Bayesian non-parametric models of parsing and translation

Trevor Cohn

Department of Computer Science
University of Sheffield

October 2009

Joint work with Phil Blunsom and Sharon Goldwater
Inducing Grammars from Text

Language is hierarchical and recursive

- natural to model as a tree (e.g., CFG)
- big challenge to induce decent trees from text

Input: Every corner of Singapore is filled with fun
POS: DT NN IN NNP VBZ VBN IN NN
SotA:
Inducing Grammars from Text

Language is hierarchical and recursive

- natural to model as a tree (e.g., CFG)
- big challenge to induce decent trees from text

Input: Every corner of Singapore is filled with fun
POS: DT NN IN NNP VBZ VBN IN NN

SotA:

```
  X
 / \
/   \/
/     /
/     X
/     X
X     X
/ \   / \ 
DT NN IN NNP VBZ X
/ \   / \ 
IN NN IN NN
```
Inducing Grammars from Text

Language is hierarchical and recursive
- natural to model as a tree (e.g., CFG)
- big challenge to induce decent trees from text

Input: Every corner of Singapore is filled with fun
POS: DT NN IN NNP VBZ VBN IN NN
Ideal:
Every corner of Singapore is filled with fun.

新加坡的每一个角落都充满乐趣。
Every corner of Singapore is filled with fun.

新加坡的每一个角落都充满着乐趣。
Learning a translation model = grammar induction

Every corner of Singapore is filled with fun

新加坡的每一个角落都充满着乐趣
Learning a translation model = grammar induction

Every corner of Singapore is filled with fun.
Learning a translation model = grammar induction

Every corner of Singapore is filled with fun.
Semi-supervised Grammar Induction

Present two semi-supervised models of grammar induction

1. Natural language parsing
   - learns a tree-substitution grammar (DOP) from a tree-banked corpus
   - learns to use large tree fragments as grammar productions

2. Machine translation
   - learns a string-to-tree transducer from parallel bitext
   - jointly infers the alignment and synchronous grammar derivation
   - informed by target syntax
Outline

1 Parsing
   - Tree substitution grammar
   - Bayesian Primer
   - The Model
   - Experiments
   - Summary

2 Machine Translation
   - Motivation
   - Model
   - Experiments
   - Summary

3 Conclusions
Outline

1 Parsing
   - Tree substitution grammar
   - Bayesian Primer
   - The Model
   - Experiments
   - Summary

2 Machine Translation
   - Motivation
   - Model
   - Experiments
   - Summary

3 Conclusions
Probabilistic Context Free Grammar’s short-comings

S
  /  
NP  VP
   /  
Amazon  VBD
       /  
  closed  PP
     /  
IN  NP
    /  
up  5 cents a share
Probabilistic Context Free Grammar’s short-comings

Amazon said up 5 cents a share
Probabilistic Context Free Grammar’s short-comings

S

NP
Amazon

VP
VBD
closed

PP
IN
up

NP
5 cents a share
Probabilistic Context Free Grammar’s short-comings

```
S
  |   |
 NP  VP
  |   |
 Amazon  VBD
  |   |
 closed  IN
  |   |
      NP
  |   |
in  5 cents a share
```
Tree-substitution grammar

- rewrite with elementary trees not just immediate children
- large elementary trees can describe interesting phenomena
  - agreement
  - sub-categorisation
  - idioms
- ... similar to what grammar engineering does for CFGs
Tree-substitution grammar

- rewrite with elementary trees not just immediate children
- large elementary trees can describe interesting phenomena
  - agreement
  - sub-categorisation
  - idioms
- ... similar to what grammar engineering does for CFGs

We present a model of TSG induction from tree-banked data
TSG example

S → NP (VP (V hates) NP)
NP → George
NP → broccoli

S → (NP George) (VP V (NP broccoli))
V → hates
TSG example

S ➔ NP (VP (V hates) NP)
NP ➔ George
NP ➔ broccoli
S ➔ (NP George) (VP V (NP broccoli))
V ➔ hates
TSG example

S \rightarrow \text{NP} (\text{VP} (\text{V} \text{hates}) \text{NP})

\text{NP} \rightarrow \text{George}

\text{NP} \rightarrow \text{broccoli}

S \rightarrow (\text{NP George}) (\text{VP V (NP broccoli)})

\text{V} \rightarrow \text{hates}
TSGs

Properties
- generalisation of CFGs
- weakly equivalent to CFGs (describe same string languages)
- can reduce TSG rules into CFG rules

Challenges
- Derivation ambiguity
- How to obtain and weight elementary trees?
TSGs

Properties
- generalisation of CFGs
- weakly equivalent to CFGs (describe same string languages)
- can reduce TSG rules into CFG rules

Challenges
- Derivation ambiguity
- How to obtain and weight elementary trees?

Previous work
- Data oriented parsing (Bod et al. 2003)
- Tree adjoining grammar induction (Chiang and Bikel, 2002)
- Adaptor Grammars (Johnson et al. 2007)
Previous work: DOP

DOP attempts to use all subtrees of the training data

- an \textit{exponential} number in the size of each tree
- various estimators for elementary tree weights
  - counting heuristic (Bod 1992)
  - MLE using EM (Bod 2000)
  - held-out estimator (Zollmann & Sima’an 2005)
  - ‘parsimony’ heuristic (Zuidema 2007)
  - ... and more

- good empirical results, but theoretically inadequate
Degeneracy of the MLE

Training:

\[ S \rightarrow N (V \text{ likes } N) \]

\[ NP \rightarrow Peter p = 1 \]

\[ NP \rightarrow Mary p = 1/2 \]

\[ \text{Likelihood} = (1 \times 1/2 \times 1/2) \times (1 \times 1/2 \times 1/2) = 1/16 \]
Degeneracy of the MLE

Training:

```
S
  |   |  
Peter | likes | Mary
```

```
S
  |   |  
Mary | likes | Peter
```

Expected Grammar:

\[
S \rightarrow N (V \text{ likes}) N
\]

\[
NP \rightarrow Peter
\]

\[
NP \rightarrow Mary
\]

Likelihood = \((1 \times \frac{1}{2} \times \frac{1}{2}) \times (1 \times \frac{1}{2} \times \frac{1}{2}) = \frac{1}{16}\)
Degeneracy of the MLE

MLE grammar:

\[
S \rightarrow (N \text{ Peter}) (V \text{ likes}) (N \text{ Mary}) \\
S \rightarrow (N \text{ Mary}) (V \text{ likes}) (N \text{ Peter})
\]

Likelihood = \( \left( \frac{1}{2} \right) \times \left( \frac{1}{2} \right) = \frac{1}{4} \)
Degeneracy of the MLE

Training:

\[
\begin{array}{ccc}
S & \rightarrow & (N \text{ Peter}) (V \text{ likes}) (N \text{ Mary}) \\
& & p = \frac{1}{2} \\
S & \rightarrow & (N \text{ Mary}) (V \text{ likes}) (N \text{ Peter}) \\
& & p = \frac{1}{2}
\end{array}
\]

Likelihood = \( \left( \frac{1}{2} \right) \times \left( \frac{1}{2} \right) = \frac{1}{4} \)

MLE will use largest units possible in order to find unique derivations
⇒ use a prior to bias towards small units
This work: a Bayesian Alternative

Generative model of an infinite grammar:
- seeks to find the simplest possible weighted grammar
- while still accurately modelling the data

Uses a prior within a Bayesian framework
- to bias towards sparse non-terminal expansions
- and bias towards small elementary trees

Performs unsupervised induction over treebanked input.
Outline

1 Parsing
   - Tree substitution grammar
   - Bayesian Primer
   - The Model
   - Experiments
   - Summary

2 Machine Translation
   - Motivation
   - Model
   - Experiments
   - Summary

3 Conclusions
Parameter estimation

- Standard approach is to find optimal parameter values, $\theta$, using Bayes’ rule:

$$P(\theta|D) \propto P(D|\theta)P(\theta)$$

- Then use either Maximum-likelihood estimate (MLE)

$$\theta^{MLE} = \arg\max_{\theta} P(D|\theta)$$

- or Maximum-a-posteriori estimate (MAP)

$$\theta^{MAP} = \arg\max_{\theta} P(D|\theta)P(\theta)$$
Bayesian integration

- But we’re not interested in $\theta$ but instead what we can do with it:
  - prediction on new data: $P(D'|\theta)$
  - estimating hidden structure of the data: $P(H|\theta, D)$

- So why not just estimate these distributions directly?
  - prediction: $P(D'|D) = \int P(D'|\theta)P(\theta|D)d\theta$
  - structure: $P(H|D) = \int P(H|\theta, D)P(\theta|D)d\theta$

**Benefits**

- Provides **robustness** to flat distributions of $\theta$
- Allows **flexibility** to choose the prior
Dirichlet Distribution

- For multinomial distributions the Dirichlet is a natural prior.

\[ \beta < 1: \text{prefer sparse solutions, } \beta > 1: \text{uniform, } \beta = 1: \text{no preference} \]
Posterior distribution is simple after integrating out the parameters

\[ P(w_{n+1}|w^n_1, \beta) = \frac{c_{w_{n+1}} + \beta}{c + W\beta} \]
Dirichlet Process

- Extension of Dirichlet to infinite distributions
  - Combines a distribution over the number and size of clusters
  - With a base distribution $P_0$ describing each cluster

$$P(w_{n+1}|w_1^n, \beta) = \frac{c_{w_{n+1}} + \beta P_0(w_{n+1})}{c + \beta}$$

- Number of items in each cluster described by the Chinese Restaurant Process
  - Restaurant has an infinite number of tables, with infinite seating capacity.
  - Customers sit at an occupied table w.p.p. number at that table
  - Sit at an empty table w.p.p. $\beta$

- Produces a power-law over cluster sizes (c.f. Zipf’s ‘law’)
Outline

1 Parsing
   - Tree substitution grammar
   - Bayesian Primer
   - The Model
   - Experiments
   - Summary

2 Machine Translation
   - Motivation
   - Model
   - Experiments
   - Summary

3 Conclusions
Model

Distribution over elementary trees, $e$, given root non-terminal $c$:

\[
G_c|\alpha_c, P_0 \sim \text{DP}(\alpha_c, P_0(\cdot|c)) \\
e|c \sim G_c
\]

- $P_0(\cdot|c)$ is the base distribution
- $\alpha_c$ is the concentration parameter
- Integrate out $G_c$

\[
p(e_i|e_{<i}, c, \alpha_c, P_0) = \frac{n_{e_i,c}^< + \alpha_c P_0(e_i|c)}{n_{<i}^c + \alpha_c}
\]

- Exchangelable: all orderings of input have same probability
CRP analogy: ‘seating’ a new NP

Original seating assignment:
CRP analogy: ‘seating’ a new NP

Share a table, prob = \( \frac{8}{8 + \alpha} \):

→ Reuse the table’s rule
CRP analogy: ‘seating’ a new NP

Sit at an empty table, prob = $\frac{\alpha}{8+\alpha}$:

→ Draw a new rule from $P_0(\cdot | c = NP)$
Base distribution, $P_0(\cdot|c)$

Generative process recursively expands each non-terminal or stops.

E.g., to generate

1. expand NP as NP PP
2. stop for child NP
3. continue for PP
4. expand PP as IN NP
5. continue for IN
6. expand IN as terminal of
7. stop for NP
Base distribution, $P_0(\cdot | c)$

Generative process recursively expands each non-terminal or stops.

E.g., to generate

```
NP
/   |
 NP  PP
```

1. expand NP as NP PP
2. stop for child NP
3. continue for PP
4. expand PP as IN NP
5. continue for IN
6. expand IN as terminal of
7. stop for NP

Expansion probabilities

- uniform ($P_0^M$) or from a MLE PCFG ($P_0^C$)
- $P_0^M$ has stronger bias towards small elementary trees
- hyper-parameters $\beta_c$ over binomial expansion decisions
Training

- given training corpus (treebank)
- introduce binary substitution variables
- define Gibbs sampler over these variables
Gibbs sampler
Gibbs sampler

Diagram with nodes labeled a, b, c, d, e, f, g.
Gibbs sampler
Gibbs sampler

\[ S \]

\[ \text{NP}, 1 \quad \text{VP}, 0 \]

\[ \text{George} \quad \text{V}, 0 \quad \text{NP}, 1 \]

\[ \text{hates} \quad \text{broccoli} \]

\[ \text{Subst} = 0 \]

\[ e_M = S \]

\[ \text{NP} \quad \text{VP} \]

\[ \text{V} \quad \text{NP} \]

\[ \text{hates} \]

\[ \text{Subst} = 1 \]

\[ e_A = S \]

\[ e_B = \text{VP} \]

\[ \text{NP} \quad \text{VP} \]

\[ \text{V} \quad \text{NP} \]

\[ \text{hates} \]
Present results for the following algorithms

**Maximum Probability Derivation (MPD)**
\[
\arg\max_d p(d|w, t, e, \alpha, \beta)
\]

**Maximum Probability Parse (MPP)**
\[
\arg\max_t p(t|w, t, e, \alpha, \beta)
\]

**Maximum Expected Rules (MER)**
find tree with max marginal probability
for each CFG rule at each span
Outline

1 Parsing
   - Tree substitution grammar
   - Bayesian Primer
   - The Model
   - Experiments
   - Summary

2 Machine Translation
   - Motivation
   - Model
   - Experiments
   - Summary

3 Conclusions
Experiments

Train on the Penn treebank
- **small**: single section, 2k trees
- **full**: sections 2–21, 33k trees
- lossy right binarisation, single count tokens replaced with UNK
- sample for 5k – 20k iterations with annealing
- measuring F1 over labelled constituents
- results are averaged over 3–5 independent runs

Baselines:
- MLE PCFG
- Berkeley parser (Petrov et al., 2006)
Results: small (devel. sec 22)

<table>
<thead>
<tr>
<th>Model</th>
<th>PCFG</th>
<th>TSG–M</th>
<th>TSG–C</th>
<th>Berkeley</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>55</td>
<td>60</td>
<td>65</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>75</td>
<td>80</td>
<td></td>
</tr>
<tr>
<td>MPD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MER</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bayesian parsing and translation

Cohn (Sheffield)

October 2009
Results on test set (sec. 23):

<table>
<thead>
<tr>
<th></th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCFG</td>
<td>70.7</td>
</tr>
<tr>
<td>Bayesian TSG</td>
<td>84.0</td>
</tr>
<tr>
<td>Berkeley</td>
<td>87.7</td>
</tr>
<tr>
<td>Berkeley-best</td>
<td>91.2</td>
</tr>
<tr>
<td>Zuidema (2007) DOP</td>
<td>83.8*</td>
</tr>
</tbody>
</table>
Induced grammars

<table>
<thead>
<tr>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP →</td>
</tr>
<tr>
<td>DT NP</td>
</tr>
<tr>
<td>NNS</td>
</tr>
<tr>
<td>DT NN</td>
</tr>
<tr>
<td>(DT the) NP</td>
</tr>
<tr>
<td>JJ NNS</td>
</tr>
<tr>
<td>NP (PP (IN of) NP)</td>
</tr>
<tr>
<td>NP PP</td>
</tr>
<tr>
<td>NP (NP (CC and) NP)</td>
</tr>
<tr>
<td>JJ NP</td>
</tr>
<tr>
<td>NN NNS</td>
</tr>
<tr>
<td>(DT the) NNS</td>
</tr>
<tr>
<td>DT (NP JJ NN)</td>
</tr>
<tr>
<td>NN</td>
</tr>
<tr>
<td>JJ NN</td>
</tr>
<tr>
<td>(NP DT NN) (PP (IN of) NP)</td>
</tr>
</tbody>
</table>
Induced grammars

<table>
<thead>
<tr>
<th>NP →</th>
</tr>
</thead>
<tbody>
<tr>
<td>DT NP</td>
</tr>
<tr>
<td>NNS</td>
</tr>
<tr>
<td>DT NN</td>
</tr>
<tr>
<td>(DT the) NP</td>
</tr>
<tr>
<td>JJ NNS</td>
</tr>
<tr>
<td>NP (PP (IN of) NP)</td>
</tr>
<tr>
<td>NP PP</td>
</tr>
<tr>
<td>NP (NP (CC and) NP)</td>
</tr>
<tr>
<td>JJ NP</td>
</tr>
<tr>
<td>NN NNS</td>
</tr>
<tr>
<td>(DT the) NNS</td>
</tr>
<tr>
<td>DT (NP JJ NN)</td>
</tr>
<tr>
<td>NN</td>
</tr>
<tr>
<td>JJ NN</td>
</tr>
<tr>
<td>(NP DT NN) (PP (IN of) NP)</td>
</tr>
</tbody>
</table>
### Induced grammars: lexicalised VP expansions

<table>
<thead>
<tr>
<th>VP →</th>
</tr>
</thead>
<tbody>
<tr>
<td>(VBD said) (SBAR (S (NP (PRP it)) VP))</td>
</tr>
<tr>
<td>(VBD said) (SBAR (S NP VP))</td>
</tr>
<tr>
<td>(VBD rose) (V̄P (NP CD (NN %)) V̄P)</td>
</tr>
<tr>
<td>(VBP want) S</td>
</tr>
<tr>
<td>(VBD said) (SBAR (S (NP (PRP he)) VP))</td>
</tr>
<tr>
<td>(VBZ plans) S</td>
</tr>
<tr>
<td>(VBD said) (SBAR S)</td>
</tr>
<tr>
<td>(VBZ says) (SBAR (S NP VP))</td>
</tr>
<tr>
<td>(VBP think) (SBAR S)</td>
</tr>
<tr>
<td>(VBD agreed) (S (VP (TO to) (VP VB V̄P)))</td>
</tr>
<tr>
<td>(VBZ includes) NP</td>
</tr>
<tr>
<td>(VBZ says) (SBAR (S (NP (PRP he)) VP))</td>
</tr>
<tr>
<td>(VBZ wants) S</td>
</tr>
<tr>
<td>(VBD closed) (V̄P (PP (IN at) NP) (V̄P , ADVP))</td>
</tr>
</tbody>
</table>
### Induced grammars: lexicalised VP expansions

<table>
<thead>
<tr>
<th>VP →</th>
</tr>
</thead>
<tbody>
<tr>
<td>(VBD said) (SBAR (S (NP (PRP it)) VP))</td>
</tr>
<tr>
<td>(VBD said) (SBAR (S NP VP))</td>
</tr>
<tr>
<td>(VBD rose) (V(\bar{P}) (NP CD (NN %)) V(\bar{P}))</td>
</tr>
<tr>
<td>(VBP want) S</td>
</tr>
<tr>
<td>(VBD said) (SBAR (S (NP (PRP he)) VP))</td>
</tr>
<tr>
<td>(VBZ plans) S</td>
</tr>
<tr>
<td>(VBD said) (SBAR S)</td>
</tr>
<tr>
<td>(VBZ says) (SBAR (S NP VP))</td>
</tr>
<tr>
<td>(VBP think) (SBAR S)</td>
</tr>
<tr>
<td>(VBD agreed) (S (VP (TO to) (VP VB V(\bar{P}))) )</td>
</tr>
<tr>
<td>(VBZ includes) NP</td>
</tr>
<tr>
<td>(VBZ says) (SBAR (S (NP (PRP he)) VP))</td>
</tr>
<tr>
<td>(VBZ wants) S</td>
</tr>
<tr>
<td>(VBD closed) (V(\bar{P}) (PP (IN at) NP) (V(\bar{P}), ADVP))</td>
</tr>
</tbody>
</table>
## Induced grammars: lexicalised VP expansions

<table>
<thead>
<tr>
<th>VP →</th>
</tr>
</thead>
<tbody>
<tr>
<td>(VBD said) (SBAR (S (NP (PRP it)) VP))</td>
</tr>
<tr>
<td>(VBD said) (SBAR (S NP VP))</td>
</tr>
<tr>
<td>(VBD rose) (VĪP (NP CD (NN %)) VĪP)</td>
</tr>
<tr>
<td>(VBP want) S</td>
</tr>
<tr>
<td>(VBD said) (SBAR (S (NP (PRP he)) VP))</td>
</tr>
<tr>
<td>(VBZ plans) S</td>
</tr>
<tr>
<td>(VBD said) (SBAR S)</td>
</tr>
<tr>
<td>(VBZ says) (SBAR (S NP VP))</td>
</tr>
<tr>
<td>(VBP think) (SBAR S)</td>
</tr>
<tr>
<td>(VBD agreed) (S (VP (TO to) (VP VB VĪP)))</td>
</tr>
<tr>
<td>(VBZ includes) NP</td>
</tr>
<tr>
<td>(VBZ says) (SBAR (S (NP (PRP he)) VP))</td>
</tr>
<tr>
<td>(VBZ wants) S</td>
</tr>
<tr>
<td>(VBD closed) (VĪP (PP (IN at) NP) (VĪP , ADVP))</td>
</tr>
</tbody>
</table>
Outline

1 Parsing
   - Tree substitution grammar
   - Bayesian Primer
   - The Model
   - Experiments
   - Summary

2 Machine Translation
   - Motivation
   - Model
   - Experiments
   - Summary

3 Conclusions
Summary

Summary:
- non-parametric Bayesian model of tree substitution grammar
- uses prior to solve MLE’s degeneracy
- promising performance

Extensions:
- improve the sampler
  - consider larger groups of variables
  - auxiliary variable ‘beam’ sampler
  - parallel tempering
- hierarchical prior
- unsupervised grammar induction (O’Donnell et al. 2009)
The Machine Translation Pipeline

Start with a parallel corpus

1. automatically align the words
2. merge directional word alignments
3. extract FST/grammar rules
4. ‘estimate’ weights for each rule

Every corner of Singapore is filled with fun

新加坡的每一个角落都充满着乐趣
The Machine Translation Pipeline

Start with a parallel corpus

1. automatically align the words
2. merge directional word alignments
3. extract FST/grammar rules
4. ‘estimate’ weights for each rule

Every corner of Singapore is filled with fun.

新加坡的每一个角落都充满着乐趣.
The Machine Translation Pipeline

Start with a parallel corpus

1. automatically align the words
2. merge directional word alignments
3. extract FST/grammar rules
4. ‘estimate’ weights for each rule

Every corner of Singapore is filled with fun

新加坡的每一个角落都充满着乐趣
The Machine Translation Pipeline

Start with a parallel corpus

1. automatically align the words
2. merge directional word alignments
3. extract FST/grammar rules
4. ‘estimate’ weights for each rule

Every corner of Singapore is filled with fun.

新加坡 的 每 一 个 角落 都 充满 着 乐趣.
The Machine Translation Pipeline

Start with a parallel corpus
1. automatically align the words
2. merge directional word alignments
3. extract FST/grammar rules
4. ‘estimate’ weights for each rule

Every corner of Singapore is filled with fun

...now we have a translation model
But...

The cascade of models & heuristics has some problems:

- word alignment doesn’t incorporate phrasal or syntax cues
  - e.g., rarely capture long distance reorderings
- ambiguity over which FST/grammar rules to extract
But...

The cascade of models & heuristics has some problems:
- word alignment doesn’t incorporate phrasal or syntax cues
  - e.g., rarely capture long distance reorderings
- ambiguity over which FST/grammar rules to extract

This work

Presents an alternative joint model for synchronous grammar estimation which:
- operates over string and tree pairs (a type of STSG)
- jointly performs alignment and grammar extraction
- in a single principled Bayesian model
Every corner of Singapore is filled with fun.

Training instance
Example: existing approach (Galley et al. 2004)

Every corner of Singapore is filled with fun.

Step 1: word alignment
Example: existing approach (Galley et al. 2004)

Every corner of Singapore is filled with fun.

Step 2: rule extraction heuristic
Example: existing approach (Galley et al. 2004)

Step 2: the rules extracted
Example: existing approach (Galley et al. 2004)

Step 3: estimate a grammar
Small word alignment errors can lead to massive rules

Every corner of Singapore is filled with fun.
Small word alignment errors can lead to massive rules.

Every corner of Singapore is filled with fun.
Our approach

Every corner of Singapore is filled with fun.

Training instance

新加坡的每一个角落都充满着乐趣。
Our approach

One step: induce rules directly, which specifies the grammar.
Outline

1 Parsing
   - Tree substitution grammar
   - Bayesian Primer
   - The Model
   - Experiments
   - Summary

2 Machine Translation
   - Motivation
   - Model
   - Experiments
   - Summary

3 Conclusions
The Model

Model grammar as a set of distributions over productions for each non-terminal symbol,

\[(e, w) \mid c \sim G_c\]

\[G_c \mid \alpha_c, P_0 \sim \text{DP}(\alpha_c, P_0(\cdot \mid c))\]

where

- \(P_0(\cdot \mid c)\) is the base distribution over (tree fragment, string) pairs
- \(\alpha_c\) is the concentration parameter controlling sparsity
- a derivation is a sequence of rules, each drawn from \(G\)

Dirichlet Process Prior

Encodes prior belief that most probability mass is on few items
I.e., that the grammar has few productions
Generative process
Generative process
Generative process
Generative process

```
S
  -- NP
    -- DT NN
      Every corner
    -- IN
      of
  -- NP
    -- NNP

Cohn (Sheffield)
Bayesian parsing and translation
October 2009 46 / 58
```
Every corner of Singapore is filled with fun.
Base distribution

A distribution over the infinite space of (tree fragment, string) pairs rewriting $c$.
- penalises larger rules cf. small rules
- tree fragment size measured in terms of CFG productions
- rule size measured in terms of terminal tokens

Decomposes $P_0(e, w|c) = P(e|c)P(w|e)$, where
- tree probability $P(e|c)$ as per before
- string probability $P(w|e)$
  - generate zero or more terminal tokens
  - insert each variable from $e$ into the string
Base distribution

A distribution over the infinite space of (tree fragment, string) pairs rewriting $c$.

- penalises larger rules cf. small rules
- tree fragment size measured in terms of CFG productions
- rule size measured in terms of terminal tokens

Decomposes $P_0(e, w|c) = P(e|c)P(w|e)$, where

- tree probability $P(e|c)$ as per before
- string probability $P(w|e)$
  - generate zero or more terminal tokens
  - insert each variable from $e$ into the string

Net effect: bias towards small grammar with small rules
Inference

Train using a Gibbs sampler

Gibbs State

We represent the derivation of a (tree, string) pair
- via an alignment variable, \( a_v \), with each node in the tree
- indicates if the node is internal to a rule or heading/footing a rule
- and specifies the aligned span of tokens in the string
- together this specifies the set of rules in the derivation

The Gibbs sampler samples updates to each \( a_v \) or pairs of \( (a_u, a_v) \).
Example Gibbs state

Every corner of Singapore is filled with fun.

Cohn (Sheffield) Bayesian parsing and translation October 2009 49 / 58
Every corner of Singapore is filled with fun.
Expand operator

Samples from all legal values for $a_v$, constrained by ancestor, descendant and sibling alignments.
Samples whether to swap values of $a_u$ and $a_v$ where $u$ and $v$ have no aligned descendants and share the same aligned ancestor.
Experiments

Corpus

Evaluated the model for Chinese $\rightarrow$ English
- trained on a subset of the NIST data set ($\approx 300k$ sentence pairs)
- ‘development’ set MT02 and test set MT03, length $\leq 20$
- segmented Chinese and parsed the English

Models

- **Baseline**: in-house implementation of Galley et al. (2004)
- using Giza++ for word alignment and grow-diag-final-and heuristic
- **Our model**: sampled for 300 iters; using the last sample
- initialised with baseline system’s derivation
- hyperparameters set by hand (not fit to the data)
Results: Learned Grammars

- **Number of Rules**
  - GKHM
  - Gibbs

- **Maximum Tree Depth**

- **Source Terminals**

Cohn (Sheffield)  Bayesian parsing and translation  October 2009  54 / 58
### Results: Grammar Rules

#### GHKM Grammar

<table>
<thead>
<tr>
<th>Rule</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>((NP \ JJ_1 \ NNS_2), 1 \ 2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>((NP \ JJ_1 \ NN_2), 1 \ 2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>((NP \ DT_1 \ JJ_2 \ NN_3), 1 \ 2 \ 3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>((NP \ PRP$\ _1 \ NN_2), 1 \ 2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>((NP \ NP_1 \ PP_2), 2 \ 1)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Gibbs Grammar

<table>
<thead>
<tr>
<th>Rule</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>((NP \ (DT \ the) \ NN_1), 1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>((NP \ (NP \ (DT \ the) \ NN_1) \ (PP \ (IN \ of) \ NP_2)), 2 \ 1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>((NP \ (DT \ the) \ NN_1), 1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>((NP \ (NP \ (DT \ the) \ JJ_1 \ NN_2) \ (PP \ (IN \ of) \ NP_3)), 3 \ 1 \ 2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>((PP \ (IN \ of) \ NP_1), 1)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Top five rules with the possessive particle and $\geq 1$ variable.
Results: Translation

**Decoder**
- in-house CYK+ syntax-based decoder
- features based on Marcu et al., (2006)
- not extensively tuned; hence length limit on testing data
- weights trained using MERT to maximise development BLEU (Och, 2003)

**Test BLEU (MT03 ≤ 20)**

<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU score</th>
</tr>
</thead>
<tbody>
<tr>
<td>GHKM</td>
<td>26.0</td>
</tr>
<tr>
<td>Our model</td>
<td>26.6</td>
</tr>
</tbody>
</table>
Outline

1. Parsing
   - Tree substitution grammar
   - Bayesian Primer
   - The Model
   - Experiments
   - Summary

2. Machine Translation
   - Motivation
   - Model
   - Experiments
   - Summary

3. Conclusions
Summary

Presented generative model of tree-to-string grammar induction
- jointly induces a weighted STSG and alignment in a single model
- addresses disconnect between word alignment models and the grammar model
- induces more parsimonious grammar than heuristic approach
- encouraging translation performance

Future work:
- improve prior over rules to incorporate, e.g., rule shape
- develop better inference techniques for faster mixing and more accurate decoding
- induce other types of synchronous model e.g., FST, SCFG, STSG, STAG
Thank you!

Questions?

References

- Inducing Compact but Accurate Tree Substitution Grammars, Trevor Sharon Goldwater and Phil Blunsom. *NAACL 2009*, Boulder, CO, USA.


- Bayesian Synchronous Grammar Induction, Phil Blunsom, Trevor Cohn and Miles Osborne. *NIPS 2008*, Vancouver, Canada.