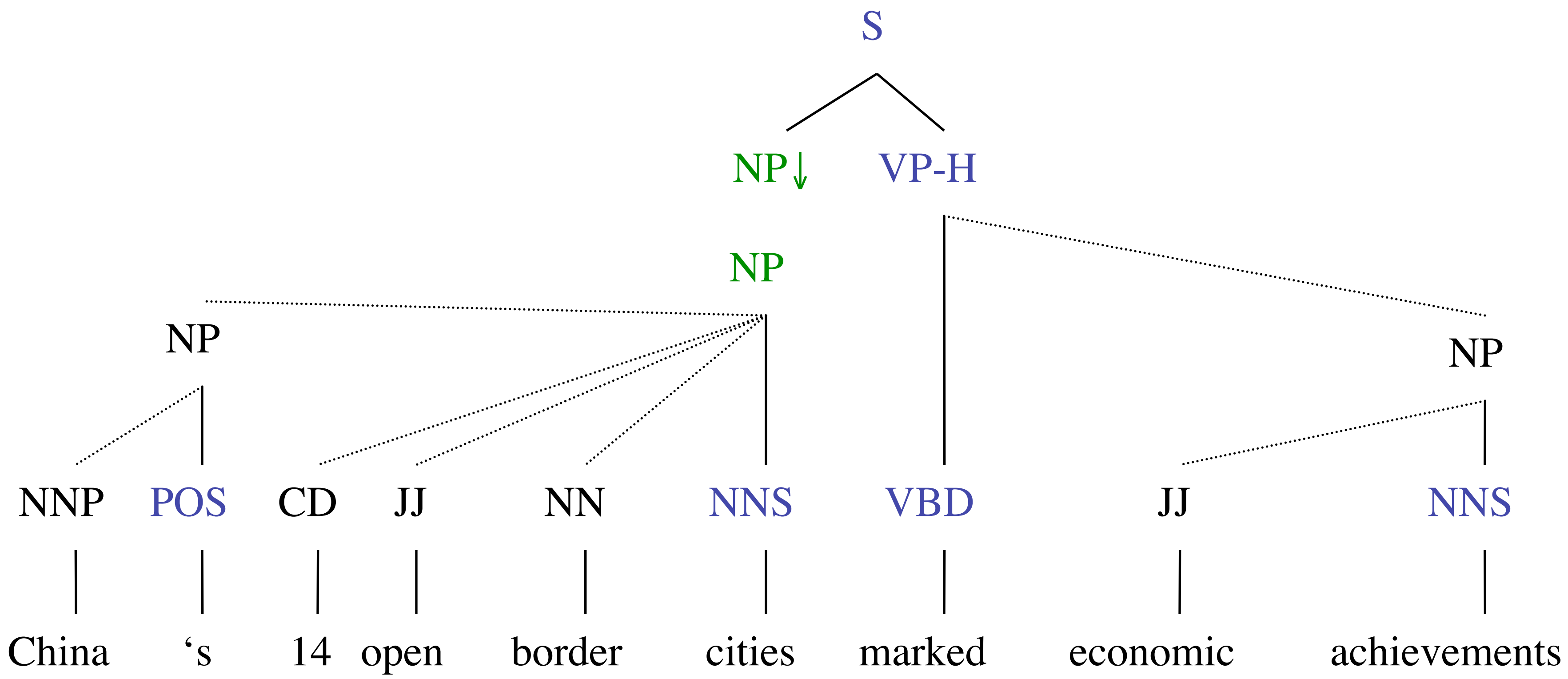
LTAG derivations from TreeBanks or phrase-structure parses



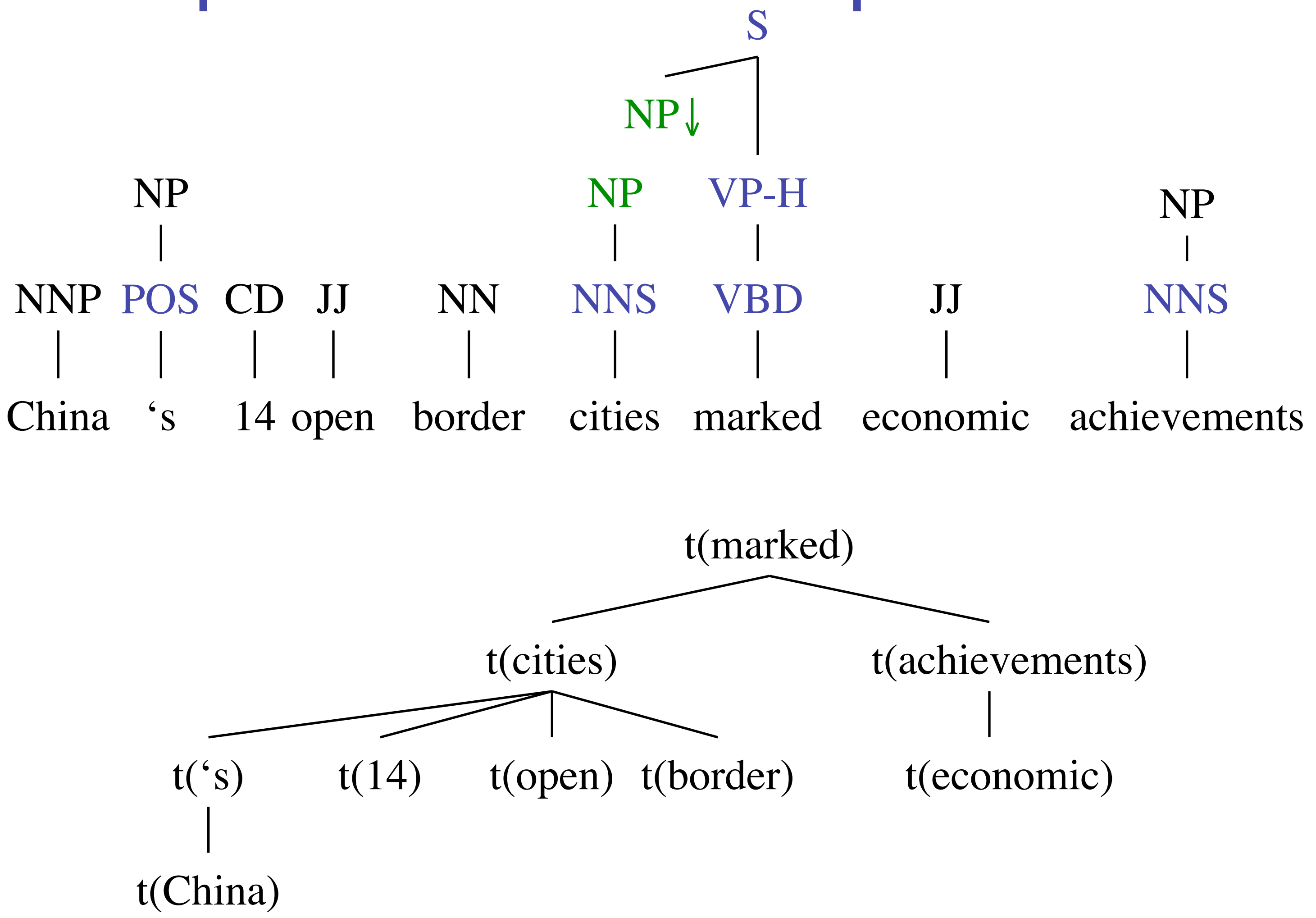
LTAG derivations from TreeBanks or phrase-structure parses



LTAG derivations from TreeBanks or phrase-structure parses



LTAG derivations from TreeBanks or phrase-structure parses

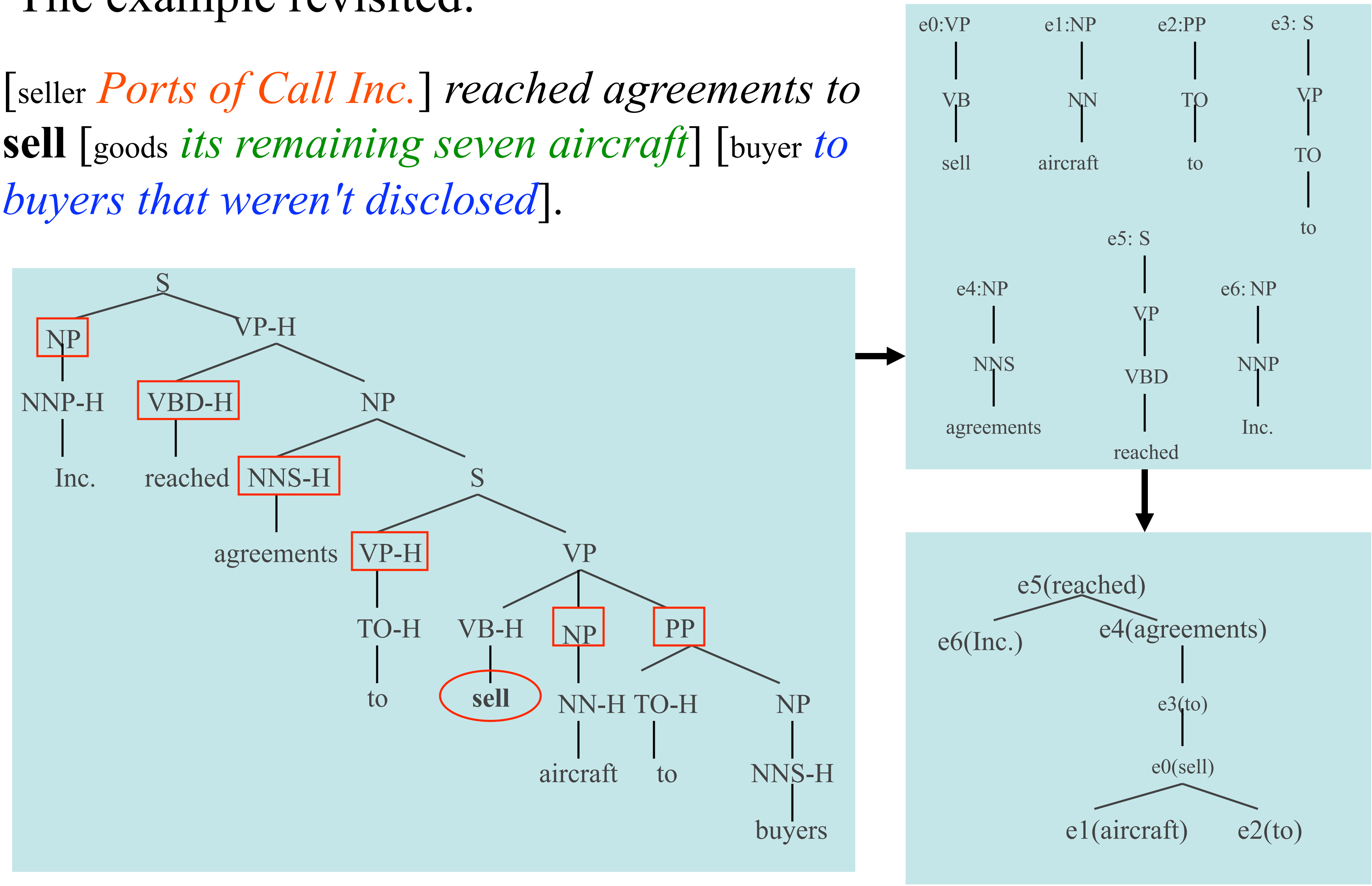


Our research focus

- Propose new source of SRL features
  - From LTAG derivation trees
  - From different types of LTAG derivations
- Increase robustness of SRL to parser errors

The example revisited:

[seller *Ports of Call Inc.*] *reached agreements to* **sell** [goods *its remaining seven aircraft*] [buyer *to buyers that weren't disclosed*].

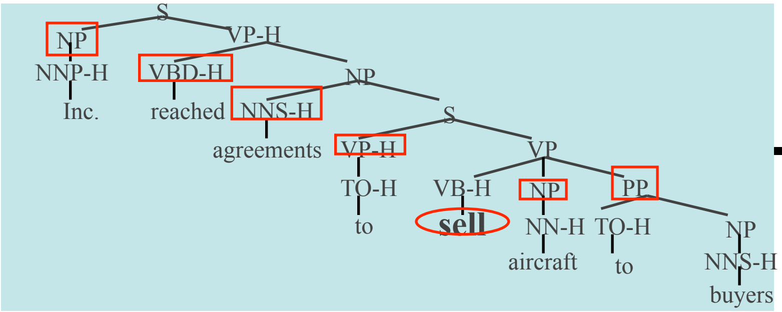


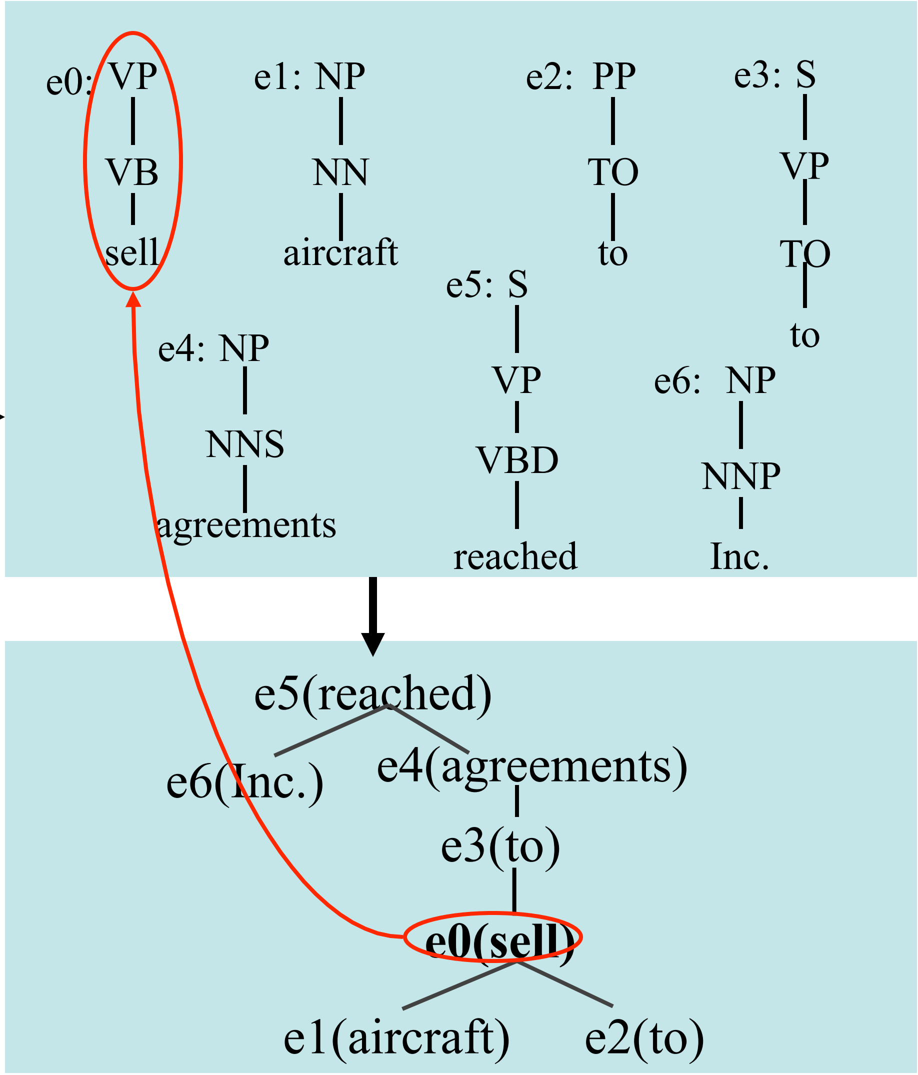
LTAG-based features

- P: Predicate elementary tree features
- A: Argument elementary tree features
- I: Intermediate elementary tree features
- R: Features capturing topological relations in LTAG derivation trees: *distance between elementary trees, relative position, modifying relations*
- S: Sub-categorization features

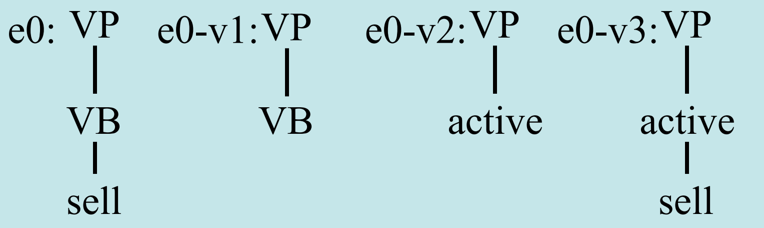
Category-P features: predicate e-tree related features

[seller *Ports of Call Inc.*] *reached agreements to* **sell** [goods *its remaining seven aircraft*] [buyer *to buyers that weren't disclosed*].



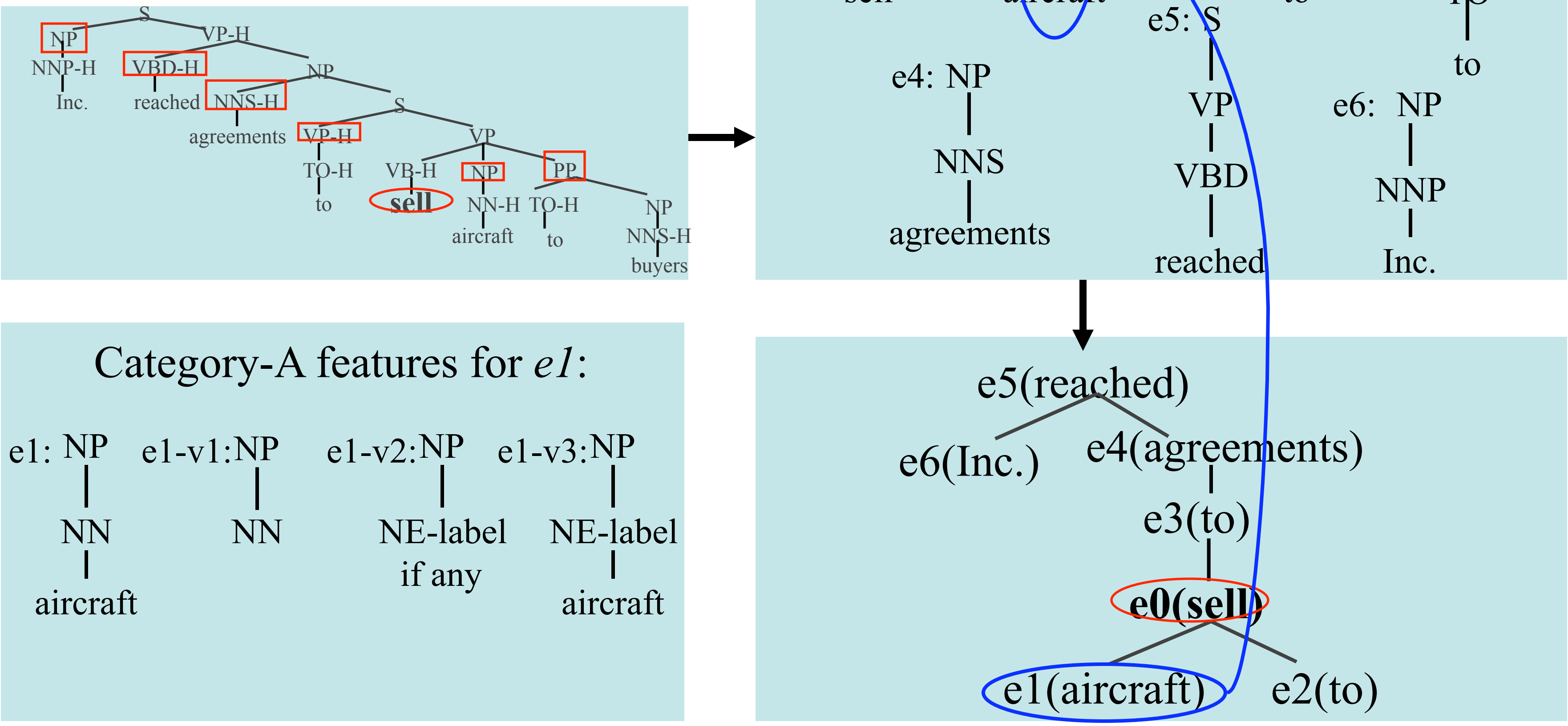


Category-P features:



Category-A features: argument e-tree related features

[seller *Ports of Call Inc.*] *reached agreements to* **sell** [goods *its remaining seven aircraft*] [buyer *to buyers that weren't disclosed*].



S NP NNP-H Inc. VP-H VBD-H reached NP NNS-H agreements S VP-H TO-H to VP VB-H sell NP NN-H aircraft PP TO-H to NP NNS-H buyers

e0: VP VB sell

e1: NP NN aircraft

e2: PP TO to

e3: S VP TO to

e4: NP NNS agreements

e5: S VP VBD reached

e6: NP NNP Inc.

e5(reached) e6(Inc.) e4(agreements) e3(to) **e0(sell)** e1(aircraft) e2(to)

Category-A features for *e1*:

e1: NP NN aircraft

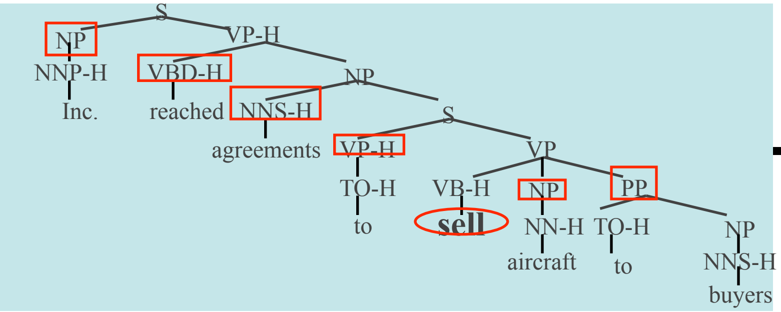
e1-v1:NP NN

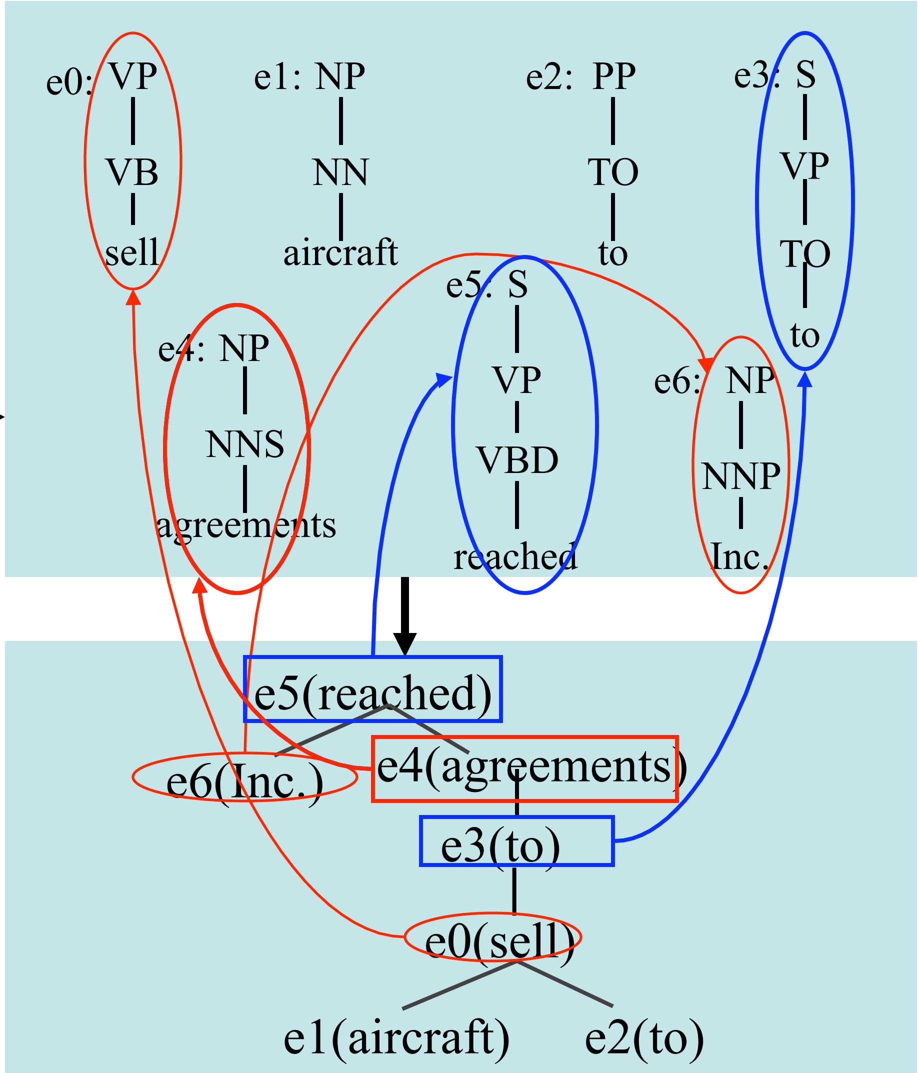
e1-v2:NP NE-label if any

e1-v3:NP NE-label aircraft

Category-I features: intermediate e-tree related features

[seller *Ports of Call Inc.*] *reached agreements to* **sell** [goods *its remaining seven aircraft*] [buyer *to buyers that weren't disclosed*].





Category-I features for *e4*:

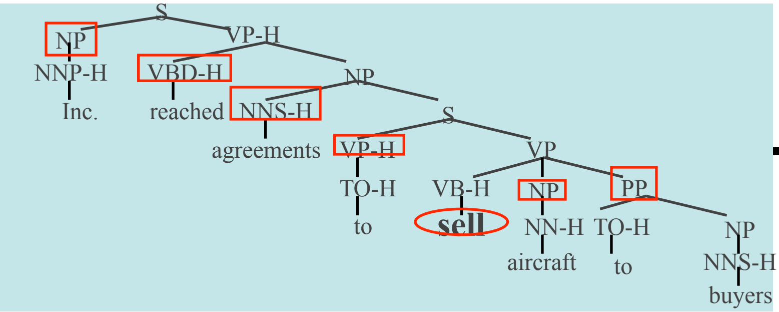
e4: NP
NNS
agreements

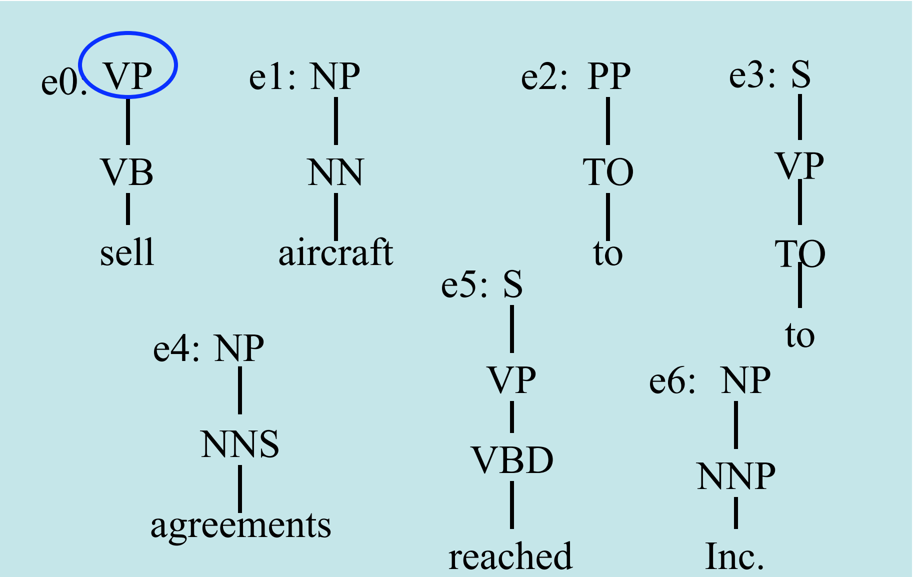
e4-v1: NP
NNS

- relative position+ modifying relation:
  left+modified
- attachment point:
  n/a
- distance:
  1 (e3)

Category-R features: topological relations between predicate etree and argument etree

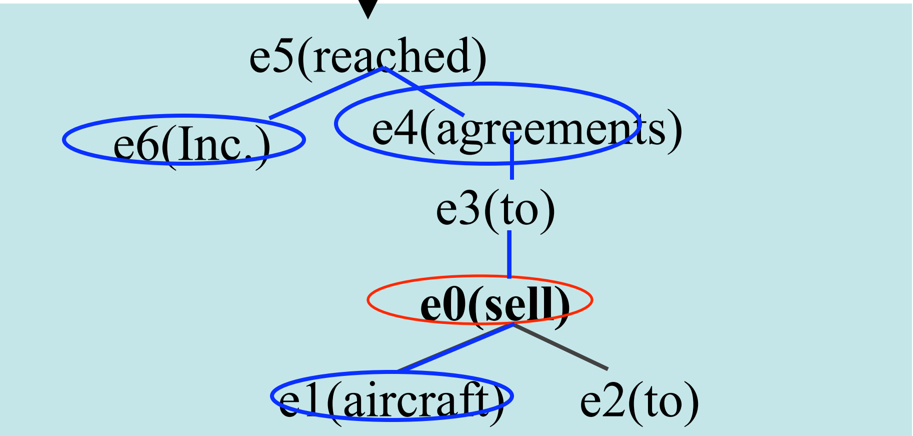
[seller *Ports of Call Inc.*] *reached agreements to* **sell** [goods *its remaining seven aircraft*] [buyer *to buyers that weren't disclosed*].





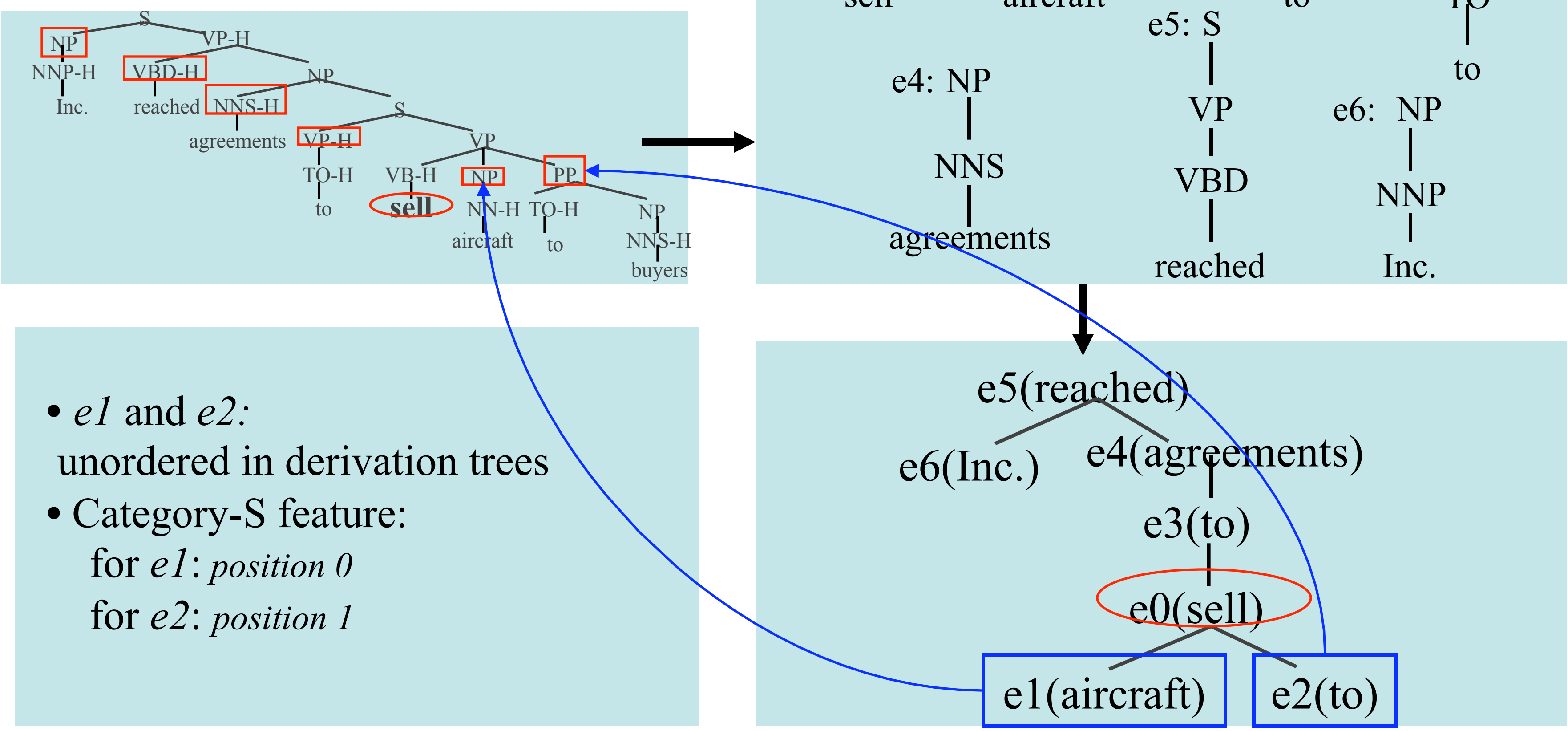
Category-R features for *e4*:
Category-R features for *e1*:

- **relative position+modifying relation:**
  right+modifying
  left+modified
- **attachment point:**
  VP
  n/a
- **distance:**
  3 (e3, e4, e5)
  0 (directly connected)



Category-S features: index of argument etree in sub-cat frame of predicate etree

[seller *Ports of Call Inc.*] *reached agreements to* **sell** [goods *its remaining seven aircraft*] [buyer *to buyers that weren't disclosed*].

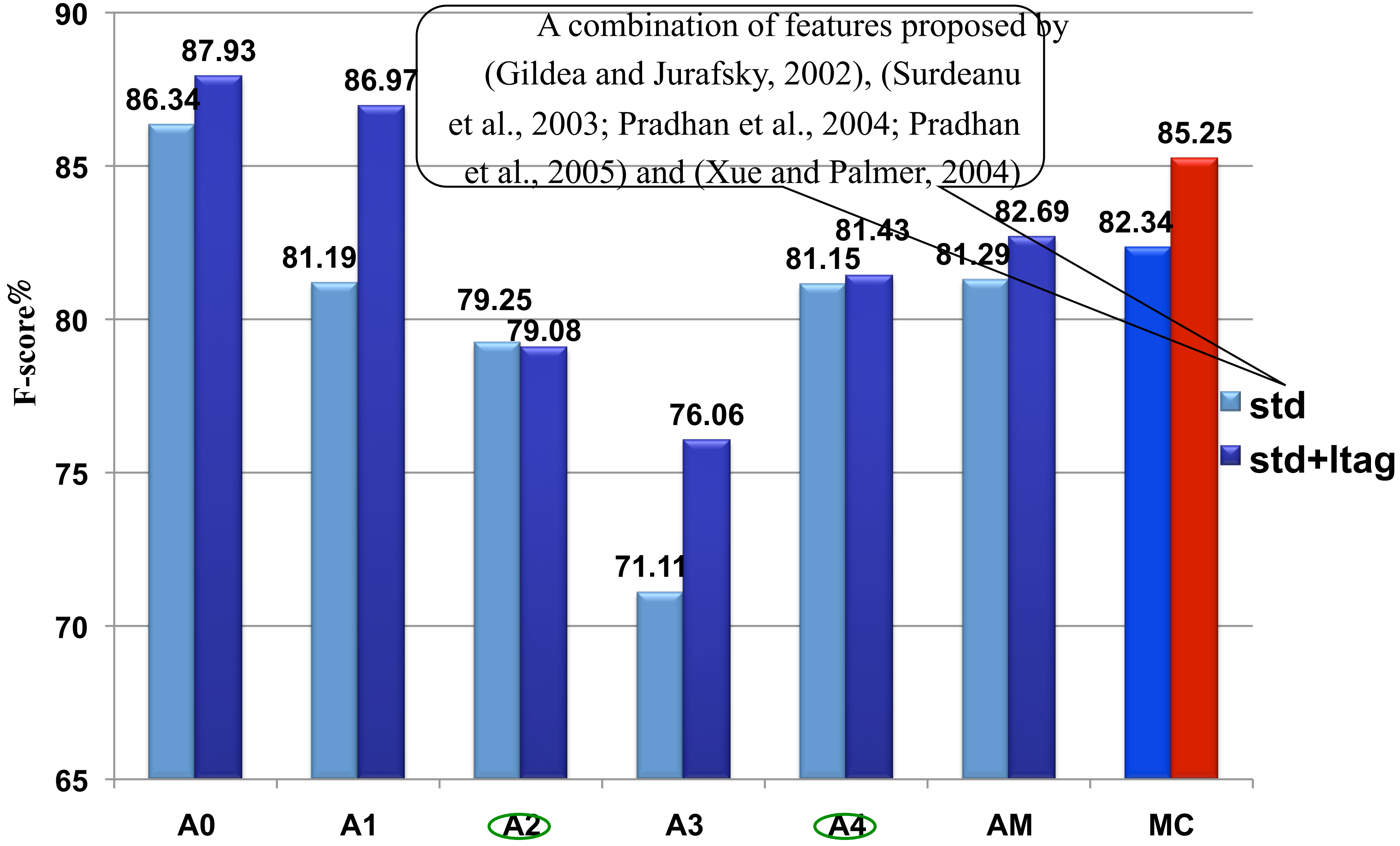


- *e1* and *e2*: unordered in derivation trees
- Category-S feature:
  for *e1*: *position 0*
  for *e2*: *position 1*

Experimental setting

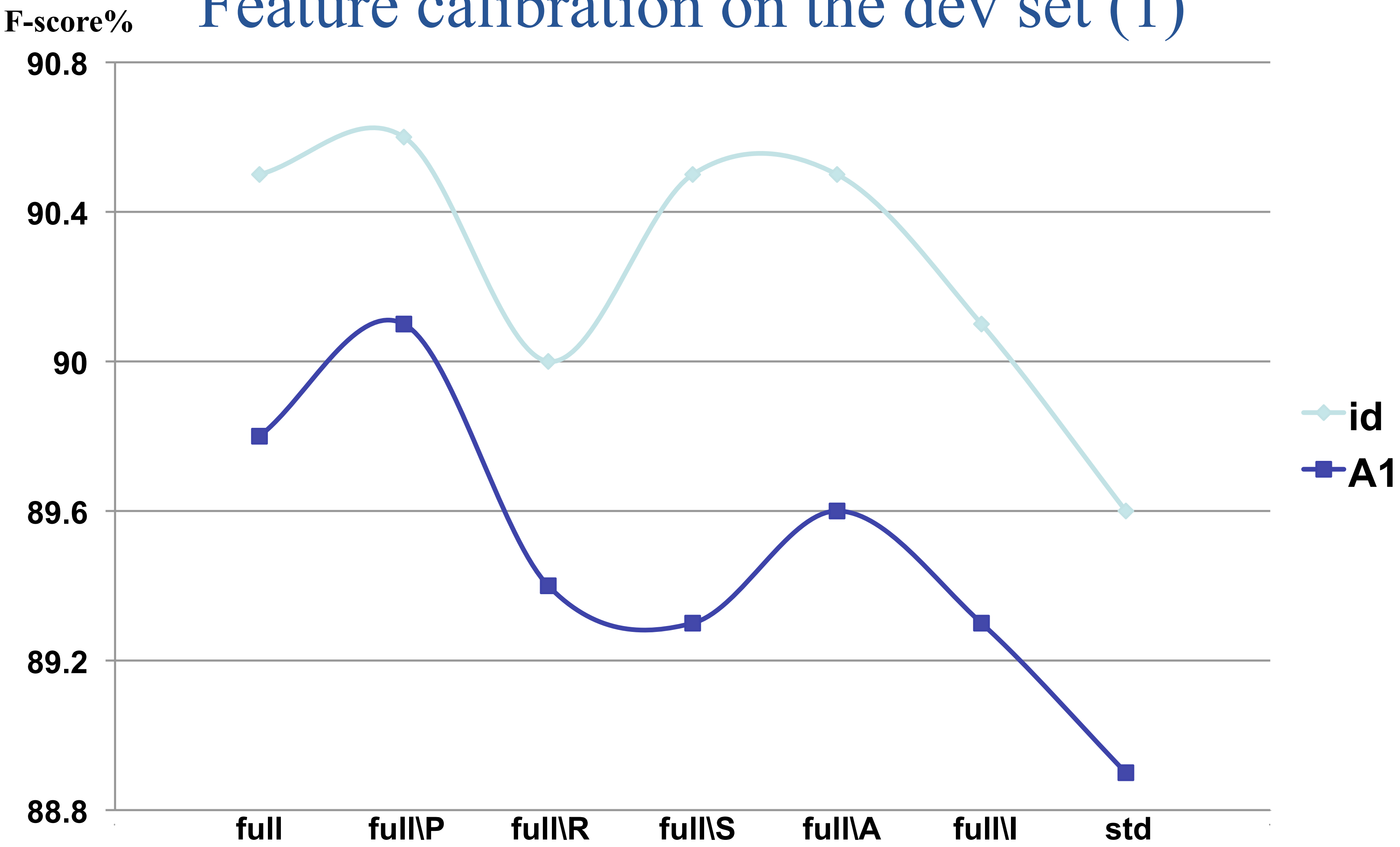
- Data:
  - PropBank is an annotated corpus with semantic roles
  - PropBank Section 02-21 for training; Section 23 for testing
  - PropBank Section 24 for feature calibration
- Argument Set Under Consideration:
  - {*A0, A1, A2, A3, A4, AM*}
- Machine learning model for identification & classification: support vector machine (SVM)
  - SVM-light (Joachims, 1999)
  - polynomial kernel with degree 3
  - 30% training data for parameter tuning
- Measures: precision/recall/F-score

Overall performance on Charniak's parser



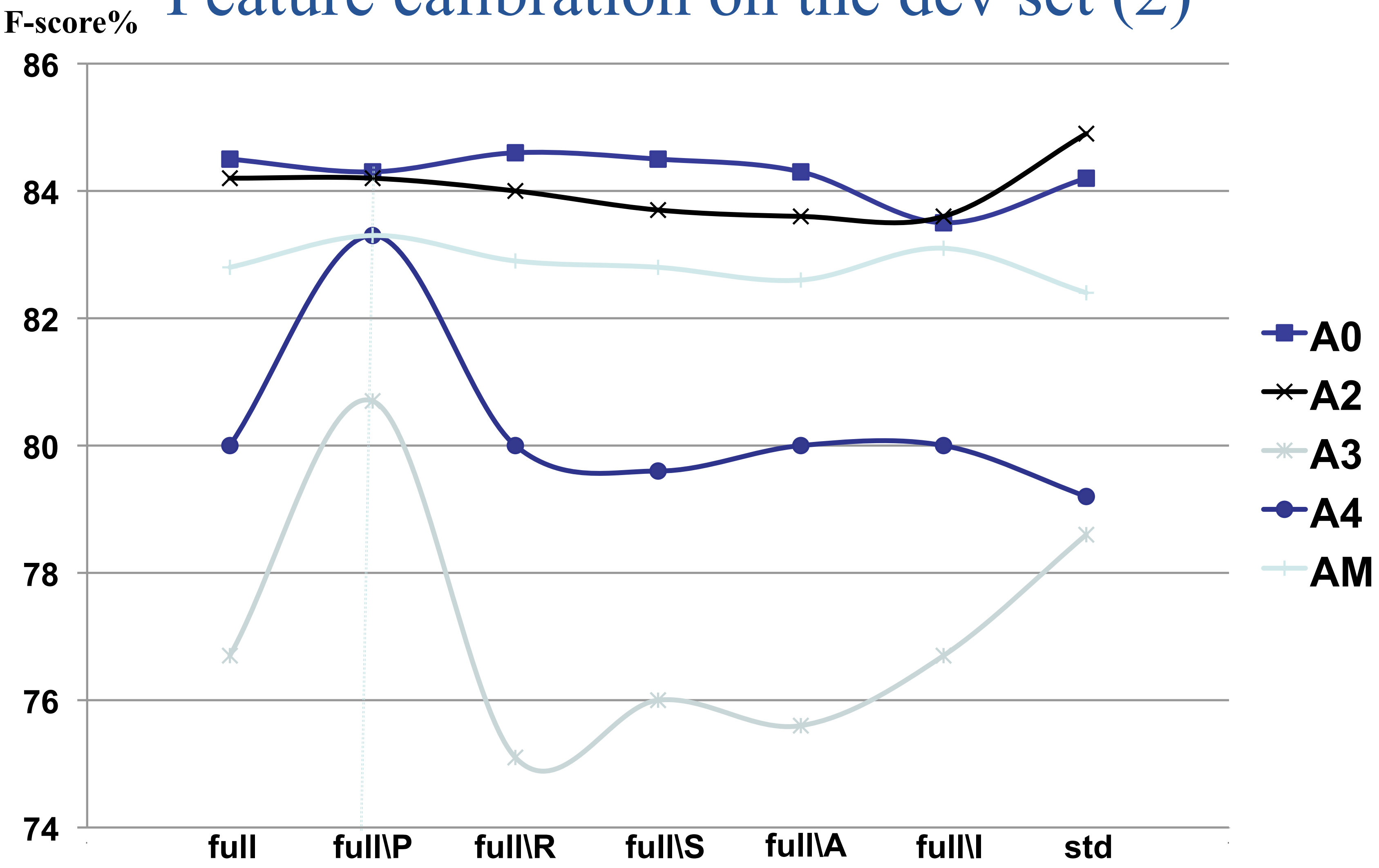
A combination of features proposed by (Gildea and Jurafsky, 2002), (Surdeanu et al., 2003; Pradhan et al., 2004; Pradhan et al., 2005) and (Xue and Palmer, 2004)

Feature calibration on the dev set (1)



P: predicate, A: argument, I: intermediate, R: topological, S: subcat

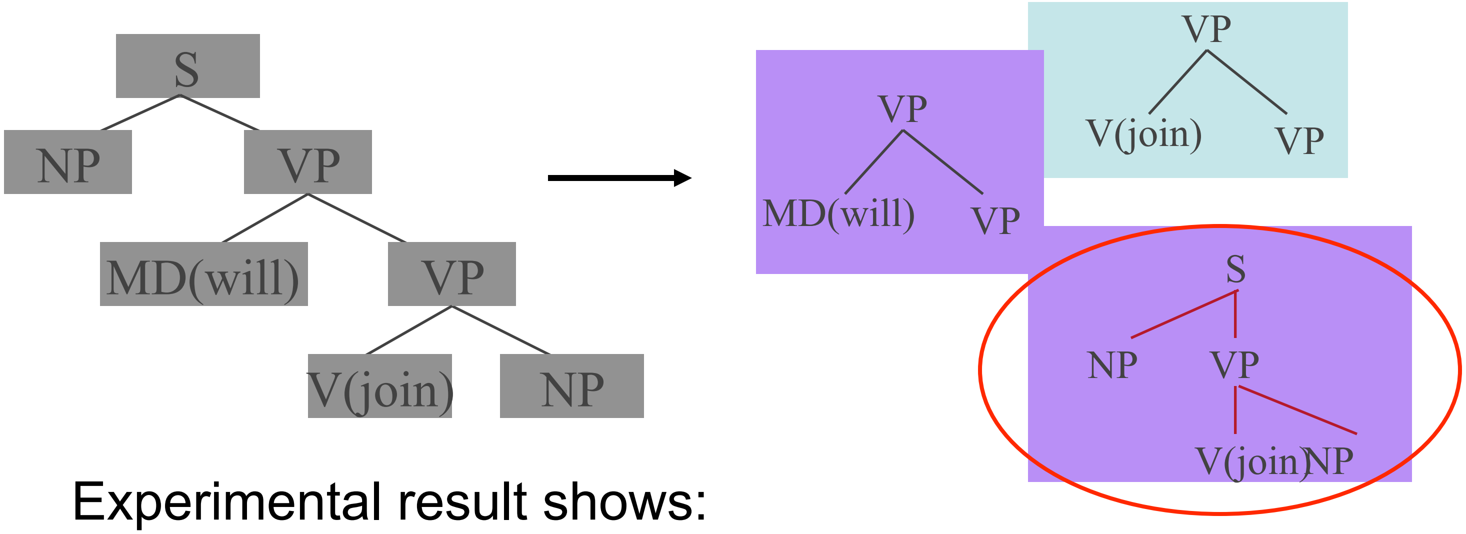
Feature calibration on the dev set (2)



P: predicate, A: argument, I: intermediate, R: topological, S: subcat

Tree kernel v.s. LTAG

What is the difference of using *subtrees provided by LTAG* and *all possible subtrees* as features?



Experimental result shows:

tree kernel based SRL v.s. LTAG feature based SRL

F-score: 83.53% → 85.25%

Experimental result

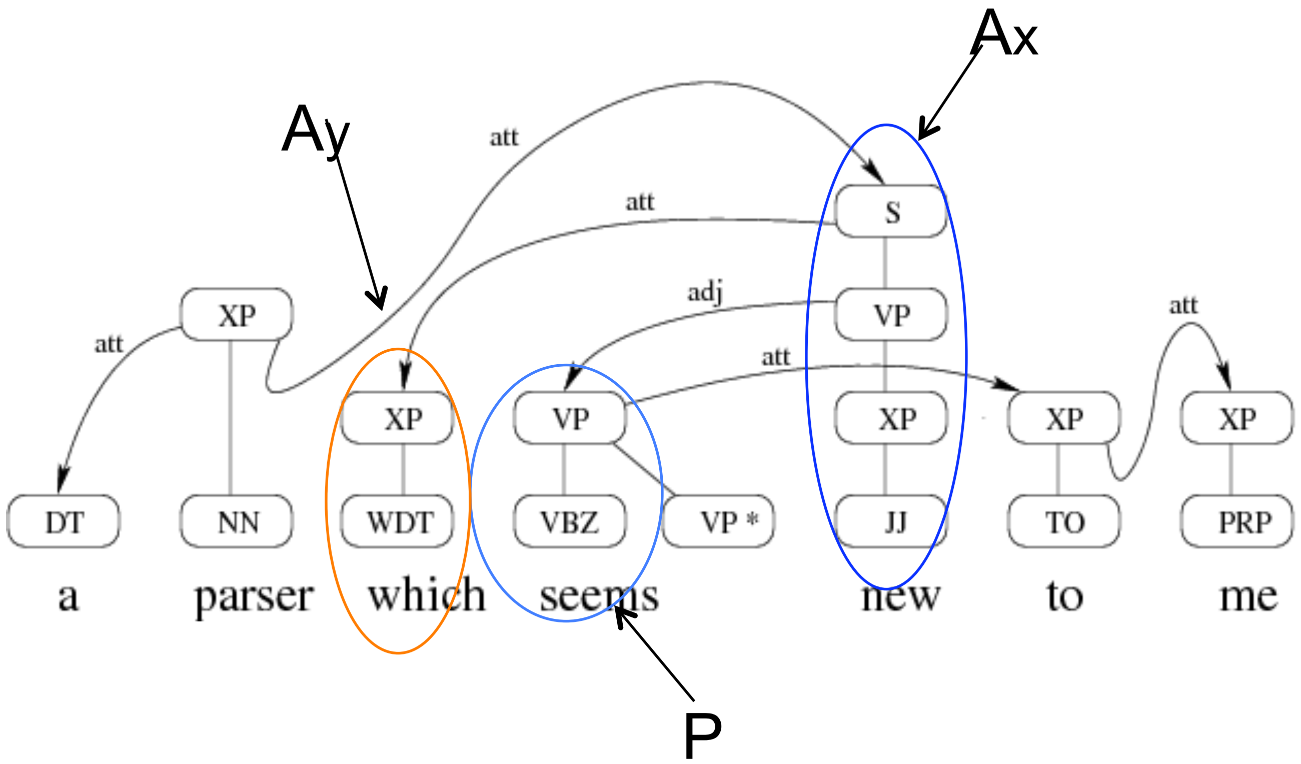
- LTAG-based features v.s. Predicate-Argument Structure features (Moschitti, 2004):
  - tree kernel over PAS + std
  - F-score: 85.25% v.s. 83.53%

  Statistically significant (using SIGF)
- CoNLL-2005 shared task:
  - std v.s. std+ltag: 74.41% v.s. 75.31% (F-score)

Using LTAG-spinal Treebank for SRL

- To explore the impact of different types of LTAG derivation trees on the SRL task:
  - The LTAG derivation trees we used are converted from constituency parses.
  - LTAG-spinal treebank (Shen & Joshi, 2005a) was extracted from the TreeBank using PropBank; therefore appears more suitable for SRL.
  - LTAG-spinal parser (Shen & Joshi, 2005b) is now available.
  - LTAG-spinal was for syntax – we use it for SRL.

Patterns in Spinal-LTAG TreeBank: P←A and P←Ax→Ay

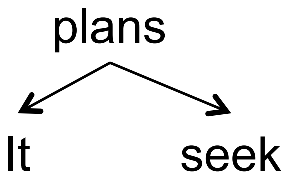


Oracle test shows that 8 patterns account for **95.5%** pred-arg pairs in TreeBank trees

Spinal-LTAG patterns

(Shen, Champollion, Joshi 2008)

1. P→A
   - (What)$_{arg1}$ will **happen** (to dividend growth)$_{arg2}$
2. P←A (relative clause, predicate adjunction)
   - (The amendment)$_{arg0}$ which **passed** today
3. P←Px→A (subject and object control)
   - (It)$_{arg0}$ plans to **seek** approval (Px = plans)



plans

It

seek

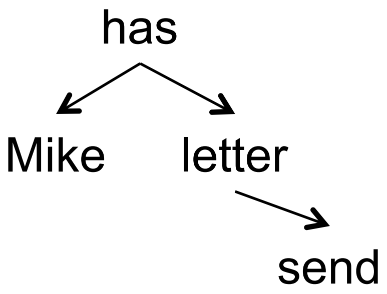
4. P←Coord→Px→A (shared arguments)
   - (Chrysotile fibers)$_{arg1}$ are curly and more easily **rejected** by the body (Px = are)

Spinal-LTAG patterns

(Shen, Champollion, Joshi 2008)

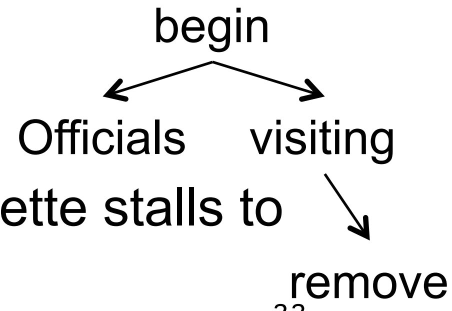
5. V←A (modifier as predicate)
   - The Dutch **publishing** (group)$_{\text{arg0}}$

6. P←Ax←Py→A
   - (Mike)$_{\text{arg1}}$ has a letter to **send** (Ax = letter, Py = has)

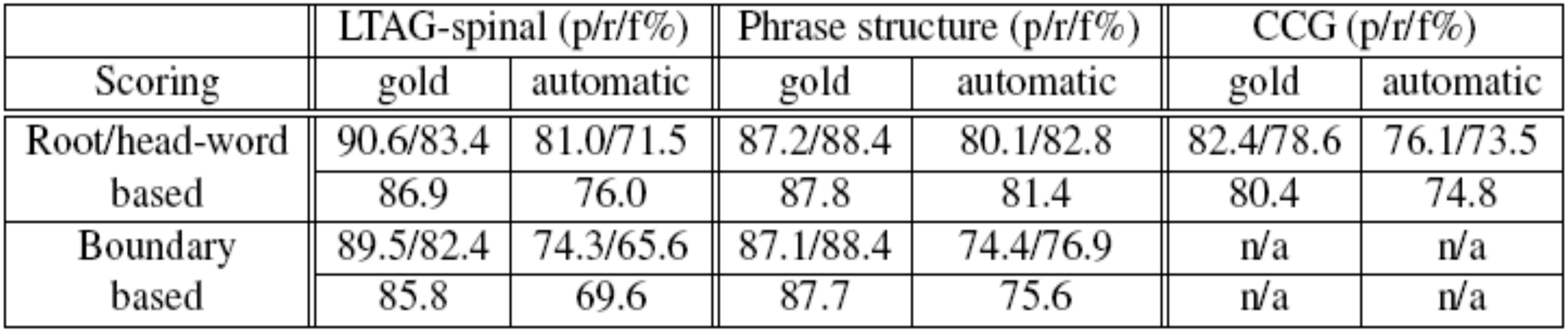


7. P←Coord←Px→A (control plus coordination)
   - (It)$_{\text{arg0}}$ expects to **obtain** regulatory approval and **complete** the transaction (Px = expects)

8. P←Px←Py→A (chained control)
   - (Officials)$_{\text{arg0}}$ began visiting about 26,000 cigarette stalls to **remove** illegal posters



Experimental results



	LTAG-spinal (p/r/f%)		Phrase structure (p/r/f%)		CCG (p/r/f%)	
Scoring	gold	automatic	gold	automatic	gold	automatic
Root/head-word	90.6/83.4	81.0/71.5	87.2/88.4	80.1/82.8	82.4/78.6	76.1/73.5
based	86.9	76.0	87.8	81.4	80.4	74.8
Boundary	89.5/82.4	74.3/65.6	87.1/88.4	74.4/76.9	n/a	n/a
based	85.8	69.6	87.7	75.6	n/a	n/a

- CCG results from (Gildea and Hockenmaier, 2003)
- For gold parses:
  - *precision*(spinal) > *precision*(phrase_structure),
  - However, *recall*(spinal) < *recall*(phase_structure).
- For automatic parses: same trend, larger gap between recall.

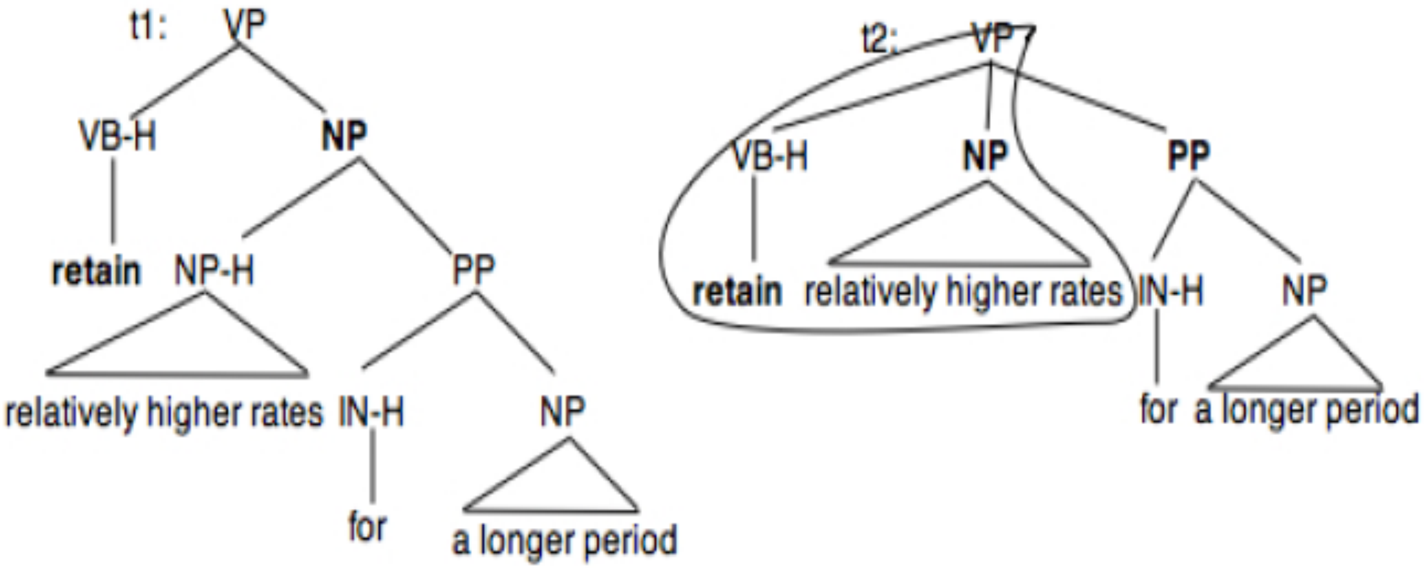
Accuracy Improvement

- Candidate selection strategy brings down the recall:
  - In automatic parses 8 patterns only capture 83.9% of pred-arg pairs (v.s. 95.5% in gold parses)
  - Increasing the recall is critical for further improvement
  - More candidates should be taken into account: all nodes along the predicate-root path should be considered.

Using Parse Forests for SRL

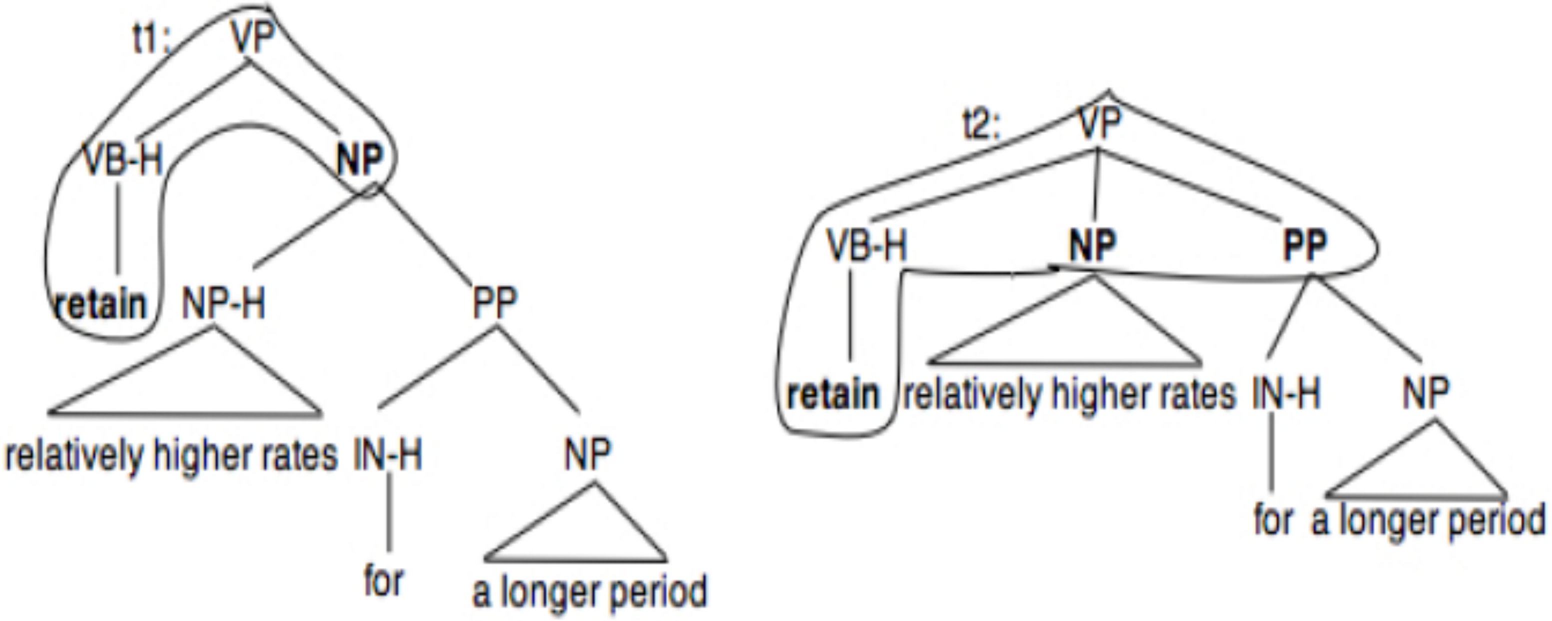
- A node that corresponds to the semantic argument exists in the i-th (i≠1) tree in the top-N parses.
- Oracle test on WSJ Section 00 shows that 98.64% of the arguments (v.s. 98.65% in gold trees) can be captured when N = 100 in automatic parses.

An example: tree 2 is better than tree 1



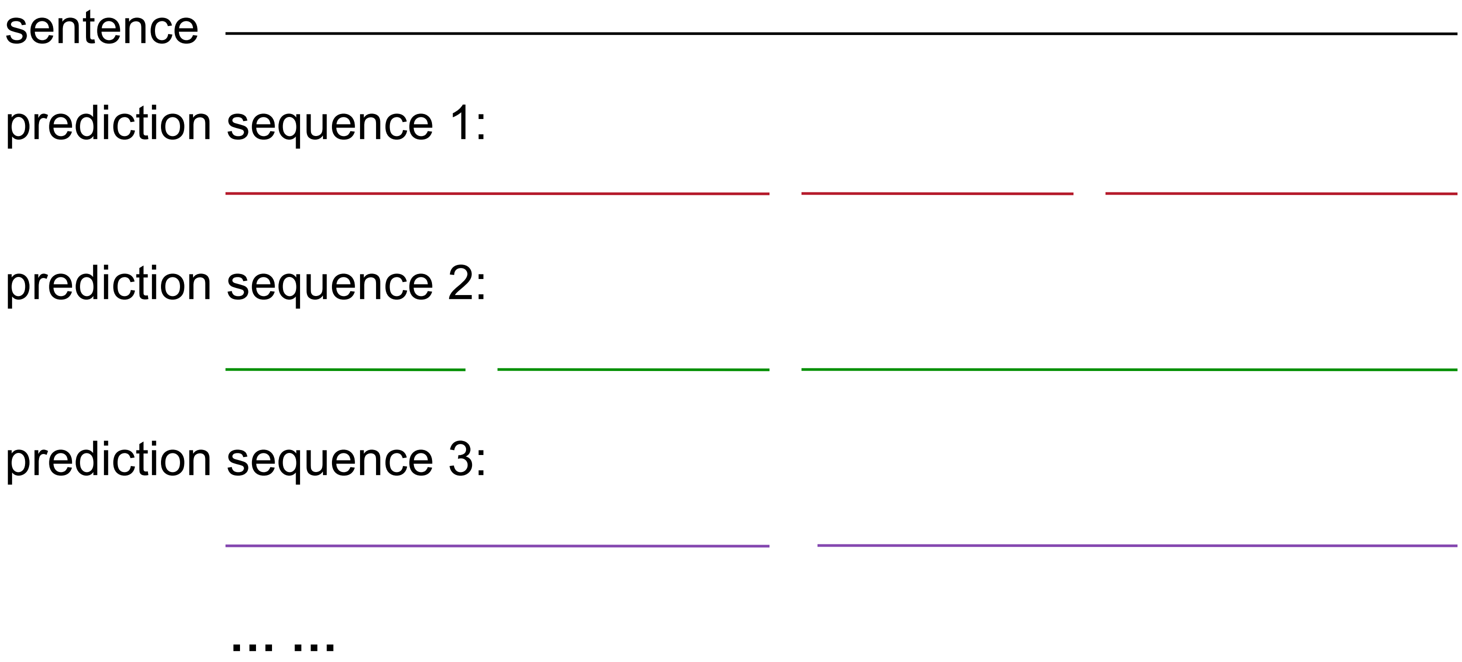
Predicate-Argument kernel based method

- Predicate-argument kernel +sub-categorization frame
- Expectation: score(*t2*) > score(*t1*)



Inference based method

- Each prediction sequence is produced based on one parse in top-*N* parses.



sentence

prediction sequence 1:

prediction sequence 2:

prediction sequence 3:

… …

Inference based method (cont'd)

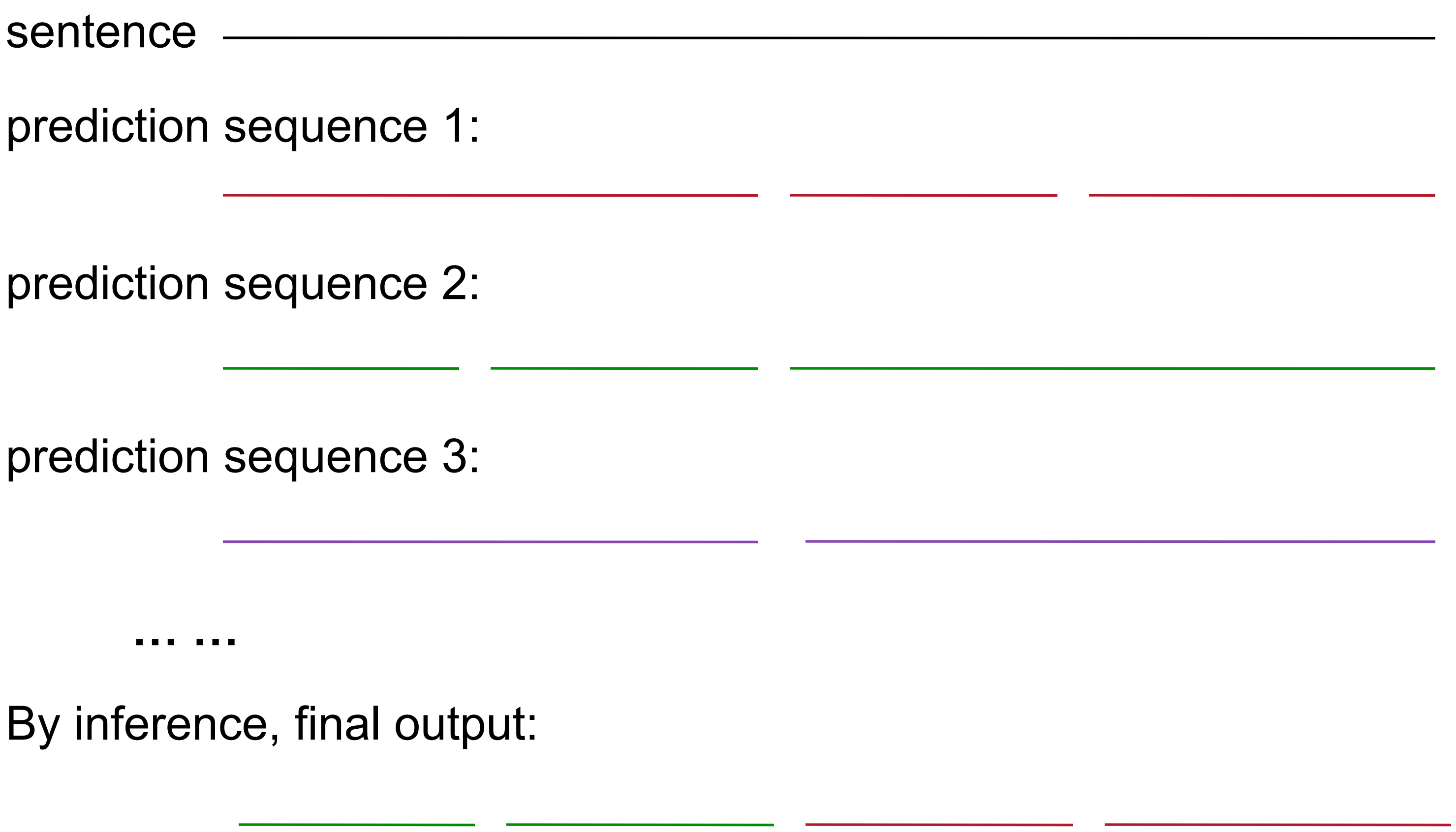
- To produce the final single prediction, an inference procedure is given to maximize the objective function as follows (Punyakanok et al, 2005a):

$$\hat{c}^{1:M} = \operatorname*{argmax}_{c^{1:M} \in L^M} \sum_{i=1}^{M} Prob(S^i = c^i),$$

L is the argument set and argument sequence is indexed from 1 to *M*

Probability of sentence portion *i* assigned semantic label *c*

Inference based method (cont'd)



sentence

prediction sequence 1:

prediction sequence 2:

prediction sequence 3:

… …

By inference, final output:

Summary

- LTAG based features can improve SRL accuracy.
- LTAG-spinal Treebank combines PropBank information with TreeBank information to create LTAG derivation trees.
- LTAG-spinal TreeBank was used to build an SRL system.
- Parse forests can increase the robustness of SRL to parser errors.