Bayesian learning of a tree substitution grammar

Matt Post and Daniel Gildea • University of Rochester • Rochester, NY, USA

TSGs

- like CFGs, but nonterminals rewrite as subtrees of arbitrary size
- rule application is still context-free
- subsume context free grammars

	+	–
CFG	have annotated data, easy to train	small domain of locality, too much independence
TSG	larger rules capture long-distance deps	no annotated data, not clear how to learn them

Learning a TSG

Problem

- we don't have a manually annotated training corpus (like a Treebank with TSG derivations) from which we could just count rules
- heuristics produce large grammars with the wrong shape (see figure in right column)
- EM overfits and requires us to maintain explicit counts of an exponential number of subtrees

Solution

- collapsed Gibbs sampling with a nonparametric prior
- *avoids overfitting*: Dirichlet Process (DP) prior discourages larger subtrees unless the data warrants it
- *avoids counting*: sample derivations instead of maintaining explicit subtree counts

Derivations as segmentations

- annotate the nodes of a parse tree with flags marking rule boundaries; this induces a set of rules (and a derivation)
- DP helps overfitting: for each nonterminal X,

$$g_X \sim DP(G_X, \alpha)$$

$$G_X(t) = \Pr_\$(|t|; p_\$) \prod_{r \in t} \Pr_{\text{MLE}}(r)$$

where

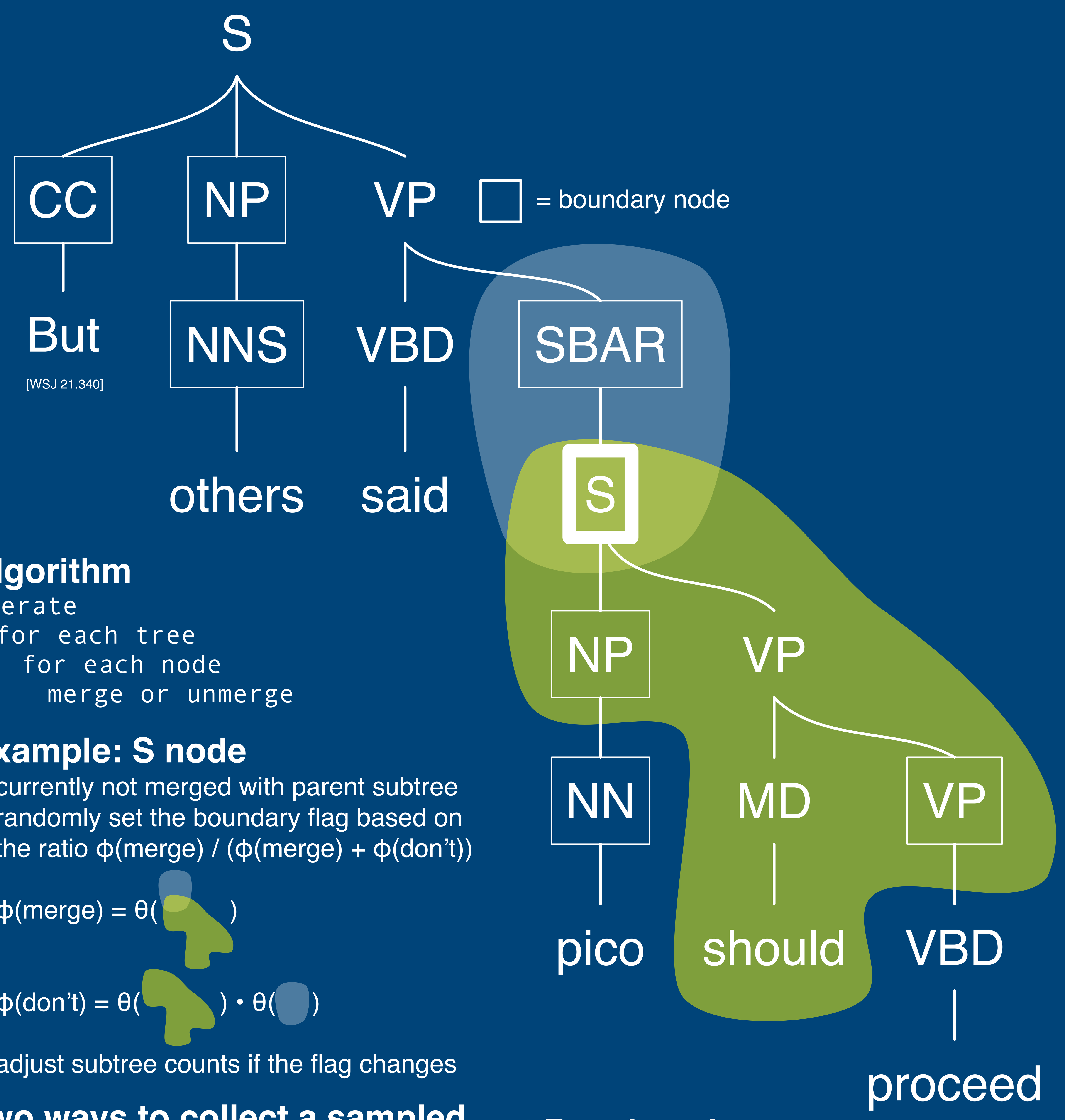
- α is a hyperparameter roughly controlling the variance
- $\Pr_\$$ is a geometric distribution on the number of PCFG rules in the subtree
- $\Pr_{\text{MLE}}$ is the Treebank PCFG prob of each rule in the subtree

- we use collapsed Gibbs sampling with a DP prior: sample a grammar g from the posterior P(g I data) based on the Chinese Restaurant Process view of the DP
- this is done by iteratively considering each node of each tree in the training data and randomly joining or splitting the subtree rooted at that node from the subtree its parent is part of
- subtree probability *t* is given as

$$\theta(t) = \frac{\text{count}_{z_t}(t) + \alpha G_{\text{root}(t)}(t)}{|z_t| + \alpha}$$

where z_t is the set of rewrites of root(t) in the current state of the rest of the corpus

- successfully used for segmentation tasks (Goldwater et al. 2009, DeNero et al. 2008)
- similar models were developed independently by Cohn et al. (2009) and O'Donnell et al. (2009)



Algorithm

```
iterate
  for each tree
    for each node
      merge or unmerge
```

Example: S node

- currently not merged with parent subtree
- randomly set the boundary flag based on the ratio ϕ(merge) / (ϕ(merge) + ϕ(don't))

ϕ(merge) = θ( )

ϕ(don't) = θ( ) • θ( )

- adjust subtree counts if the flag changes

Two ways to collect a sampled grammar

Subtree probabilities are set using relative frequency, but counts can be collected in two ways after running the sampler for *i* iterations

sum	$(\alpha, p_\$, \leq i)$	extract counts from the derivations at the end of iterations 1...i
point	$(\alpha, p_\$, i)$	extract counts from the i^{th} derivation

Baseline grammars

- Treebank PCFG
- Bod (2001) "minimal subset": all rules of height 1, plus 400K subtrees sampled at each height $2 \leq h \leq 14$, minus unlexicalized subtrees of $h \geq 6$, minus lexicalized subtrees with more than 12 words
- spinal extraction heuristic: extract as one subtree the sequence of CFG rules from leaf upward sharing a head

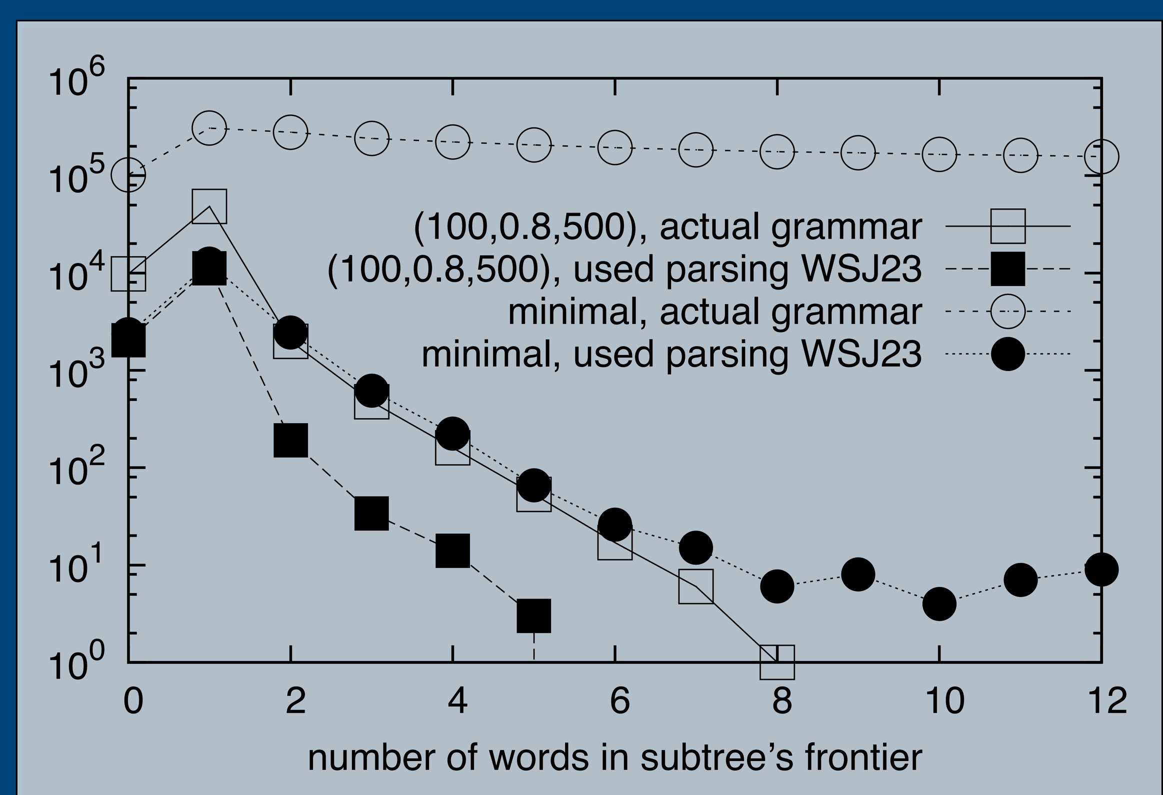
Results: accuracy

We approximated the most probable parse with the most probable derivation.

grammar	size	LP	LR	F_1
PCFG	46K	75.37	70.05	72.61
spinal	190K	80.30	78.10	**79.18**
minimal subset	2.56M	76.40	78.29	77.33
$(100, 0.7, 500)$	61K	81.95	81.76	81.85
$(100, 0.8, 500)$	60K	82.73	82.21	**82.46**
$(100, 0.9, 500)$	59K	82.57	81.53	82.04
$(100, 0.7, \leq 500)$	2.05M	82.81	82.01	82.40
$(100, 0.8, \leq 500)$	1.13M	83.06	82.10	**82.57**
$(100, 0.9, \leq 500)$	528K	83.17	81.91	82.53

Results: shape

The rules used when parsing are a better fit with the rules available in the grammar.



Results: sampled rules

(S (VP (TO to) VP))	0.106
(S (VP (TO to) (VP VBD NP)))	0.035
(S NP (VP (VBD said) SBAR) .)	0.029
(SBAR (IN that) S)	0.137
(SBAR S)	0.036
(SBAR (WHNP (WDT that)) S)	0.035
(SINV `` S , '' (VP (VBZ says)) NP .)	0.181
(SINV `` S , '' (VP (VBD said)) NP .)	0.176

Conclusion

We can efficiently learn compact TSGs that outperform heuristic approaches, and which take the expected shape in terms of histograms of subtree sizes.

UNIVERSITY *of* ROCHESTER