Representation and Learning of Grammars Beyond the Language Domain

Kewei Tu (ShanghaiTech University)

http://sist.shanghaitech.edu.cn/faculty/tukw/
Outline

- Extending grammars beyond the language domain
- Grammar-based probabilistic modeling
- Unsupervised learning of grammars
Outline

- Extending grammars beyond the language domain
  - And-Or Grammars [Zhu & Mumford, 2006], [Tu, IJCAI 2016]
- Grammar-based probabilistic modeling
- Unsupervised learning of grammars
### Grammars of language

<table>
<thead>
<tr>
<th>Grammar</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \rightarrow NP \ VP$</td>
<td>.80</td>
</tr>
<tr>
<td>$S \rightarrow Aux \ NP \ VP$</td>
<td>.15</td>
</tr>
<tr>
<td>$S \rightarrow VP$</td>
<td>.05</td>
</tr>
<tr>
<td>$NP \rightarrow Pronoun$</td>
<td>.35</td>
</tr>
<tr>
<td>$NP \rightarrow Proper-Noun$</td>
<td>.30</td>
</tr>
<tr>
<td>$NP \rightarrow Det \ Nominal$</td>
<td>.20</td>
</tr>
<tr>
<td>$NP \rightarrow Nominal$</td>
<td>.15</td>
</tr>
<tr>
<td>Nominal $\rightarrow$ Noun</td>
<td>.75</td>
</tr>
<tr>
<td>Nominal $\rightarrow$ Nominal Noun</td>
<td>.20</td>
</tr>
<tr>
<td>Nominal $\rightarrow$ Nominal PP</td>
<td>.05</td>
</tr>
<tr>
<td>$VP \rightarrow$ Verb</td>
<td>.35</td>
</tr>
<tr>
<td>$VP \rightarrow$ Verb NP</td>
<td>.20</td>
</tr>
<tr>
<td>$VP \rightarrow$ Verb NP PP</td>
<td>.10</td>
</tr>
<tr>
<td>$VP \rightarrow$ Verb PP</td>
<td>.15</td>
</tr>
<tr>
<td>$VP \rightarrow$ Verb NP NP</td>
<td>.05</td>
</tr>
<tr>
<td>$VP \rightarrow$ VP PP</td>
<td>.15</td>
</tr>
<tr>
<td>$PP \rightarrow$ Preposition NP</td>
<td>1.0</td>
</tr>
</tbody>
</table>
And-Or Normal Form of Stochastic CFG

- Two types of non-terminals
  - And-nodes, Or-nodes

- Two types of production rules
  - And-rule (composition): $A \rightarrow N_1 N_2 \cdots$
  - Or-rule (alternative configurations): $O \rightarrow N_1|N_2|t_1|t_2|\cdots$

**SCFG**

- $S \rightarrow a \ (0.4) \ | \ AB \ (0.6)$
- $A \rightarrow a \ (1.0)$
- $B \rightarrow b_1 \ (0.2) \ | \ b_2 \ (0.5) \ | \ b_3 \ (0.3)$

**The AND-OR Form**

- $\text{OR}_S \rightarrow a \ (0.4) \ | \ \text{AND}_{AB} \ (0.6)$
- $\text{AND}_{AB} \rightarrow \text{OR}_A \text{OR}_B$
- $\text{OR}_A \rightarrow a \ (1.0)$
- $\text{OR}_B \rightarrow b_1 \ (0.2) \ | \ b_2 \ (0.5) \ | \ b_3 \ (0.3)$

[Tu & Honavar, 2008]
And-Or Normal Form of Stochastic CFG

And-node  Or-node

Phrase

NP  Verb  NP

start

“Concatenating” relation

Word/Letter
And-Or Normal Form of Stochastic CFG

```
They start the book today.
```

“They start the book today.”
Stochastic Context-free And-Or Grammar (AOG)

And-node  Or-node

High-level pattern

Relation

Atomic pattern
AOG of Images

Images of Animal face

Dog face

Eyes

Ears

Nose

Spatial relation (e.g., relative position)

Visual word (e.g., oriented line)
AOG of Images

Images of Animal face

And-node  Or-node

Dog face

Eyes  Ears  Nose

……

……

……

……
Applications – indoor scene parsing

(a) Grammar

(b) Parse Tree

[Zhao & Zhu, NIPS 2011]
AOG of Events

- Event or sub-event
  - Open door
  - Walk
  - Close door

- Temporal relation

- Atomic action

- Office activity
Applications – video event parsing

(a) Grammar

(b) Parse Tree

[Pei, Jia & Zhu, ICCV 2011]
With some additional constraints, we get a decomposable sum-product network [Poon & Domingos, 2011]
Formal Definition

- A stochastic context-free And-Or grammar is a 5-tuple
  - A set $\Sigma$ of terminals (atomic patterns)
  - A set $N$ of nonterminals (high-level patterns)
    - Two disjoint subsets: And-nodes, Or-nodes
  - A start symbol $S \in N$ (a complete pattern)
  - A function $\theta$ that maps an instance of a terminal or nonterminal $x$ to a parameter $\theta_x$
  - A set $R$ of production rules
    - Or-rule $\langle r, p \rangle$
      - $r$: $O \rightarrow x$
      - $p$ is the probability of $O$ producing $x$
    - And-rule $\langle r, t, f \rangle$
      - $r$: $A \rightarrow \{x_1, x_2, \ldots, x_n\}$ for some $n \geq 2$
      - $t(\theta_{x_1}, \theta_{x_2}, \ldots, \theta_{x_n})$ is a parameter relation between child nodes
      - $\theta_A = f(\theta_{x_1}, \theta_{x_2}, \ldots, \theta_{x_n})$ is a parameter function
Special cases of AOG

**Natural Language Processing**
- Stochastic Context-free Grammars
- Hidden Markov Models
- Linear Context-free Rewriting System
- Constraint-based Grammar Formalisms
- etc.

**Computer Vision**
- Pictorial Structures Models
- Deformable Part Models
- Flexible Mixture-of-Parts Models
- etc.

**Machine Learning**
- Sum-Product Networks
- Naïve Bayes
- Biclustering
- Thin Junction Trees
- Mixtures of Trees
- Latent Tree Models
- etc.
Logic Interpretation

- Stochastic context-free AOG can be seen as a subset of first-order probabilistic logic
  - grammar rules $\rightarrow$ material implications
  - terminals, nonterminals $\rightarrow$ unary relations
  - A possible-world semantics

- Close connection with the field of statistical relational learning
  - Resembles tractable Markov logic [Domingos & Webb, 2012]
  - A special case of stochastic logic programs [Muggleton, 1996]
Outline

- Extending grammars beyond the language domain
- Grammar-based probabilistic modeling
  - Sum-product networks
  - Latent Dependency Forest Models
- Unsupervised learning of grammars
Sum-Product Network (SPN) [Poon & Domingos, UAI 2011]

- A new type of probabilistic models
- Linear-time marginal inference
- Capable of modeling context-specific independence

Additional structural restrictions:
- Completeness at sum nodes
- Decomposability at product nodes
Reduction to AOG

[Tu, IJCAI 2016]

- Reduction
  - Sum $\leftrightarrow$ Or-node
  - Product $\leftrightarrow$ And-node
  - Indicator $\leftrightarrow$ Terminal
  - No relation in And-rules

- Probability computation in SPN $\leftrightarrow$ Marginal computation in AOG (the inside algorithm)
Benefit of the Reduction

A lot of methods/ideas for inference and learning of grammars in NLP

These methods/ideas may be borrowed for inference and learning of SPNs
Example: Approximate MAP inference of SPNs based on k-best parsing [Mei, Jiang & Tu, AAAI 2018]

- Motivation: MAP solution often corresponds to parses with large value
  - Perform an extension of k-best parsing on the SPN
  - For each parse, compute the (conditional) probability of the assignment
  - Return the assignment with the highest probability
Grammars $\Leftrightarrow$ Probabilistic Models

Context-free Grammar

Dependency Grammar (Non-projective)

Latent Dependency Forest Model
Latent Dependency Forest Model (LDFM)
[Chu, Jiang & Tu, AAAI 2017] [Jiang, Zhou & Tu, under review]

**Dependency rules:**

- Root $\rightarrow$ $X_1=T$ [0.2]
- Root $\rightarrow$ $X_1=F$ [0.1]
- Root $\rightarrow$ $X_2=T$ [0.4]
- Root $\rightarrow$ $X_2=F$ [0.3]
- $X_1=T \rightarrow X_2=T$ [0.6]
- $X_1=T \rightarrow X_2=F$ [0.4]
- $X_1=F \rightarrow X_2=T$ [0.1]
- $X_1=F \rightarrow X_2=F$ [0.9]

**Given $X_1=T$, $X_2=F$, we have three parses:**

- Root
  - $X_1=T$ $X_2=F$, $P(X_1 = T, X_2 = F) \propto w_1 + w_2 + w_3 = 0.29$

**Features**

- Capable of modeling some context-specific independence
- Tractable computation of unnormalized probability
  - (Projective LDFM) Tractable joint and marginal inference
- Learning is easy: no structure search
Outline

- Extending grammars beyond the language domain
- Grammar-based probabilistic modeling
- Unsupervised learning of grammars
Learning a grammar

Supervised Methods

- Rely on treebanks (data samples annotated with parse trees)

Nonexistent for most types of data!
Learning a grammar

- **Supervised Methods**
  - Rely on treebanks (data samples annotated with parse trees)

- **Unsupervised Methods**
  - Do not require annotated data
Unsupervised grammar learning

- Two tasks
  - Structure search
    - Try to find an optimal set of grammar rules
  - Parameter learning
    - Given a set of grammar rules, try to learn their probabilities

Extremely difficult on real data

Difficult but doable, a lot of work in NLP over the past decade
Unsupervised parameter learning

- Expectation-maximization
  - Given grammar rules, initialize their probabilities
  - E-step
    - Parse the training samples using the current grammar
  - M-step
    - Update the grammar rule probabilities to maximize expected likelihood of the parses
  - Repeat until convergence

- Next
  - Our work in unsupervised parameter learning of natural language grammars
Unambiguity regularization [Tu & Honavar, EMNLP 2012]

- The ambiguity of natural language is remarkably low!
- Idea: incorporate unambiguity bias into learning
  - Measure ambiguity by entropy of the parse given the sentence and the grammar
  - Add it into the objective function using posterior regularization [Ganchev et al. (2010)]

\[
J(\theta) = \log p(\theta|X) - \min_q \left( \text{KL}(q(Z)||p_\theta(Z|X)) + \sigma \sum_i H(z_i) \right)
\]

- Log posterior of the grammar
- Entropy of a proxy distribution \( q \) of the parses given the sentences & grammar
- \( \sigma \) controls regularization strength
Unambiguity regularization [Tu & Honavar, EMNLP 2012]

- Coordinate Ascent
- Algorithm behavior depends on regularization strength $\sigma$
  - $\sigma = 0$: equivalent to standard EM
  - $\sigma \geq 1$: equivalent to hard EM
  - $0 < \sigma < 1$: softmax-EM (E-step produces softmax of parse distribution)

Very easy to implement: Just exponentiate all the rule probabilities before the E-step
Experimental results

Dependency Accuracy on WSJ10 Testset (Training with WSJ10, no lexicalization)

Generative Approaches
Unsupervised Neural Dependency Parsing

[Jiang, Han & Tu, EMNLP 2016]

- Represent symbols with vectors
- Learn ANN to predict grammar rule probabilities

\[ P(\text{child} \mid \text{head}, \text{direction}, \text{valency}) \]

**Softmax Layer:**
\[ p = \text{Softmax}(W_c h) \]

**Hidden Layer:**
\[ h = \tanh(W_{dir}[v_h, v_{val}]) \]

**Continuous Representation:**
\[ [v_h, v_{val}] \]

**Inputs:**
- Head POS Tag
- Valency

**Output values**
Unsupervised Neural Dependency Parsing

[Jiang, Han & Tu, EMNLP 2016]

- Learning by expectation-maximization

- Dynamic Programming
  - Forward Evaluating
  - Count Normalizing

- Neural Network Training
Experimental results

Dependency Accuracy on WSJ10 Testset (Training with WSJ10, no lexicalization)

Generative Approaches
CRF-autoencoder [Cai, Jiang & Tu, EMNLP 2017]

- Use a discriminative parser in unsupervised grammar learning
  - Based on the CRF-autoencoder framework [Ammar, Dyer & Smith, 2014]

A discriminative parser (MSTParser)

Generating child word from head word

Diagram:
- Encoder
- Decoder
  - $y_1$ to $\hat{x}_1$ (These)
  - $y_2$ to $\hat{x}_2$ (stocks)
  - $y_3$ to $\hat{x}_3$ (eventually)
  - $y_4$ to $\hat{x}_4$ (reopened)
CRF-autoencoder [Cai, Jiang & Tu, EMNLP 2017]

- Learning objective
  - Reconstruction probability
- Optimization algorithm
  - Coordinate ascent: alternately optimize encoder and decoder
Experimental results

Dependency Accuracy on WSJ10 Testset
(Training with WSJ10, no lexicalization)

Generative Approaches

- DMV (2004)
- LN Families (2009)
- PR-S (2010)
- EVG (2009)
- TSG-DMV (2010)
- UR-A E-DMV (2012)
- N E-DMV good init (2016)

Discriminative Approaches

- Convex MST (2015)
- CRF-AE (2017)
Summary
Summary

- Extending grammars beyond the language domain
  - Stochastic And-Or grammars
- Grammar-based probabilistic modeling
  - Sum-product networks
  - Latent Dependency Forest Models
- Unsupervised learning of grammars
  - Unambiguity regularization
  - Unsupervised neural dependency parsing
  - CRF autoencoder
Thank you!

Q&A