Inducing a Discriminative Parser to Optimize Machine Translation Reordering

Graham Neubig[1,2,3], Taro Watanabe[2], Shinsuke Mori[1]

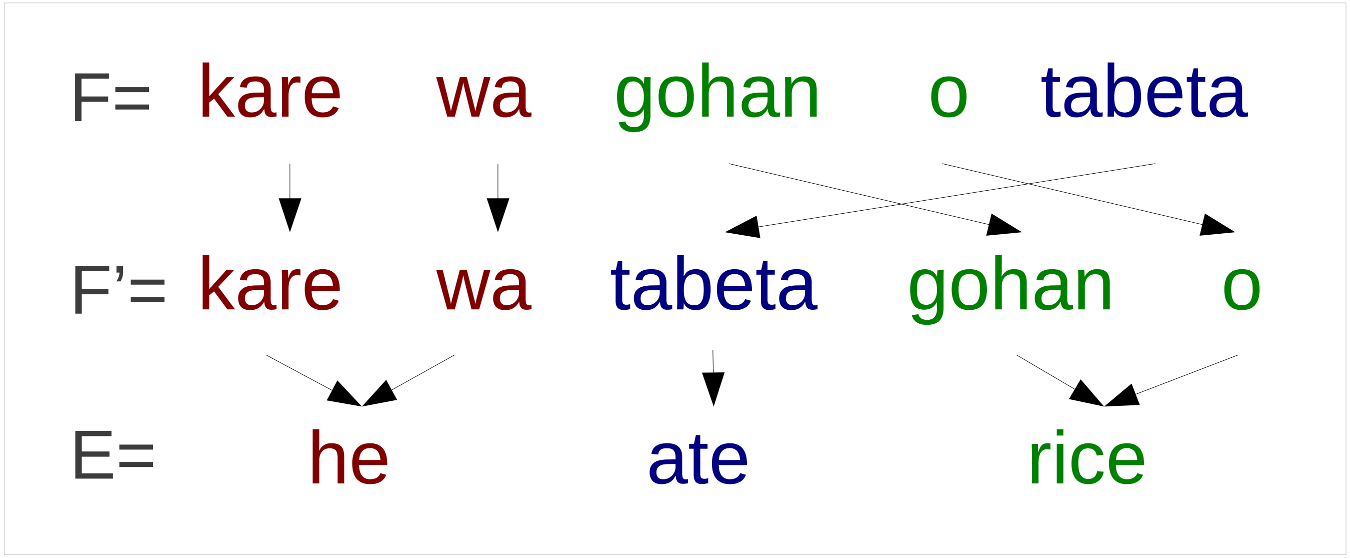
1 KYOTO UNIVERSITY KYOTO JAPAN FOUNDED 1897

2 NICT

3 now at Nara Institute of Science and Technology NAIST

Preordering

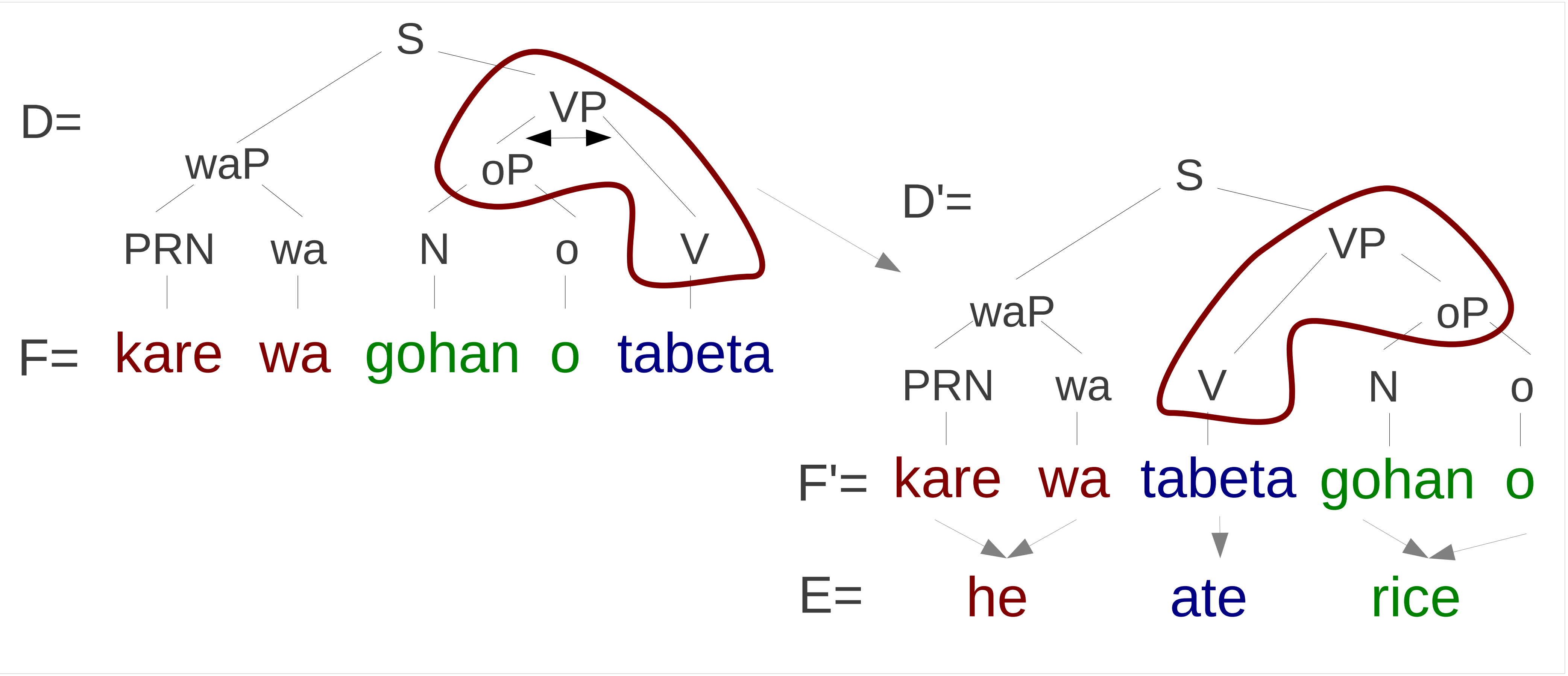
- Long-distance reordering is a weak point of SMT
- Preordering first reorders, then translates



- A good preorderer will effectively find F' given F

Syntactic Preordering

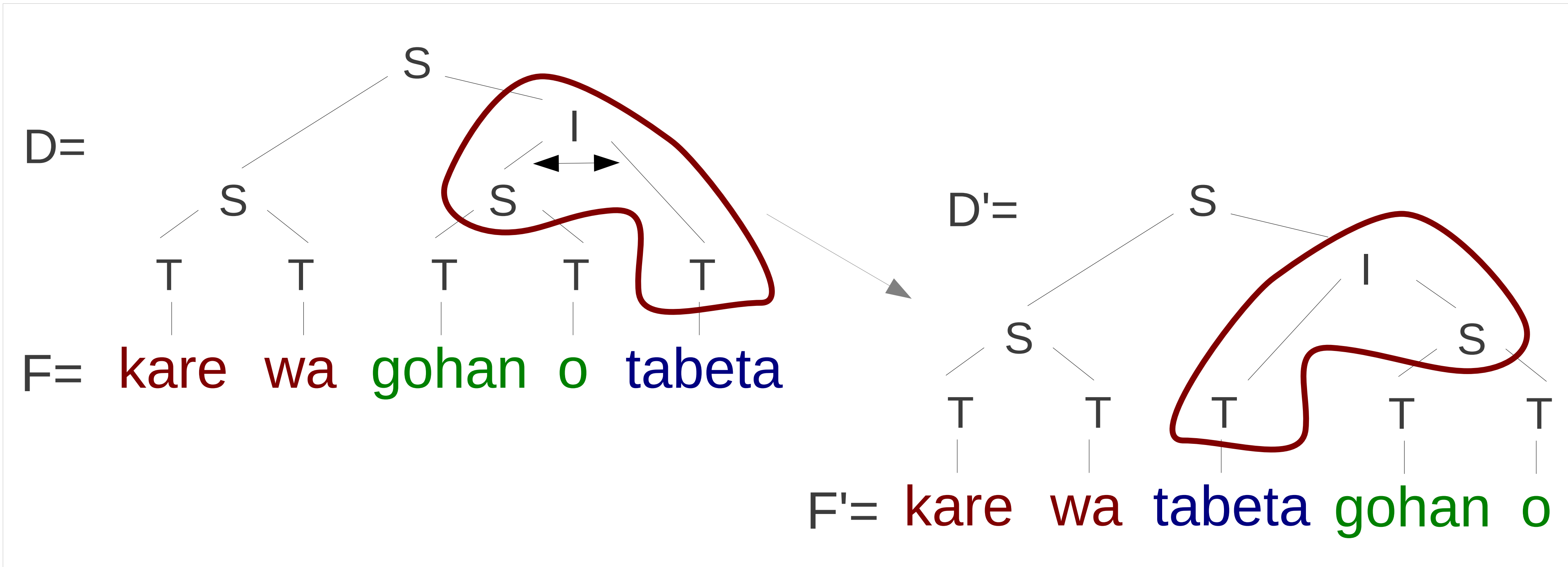
- Define rules over a syntactic parse of the source



- What if we don't have a parser in the source language?

Bracketing Transduction Grammars [Wu 97]

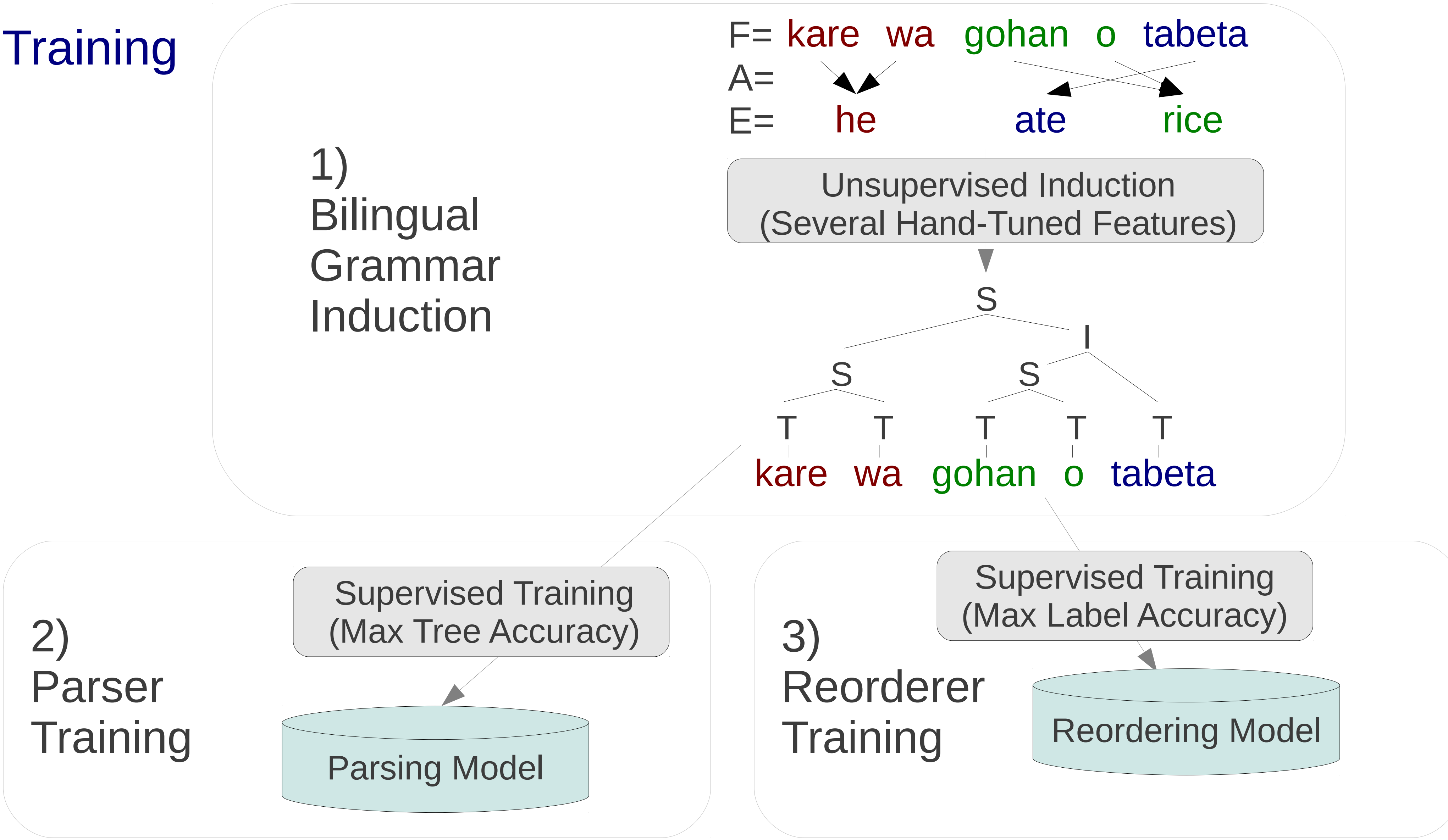
- Binary CFGs with only straight (S) and inverted (I) non-terminals, and pre-terminals (T)



- Language independent
- BTG tree uniquely defines a reordering

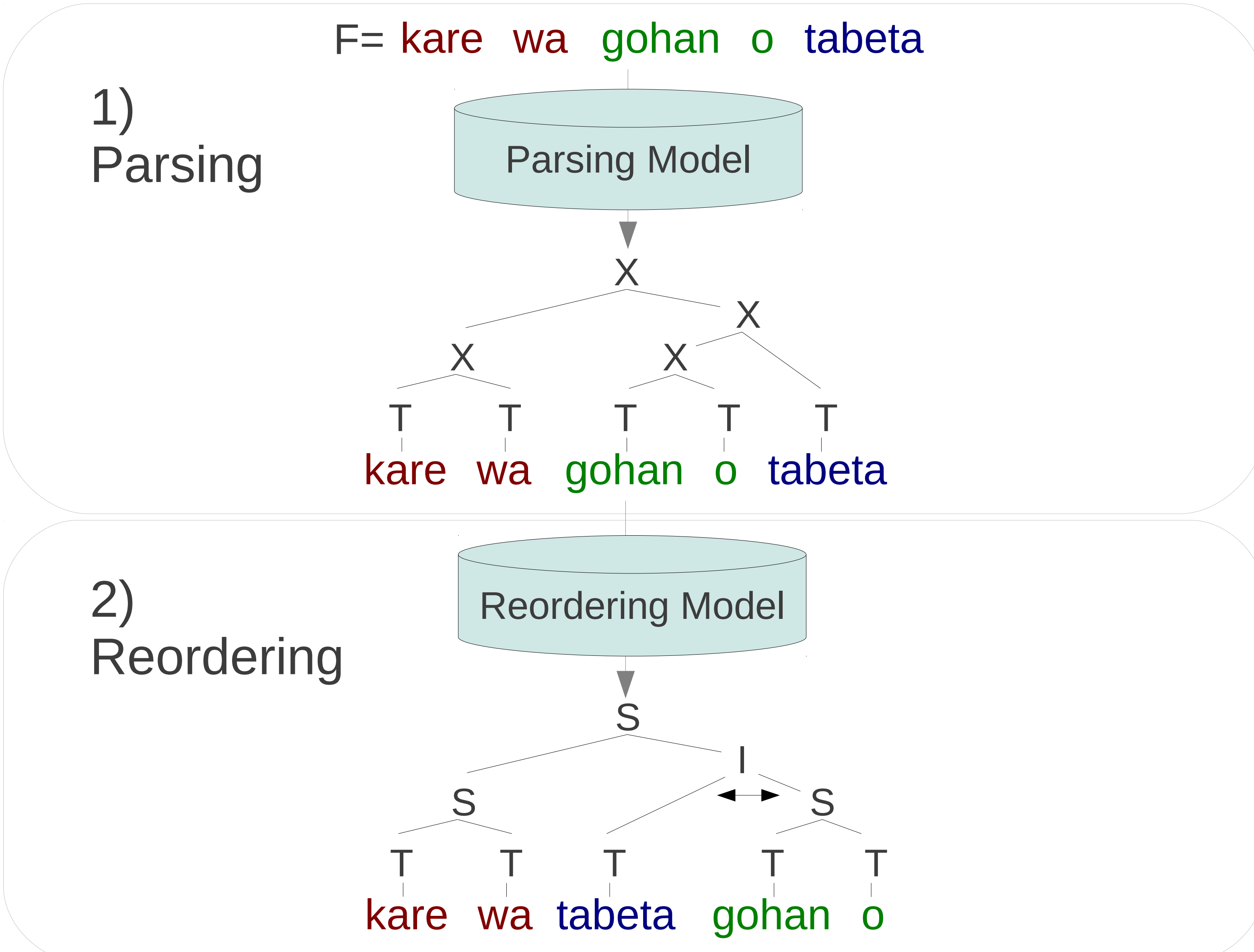
3-Step BTG Grammar Training for Reordering [DeNero+ 11]

Training



3-Step BTG Grammar Induction for Reordering [DeNero+ 11]

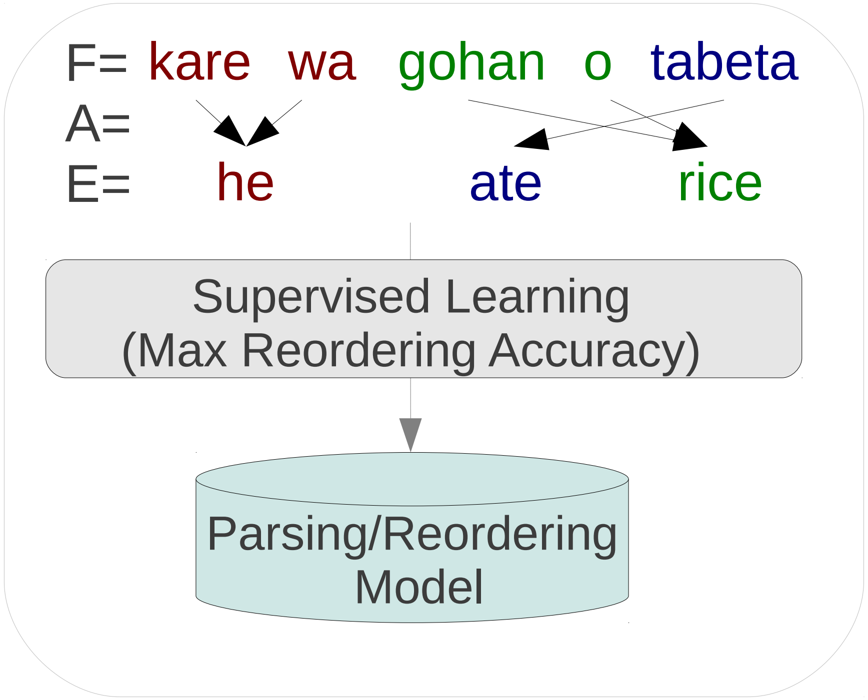
Testing



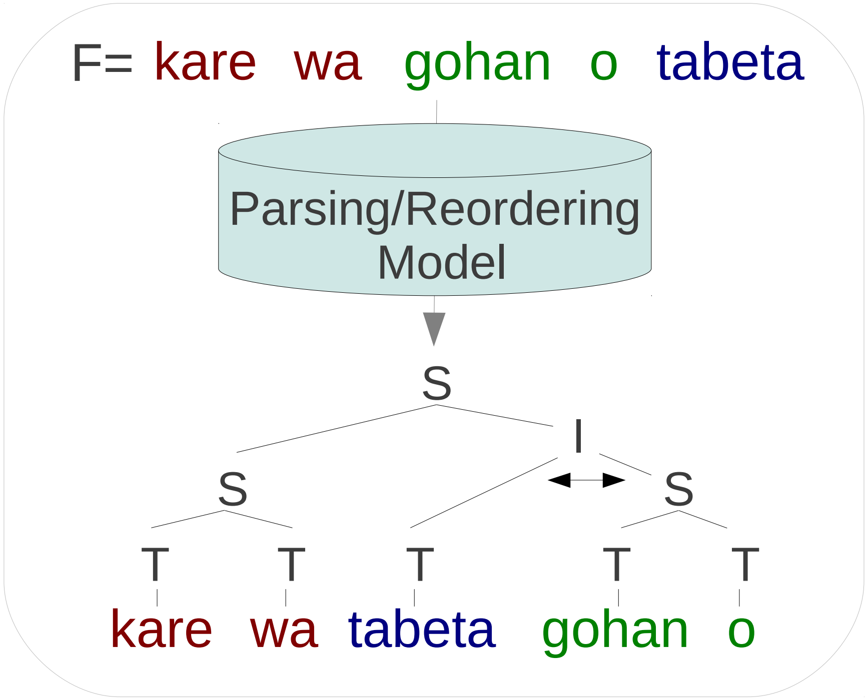
Our Work: Inducing a Parser to Optimize Reordering

- What if we can reduce three steps to one, and directly maximize ordering accuracy?

Training



Testing



Optimization Framework

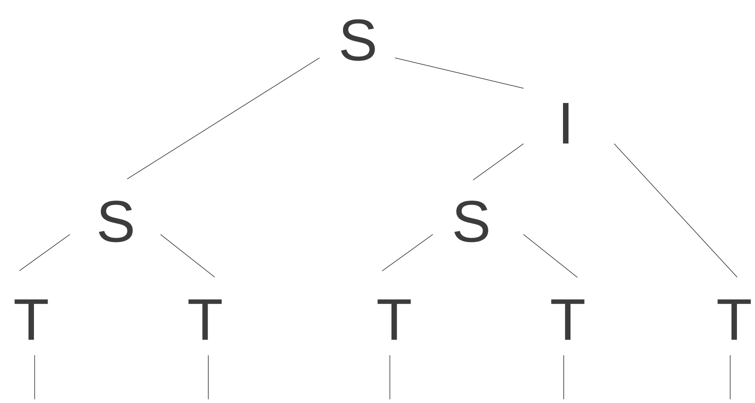
Optimization Framework

- Input: Source sentence F

  F= kare wa gohan o tabeta
- Output: Reordered source sentence F'

  F'= kare wa tabeta gohan o
- Latent: Bracketing transduction grammar derivation D

  D=



Scores and Losses

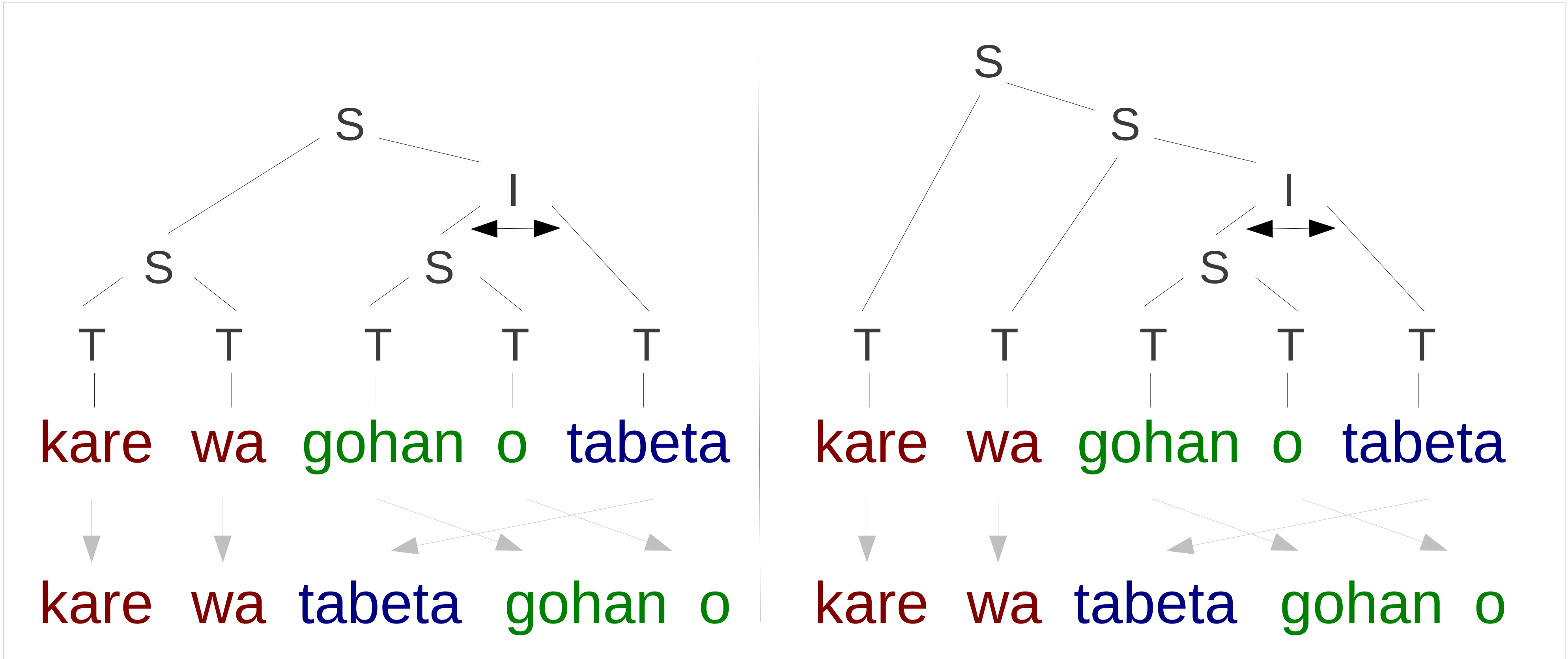
- Define a score over source sentences and derivations

$$S(F, D; \boldsymbol{w}) = \sum_i w_i * \phi_i(F, D)$$

- Optimization finds a weight vector that minimizes loss

$$\underset{\boldsymbol{w}}{\operatorname{argmin}} \sum_{F, F'} L\left(F'^{*}, \underset{F' \leftarrow F, D}{\operatorname{argmax}} S(F, D; \boldsymbol{w})\right)$$

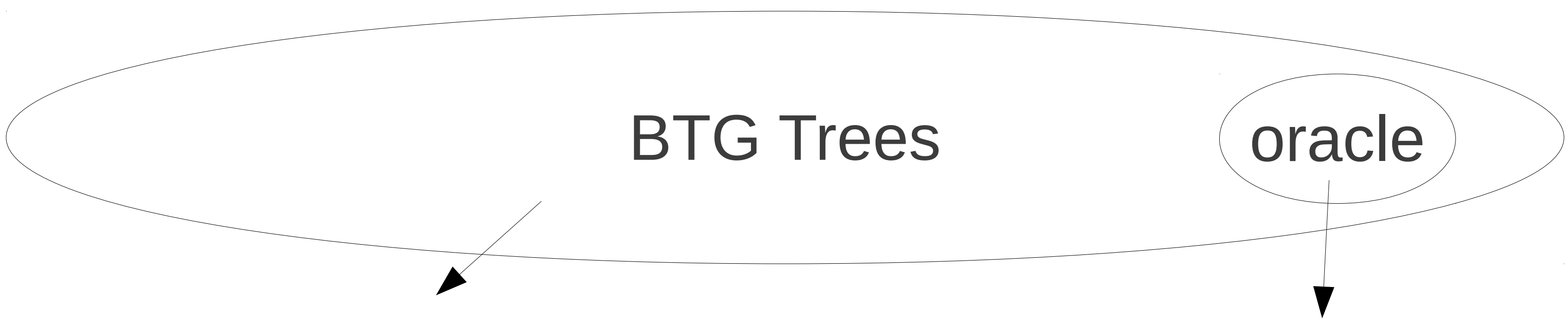
Note: Latent Variable Ambiguity



- Out of these, we want easy-to-reproduce trees
- [DeNero+ 11] finds trees with bilingual parsing model
- Our model discovers trees during training

Training: Latent Online Learning

- Find
  - model parse of maximal score and
  - oracle parse of maximal score among parses of minimal loss



$$D = \underset{\tilde{D}}{\operatorname{argmax}}\, S(F, \tilde{D}) \qquad \hat{D} = \underset{\tilde{D} \in \operatorname{argmin}_D L(F, D)}{\operatorname{argmax}} S(F, \tilde{D})$$

- Adjust weights (example: perceptron)

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \phi(F, \hat{D}) - \phi(F, D)$$

Considering Loss in Online Learning

- Consider loss (how bad is the mistake?)

kare wa tabeta gohan o	reference (L=0)
kare wa gohan tabeta o	L=1
o gohan tabeta wa kare	L=8

- Make it easy to choose trees with high loss in training

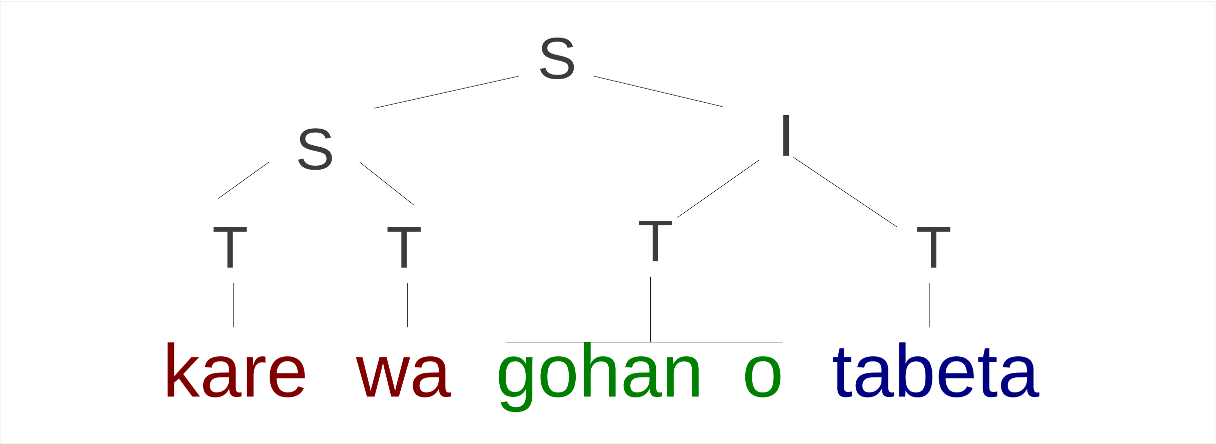
→To avoid high-loss trees, must give a large penalty

$$D = \underset{\tilde{D}}{\operatorname{argmax}}\, S(F, \tilde{D}) + L(F, \tilde{D})$$

Parser

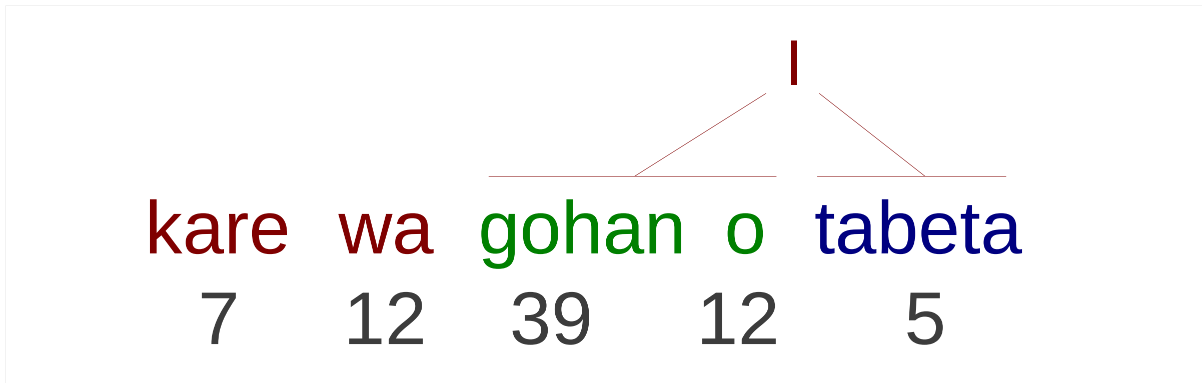
Parsing Setup: Standard Discriminative Parser

- Features independent with respect to each node
  - Parsing, reordering possible in $O(n^3)$ with CKY
- Multi-word pre-terminals allowed



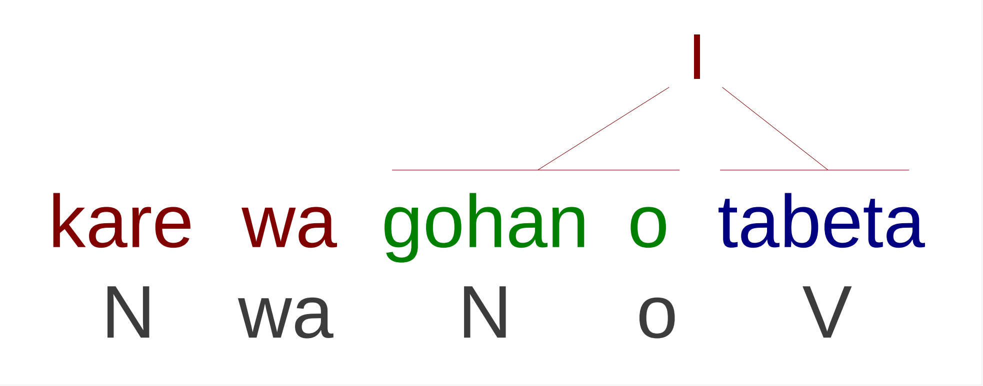
Language Independent Features

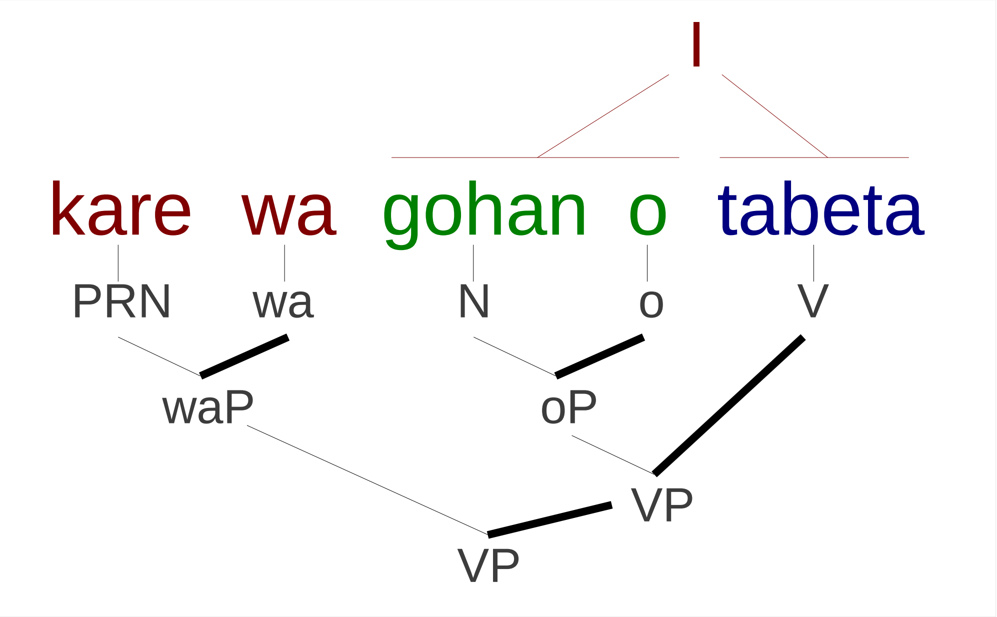
- No linguistic analysis necessary



- **Lexical:** Left, right, inside, outside, boundary words
- **Class:** Same as lexical but induced classes
- **Phrase Table:** Whether span exists in phrase table
- **Balance:** Left branching or right branching?

Language Dependent Features





- POS Features: Same as lexical, but over POSs
- CFG Features: Whether nodes match supervised parser's spans

Reordering Losses

Reordering Losses [Talbot+ 11]: Chunk Fragmentation

- How many chunks are necessary to reproduce reference?

System Reordering: <s> kare wa | gohan o | tabeta | </s>

Reference Reordering: <s> kare wa | tabeta | gohan o | </s>

<u>Loss:</u>

$$L_{chunk}(F, \tilde{D}) = \text{Number of Chunks - 1}$$

<u>Accuracy:</u>

$$A_{chunk}(F, \tilde{D}) = \text{1 - (Number of Chunks - 1)/(J+1)}$$

Reordering Losses [Talbot+ 11]: Kendall's Tau

- How many pairs of reversed words?

System Reordering:	kare	wa	gohan	o	tabeta
Reference Reordering:	kare	wa	tabeta	gohan	o

Loss:

$$L_{tau}(F, \tilde{D}) = \text{ Reversed Words}$$

Accuracy:

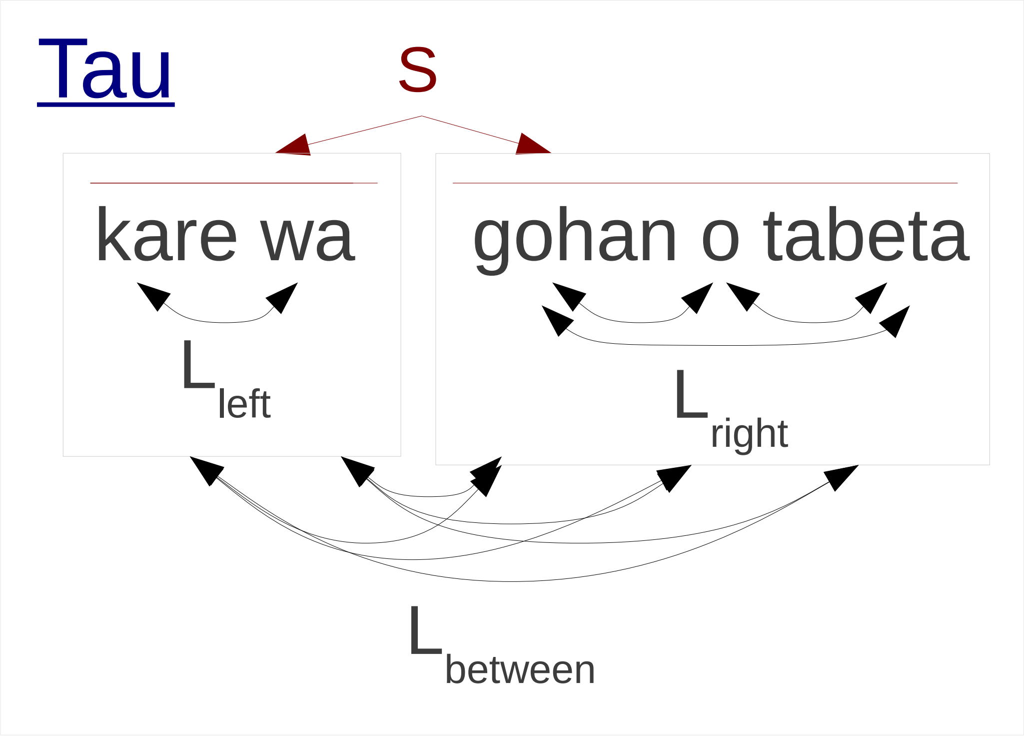
$$A_{tau}(F, \tilde{D}) = \text{ 1 - Reversed Word/Potential Reversed Words}$$

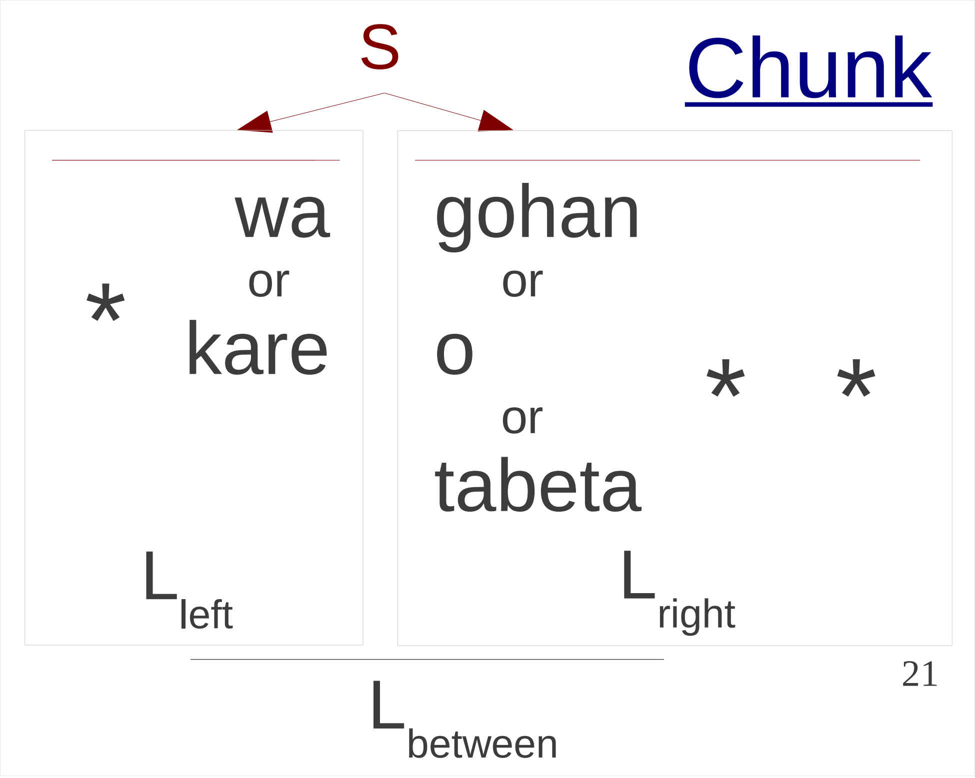
Calculating Loss by Node

- Large-margin training, must calculate loss efficiently

$$D = \operatorname*{argmax}_{\tilde{D}} S(F, \tilde{D}) + L(F, \tilde{D})$$

- Can factor loss by node as well (detail in paper)





Experiments

Experimental Setup

- English-Japanese and Japanese-English translation
- Data from the Kyoto Free Translation Task

	sent.	word (ja)	word (en)
RM-train	602	14.5k	14.3k
RM-test	555	11.2k	10.4k
LM/TM	329k	6.08M	5.91M
tune	1166	26.8k	24.3k
test	1160	28.5k	26.7k

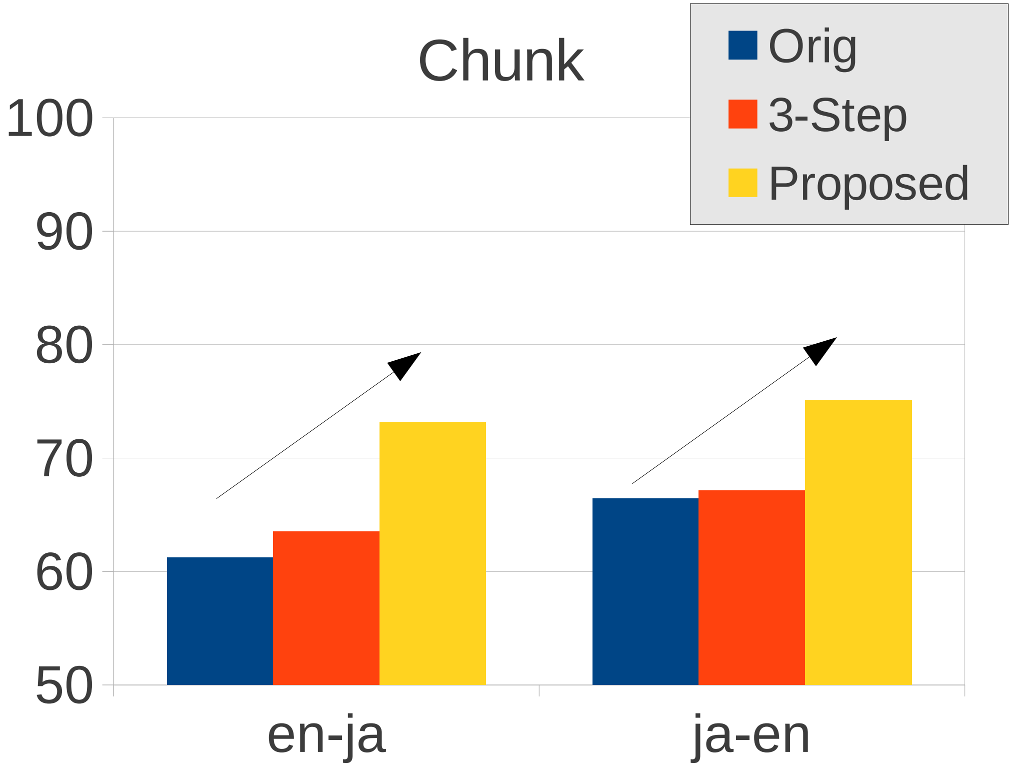
Manually Aligned

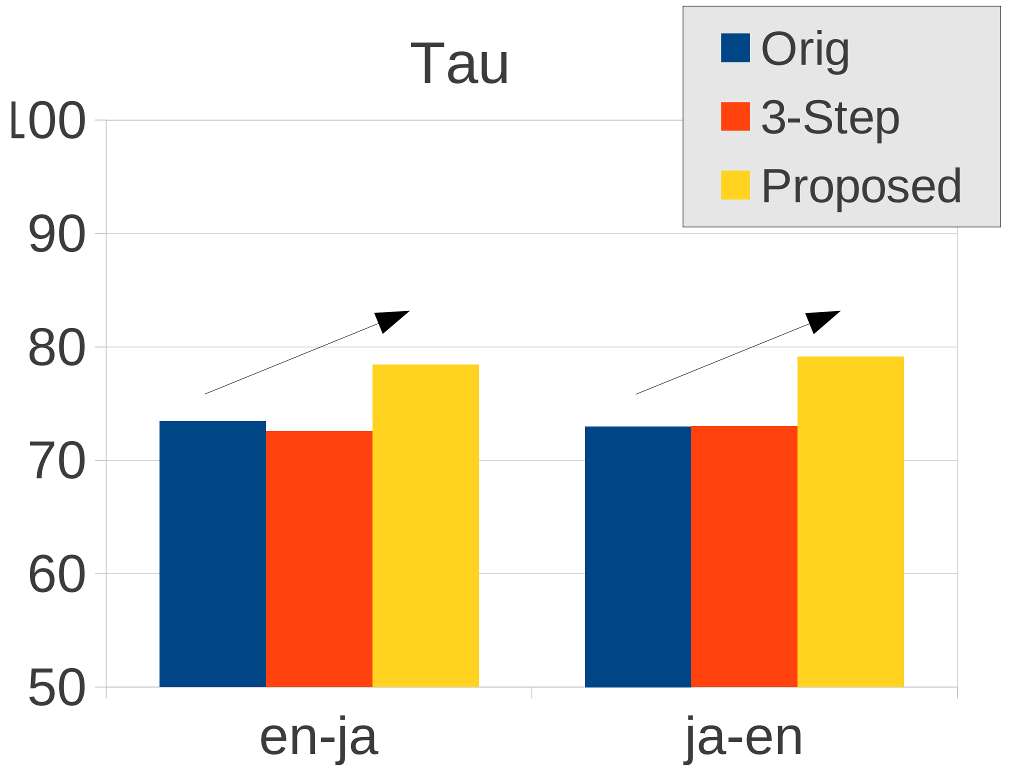
Experimental Setup

- Reordering Model Training:
  - 500 iterations
  - Using Pegasos with regularization constant 10^{-3}
  - Default: chunk fragmentation loss, standard features
- Translation: Moses with lexicalized reordering
- Compare: Original order, 3-step training, the proposed method

Result:
Proposed Model Improves Reordering

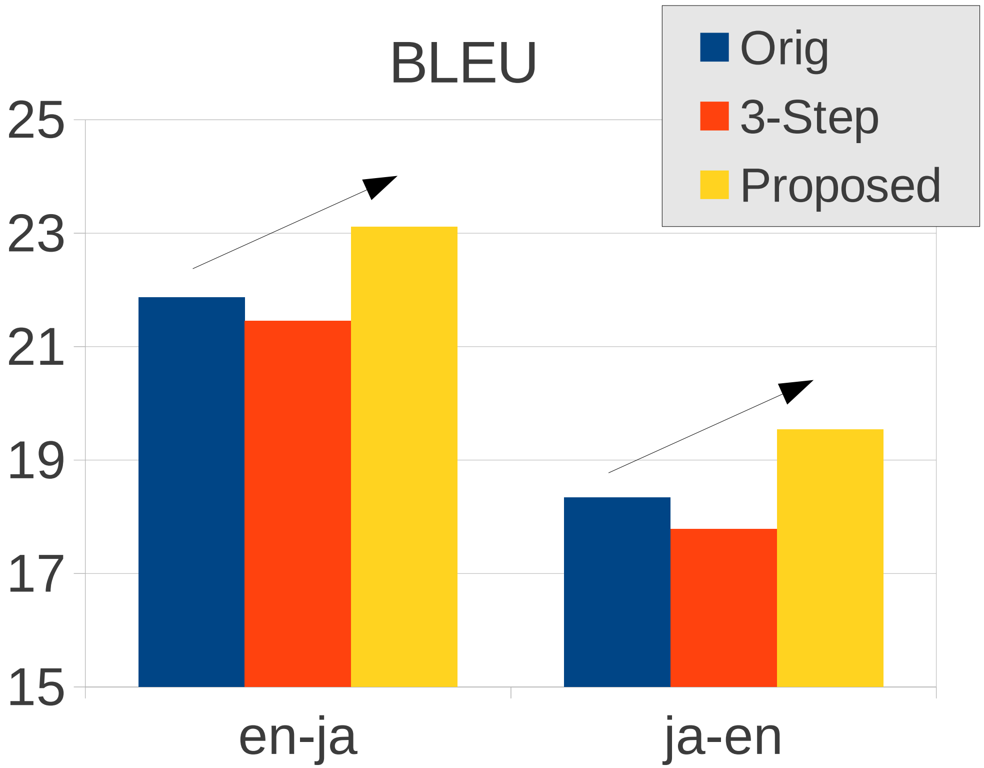
- Results for chunk fragmentation/Kendall's Tau

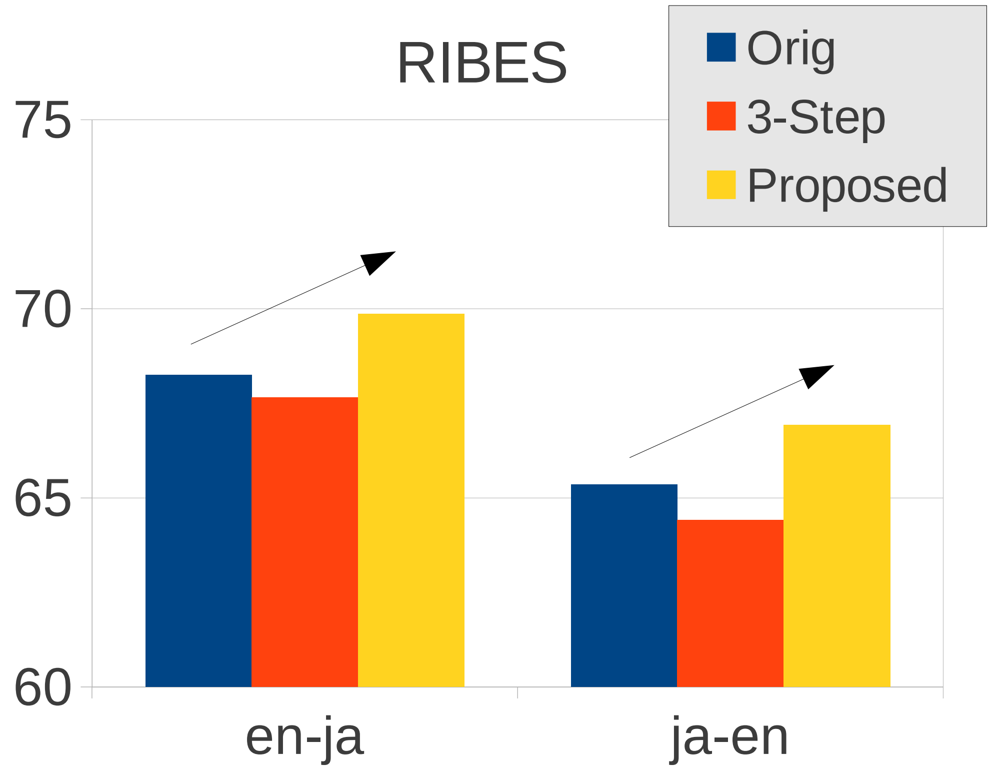




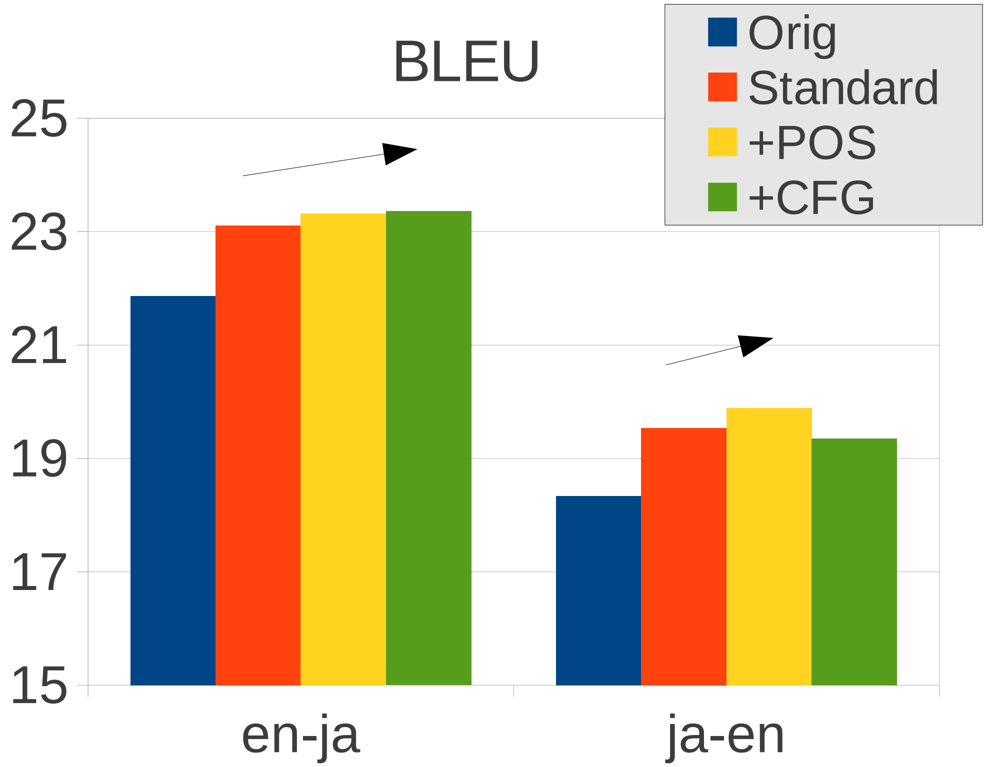
Result:
Proposed Model Improves Translation

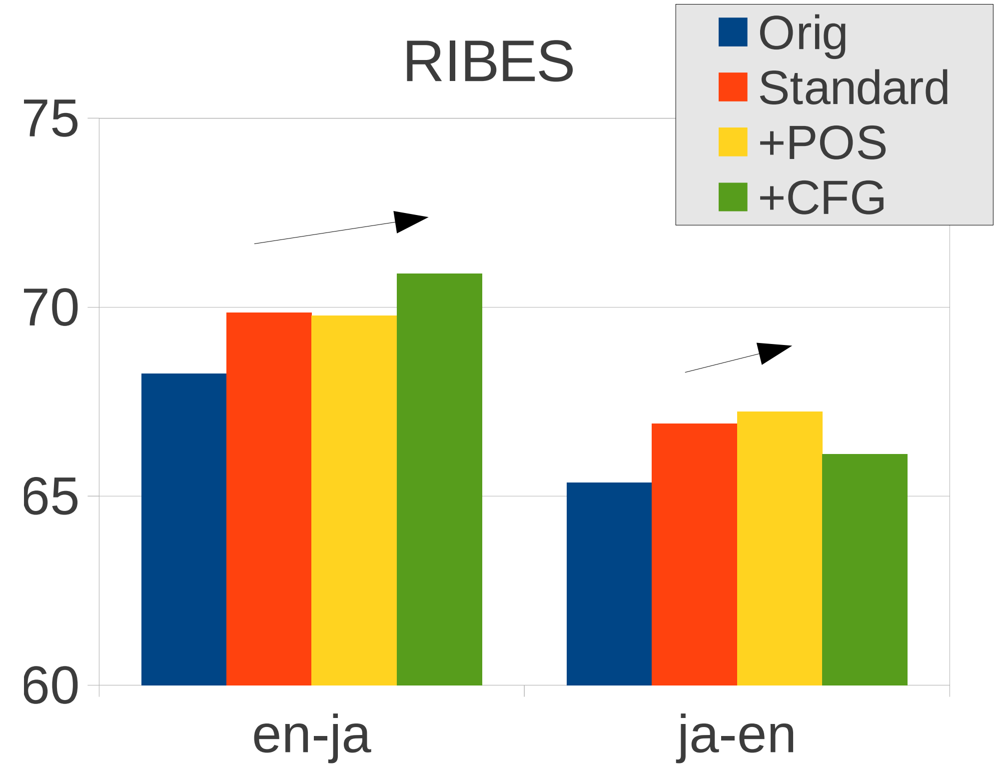
- Results for BLEU and RIBES:





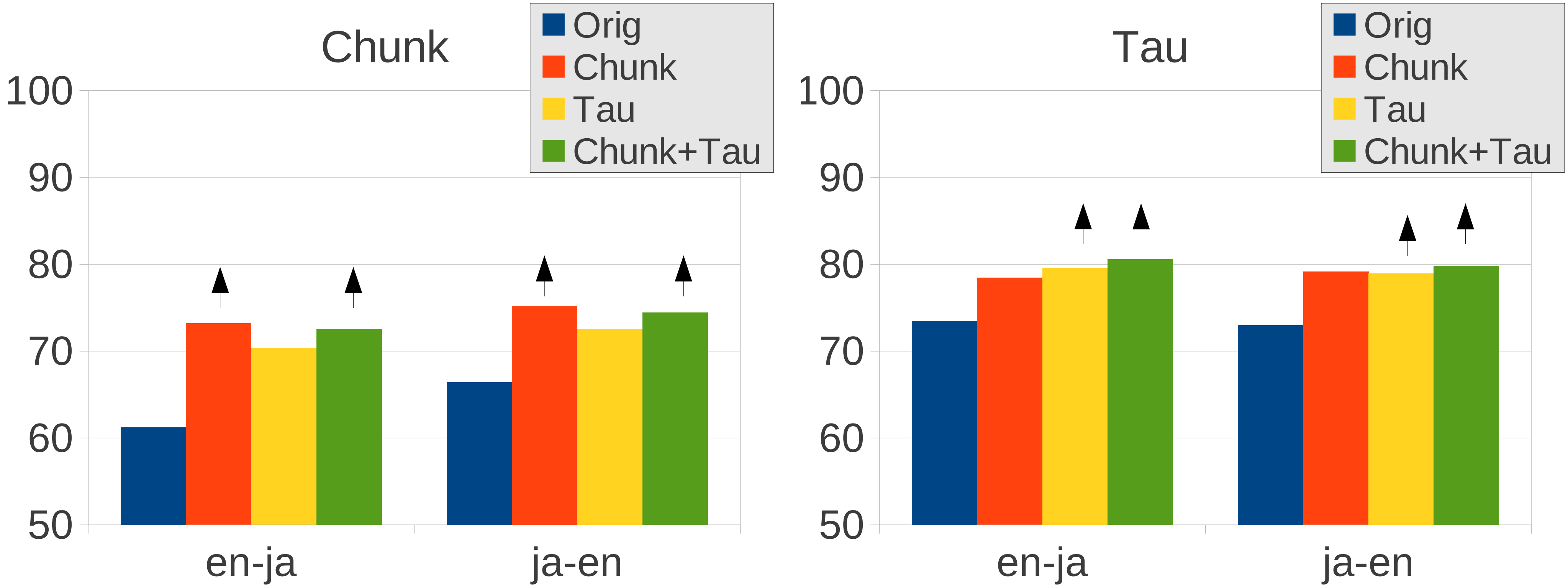
Result:
Adding Linguistic Info (Generally) Helps





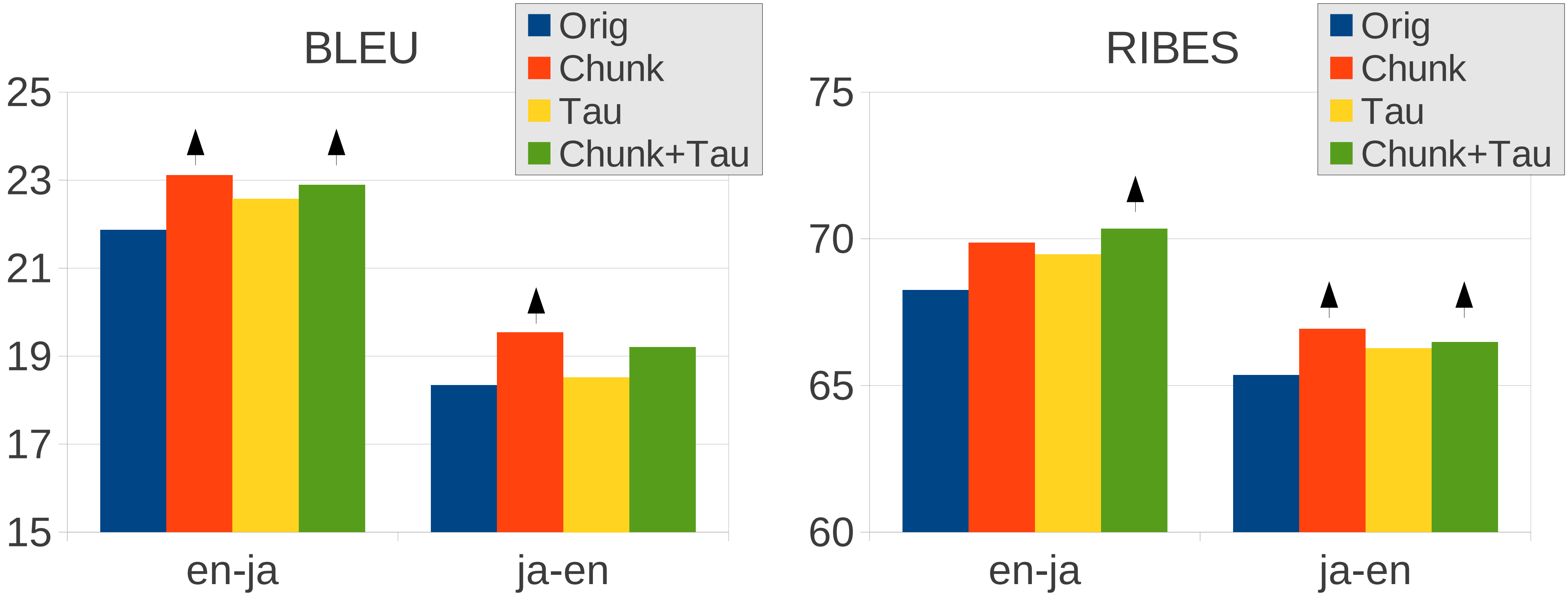
Result:
Training Loss Affects Reordering

- Optimized criterion is higher on test set as well

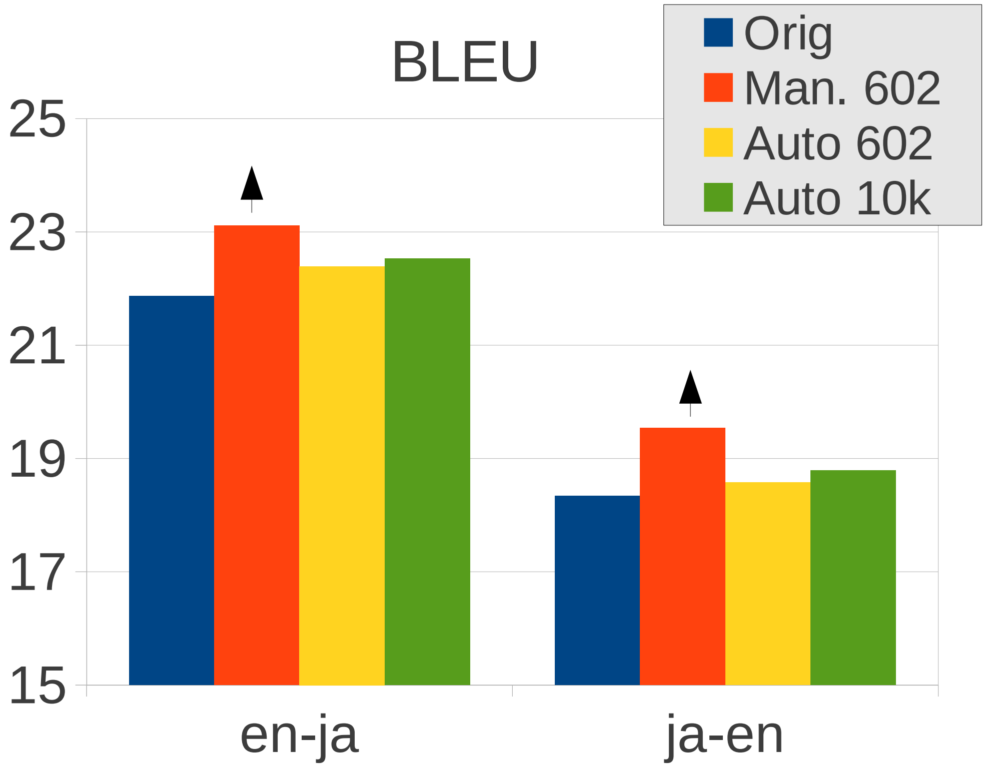


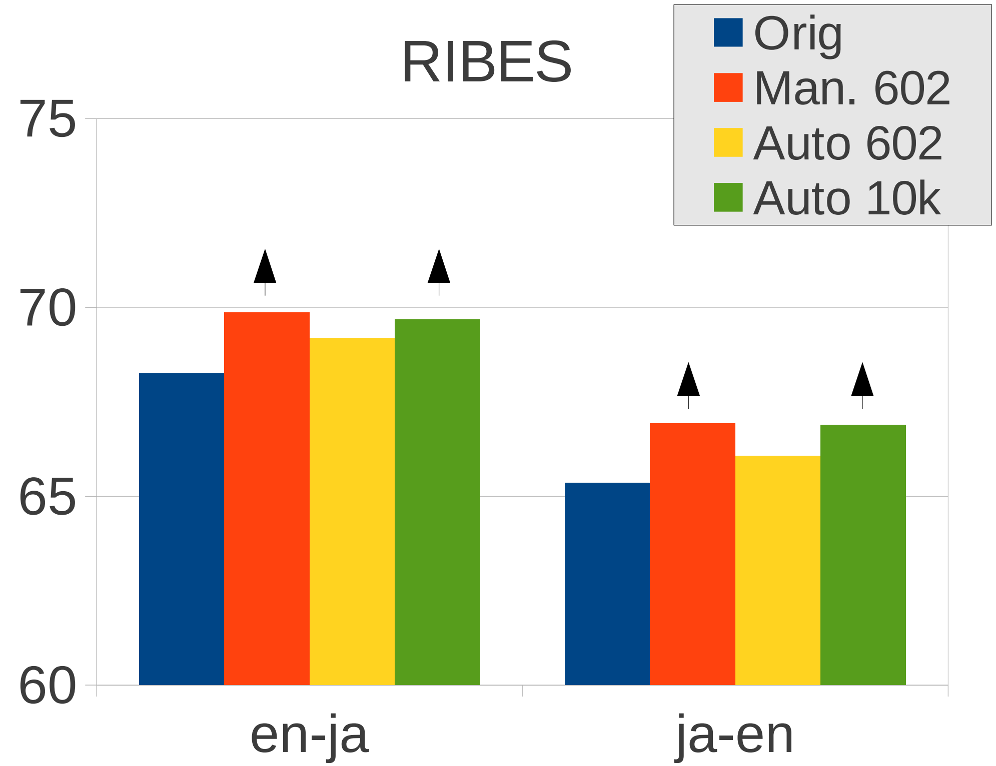
Result: Training Loss Affects Translation

- Optimizing chunk fragmentation generally gives best results

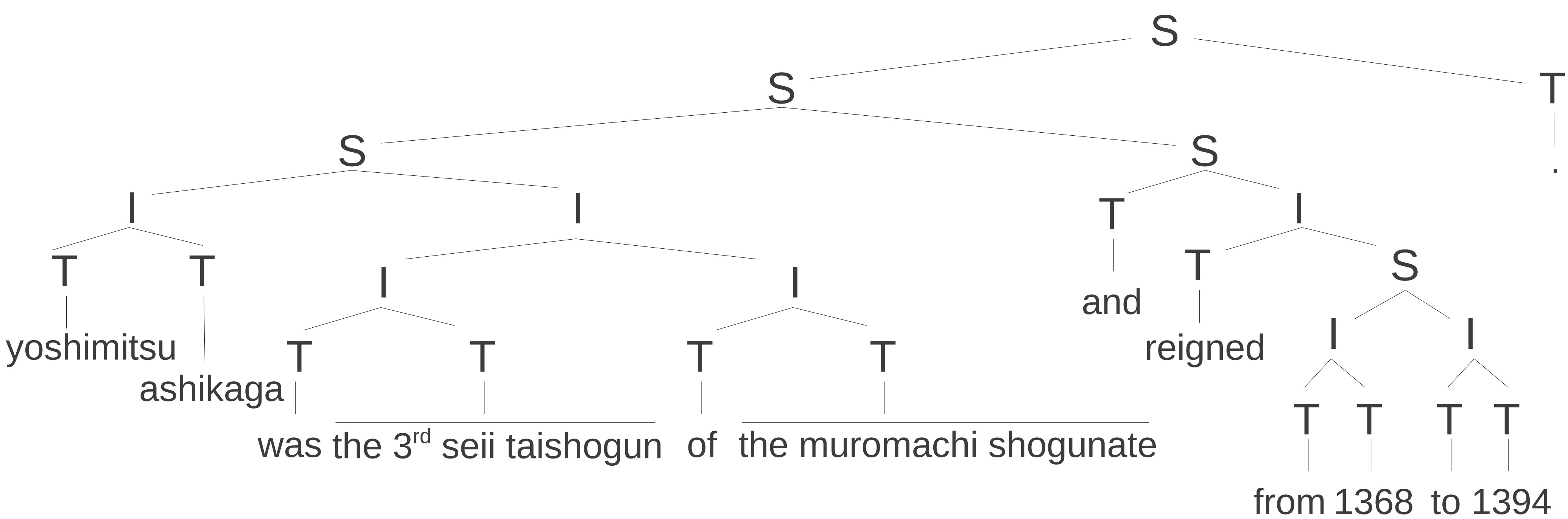


Result: Automatic Alignments, Better than Nothing, Worse than Manual

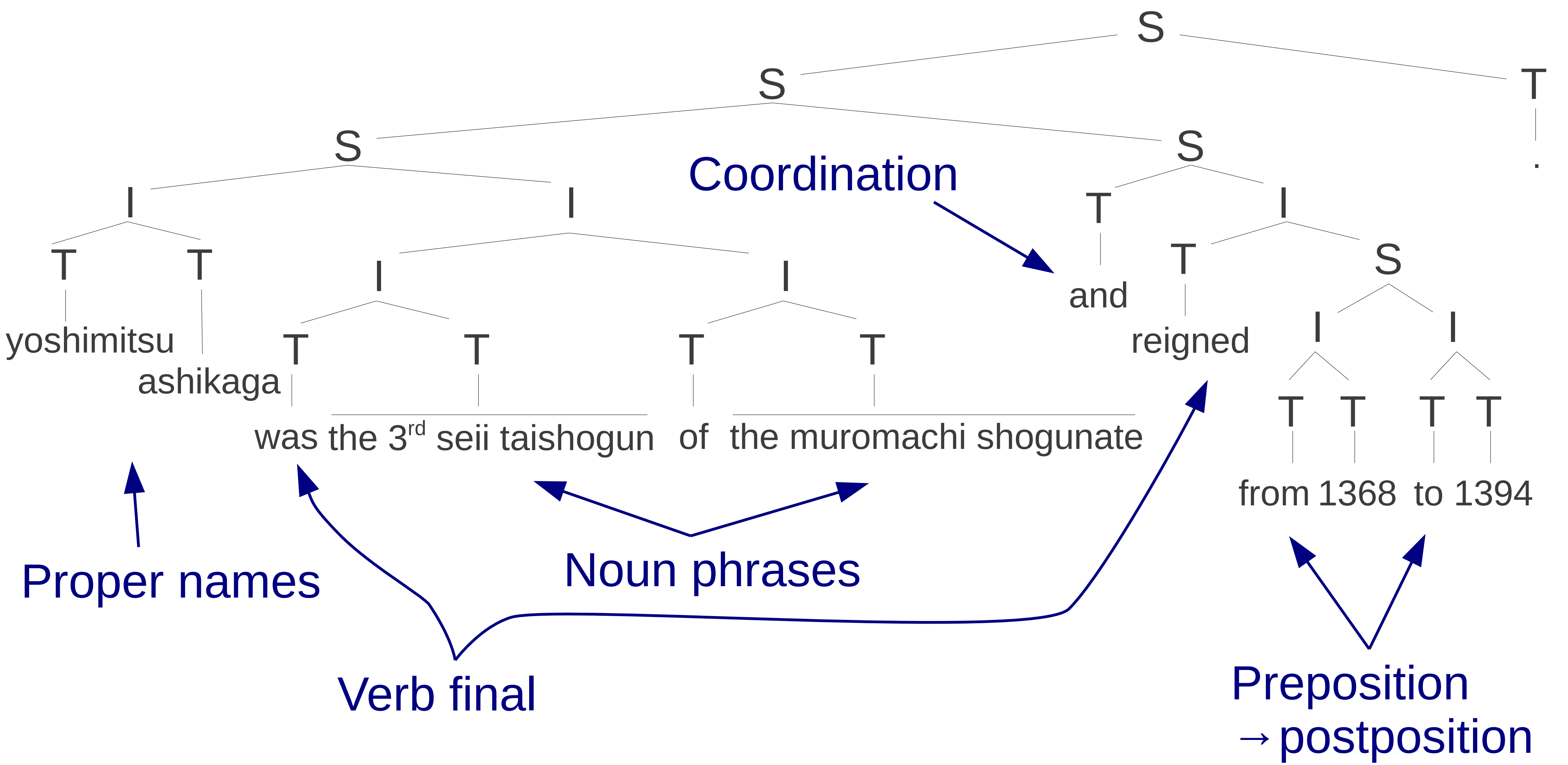




Parsing Result



Parsing Result



Conclusion

- Presented a method to induce a discriminative parser to optimize machine translation reordering
- Favorable results for English ↔ Japanese
- Future Work:
  - Development of better features
  - Incorporation into tree-to-string translation
  - Probabilistic inference

Will be!

^ Available Open Source: http://www.phontron.com/lader

Thank you!