Language modeling with tree substitution grammars

Matt Post
Daniel Gildea

UNIVERSITY of ROCHESTER

NIPS workshop on Grammar Induction, Representation of Language, and Language Learning

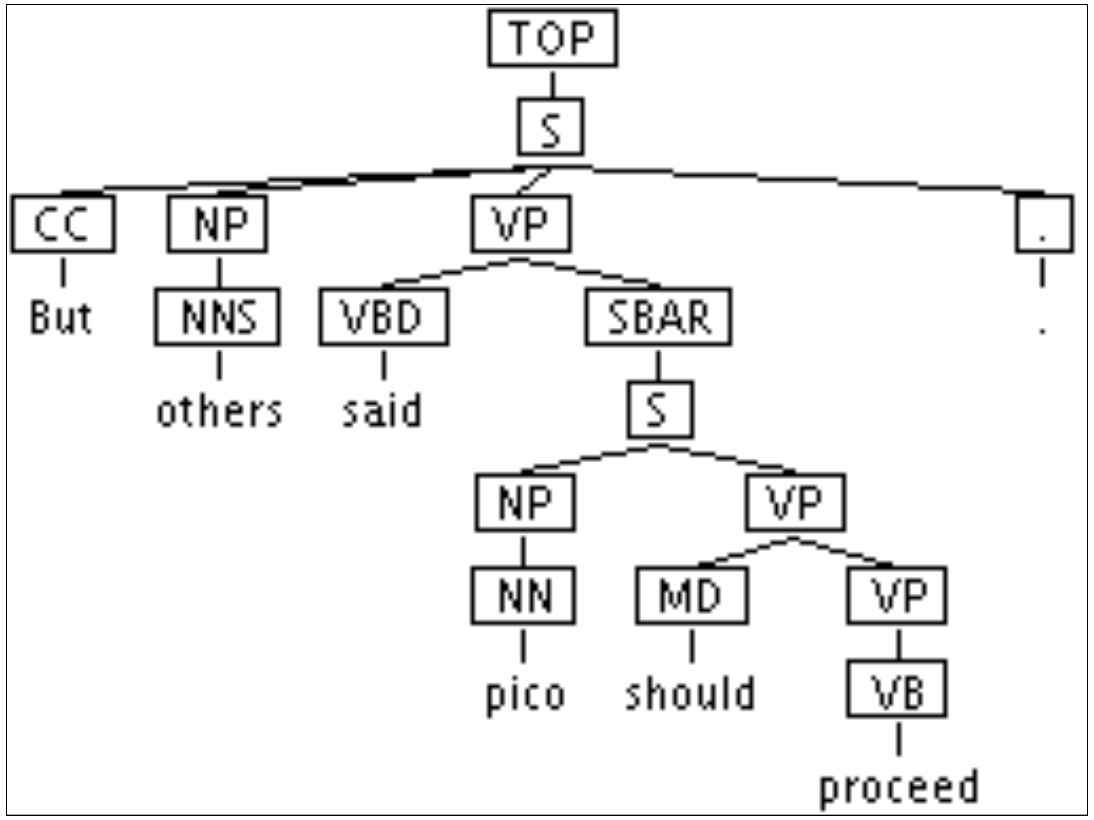
December 11, 2009

1.

certain learned TSGs (a) have lower perplexity and (b) are roughly the same size as the standard Treebank CFG

Standard CFG

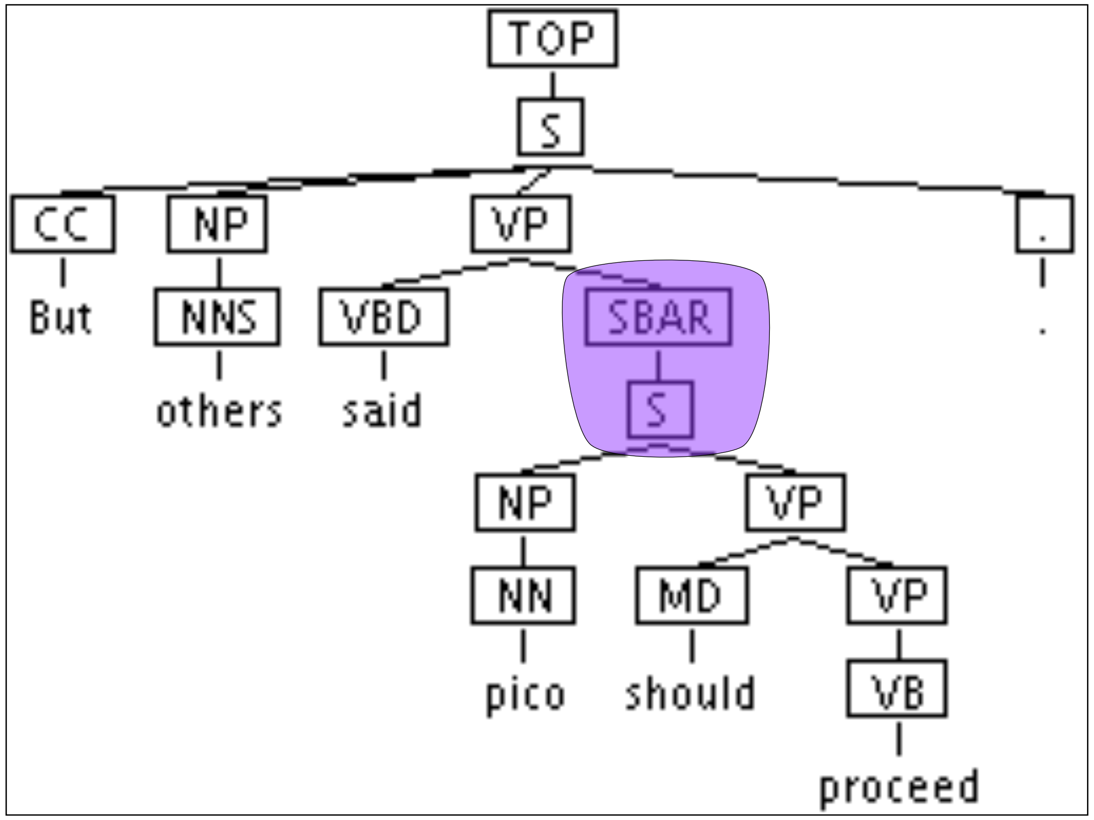
Nonterminals rewrite as sequence of child nonterminals



parse tree from training data

Standard CFG

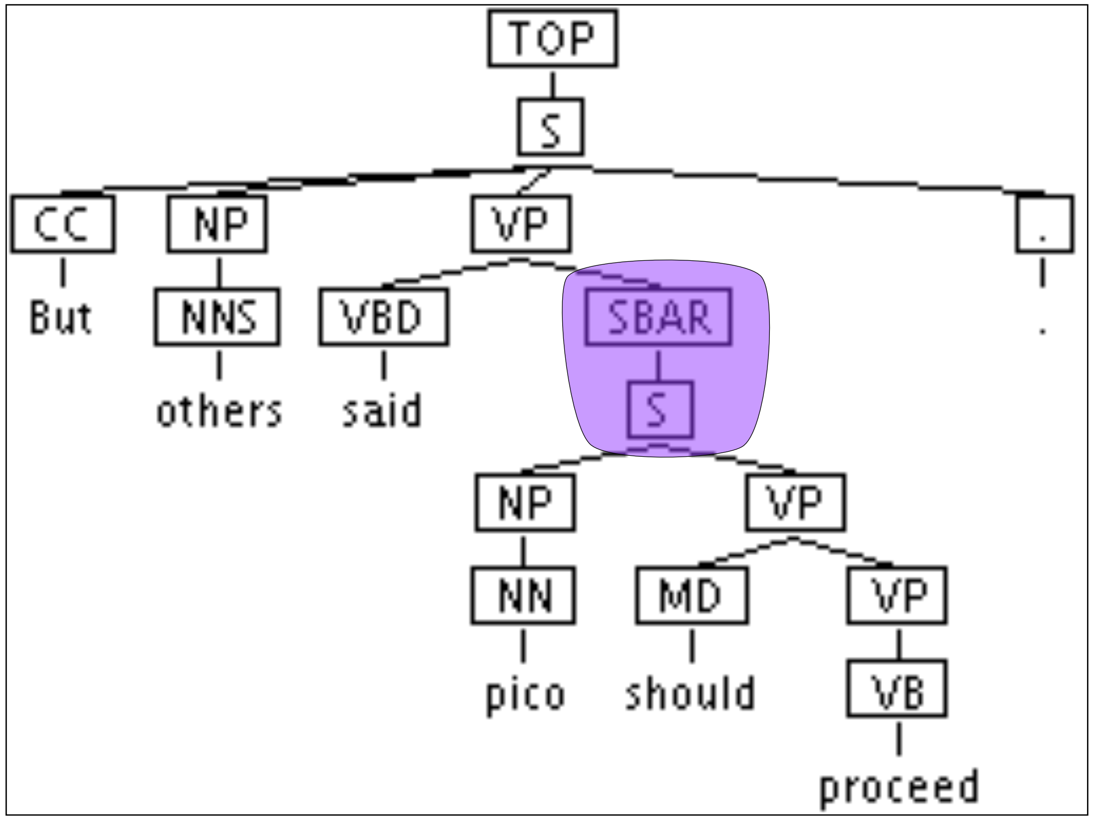
Nonterminals rewrite as sequence of child nonterminals



parse tree from training data

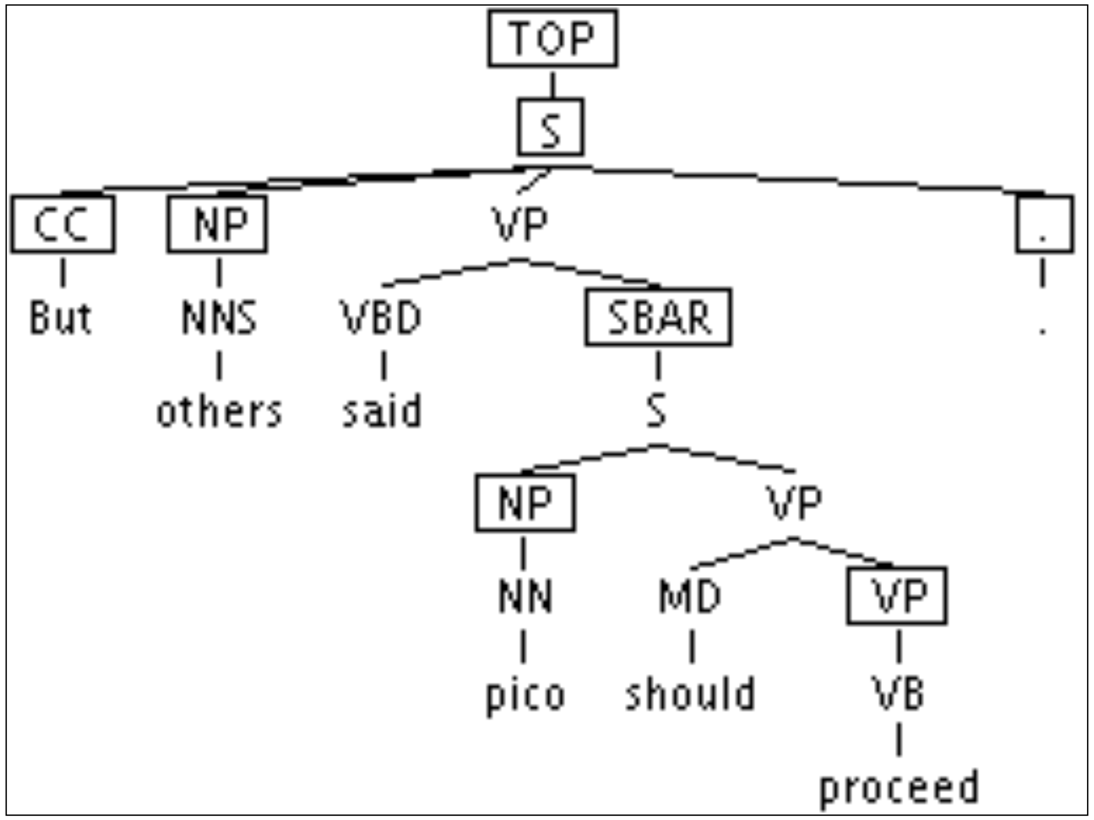
Standard CFG

Nonterminals rewrite as sequence of child nonterminals



TSG

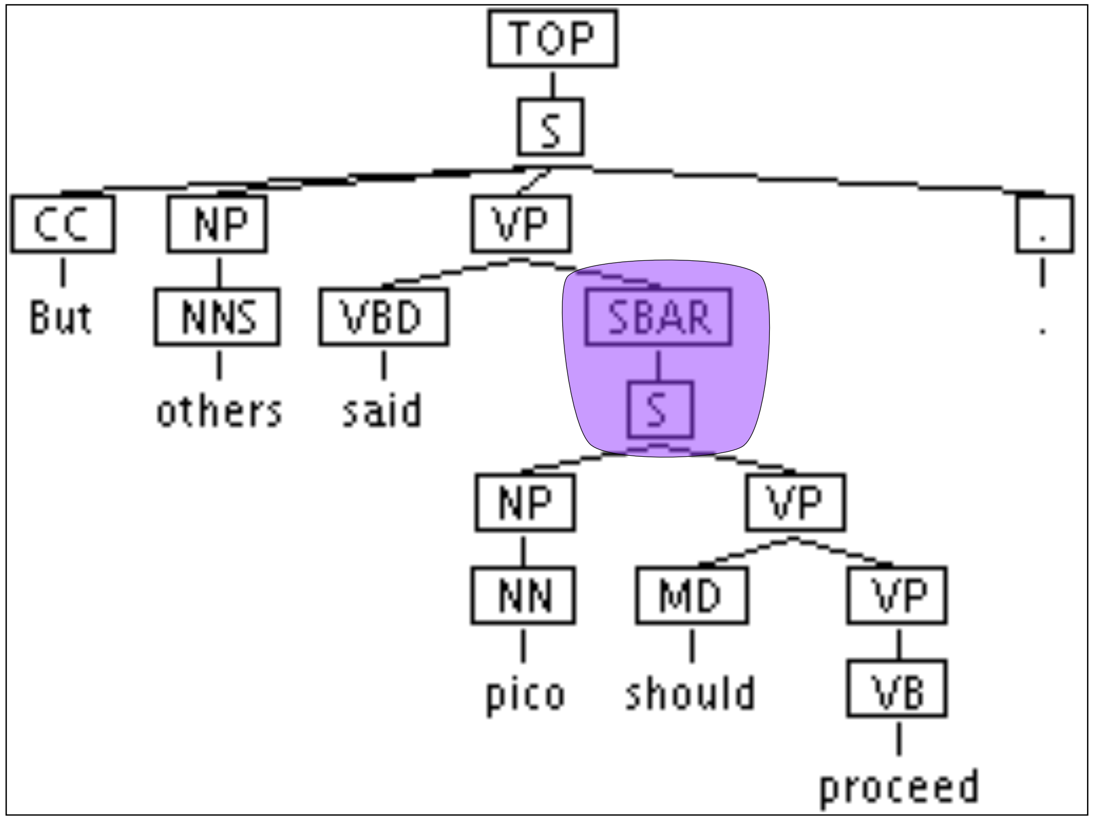
Nonterminals rewrite as sequence of child subtrees, each of arbitrary size



parse tree from training data

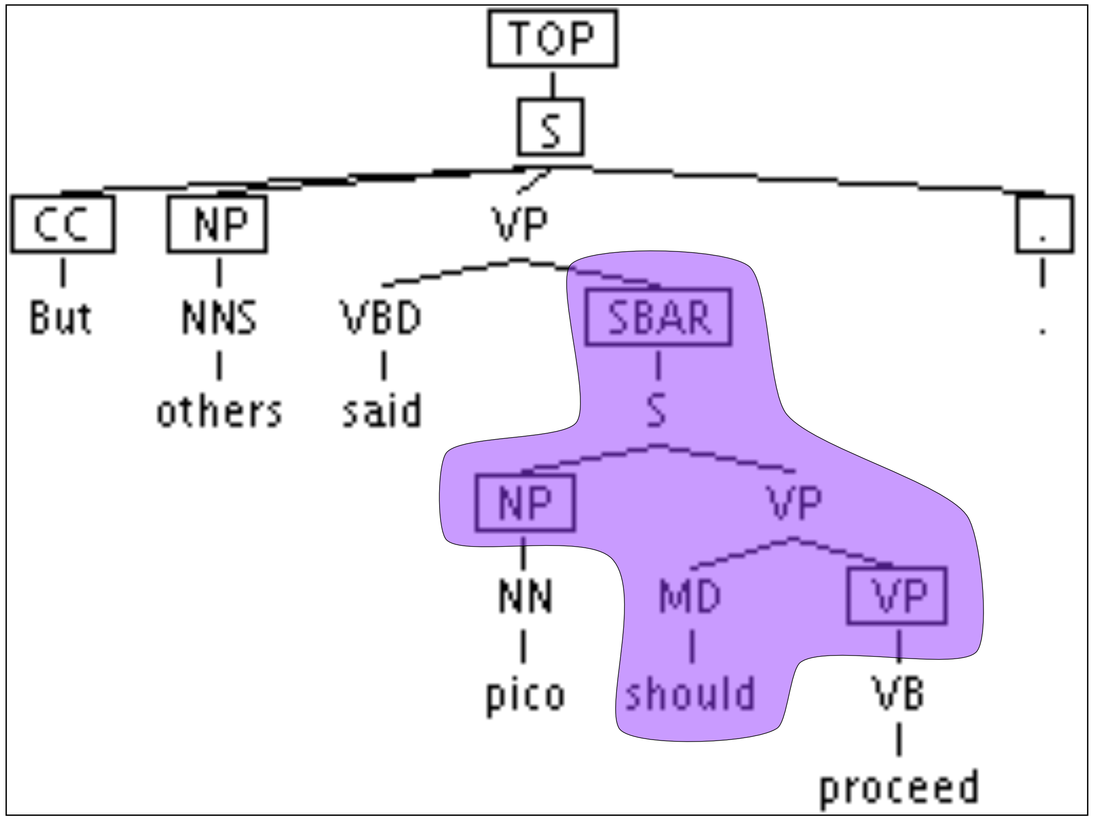
Standard CFG

Nonterminals rewrite as sequence of child nonterminals



TSG

Nonterminals rewrite as sequence of child subtrees, each of arbitrary size



parse tree from training data

how do we learn a TSG?

how do we learn a TSG?

DOP

use *all* of the
subtrees
in the training data

(Bod, 2001)

how do we learn a TSG?

DOP

use *all* of the
subtrees
in the training data

(Bod, 2001)

EM

guess the derivations
and count

Overview: Post & Gildea (2009)

(see also Cohn et al. (2009), Tenenbaum et al. (2009))

overfitting
use a Dirichlet Process prior that discourages large subtrees

$$
\begin{aligned}
g_X &\sim DP(\alpha, G_X) \\
G_X(t) &= \Pr_{\$}(|t|; p_{\$}) \prod_{r \in t} \Pr_{\text{MLE}}(r)
\end{aligned}
$$

Overview: Post & Gildea (2009)

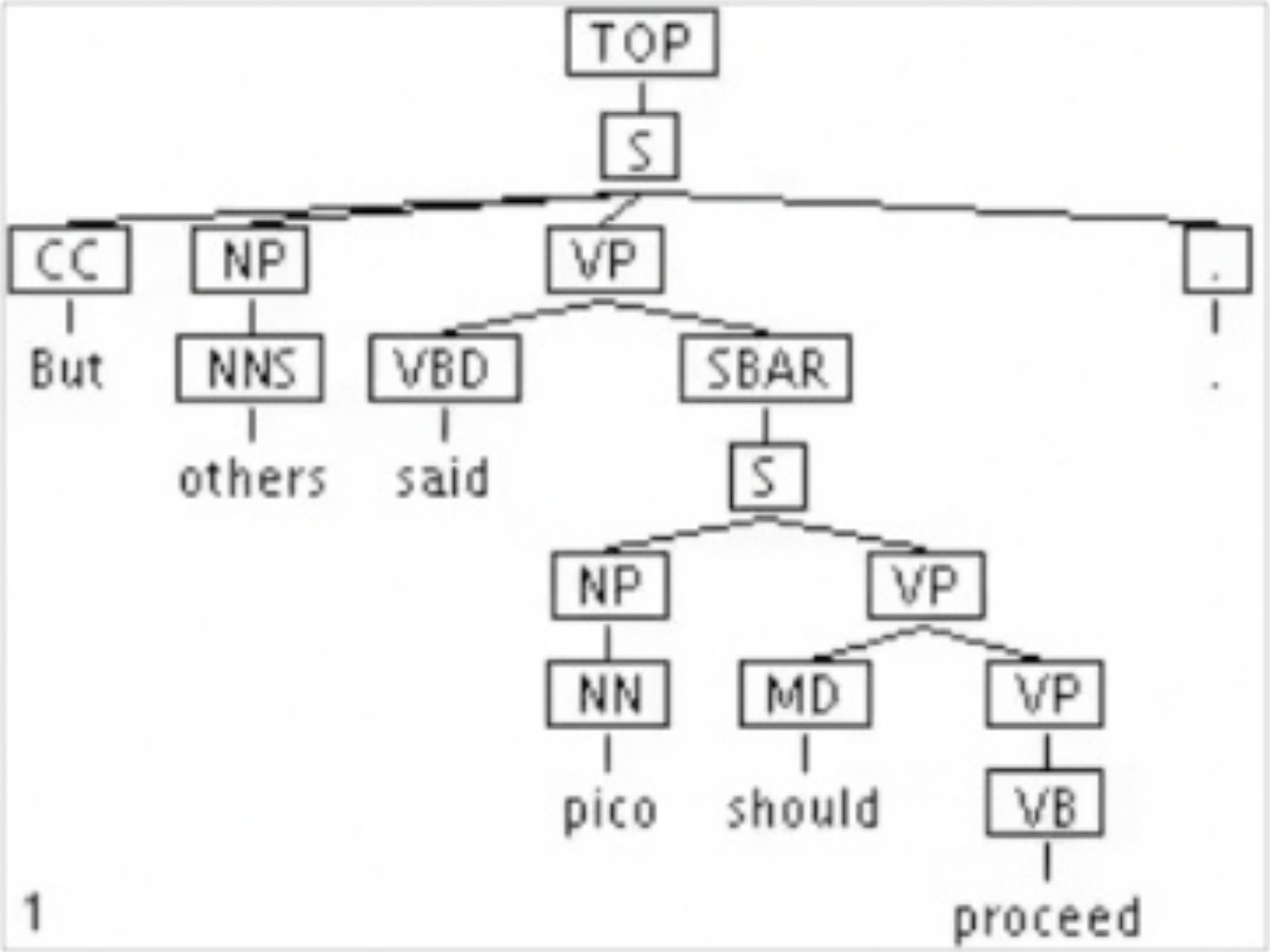
(see also Cohn et al. (2009), Tenenbaum et al. (2009))

overfitting
use a Dirichlet Process prior that discourages large subtrees

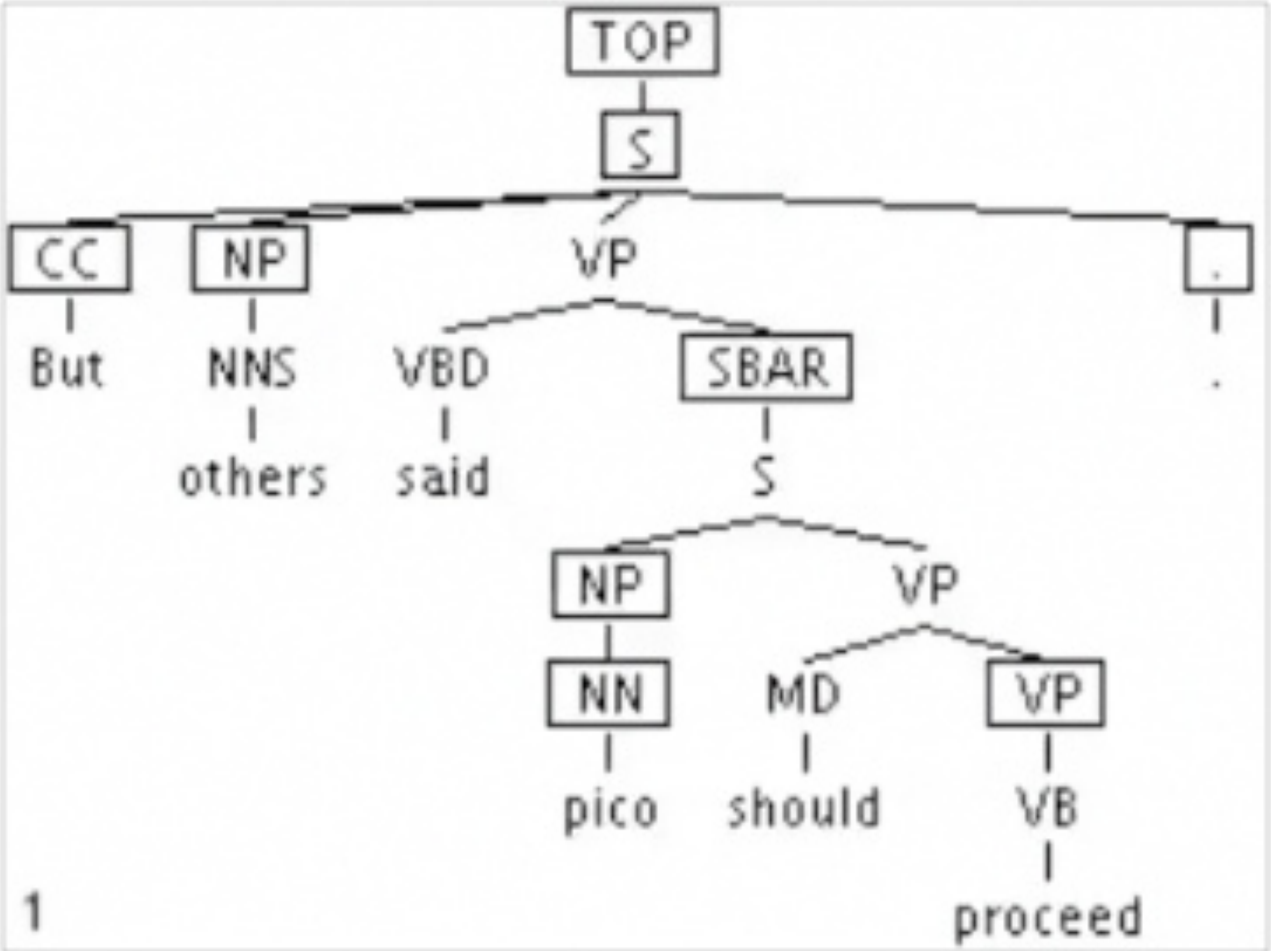
$$\begin{aligned} g_X &\sim DP(\alpha, G_X) \\ G_X(t) &= \Pr_{\$}(|t|; p_{\$}) \prod_{r \in t} \Pr_{\mathrm{MLE}}(r) \end{aligned}$$

space efficiency
only maintain counts of subtrees from the set of existing derivations in the training data

Collapsed Gibbs sampling
(Goldwater et al., 2009)

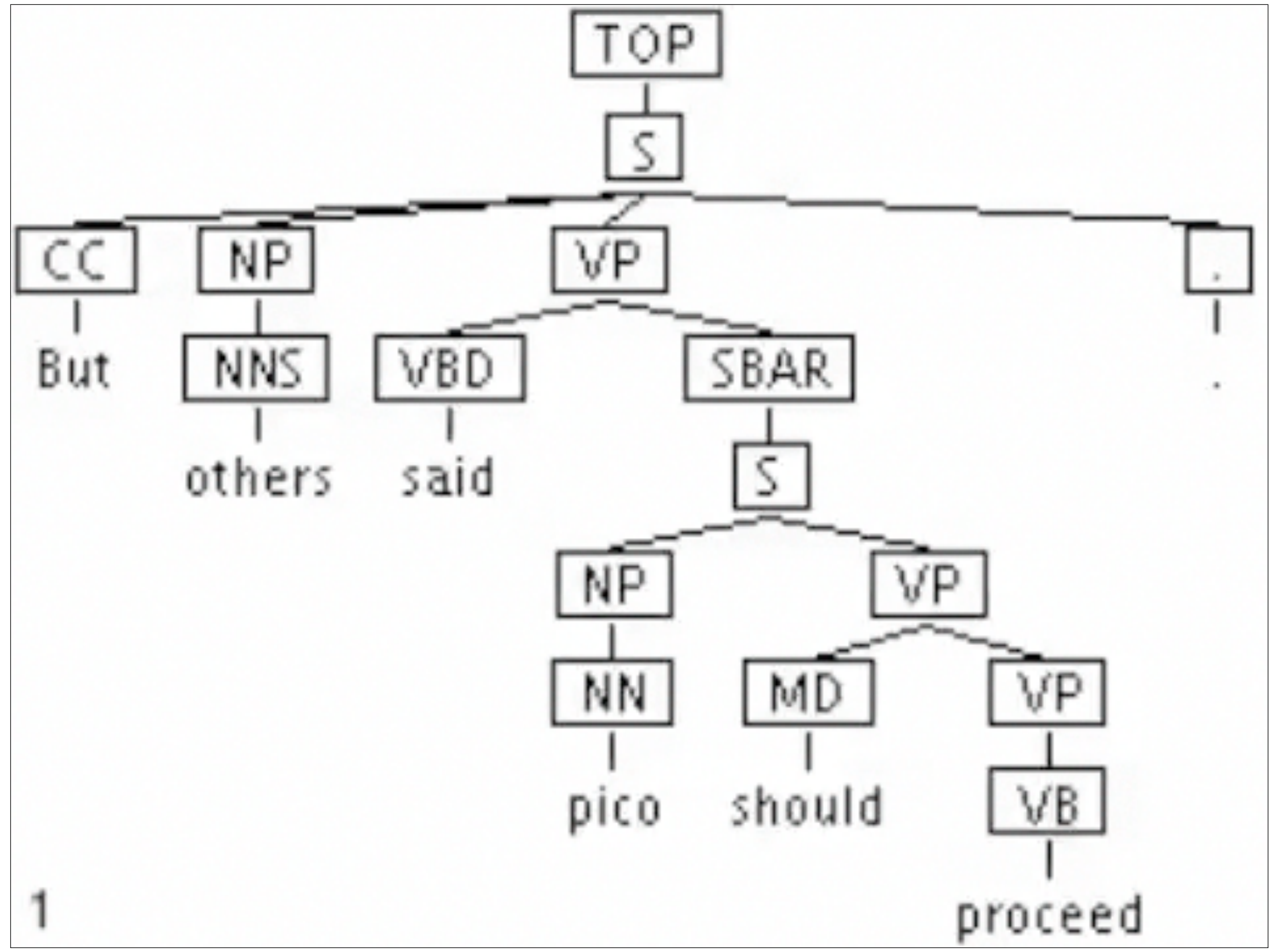


Treebank initialization

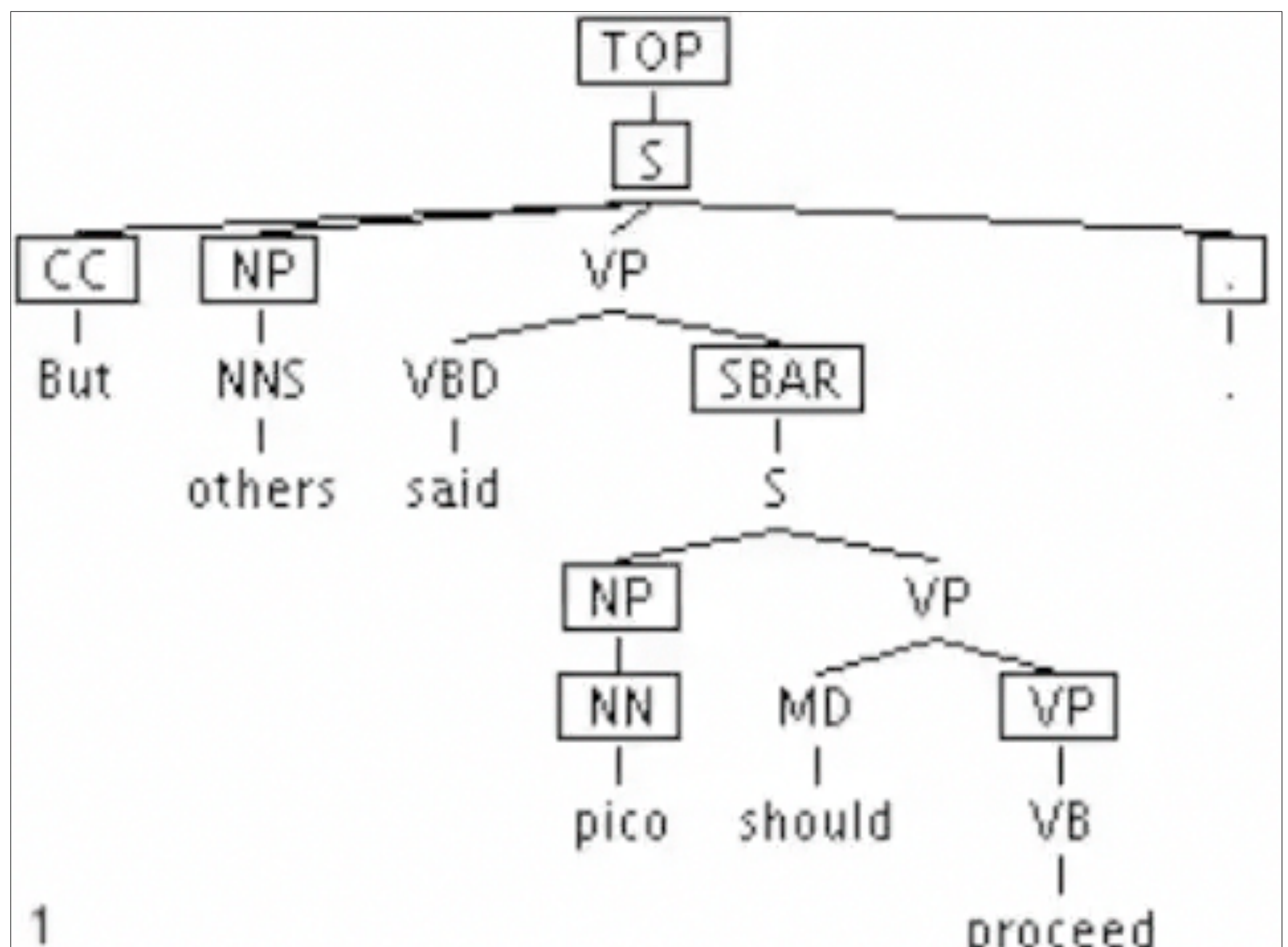


spinal initialization

Treebank initialization



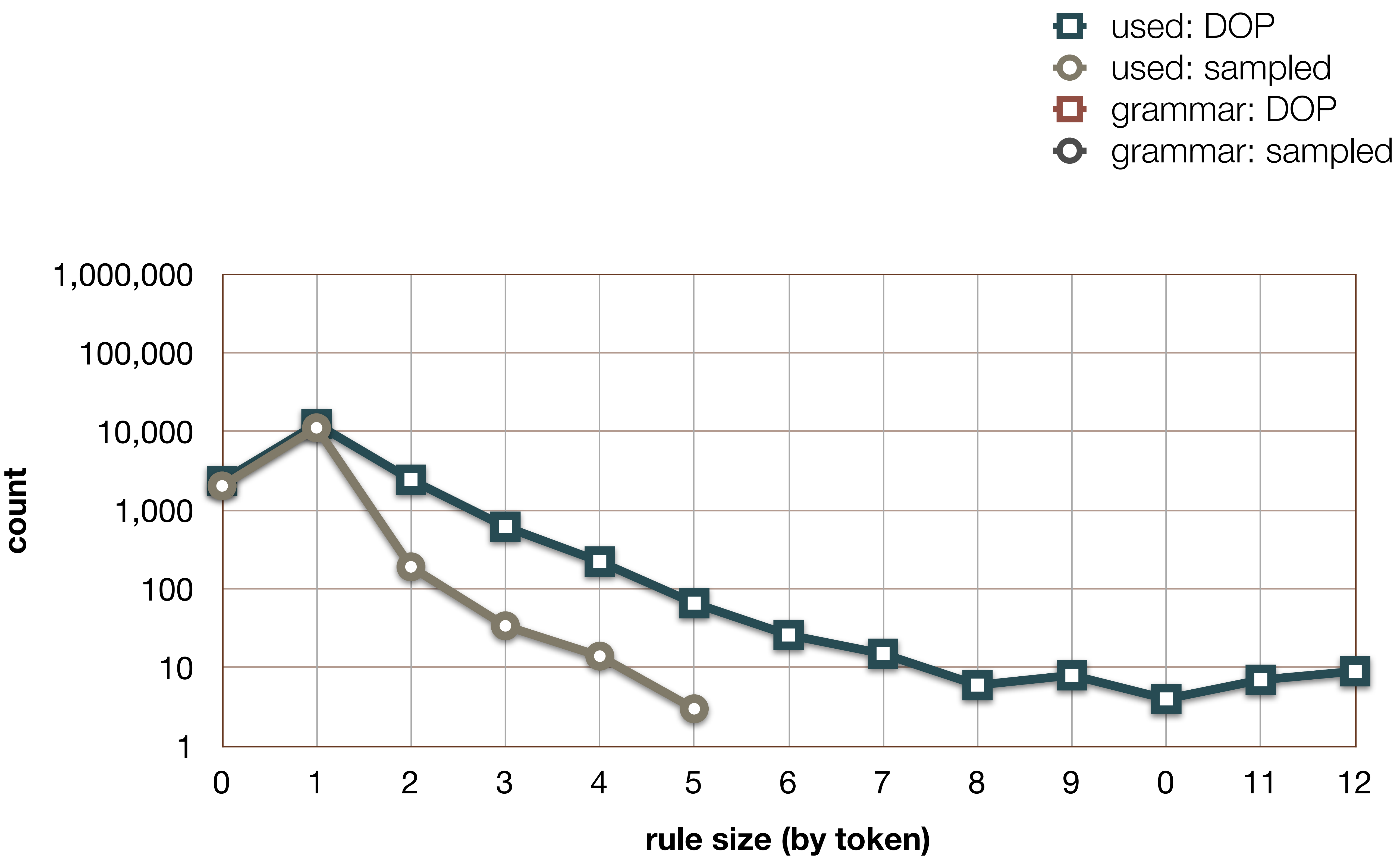
spinal initialization



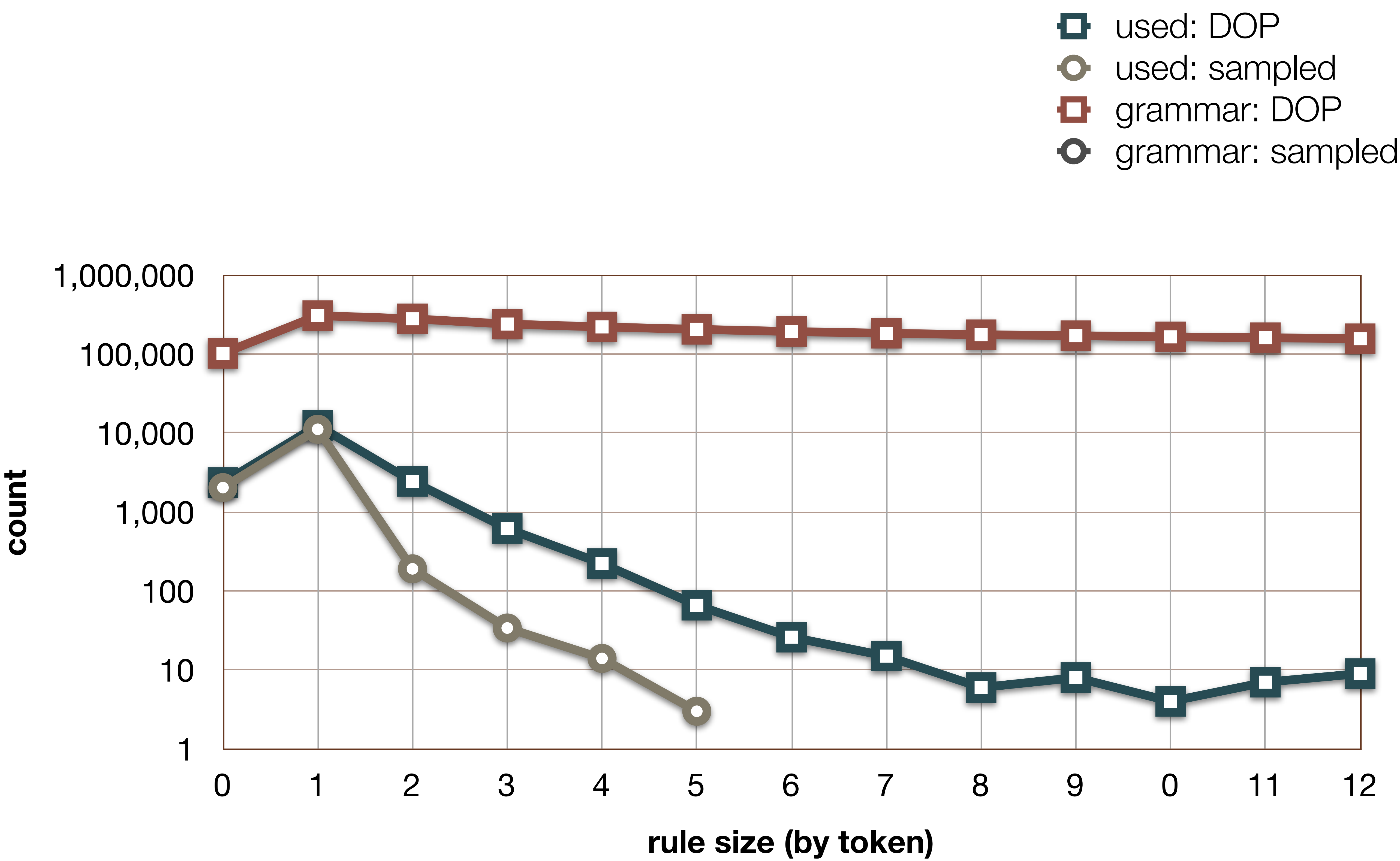
Overview: Post & Gildea (2009)

used: DOP
used: sampled
grammar: DOP
grammar: sampled

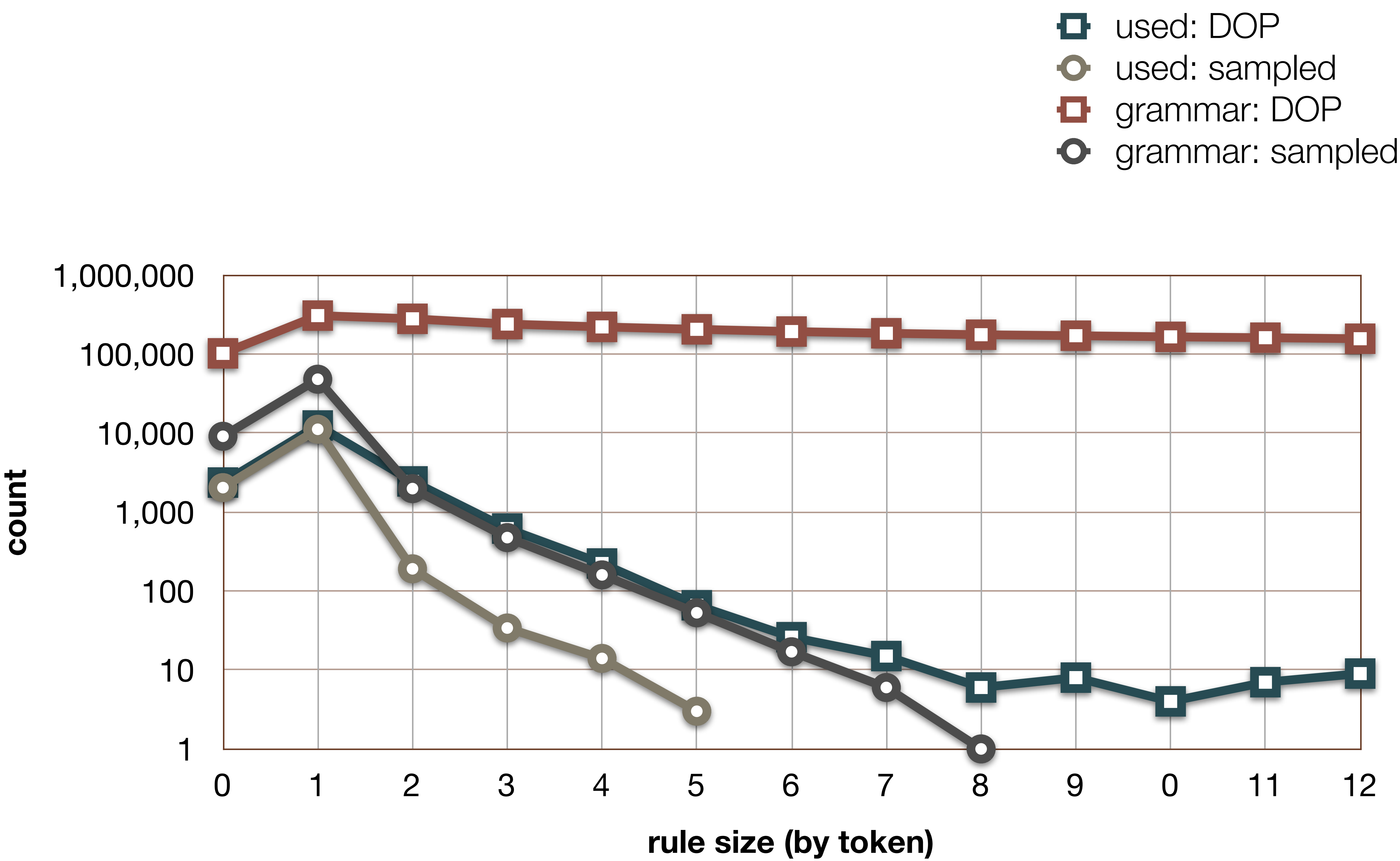
Overview: Post & Gildea (2009)



Overview: Post & Gildea (2009)



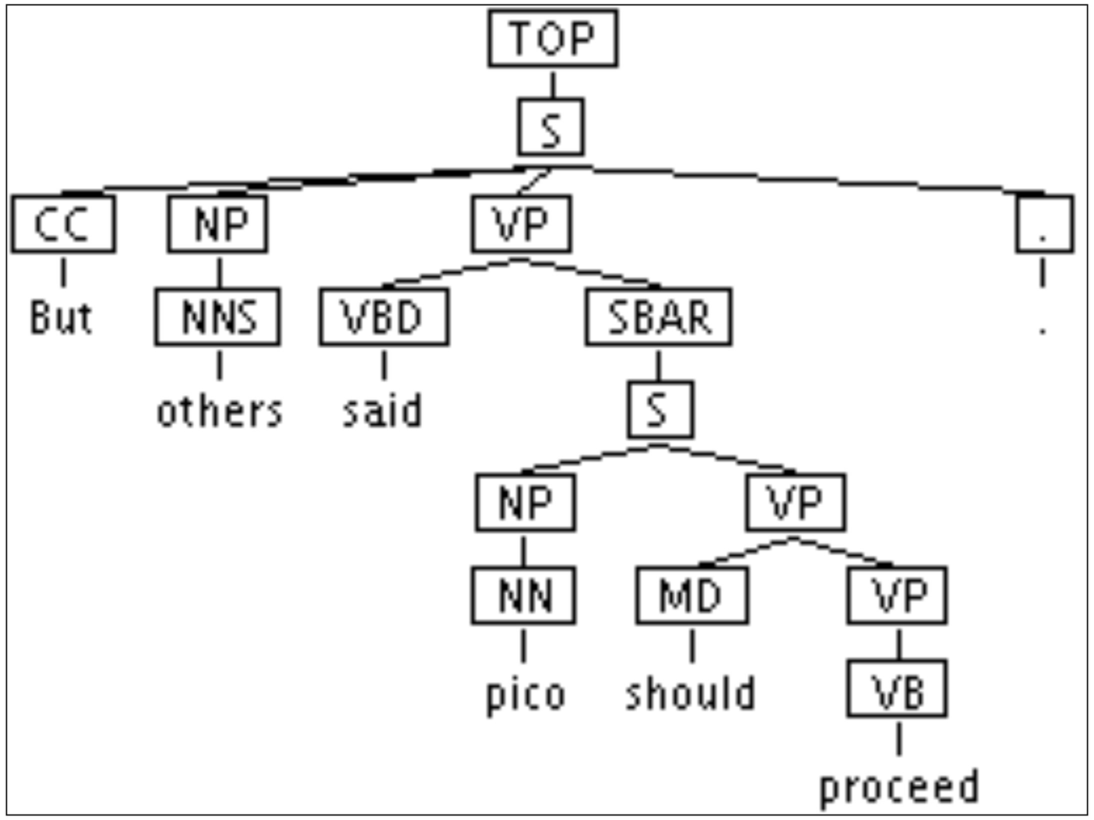
Overview: Post & Gildea (2009)



Experiments

Treebank grammar

All rules have a depth of one

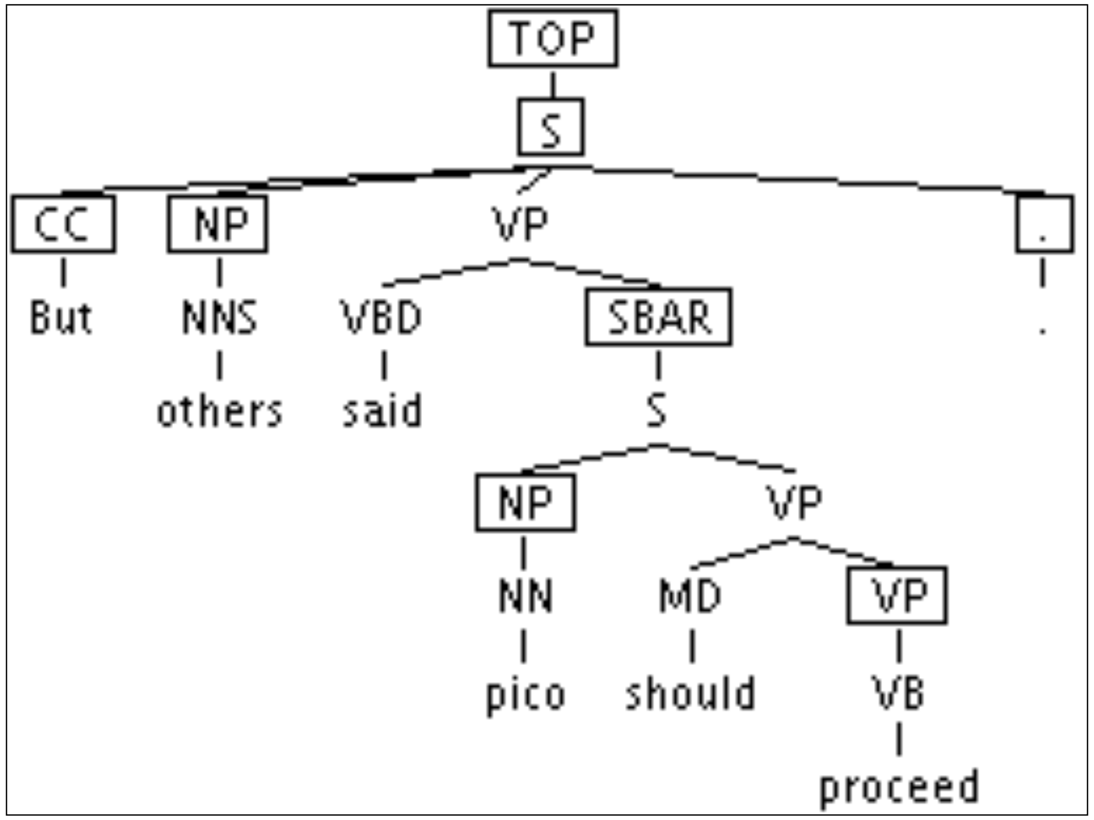


Treebank grammar

All rules have a depth of one

“spinal” grammar

TSG subtrees induced by maximally projecting each word

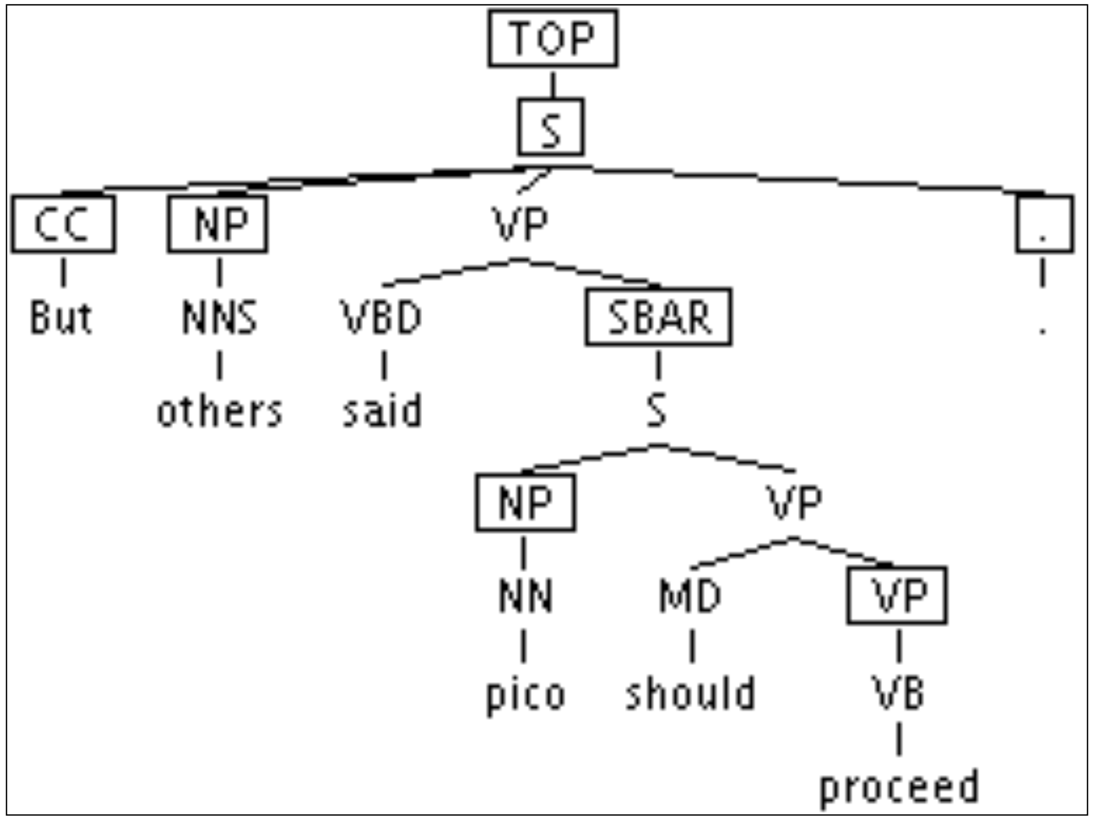


Treebank grammar

All rules have a depth of one

"spinal" grammar

TSG subtrees induced by maximally projecting each word



Treebank grammar

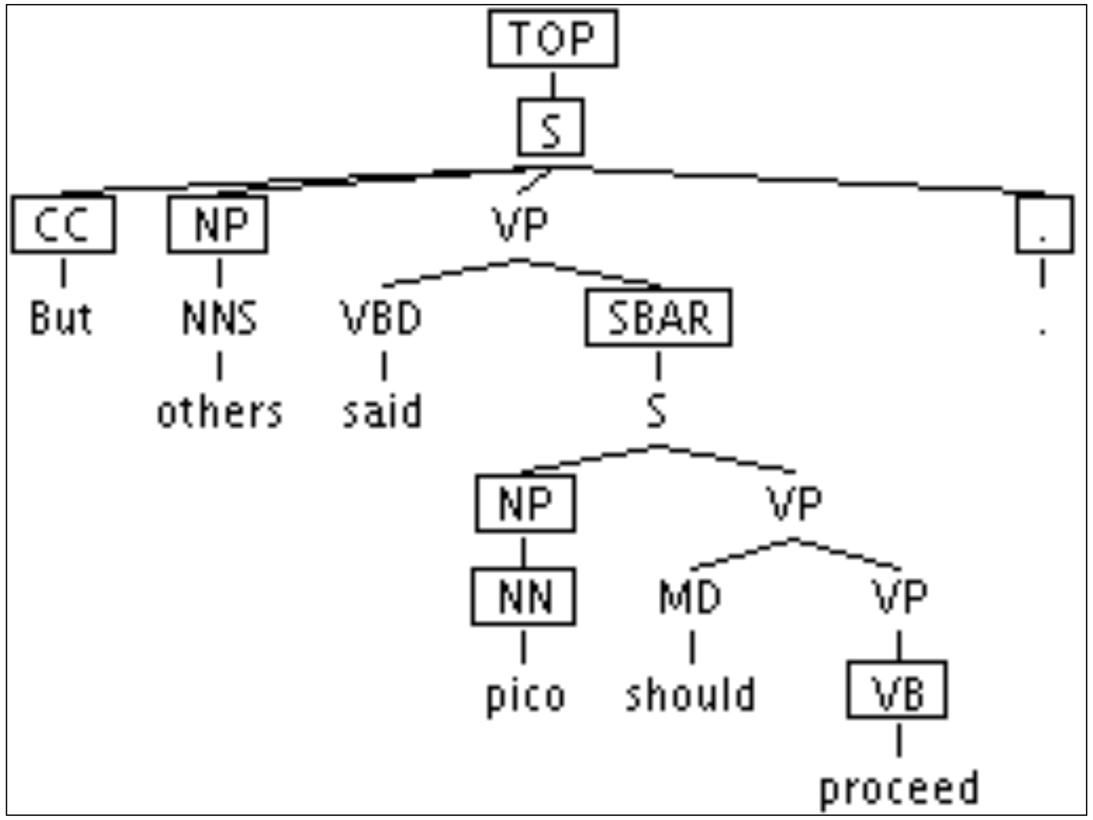
All rules have a depth of one

“spinal” grammar

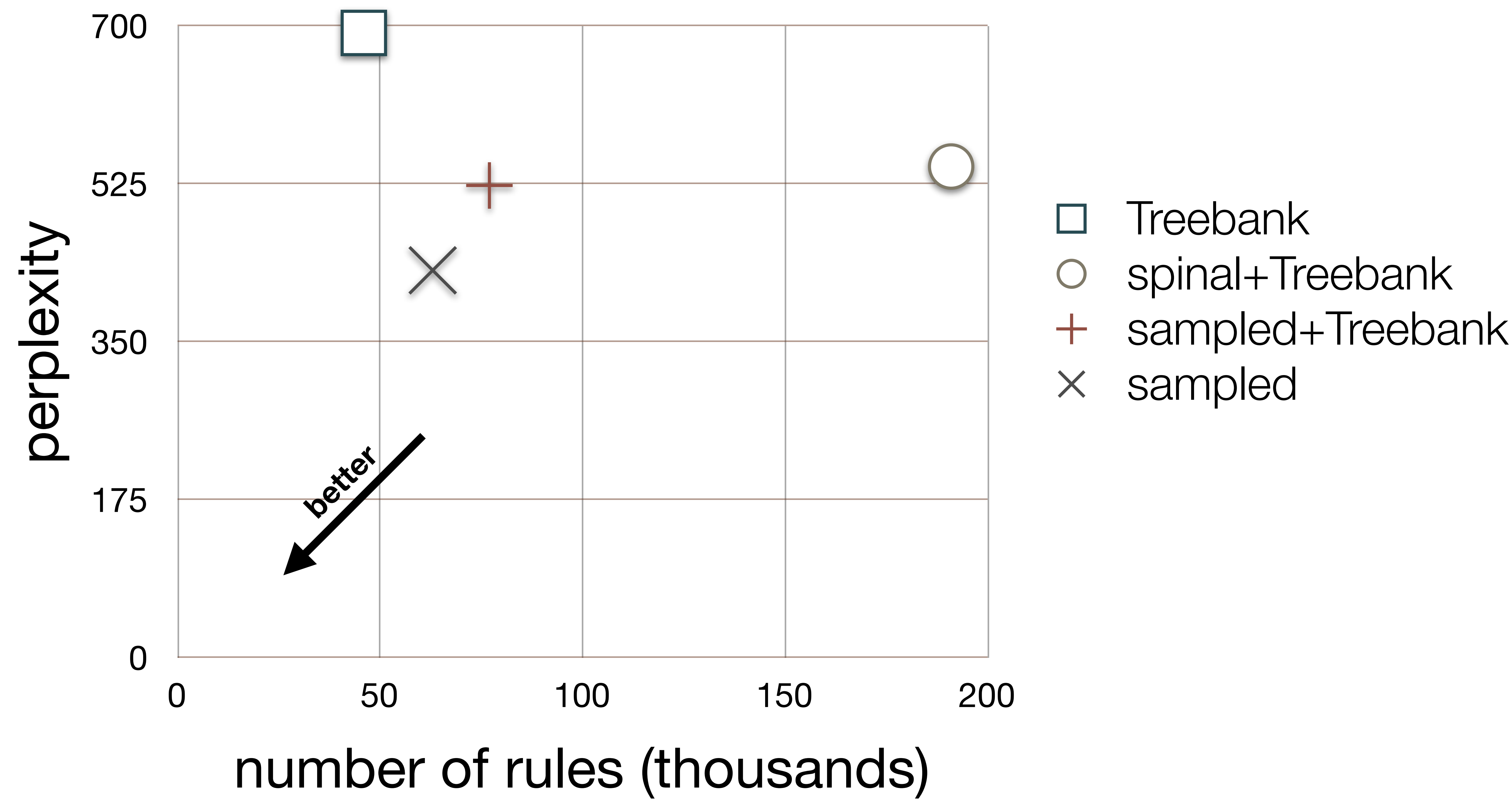
TSG subtrees induced by maximally projecting each word

sampled grammar

TSG subtrees induced with a collapsed Gibbs sampler and Dirichlet Process prior



Perplexity



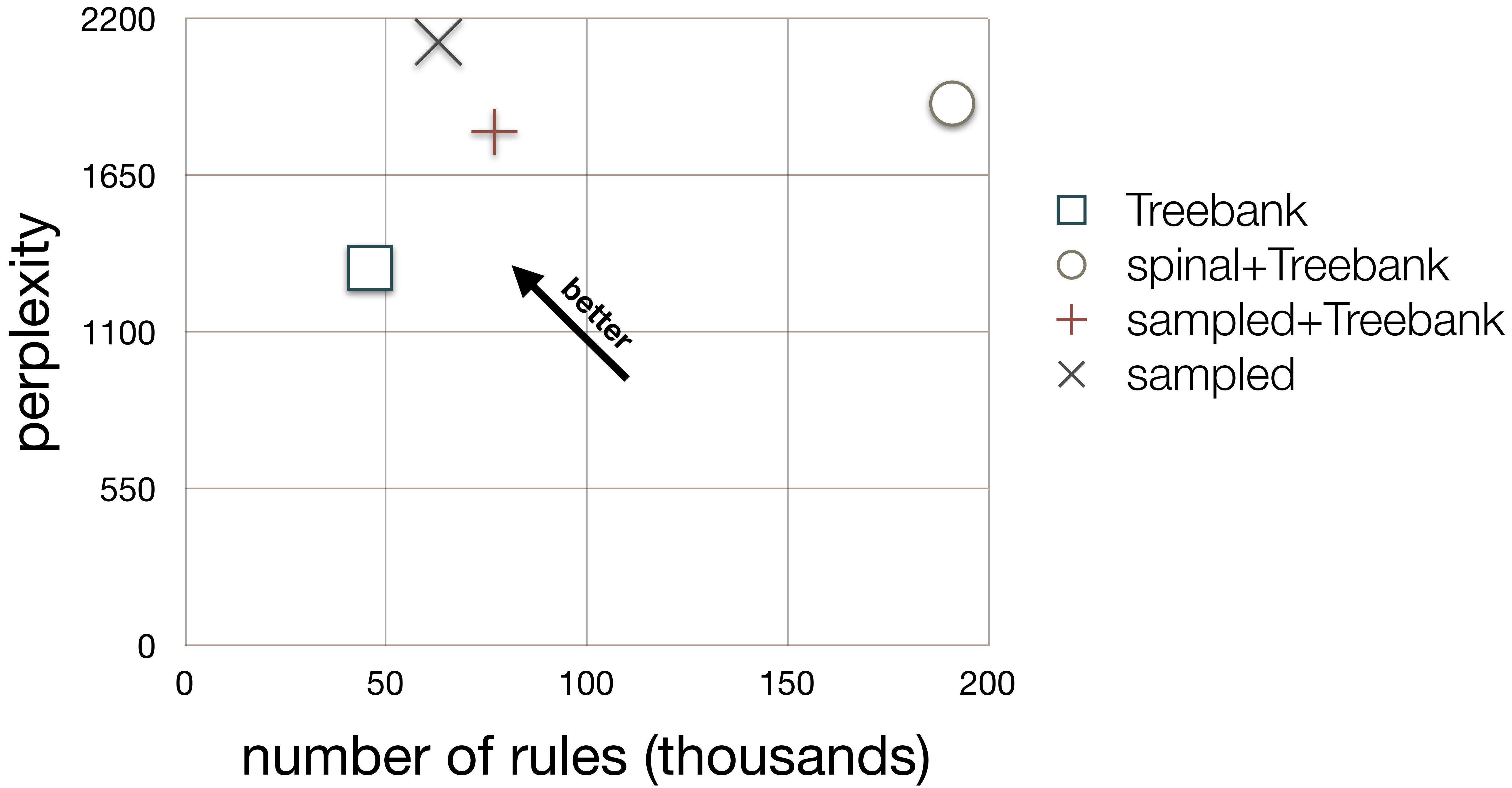
700
525
350
175
0
0
50
100
150
200
perplexity
number of rules (thousands)
better
Treebank
spinal+Treebank
sampled+Treebank
sampled

Pseudorandom text

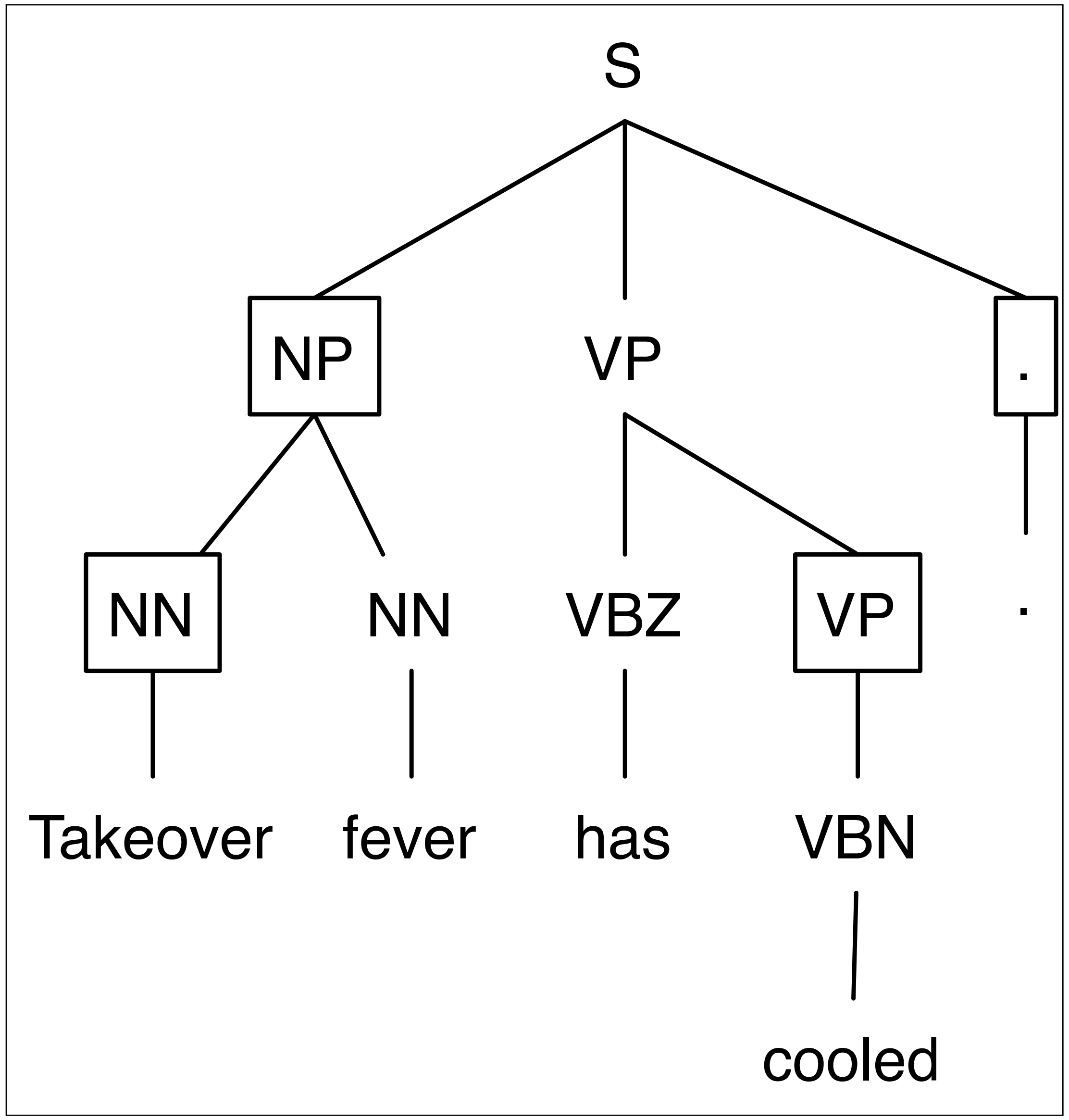
Okanohara and Tsujii (2007)

[**banks investment Big**]$_{\text{NP}}$ refused to step up to [**plate the**]$_{\text{NP}}$ to support [**traders floor beleaguered the**]$_{\text{NP}}$ by buying [[**of stock**]$_{\text{PP}}$ [***blocks big***]$_{\text{NP}}$]$_{\text{NP}}$, traders say .

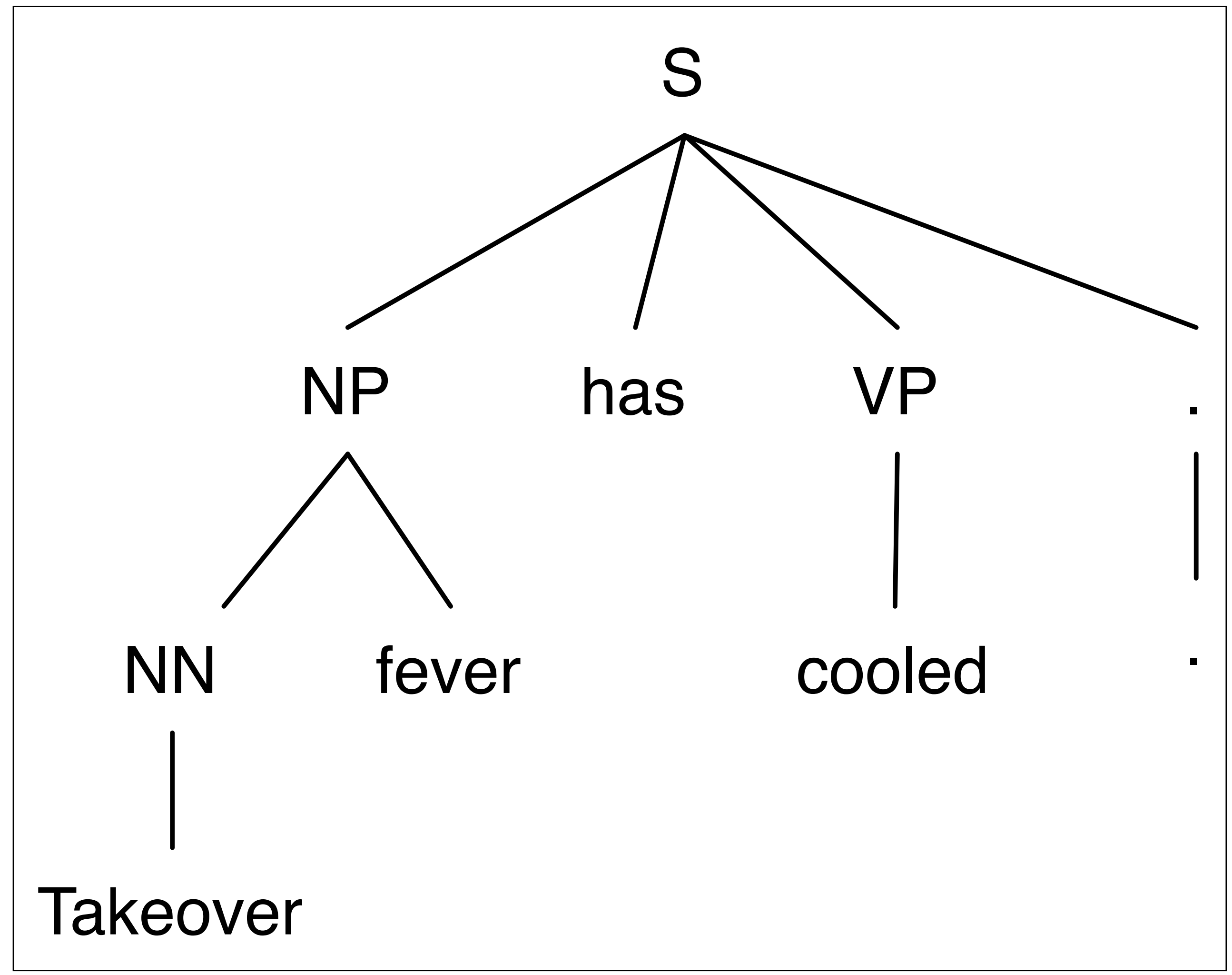
Perplexity on pseudo-negative text



Flattening



TSG derivation in training corpus



internal nodes removed

2.

sampled TSGs lead to perplexity improvements with a bilexical parser, suggesting they are improving Treebank structure

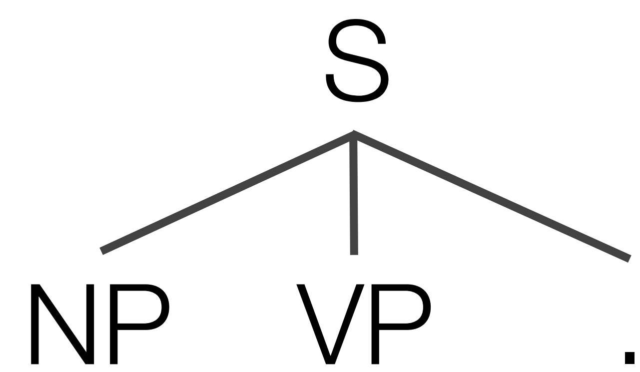
CFG parsing

S

CFG parsing

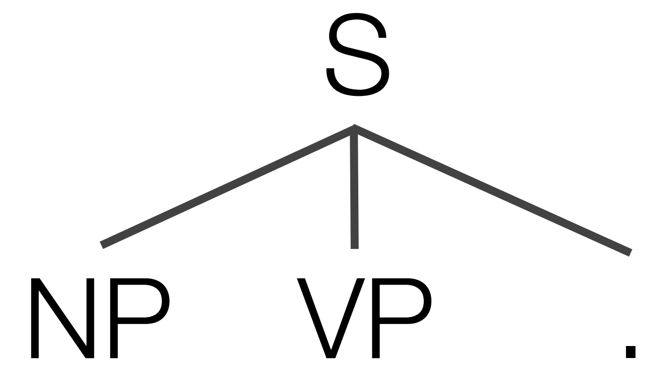
1. Replace each nonterminal with its children in a single act

   Pr(*rhs*| P)



CFG parsing

1. Replace each nonterminal with its children in a single act
   Pr(*rhs*| P)
2. Recurse



Bilexical parsing (Collins Model 1)

S

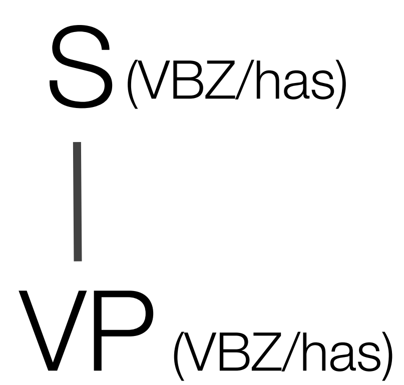
Bilexical parsing (Collins Model 1)

S (VBZ/has)

1. Generate the head word and tag

   Pr(h,t | P)

Bilexical parsing (Collins Model 1)



1. Generate the head word and tag

   Pr(h,t | P)

2. Generate the head child

   Pr(H | P,h,t)

Bilexical parsing (Collins Model 1)

S (VBZ/has)

VP (VBZ/has)

1. Generate the head word and tag

   Pr(h,t | P)

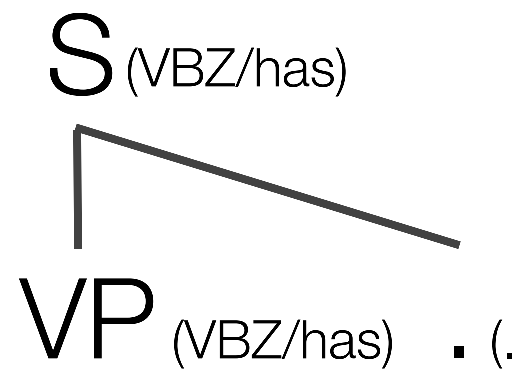
2. Generate the head child

   Pr(H | P,h,t)

3. Generate the sibling head labels and tags

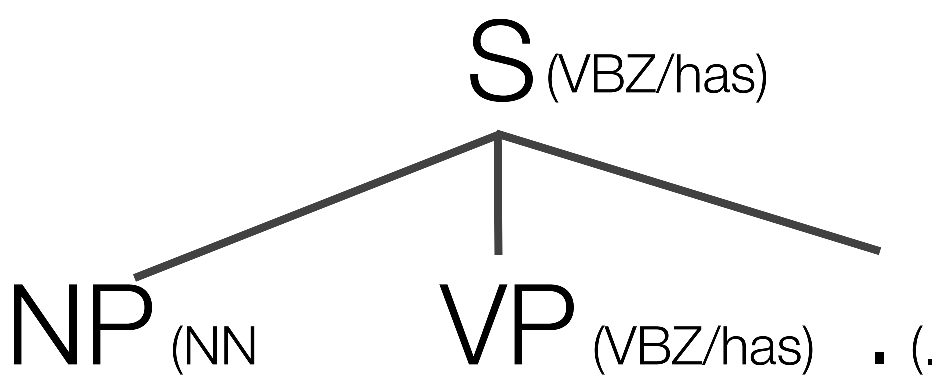
   Pr(C,c_t | P,h,t,H)

Bilexical parsing (Collins Model 1)



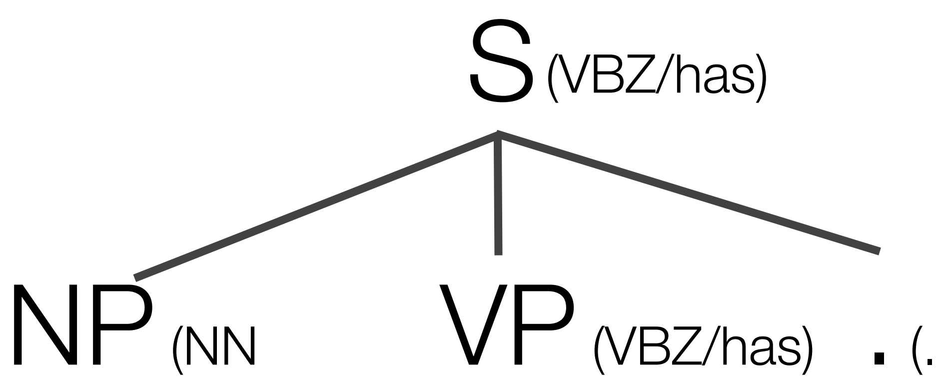
1. Generate the head word and tag
   $\Pr(h,t \mid P)$
2. Generate the head child
   $\Pr(H \mid P,h,t)$
3. Generate the sibling head labels and tags
   $\Pr(C,c_t \mid P,h,t,H)$

Bilexical parsing (Collins Model 1)



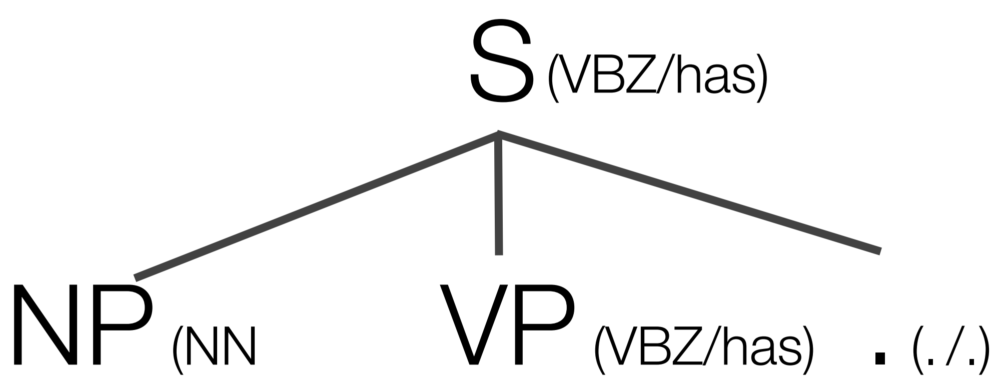
1. Generate the head word and tag
   $\Pr(h,t \mid P)$
2. Generate the head child
   $\Pr(H \mid P,h,t)$
3. Generate the sibling head labels and tags
   $\Pr(C,c_t \mid P,h,t,H)$

Bilexical parsing (Collins Model 1)



1. Generate the head word and tag
   $\Pr(h,t \mid P)$
2. Generate the head child
   $\Pr(H \mid P,h,t)$
3. Generate the sibling head labels and tags
   $\Pr(C,c_t \mid P,h,t,H)$
4. Generate the sibling head word
   $\Pr(c_w \mid P,h,t,H,c_h,c_t)$

Bilexical parsing (Collins Model 1)



1. Generate the head word and tag

   $\Pr(h,t \mid P)$

2. Generate the head child

   $\Pr(H \mid P,h,t)$

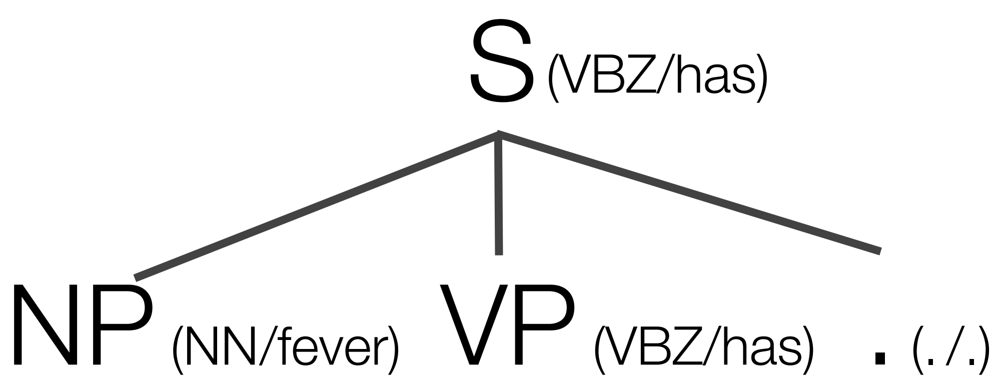
3. Generate the sibling head labels and tags

   $\Pr(C,c_t \mid P,h,t,H)$

4. Generate the sibling head word

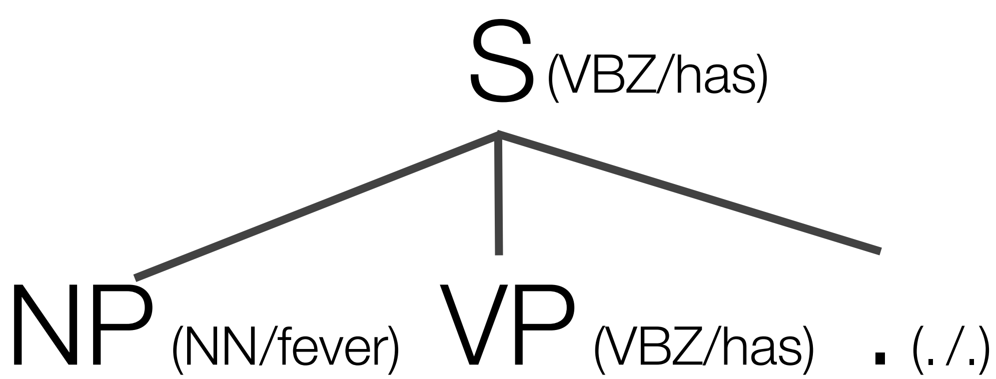
   $\Pr(c_w \mid P,h,t,H,c_h,c_t)$

Bilexical parsing (Collins Model 1)



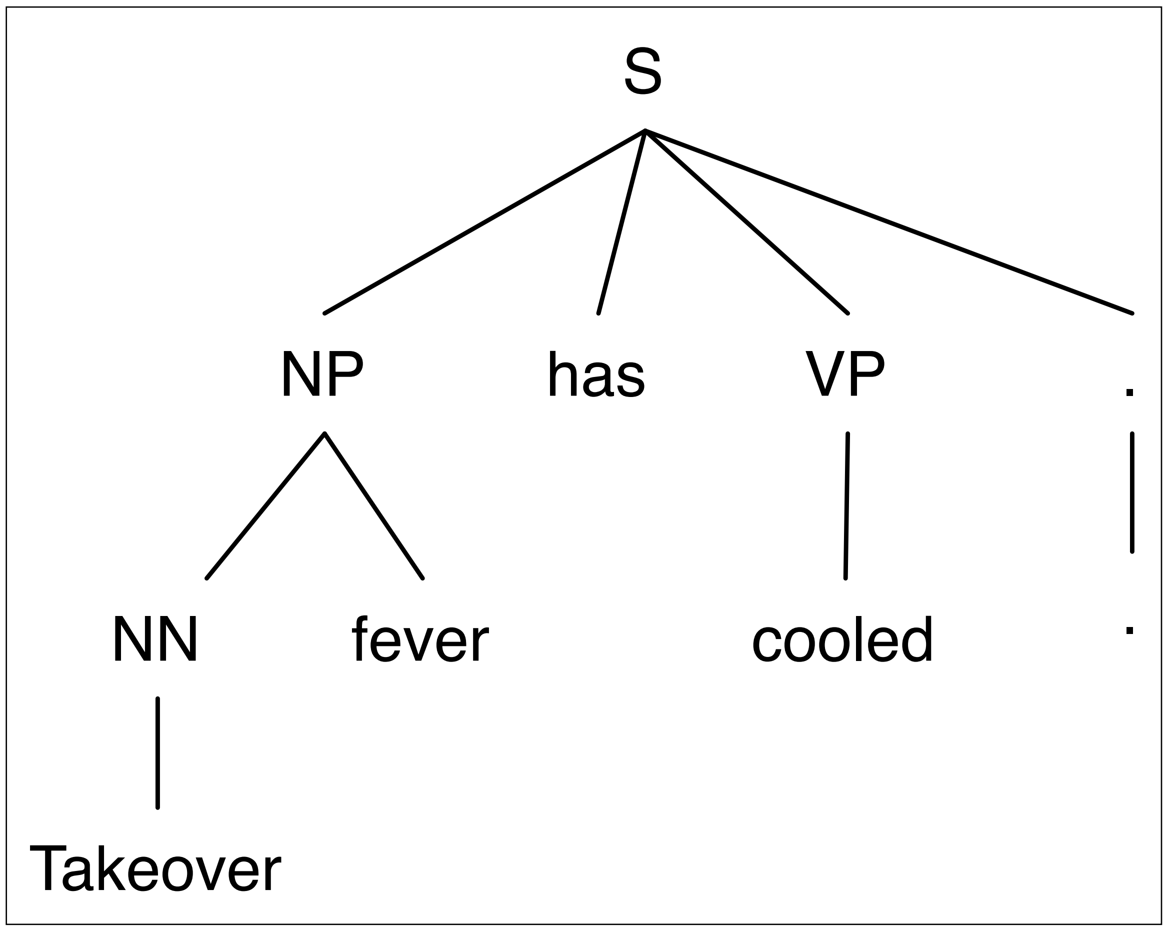
1. Generate the head word and tag
   $\Pr(h,t \mid P)$
2. Generate the head child
   $\Pr(H \mid P,h,t)$
3. Generate the sibling head labels and tags
   $\Pr(C,c_t \mid P,h,t,H)$
4. Generate the sibling head word
   $\Pr(c_w \mid P,h,t,H,c_h,c_t)$

Bilexical parsing (Collins Model 1)

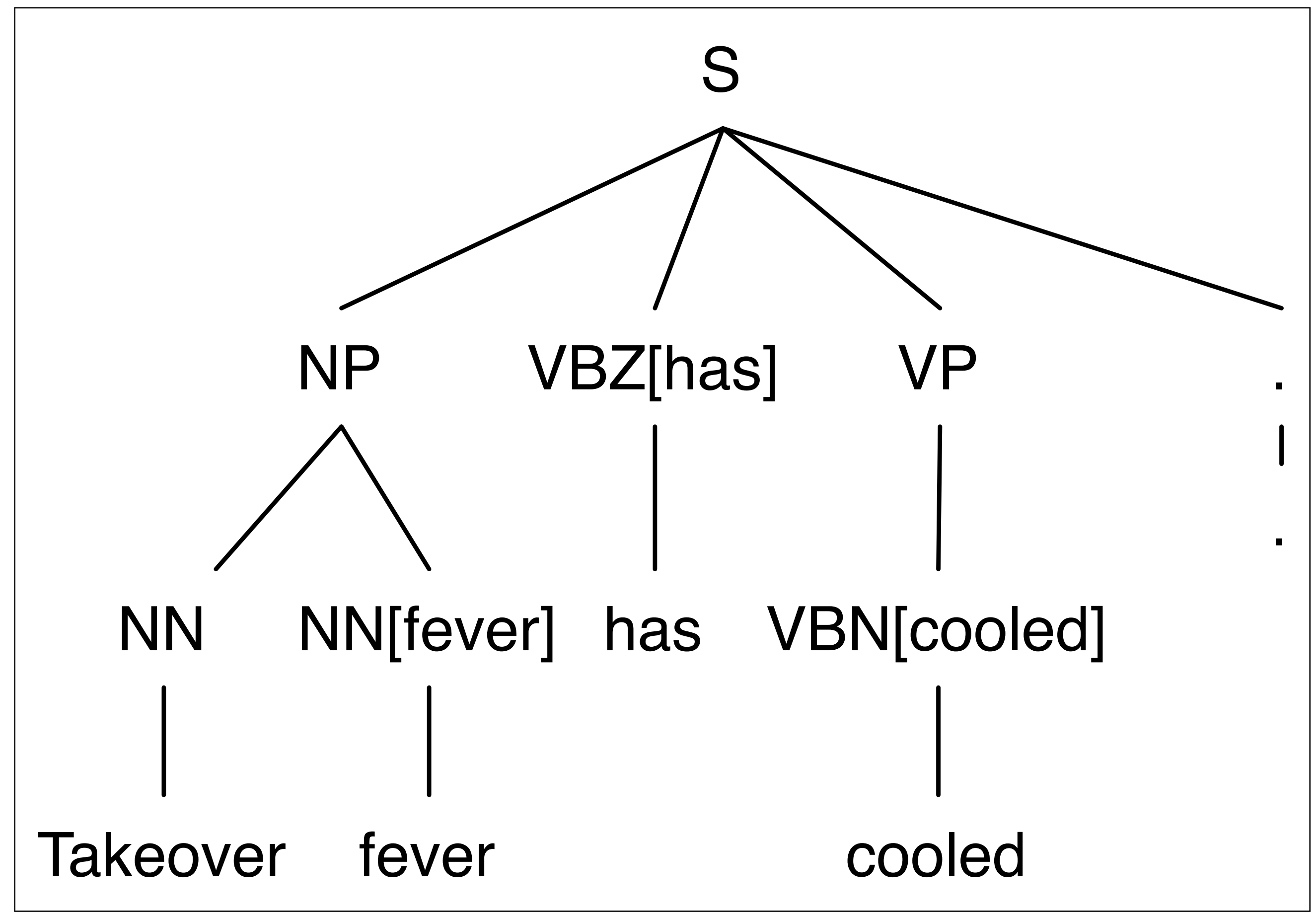


1. Generate the head word and tag
   $\Pr(h,t \mid P)$
2. Generate the head child
   $\Pr(H \mid P,h,t)$
3. Generate the sibling head labels and tags
   $\Pr(C,c_t \mid P,h,t,H)$
4. Generate the sibling head word
   $\Pr(c_w \mid P,h,t,H,c_h,c_t)$
5. Recurse

Raising

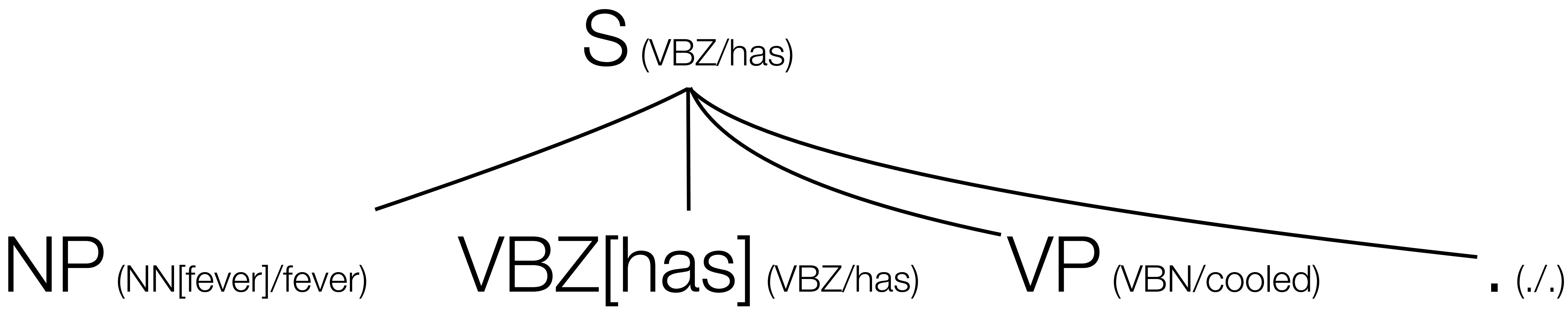


internal nodes removed

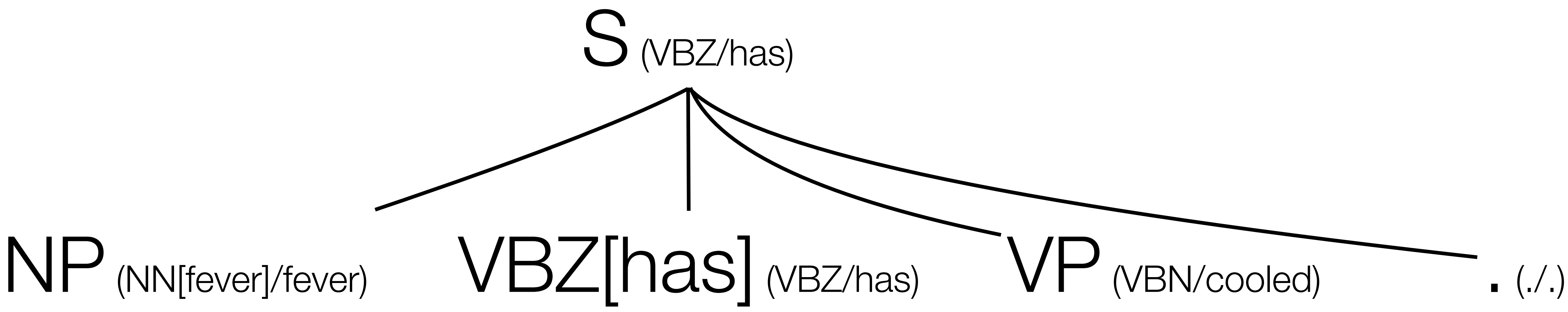


preterminals reintroduced

Conflict

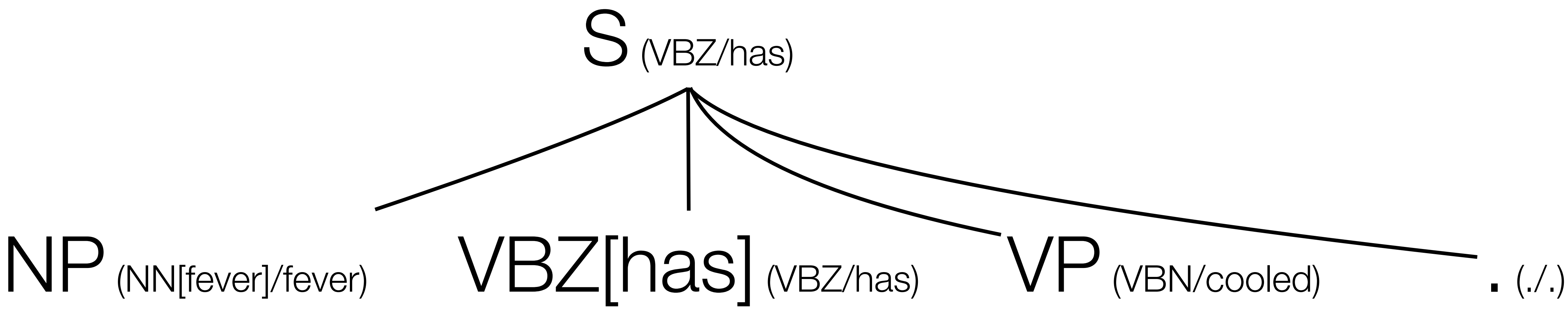


Conflict



three-level
interpolation

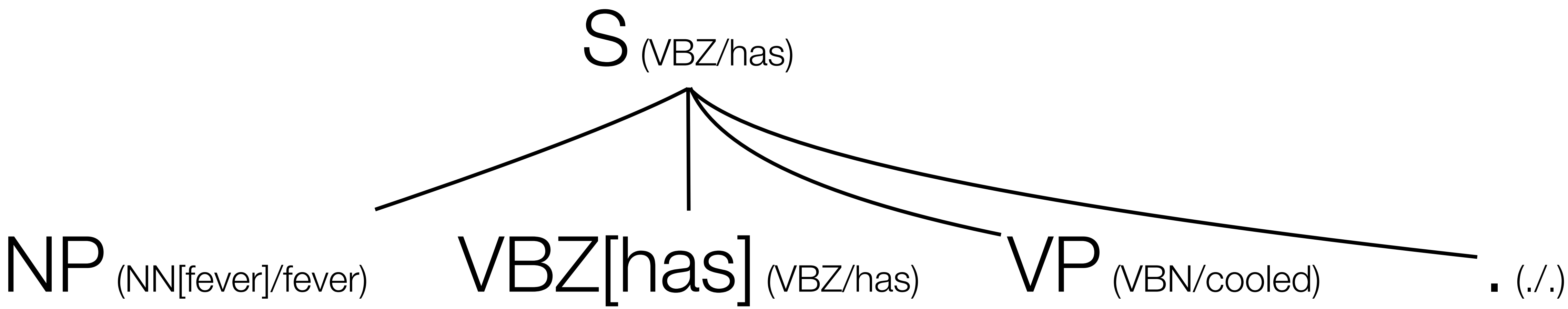
Conflict



$P(NP, NN[fever] \mid S, VBZ[has], VBZ, has, \leftarrow)$

three-level interpolation

Conflict

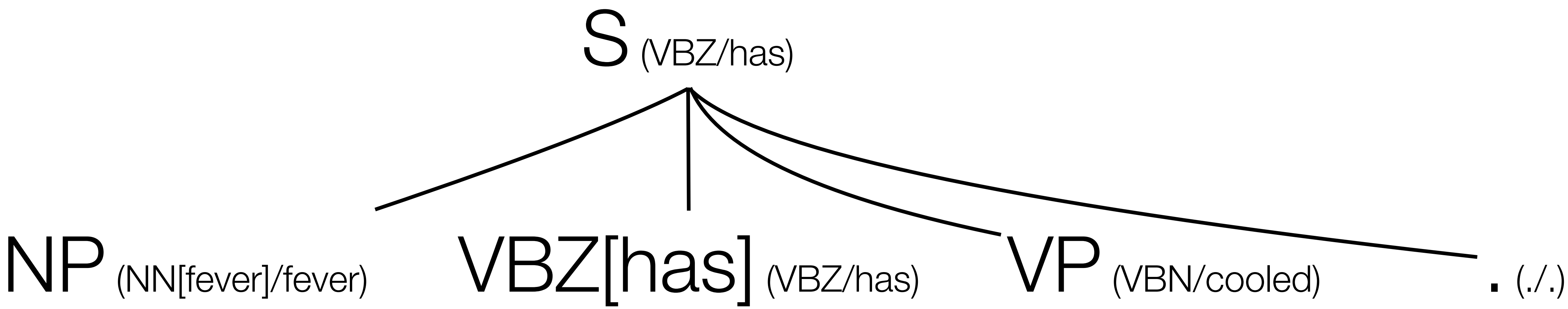


three-level interpolation

$P(\text{NP},\text{NN[fever]} \mid \text{S},\text{VBZ[has]},\text{VBZ},\text{has},\leftarrow)$

$P(\text{NP},\text{NN[fever]} \mid \text{S},\text{VBZ[has]},\text{VBZ},\leftarrow)$

Conflict



three-level interpolation

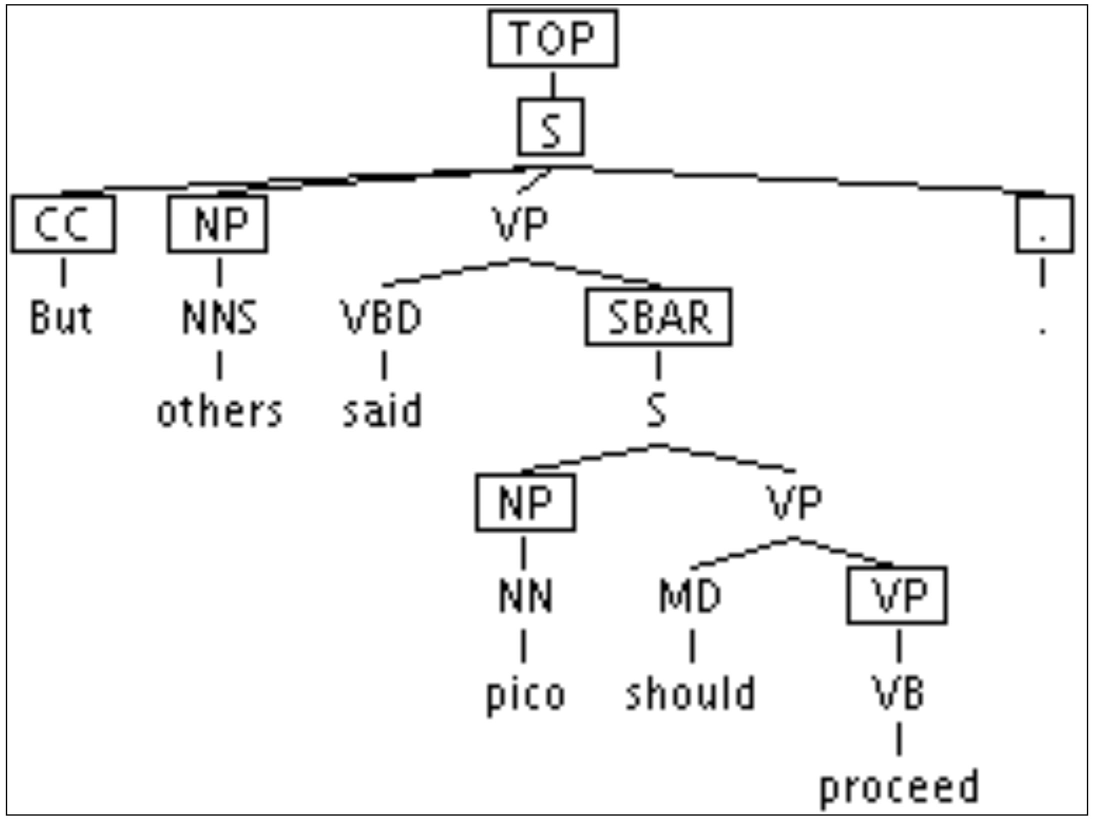
$$P(\mathrm{NP},\mathrm{NN[fever]} \mid \mathrm{S},\mathrm{VBZ[has]},\mathrm{VBZ},\mathrm{has},\leftarrow)$$

$$P(\mathrm{NP},\mathrm{NN[fever]} \mid \mathrm{S},\mathrm{VBZ[has]},\mathrm{VBZ},\leftarrow)$$

$$P(\mathrm{NP},\mathrm{NN[fever]} \mid \mathrm{S},\mathrm{VBZ[has]},\leftarrow)$$

TSG

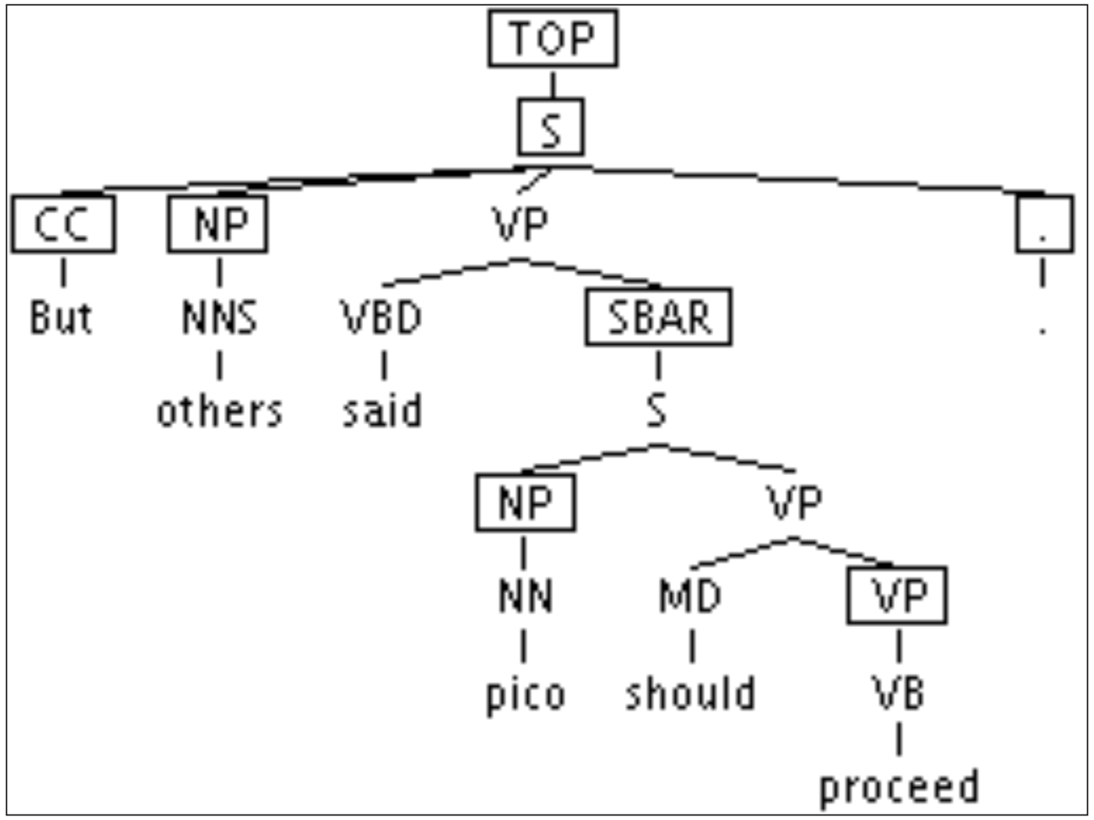
Subtrees can extend down to the leaves of the parse tree



parse tree from training data

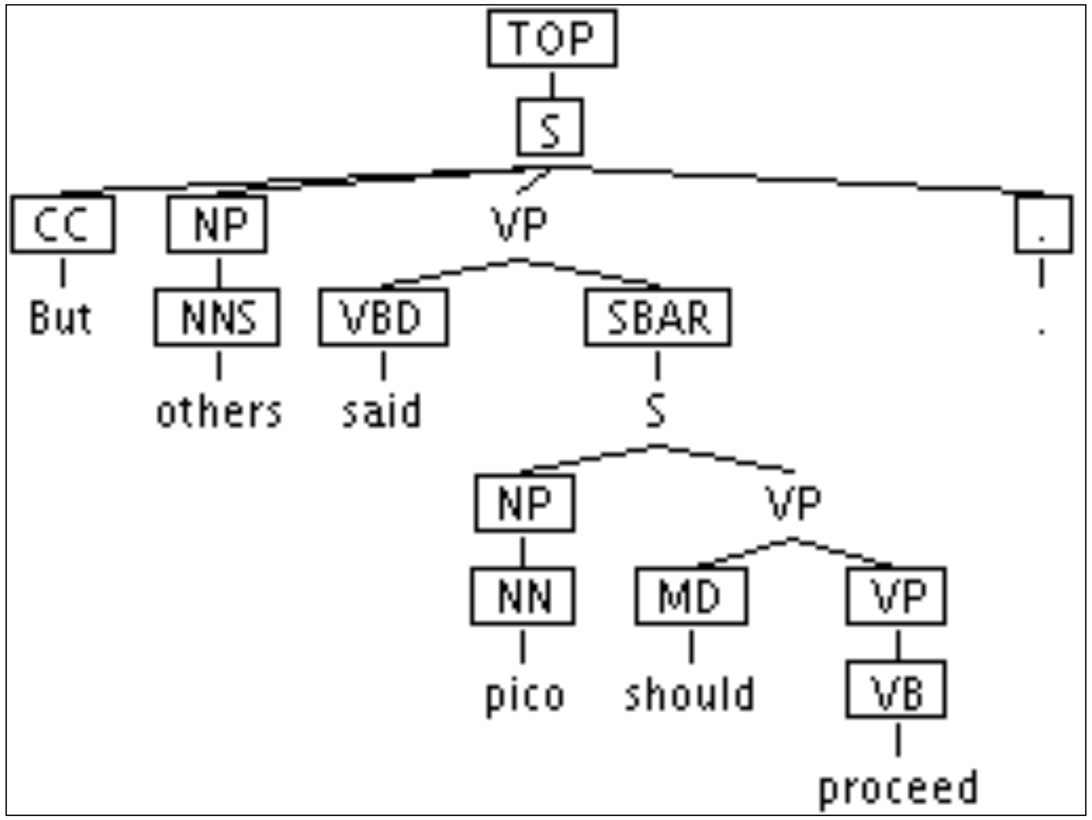
TSG

Subtrees can extend down to the leaves of the parse tree



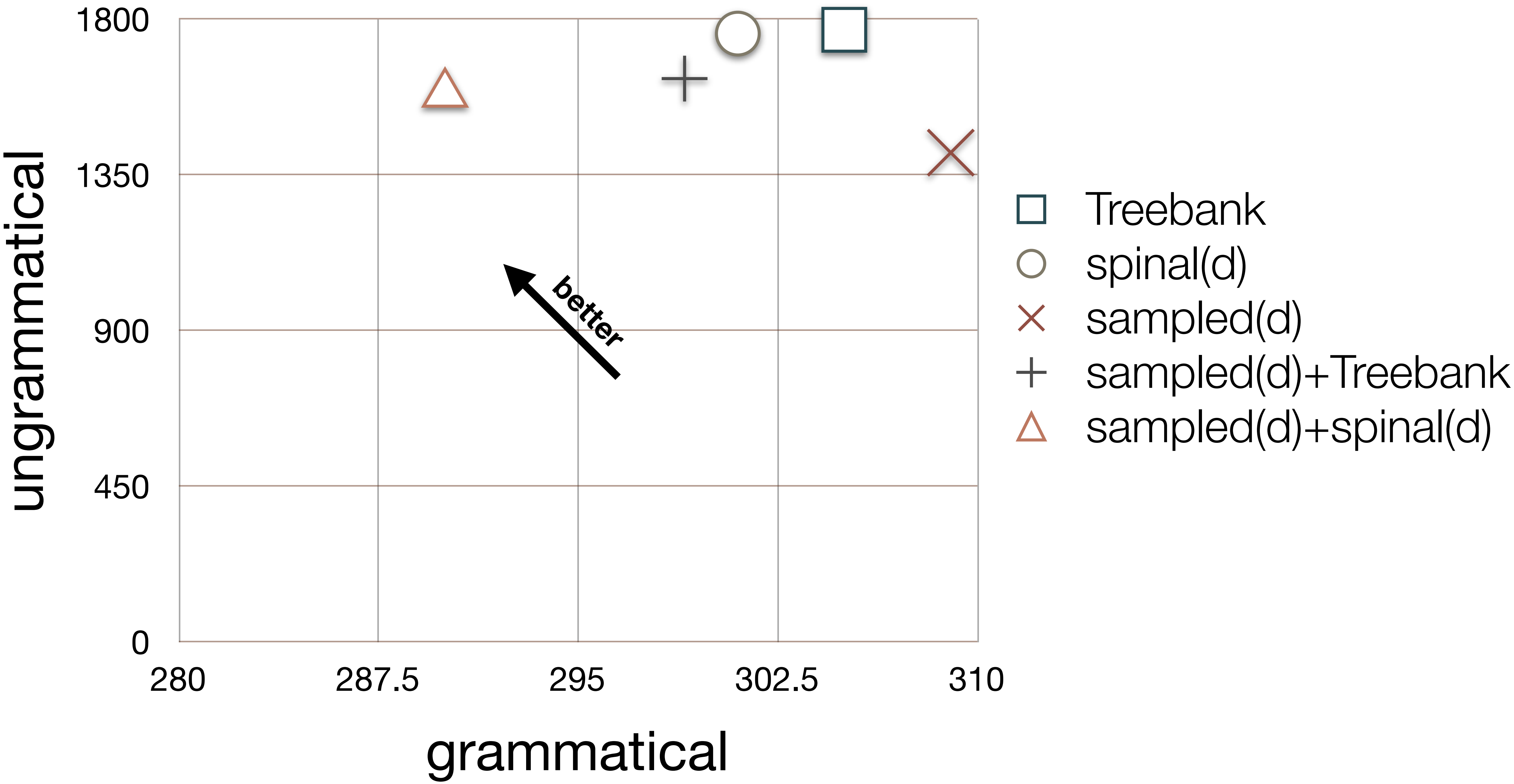
Detached TSG

Words are detached from TSG subtrees at the preterminal before flattening and raising



parse tree from training data

Perplexities



QUESTIONS

References

Trevor Cohn, Sharon Goldwater, and Phil Blunsom. 2009. Inducing compact but accurate tree-substitution grammars. In *Proc. NAACL*.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2009. A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1): 21–54.

Daisuke Okanohara and Jun’ichi Tsujii. 2007. A discriminative language model with pseudo-negative samples. In Proc ACL.

Matt Post and Daniel Gildea. 2009. Bayesian learning of a tree substitution grammar. In *Proc ACL*.

Joshua B Tenenbaum, Noah D. Goodman, and Timothy J. O'Donnell. 2009. Fragment Grammars: Exploring Computation and Reuse in Language. MIT Technical Report.

State-split grammars