Decoding and Inference with Syntactic Translation Models

April 8, 2014
CFGs

\[
\begin{align*}
S & \rightarrow \ NP \ VP \\
VP & \rightarrow \ NP \ V \\
V & \rightarrow \ \text{tabeta} \\
NP & \rightarrow \ \text{jon-ga} \\
NP & \rightarrow \ \text{ringo-o} \\
\end{align*}
\]

Output: jon-ga ringo-o tabeta
Synchronous CFGs

\[
S \rightarrow NP \ VP
\]
\[
VP \rightarrow NP \ V
\]
\[
V \rightarrow tabeta
\]
\[
NP \rightarrow jon-ga
\]
\[
NP \rightarrow ringo-o
\]
Synchronous CFGs

\[
\begin{align*}
S & \rightarrow NP \ VP : \ \boxed{1} \ \boxed{2} \\
VP & \rightarrow NP \ V : \ \boxed{2} \ \boxed{1}
\end{align*}
\]
(monotonic)

\[
\begin{align*}
V & \rightarrow \text{tabeta} : \ \text{ate} \\
NP & \rightarrow \text{jon-ga} : \ \text{John} \\
NP & \rightarrow \text{ringo-o} : \ \text{an apple}
\end{align*}
\]
(inverted)
Synchronous CFGs

\[
\begin{align*}
S & \rightarrow \text{NP VP} : 1 \ 2 \quad \text{(monotonic)} \\
\text{VP} & \rightarrow \text{NP V} : 2 \ 1 \quad \text{(inverted)} \\
\text{V} & \rightarrow \text{tabeta} : \text{ate} \\
\text{NP} & \rightarrow \text{jon-ga} : \text{John} \\
\text{NP} & \rightarrow \text{ringo-o} : \text{an apple}
\end{align*}
\]
Synchronous generation

Output: (jon-ga ringo-o tabeta : John ate an apple)
Translation as parsing

Parse source

NP  VP
jon-ga  ringo-o  tabeta

Project to target

NP  VP
NP  V  NP
John  ate  an apple
A closer look at parsing

- Parsing is usually done with dynamic programming
- Share common computations and structure
- Represent exponential number of alternatives in polynomial space
- With SCFGs there are two kinds of ambiguity
  - source parse ambiguity
  - translation ambiguity
- parse forests can represent both
A closer look at parsing

• Any monolingual parser can be used (most often: CKY / “dotted” CKY variants)

• Parsing complexity is $O(|n^3|)$
  • cubic in the length of the sentence ($n^3$)
  • cubic in the number of non-terminals ($|G|^3$)
    • adding nonterminal types increases parsing complexity substantially!

• With few NTs, exhaustive parsing is tractable
“If $A$ and $B$ are true with weights $u$ and $v$, and phi is also true, then $C$ is true with weight $w$.\)
Example: CKY

Inputs:

\[ f = \langle f_1, f_2, \ldots, f_e \rangle \]

\[ G \quad \text{Context-free grammar in Chomsky normal form.} \]

Item form:

\[ [X, i, j] \quad \text{A subtree rooted with NT type } X \text{ spanning } i \text{ to } j \text{ has been recognized.} \]
Example: CKY

Goal:

\[ [S, 0, \ell] \]

Axioms:

\[
[X, i - 1, i] : w \quad (X \xrightarrow{w} f_i) \in G
\]

Inference rules:

\[
[X, i, k] : u \\ [Y, k, j] : v \\
\overline{[Z, i, j] : u \times v \times w} \\
(Z \xrightarrow{w} XY) \in G
\]
I saw her duck.
I saw her duck
I saw her duck.
S → PRP VP
VP → V NP
VP → V SBAR
SBAR → PRP V
NP → PRP NN
V → saw
NN → duck
V → duck
PRP → I
PRP → her

I saw her duck
I saw her duck.
S → PRP VP
VP → V NP
VP → V SBAR
SBAR → PRP V
NN → PRP NN
V → saw
NN → duck
V → duck
PRP → I
PRP → her

I
saw
her
duck
I saw her duck.
I saw her duck.
I saw her duck.
I saw her duck.
I saw her duck.
What is this object?

I saw her duck
Semantics of hypergraphs

- Generalization of directed graphs
- Special node designated the “goal”
- Every edge has a single head and 0 or more tails (the arity of the edge is the number of tails)
- Node labels correspond to LHS’s of CFG rules
- A derivation is the generalization of the graph concept of path to hypergraphs
- Weights multiply along edges in the derivation, and add at nodes (cf. semiring parsing)
Edge labels

- Edge labels may be a mix of terminals and substitution sites (non-terminals)
- In translation hypergraphs, edges are labeled in both the source and target languages
- The number of substitution sites must be equal to the arity of the edge and must be the same in both languages
- The two languages may have different orders of the substitution sites
- There is no restriction on the number of terminal symbols
Edge labels

\{ (la lectura \ de\ ayer : yesterday 's reading),
 (la lectura \ de\ ayer : reading from yesterday) \}
A Lingua Franca for MT

- Translation hypergraphs are a *lingua franca* for translation search spaces
- Note that FST lattices are a special case
- Decoding problem: how do I build a translation hypergraph?
- For SCFG-translation: just parse
Tree-to-string Translation

- How do we generate a hypergraph for a tree-to-string translation model?
  - Simple linear-time (given a fixed translation model) top-down matching algorithm
  - Recursively cover “uncovered” sites in tree
  - Each node in the input tree becomes a node in the translation forest
- For details, Huang et al. (AMTA, 2006) and Huang et al. (EMNLP, 2010)
\[
\begin{align*}
S(x_1:NP \ x_2:VP) & \rightarrow x_1 \ x_2 \\
VP(x_1:NP \ x_2:V) & \rightarrow x_2 \ x_1 \\
\text{tabeta} & \rightarrow \text{ate} \\
\text{ringo-o} & \rightarrow \text{an apple} \\
\text{jon-ga} & \rightarrow \text{John}
\end{align*}
\]
\[ S(x_1:NP \ x_2:VP) \rightarrow x_1 \ x_2 \]

\[ VP(x_1:NP \ x_2:V) \rightarrow x_2 \ x_1 \]

\[ \text{tabeta} \rightarrow \text{ate} \]

\[ \text{ringo-o} \rightarrow \text{an apple} \]

\[ \text{jon-ga} \rightarrow \text{John} \]
S(\(x_1:NP\) \(x_2:VP\)) \(\rightarrow\) \(x_1\) \(x_2\)

VP(\(x_1:NP\) \(x_2:V\)) \(\rightarrow\) \(x_2\) \(x_1\)

\textit{tabeta} \(\rightarrow\) \textit{ate}

\textit{ringo-o} \(\rightarrow\) \textit{an apple}

\textit{jon-ga} \(\rightarrow\) \textit{John}
S(x₁:NP x₂:VP) → x₁ x₂
VP(x₁:NP x₂:V) → x₂ x₁

tabeta → ate
ringo-o → an apple
jon-ga → John
\[
S(x_1:NP \ x_2:VP) \rightarrow x_1 \ x_2 \\
VP(x_1:NP \ x_2:V) \rightarrow x_2 \ x_1
\]
\[
tabeta \rightarrow \text{ate}
\]
\[
ringo-o \rightarrow \text{an apple}
\]
\[
jon-ga \rightarrow \text{John}
\]
S(\(x_1:NP\) \(x_2:VP\)) \(\rightarrow\) \(x_1\) \(x_2\)

VP(\(x_1:NP\) \(x_2:V\)) \(\rightarrow\) \(x_2\) \(x_1\)

\textit{tabeta} \(\rightarrow\) \textit{ate}

\textit{ringo-o} \(\rightarrow\) \textit{an apple}

\textit{jon-ga} \(\rightarrow\) \textit{John}
\[
S(x_1:NP \ x_2:VP) \rightarrow x_1 \ x_2 \\
VP(x_1:NP \ x_2:V) \rightarrow x_2 \ x_1
\]

```
tabela \rightarrow ate
ringo-o \rightarrow an \ apple
jon-ga \rightarrow John
```
Working With Hypergraphs
Derivations

d_1 = e_4 e_1 e_3 \quad y(d_1) = yesterday’s reading

d_2 = e_5 e_1 e_3 \quad y(d_2) = reading from yesterday

d_3 = e_4 e_2 e_3 \quad y(d_3) = yesterday’s lecture

d_4 = e_5 e_2 e_3 \quad y(d_4) = lecture from yesterday
Derivations

\[ w[d_1] = 0.4 \cdot 0.8 \cdot 0.5 = 0.16 \]

\[ w[d_2] = 0.6 \cdot 0.8 \cdot 0.5 = 0.24 \]

\[ w[d_3] = 0.4 \cdot 0.2 \cdot 0.5 = 0.04 \]

\[ w[d_4] = 0.6 \cdot 0.2 \cdot 0.5 = 0.06 \]
Best Path

- $e_1$ with weight 0.8 leads to reading.
- $e_2$ with weight 0.2 leads to lecture.
- $e_3$ with weight 0.5 leads to yesterday.
- $e_4$ with weight 0.4 connects to 2's 1.
- $e_5$ with weight 0.6 connects from 1 to 2.

Nodes connect with arrows indicating the path.
Best Path

0.8 e_1 \rightarrow \text{reading} \rightarrow \text{lecture} \rightarrow 2 \text{'s 1}

0.2 e_2 \rightarrow \text{lecture} \rightarrow 0.4 e_4 \rightarrow \text{from 2}

0.5 e_3 \rightarrow \text{yesterday} \rightarrow 0.6 e_5 \rightarrow 1
Best Path

0.8 \( e_1 \) reading

0.2 \( e_2 \) lecture

0.5 \( e_3 \) yesterday

\( 2 \)'s 1

0.4 \( e_4 \)

0.6 \( e_5 \) from 2
Best Path

0.8 \( e_1 \) reading

0.2 \( e_2 \) lecture

0.5 \( e_3 \) yesterday

0.2

0.4 \( e_4 \)

0.6 \( e_5 \)

1 from 2

2's 1
Best Path

\[ 0.8 e_1 \rightarrow \text{reading} \rightarrow 0.8 \]
\[ 0.2 e_2 \rightarrow \text{lecture} \rightarrow 0.8 \]
\[ 0.5 e_3 \rightarrow \text{yesterday} \rightarrow 0.8 \]

\[ 0.4 e_4 \rightarrow 2 \text{'s} 1 \]
\[ 0.6 e_5 \rightarrow 1 \text{ from} 2 \]
Best Path

0.8 $e_1$ reading
0.2 $e_2$ lecture
0.5 $e_3$ yesterday

0.8

0.4 $e_4$

0.6 $e_5$

2's 1
1 from 2
Best Path

- Reading: $0.8 e_1$
- Lecture: $0.2 e_2$
- Yesterday: $0.5 e_3$

0.8

- $0.4 e_4$
- $0.6 e_5$

2's 1
1 from 2
Best Path

0.8 $e_1$ reading

0.2 $e_2$ lecture

0.5 $e_3$ yesterday

0.8

0.4 $e_4$

0.6 $e_5$

2's 1

1 from 2
Best Path

0.8 \( e_1 \) reading

0.2 \( e_2 \) lecture

0.5 \( e_3 \) yesterday

0.8

0.5

0.4 \( e_4 \)

0.6 \( e_5 \)

2's 1

1 from 2
Best Path

0.8 \( e_1 \) reading
0.2 \( e_2 \) lecture
0.5 \( e_3 \) yesterday

0.8

0.5

0.4 \( e_4 \)
0.6 \( e_5 \)

2's 1
1 from 2

---

reading
lecture
yesterday

---
Best Path

0.8 $e_1$ reading

0.2 $e_2$ lecture

0.5 $e_3$ yesterday

0.8

0.5

0.4 $e_4$

0.6 $e_5$

1 from 2

2's 1
Best Path

$e_1 \rightarrow 0.8 \rightarrow$ reading

$e_2 \rightarrow 0.2 \rightarrow$ lecture

$e_3 \rightarrow 0.5 \rightarrow$ yesterday

$0.8 \leftarrow 2 \text{'s 1}$

$0.4 e_4 \leftarrow 0.6 e_5 \leftarrow 1 \text{ from 2}$
Best Path

\[ 0.8 \times 0.5 \times 0.4 = 0.16 \]
Best Path

0.8 $e_1$ reading
0.2 $e_2$ lecture
0.5 $e_3$ yesterday

0.8

0.5

0.4 $e_4$

0.6 $e_5$

0.16

1 from 2

1 's 2
Best Path

0.8 \times 0.5 \times 0.6 = 0.24
Best Path

\[
0.8 \times 0.5 \times 0.6 = 0.24
\]
Best Path

![Diagram with nodes and edges labeled with probabilities and words: reading, lecture, yesterday, 2's 1, 0.8, 0.5, 0.8, 0.4, 0.6, 0.24.](image)
Best yield: reading from yesterday
Best path: 0.24
Best Path

Best yield: reading from yesterday

Best path: 0.24

\[ d_1 = e_4 e_1 e_3 \quad w[d_1] = 0.4 \cdot 0.8 \cdot 0.5 = 0.16 \]
\[ d_2 = e_5 e_1 e_3 \quad w[d_2] = 0.6 \cdot 0.8 \cdot 0.5 = 0.24 \]
\[ d_3 = e_4 e_2 e_3 \quad w[d_3] = 0.4 \cdot 0.2 \cdot 0.5 = 0.04 \]
\[ d_4 = e_5 e_2 e_3 \quad w[d_3] = 0.6 \cdot 0.2 \cdot 0.5 = 0.06 \]
Other Algorithms

• Given a weighted hypergraph

• In the Viterbi (Inside) algorithm, there are two operations

  • Multiplication (extend path)
  • Maximization (chose between paths)

• Semirings generalize these to compute other quantities
# Semirings

<table>
<thead>
<tr>
<th>semiring</th>
<th>( \mathbb{K} )</th>
<th>( \oplus )</th>
<th>( \otimes )</th>
<th>( 0 )</th>
<th>( 1 )</th>
<th>notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>{0,1}</td>
<td>( \lor )</td>
<td>( \land )</td>
<td>0</td>
<td>1</td>
<td>idempotent</td>
</tr>
<tr>
<td>count</td>
<td>( \mathbb{N}_0 \cup {\infty} )</td>
<td>+</td>
<td>( \times )</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>probability</td>
<td>( \mathbb{R}_+ \cup {\infty} )</td>
<td>+</td>
<td>( \times )</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>tropical</td>
<td>( \mathbb{R} \cup {-\infty, \infty} )</td>
<td><strong>max</strong></td>
<td>+</td>
<td>(-\infty)</td>
<td>0</td>
<td>idempotent</td>
</tr>
<tr>
<td>log</td>
<td>( \mathbb{R} \cup {-\infty, \infty} )</td>
<td>( \oplus_{\text{log}} )</td>
<td>+</td>
<td>(-\infty)</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
Inside Algorithm

\[ \alpha(q_{goal}) = \bigoplus_{d \in \mathcal{G}} \bigotimes_{e \in d} w(e) \]

1: function \text{INSIDE}(\mathcal{G}, K) \quad \triangleright \mathcal{G} \text{ is an acyclic hypergraph and } K \text{ is a semiring}
2: \text{for } q \text{ in topological order in } \mathcal{G} \text{ do}
3: \quad \text{if } B(q) = \emptyset \text{ then}
4: \quad \quad \alpha(q) \leftarrow 1 \quad \triangleright \text{ assume states with no in-edges are axioms}
5: \quad \text{else}
6: \quad \quad \alpha(q) \leftarrow 0
7: \quad \text{for all } e \in B(q) \text{ do}
8: \quad \quad k \leftarrow w(e)
9: \quad \text{for all } r \in t(e) \text{ do}
10: \quad \quad \quad k \leftarrow k \otimes \alpha(r) \quad \triangleright \text{ all in-coming edges to node } q
11: \quad \quad \alpha(q) \leftarrow \alpha(q) \oplus k
12: \quad \text{return } \alpha
13: \text{for all } r \in t(e) \text{ do}
14: \quad \quad k \leftarrow k \otimes \alpha(r) \quad \triangleright \text{ all tail (previous) nodes of edge } e
15: \quad \alpha(q) \leftarrow \alpha(q) \oplus k
16: \text{return } \alpha
Count Derivations
Count Derivations

\[
e_1 \rightarrow \text{reading}
\]
\[
e_2 \rightarrow \text{lecture}
\]
\[
e_3 \rightarrow \text{yesterday}
\]
\[
2 \text{'s} 1
\]
\[
e_4
\]
\[
e_5
\]
\[
1 \text{ from } 2
\]
Count Derivations

\[ e_1 \quad \text{reading} \quad 2 \]

\[ e_2 \quad \text{lecture} \quad 2 \]

\[ e_3 \quad \text{yesterday} \quad 1 \]

\[ 2 \text{'s} 1 \]

\[ e_4 \quad e_5 \quad 1 \text{ from } 2 \]
Count Derivations

\[ 2 \times 1 \times 1 = 2 \]
Count Derivations

\[2 \times 1 \times 1 = 2\]
Count Derivations

e_1 \quad \text{reading}
e_2 \quad \text{lecture}
e_3 \quad \text{yesterday}

2 \quad \text{2's 1}

4 \quad \text{1 from 2}
Inside-Outside

1: function OUTSIDE($G, K, \alpha$)  
   2: for all $q \in \mathcal{G}$ do  
   3: \hspace{1em} $\beta(q) \leftarrow \vec{0}$  
   4: \hspace{1em} $\beta(q_{goal}) = \vec{1}$  
   5: \hspace{1em} for $q$ in reverse topological order in $G$ do  
   6: \hspace{2em} for all $e \in B(q)$ do  
   7: \hspace{3em} for all $r \in t(e)$ do  
   8: \hspace{4em} $k \leftarrow w(e) \otimes \beta(q)$  
   9: \hspace{4em} for all $s \in t(e)$ do  
   10: \hspace{5em} if $r \neq s$ then  
   11: \hspace{6em} $k \leftarrow k \otimes \alpha(s)$  
   12: \hspace{5em} $\beta(r) \leftarrow \beta(r) \oplus k$  
   13: \hspace{1em} return $\beta$

1: function INSIDEOUTSIDE($G, K$)  
   2: $\alpha \leftarrow $ INSIDE($G, K$)  
   3: $\beta \leftarrow $ OUTSIDE($G, K, \alpha$)  
   4: for edge $e$ in $G$ do  
   5: \hspace{1em} $\gamma(e) \leftarrow w(e) \otimes \beta(n(e))$  
   6: \hspace{1em} for all $q \in t(e)$ do  
   7: \hspace{2em} $\gamma(e) \leftarrow \gamma(e) \otimes \alpha(q)$  
   8: \hspace{1em} return $\gamma$

$\alpha$ is the result of INSIDE($G, K$)  
all in-coming edges to node $q$  
all tail (previous) nodes of edge $e$  
all tail (previous) nodes of edge $e$, again  
incorporate inside score  
compute edge marginals  
edge weight and outside score of edge’s head node  
inside score of tail nodes  
$\gamma(e)$ is the edge marginal of $e$
Inside-Outside

- Compute lots of interesting quantities
  - The score of the best path through each edge
  - The total number of derivations that contain an edge
  - The total score of all derivations going through an edge
Inference algorithms

- **Viterbi**  $O(|E| + |V|)$
  - Find the maximum weighted derivation
  - Requires a partial ordering of weights

- **Inside - outside**  $O(|E| + |V|)$
  - Compute the marginal (sum) weight of all derivations passing through each edge/node

- **k-best derivations**  $O(|E| + |D_{max}|k \log k)$
  - Enumerate the $k$-best derivations in the hypergraph
  - See IWPT paper by Huang and Chiang (2005)
Things to keep in mind

Bound on the number of edges (SCFG):

\[ |E| \in O(n^3|G|^3) \]

Bound on the number of nodes:

\[ |V| \in O(n^2|G|) \]
Next time

What about the LM?