Lecture 12: Deep semantics dependencies & semantic roles
StatNLP, Winter 2008

Roger Levy

UCSD Linguistics
Interpreting language

- We’ve seen how to recover hierarchical structure from sentences via parsing.
- Why on earth would we want that kind of structure?
- Insight: that structure is a major cue to sentence meaning.
- We’ll look in the next few classes into how to recover aspects of those meanings.
Today we’ll look at two interrelated problems

1: **Semantic Roles**
   - Different verbs/nouns assign different semantic properties to the phrases that stand in structural relations with them
   - We want to figure out how those semantic properties are assigned, and recover them for input text

2: **Discontinuous/nonlocal Dependencies**
   - In some cases, a given role is assigned to a semantic unit that is discontinuous in “surface” phrase structure
   - Role assignment & interpretation can be improved by “reuniting” the fragments in these discontinuities
Semantic Role Labeling (SRL)

- Characterize clauses as relations with roles:

  \[ [\text{Judge She}] \ \text{blames} \ [\text{Evaluator the Government}] \ [\text{Reason for failing to do enough to help}] \].

  Holman would characterise this as \textit{blaming} \ [\text{Evaluator the poor}] .

  The letter quotes Black as saying that \[ [\text{Judge white and Navajo ranchers}] \ \text{misrepresent their livestock losses and blame} \ [\text{Reason everything}] \ [\text{Evaluator on coyotes}] \].

- We want to know more than which NP is the subject:
- Relations like \textit{subject} are syntactic; relations like \textit{agent} or \textit{message} are semantic
- Typical pipeline:
  - Parse, then label roles
  - Almost all errors locked in by parser
  - Really, SRL is (or should be) quite a lot easier than parsing
SRL: applications

- You don’t have to look very far to see where SRL can be useful
- Example: Information Extraction: who bought what?

- Home Depot sells **lumber**
- **Home Depot** was sold by its parent company
- Home Depot sold **its subsidiary company** for $10 million
- **Home Depot** sold for $100 billion
- **Home Depot’s lumber** sold well last year.
SRL Example

He heard the sound of liquid slurping in a metal container as Farrell approached him from behind.

Gildea & Jurafsky 2002
Roles: PropBank / FrameNet

- **FrameNet**: roles shared between verbs
- **PropBank**: each verb has its own roles
- **PropBank** more used, because it's layered over the treebank (and so has greater coverage, plus parses)
- **Note**: some linguistic theories postulate even fewer roles than FrameNet (e.g. 5-20 total: agent, patient, instrument, etc.)
PropBank Example

**fall.01**

- **sense:** move downward
- **roles:**
  - Arg1: thing falling
  - Arg2: extent, distance fallen
  - Arg3: start point
  - Arg4: end point

Sales fell to $251.2 million from $278.7 million.

- **arg1:** Sales
- **rel:** fell
- **arg4:** to $251.2 million
- **arg3:** from $278.7 million
PropBank Example

\textbf{rotate.02}  
sense: shift from one thing to another

roles:  
Arg0: causer of shift
Arg1: thing being changed
Arg2: old thing
Arg3: new thing

Many of Wednesday’s winners were losers yesterday as investors quickly took profits and rotated their buying to other issues, traders said. (wsj_1723)

arg0: investors
rel: rotated
arg1: their buying
arg3: to other issues

- http://www.cs.rochester.edu/~gildea/PropBank/Sort/
FrameNet example

Shared Arguments

S

NP-SBJ
JJ massive JJ internal NN debt

VP
VBZ has

S
VBN

forced

NP-SBJ-1
DT the NN government

VP

TO to

VP
VB borrow

ADVP-MNR RB massively

force

arg0
massive internal debt

arg1
the government

arg2
borrow

MNR
massively
Path Features

Path | Description
--- | ---
VB↑VP↓PP | PP argument/adjunct
VB↑VP↑S↓NP | subject
VB↑VP↑NP | object
VB↑VP↑VP↑S↓NP | subject (embedded VP)
VB↑VP↑ADVP | adverbial adjunct
NN↑NP↑NP↓PP | prepositional complement of noun
Results

• Features:
  • Path from target to filler
  • Filler’s syntactic type, headword, case
  • Target’s identity
  • Sentence voice, etc.
  • Lots of other second-order features

• Gold vs parsed source trees

  • SRL is fairly easy on gold trees
  • Harder on automatic parses

<table>
<thead>
<tr>
<th></th>
<th>Core</th>
<th>Argm</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>92.2</td>
<td>89.9</td>
</tr>
<tr>
<td>Acc.</td>
<td>80.7</td>
<td>71.8</td>
</tr>
<tr>
<td>F1</td>
<td>84.1</td>
<td>81.4</td>
</tr>
<tr>
<td>Acc.</td>
<td>66.5</td>
<td>55.6</td>
</tr>
</tbody>
</table>
A man arrived who I knew.
Who was believed to know the answer?
The story was about the children that the witch wanted to eat.
Cries of pain could be heard from below.
We discussed plans yesterday to redecorate the house.
These sentences have a common property.

They all involve a unified semantic proposition broken up into two pieces across the sentence.

The intervening material is part of an unrelated proposition.

A man arrived who I knew.

Who was believed to know the answer?

The story was about the children that the witch wanted to eat.

Cries of pain could be heard from below.

We discussed plans yesterday to redecorate the house.
What is the state of the art in robust computation of linguistic meaning?

- **Probabilistic context-free grammars** trained on syntactically annotated corpora (*Treebanks*) yield robust, high-quality syntactic parse trees.

Nodes of these parse trees are often reliable indicators of *phrases* corresponding to *semantic units* (Gildea & Palmer 2002).

```
a man arrived yesterday
```

```
Total: 0.7*0.35*0.15*0.3*0.03*0.02*0.4*0.07 = 1.85 \times 10^{-7}
```
Dependency trees

- Alternatively, syntactic parse trees can directly induce dependency trees
- Can be interpreted pseudo-propositionally; high utility for Question Answering (Pasca and Harabagiu 2001)
A man who I knew arrived.

I ate what you cooked.
Parse trees to dependency trees

A man who I knew arrived.

A man arrived who I knew.
Limits of context-free parse trees
Recovering correct dependency trees

- Context-free trees can’t transparently encode such non-local semantic dependency relations
- Treebanks typically do encode non-local dependencies in some form
- But nobody has used them, probably because:
  - it’s not so clear how to do probability models for non-local dependencies
  - Linguists think non-local dependencies are relatively rare, in English (unquantified, though!)
  - Nobody’s really tried to do much deep semantic interpretation of parse trees yet, anyway
Today’s agenda

- How to **quantify** the frequency and difficulty of non-local dependencies?
- Are non-local dependencies really a big problem for English? For other languages?
- How can we reliably and robustly **recover** non-local dependencies so that:
  - The assumptions and representation structure are as theory-neutral as possible?
  - The recovery algorithm is independent of future semantic analysis we might want to apply?
Quantifying non-local dependencies

• Kruijff (2002) compared English, German, Dutch, and Czech treebanks for frequency of nodes-with-holes. Result:
  
  English < {Dutch,German} < Czech

• This metric gives no similarity standard of comparison of two trees
• Also, this metric can be sensitive to the details of syntactic tree construction
• Typed dependency evaluation provides an alternative measure; initial results suggest good correlation
Previously proposed evaluation metric (Johnson 2002) requires correctly identified *antecedent* and correct *string position* of relocation.

This metric underdetermines dependency relations.
Proposed new metric

Given a sentence $S$ and a context-free parse tree $T$ for $S$, compare the dependency tree $D$ induced from $T$ with the correct dependency tree for $S$

Dependency trees can be scored by the edges connecting different words
Proposed new metric

- Dependencies induced from a CF tree will be counted as wrong when the true dependency is non-local.
Two types of non-local dependencies

- Treebanks give us gold-standard dependency trees:
  - Most dependencies can be read off the context-free tree structure of the annotation
  - Non-local dependencies can be obtained from special treebank-specific annotations

- Two types of non-local dependency:
  - *Dislocated* dependency
    - *A man arrived who I knew*
      - *who I knew* is related to *man*, NOT to *arrived*
  - *Shared* dependency
    - *I promised to eat it*
      - *I* is related to BOTH *promised* AND *eat*
Nonlocal dependency annotation in treebanks

- Penn treebanks annotate:
  - null complementizers (which can mediate relativization)
  - dislocated dependencies
  - sharing dependencies
Nonlocal dependencies in treebanks

- The NEGRA treebank (German) permits crossing dependencies during annotation, then algorithmically maps to a context-free representation.
- Null complementizers and shared dependencies are unannotated.
Nonlocal dependency, quantified

Context free dependency accuracy by syntactic category

Accuracy

- ADVP
- SBAR
- ADJP
- VP
- S
- NP
- Overall

CF deps
Nonlocal dependency, quantified

Dependency accuracy of state-of-the-art context-free parser, English

(parsing done with state-of-the-art Charniak 2000 parser)
Cross-linguistic comparison

Underlying dependency accuracy, German vs English

(parsing done with vanilla PCFG; sharing dependencies and relativizations excluded from non-local dependency)
Underlying dependency evaluation: conclusions

- Non-local dependency errors increase in prominence as parser quality improves.
- In German, non-local dependency is a much more serious problem than for English.
- Context-free assumption less accurate on gold-standard trees for categories involving combinations of phrases (SBAR, S, VP) than for lower-level phrasal categories (NP, ADVP, ADJP).
Recovery of non-local dependencies

- Context-free trees can directly recover only local (non-crossing) dependencies. To recover non-local dependencies, we have three options:
  - Treat the parsing task as initially context-free, then correct the parse tree post-hoc (Johnson 2002; present work)
  - Incorporate non-local dependency into the category structure of parse trees (Collins 1999; Dienes & Dubey 2003).
  - Incorporate non-local dependency into the edge structure of parse trees (Plaehn 2000; elsewhere in my thesis)
Linguistically motivated tree reshaping

1. Null complementizers (mediate relativization)
   A. Identify sites for null node insertion
   B. Find best daughter position and insert.

2. Dislocated dependencies
   A. Identify dislocated nodes
   B. Find original/“deep” mother node
   C. Find best daughter position in mother and insert

3. Shared dependencies
   A. Identify sites of nonlocal shared dependency
   B. Identify best daughter position and insert.
   C. Find controller for each control locus.
Sony was quick to see the many negative aspects yesterday.
A context-free parse

Sony was quick yesterday to point out the many negative aspects.

It sees
1a: null insertion sites

Sony was quick yesterday to point out the many negative aspects it sees.
1b: location of null insertions

```
(S
  (NP (NNP Sony) (VBD was) (ADJP yesterday (JJ quick)))
  (TO to)
  (VP (VB point) (RP out)
    (NP (DT the) (JJ many) (JJ negative) (NNS aspects))
  )
  (SBAR (WHNP (NP (PRP it)) (VBZ sees)))
)
```
2a: identify dislocations
2b: identify origin sites
2c: insert dislocations in origins
2c: insert dislocations in origins

```
(S
  (NP
    (NNP Sony) (VBD was)
    (ADJP (JJ quick))
    (TO to)
    (VB point)
    (RP out)
    (DT the) (JJ many) (JJ negative) (NNS aspects)
  )
  (SBAR
    (S
      (NP
        (PRP it)
      )
      (VBZ sees)
    )
    (WHNP 0)
  )
)
```
3a: identify sites of non-local shared dependency
3b: insert non-local shared dependency sites
3c: find controllers of shared dependencies
End.
Application of maximum entropy models

- Discriminative classification model with well-founded probabilistic interpretation, close relationship to log-linear models and logistic regression
- Node-by-node discriminative classification of the form

\[ P(c_j \mid \bar{x}) = \frac{e^{\lambda_j \bar{x}}}{\sum_i e^{\lambda_i \bar{x}}} \]

- Quadratic regularization and thresholding by feature token count to prevent overfitting
- In node relocation and controller identification, candidate nodes are ranked by binary (yes/no) classification scores and the highest-ranked node is selected
Larger feature space

- Words in strings have only one primitive measure of proximity: linear order.
- Tree nodes have several: precedence, sisterhood, domination.
- Results in much richer feature space to explore.
Feature types

- Syntactic categories, mothers, grandmothers (infinitive VP)
  - Head words (*wanted* vs *to* vs *eat*)
- Presence of daughters (NP under S)
- Syntactic path:
  \[<↑\text{SBAR}, ↓\text{S}, ↓\text{VP}, ↓\text{S}, ↓\text{VP}>\]
Evaluation on new metric: gold-standard input trees
Evaluation on new metric: parsed input trees

![Accuracy Chart]

Legend:
- Johnson 2002, parsed input trees
- Present work, parsed input trees
- Parsed CF output trees

Accuracy (F1)
English-language evaluation

- Deep dependency error reduction of 76% over baseline gold-standard context-free dependencies, and 42% over Johnson 2002 (95.5%/98.1%/91.9%)
- Performance particularly good on individual subtasks -- over 50% error reduction (95.2%) over Johnson 2002 on identification of non-local shared dependency sites
- Relative error reduction degraded across the board on parsed trees -- performance on parsed trees doesn’t match Johnson 2002 (could be due to overfitting on gold-standard dataset)
Cross-linguistic comparison: dislocated dependencies only

• Compare performance of classifier on NEGRA and similar-sized subset of WSJ (~350K words)
• NEGRA has no annotations of null complementizers or sharing dependencies, so evaluate only on dislocations and exclude relativization
• WSJ context-free parsing is far more accurate than NEGRA parsing, so to even the field, use unlexicalized, untuned PCFGs for parsing
Algorithm performance: cross-linguistic comparison

![Bar chart showing performance comparison between CF deps and Corrected deps for German and English gold and parsed data.](chart.png)
Major error types -- German

- Major ambiguity between local and non-local dependency for clause-final VP and S nodes

“The RMV will not begin to be formed until much later.”
Major error types -- German

- Scrambling -- clause-initial NPs don’t always belong in S

“The researcher Otto Schwabe documented the history of the excavation.”