Lecture 13a: Chunks

Announcements

Code Freeze Day!
- From here on out, don’t change your code
- Exceptions:
  - Bugfixes: don’t tell me why your code crashed, just fix it.
  - Checkpointed before-and-after:
    - If you tell me how the frozen version of your code works
    - Then you can also say how a (small) modification changes performance

Announcements (II)

Paper: Due one week from today
Four pages (no longer)
Three parts
- Introduction: describe your motivation and program
- Why did you design it the way you did
- How is it supposed to work
- Predicted limitations and risks
- Performance
- How well does it work on the problems given?
- Report relevant quantitative metrics for your design
- You may supplement with additional problems of your own design
- Conclusion
- Do your predictions match the results? If not, why not?

Announcements (III)

In-class presentation: one week from today
- 5 minutes maximum (timed)
- Powerpoint or pdf slides
- Email to me (draper@colostate.edu) the night before
- Same 3-part structure as the paper

Project #3 Preview

Project #3 is OPTIONAL
- Do it if unhappy with existing grades
- Accepted through Wednesday, May 9th
- I will try to be quick about Project #2 grades

Task: Build an NLP interface to your Blocks World Simulator
- Control it through a series of English language commands
- Can use existing parsers and other tools

Pipeline of NLP Tools

Scraping (not covered here)
Sentence splitting
Tokenization
(Stemming)
Part-of-speech tagging
Shallow parsing

Named entity recognition
Syntactic parsing
(Semantic Role Labeling)

---
POS as HMM

What are the states?
- POS tags
  - CC/CD/. . ./NN/. . ./WB
  - Because the goal is to find the most likely sequence of tags

What are the observations?
- Words

What is in the Transition Table?
- Maps POS tags to POS tags
- Probabilities
  - How likely is a singular noun (NN) to be followed by an adjective (JJ)?
- Trained using a labeled corpus

What about the observation matrix?
- Maps words (observations) to states (POS tags)
- Entries: P(POS tag | word)
- Trained on labeled corpus

POS tagging with Hidden Markov Models

\[
P(t_1\ldots t_n | w_1\ldots w_n) = \frac{P(w_1\ldots w_n | t_1\ldots t_n)P(t_1\ldots t_n)}{P(w_1\ldots w_n)} \\
\propto P(w_1\ldots w_n | t_1\ldots t_n)P(t_1\ldots t_n) \\
\approx \prod_{i=1}^{n} P(w_i | t_i)P(t_i | t_{i-1})
\]

POS tagging algorithms

Performance on the Wall Street Journal corpus

<table>
<thead>
<tr>
<th>Method</th>
<th>Training Cost</th>
<th>Speed</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dependency Net (2003)</td>
<td>Low</td>
<td>Low</td>
<td>97.2</td>
</tr>
<tr>
<td>Conditional Random Fields</td>
<td>High</td>
<td>High</td>
<td>97.1</td>
</tr>
<tr>
<td>Support vector machines (2003)</td>
<td>High</td>
<td>High</td>
<td>97.1</td>
</tr>
<tr>
<td>Bidirectional MEMM (2005)</td>
<td>Low</td>
<td></td>
<td>96.6</td>
</tr>
<tr>
<td>Brill’s tagger (1995)</td>
<td>Very low</td>
<td>High</td>
<td>96.7</td>
</tr>
<tr>
<td>HMM (2000)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Chunking (shallow parsing)

He reckons the current account deficit will narrow to only 1.8 billion in September.

A chunker (shallow parser) segments a sentence into non-recursive phrases.

The Noun Phrase (NP)

Examples:
- He
- Barak Obama
- The President
- The former Congressman from Illinois

They can all appear in a similar context: ___ was born in Hawaii.

Prepositional Phrases

Examples:
- the man in the white suit
- Come and look at my paintings
- Are you fond of animals?
- Put that thing on the floor
Verb Phrases
Examples:
- He went
- He was trying to keep his temper.
- She quickly showed me the way to hide.

Chunking
Text chunking is dividing sentences into non-overlapping phrases.

Noun phrase chunking deals with extracting the noun phrases from a sentence.
While NP chunking is much simpler than parsing, it is still a challenging task to build an accurate and very efficient NP chunker.

What is it good for
Chunking is useful in many applications:
- Information Retrieval & Question Answering
- Machine Translation
- Preprocessing before full syntactic analysis
- Text to speech
- ...

What kind of structures should a partial parser identify?
Different structures useful for different tasks:
- Partial constituent structure
  \[[I \text{ saw } [a \text{ tall man } \text{ in the park}]]\]
- Prosodic segments
  \[[I \text{ saw } [a \text{ tall man } \text{ in the park}]]\]
- Content word groups
  \[[I \text{ saw } [a \text{ tall man } \text{ in the park}]]\]

Chunk Parsing
Goal: divide a sentence into a sequence of chunks.

Chunks are non-overlapping regions of a text:
- [I saw [a tall man] in [the park]].

Chunks are non-recursive
- a chunk can not contain other chunks

Chunks are non-exhaustive
- not all words must be included in chunks

Chunk Parsing Examples
Noun-phrase chunking:
- [I saw [a tall man] in [the park]].

Verb-phrase chunking:
- The man who [was in the park] [saw me].

Prosodic chunking:
- [I saw [a tall man] in the park].
Chunks and Constituency

Constituents: [a tall man in [the park]].
Chunks: [a tall man] in [the park].

Chunks are not constituents
- Constituents are recursive

Chunks are typically subsequences of Constituents
- Chunks do not cross constituent boundaries

Chunk Parsing: Accuracy

Chunk parsing achieves higher accuracy
- Smaller solution space
- Less word-order flexibility within chunks than between chunks
- Better locality:
  - Fewer long-range dependencies
  - Less context dependence
- No need to resolve attachment ambiguity
- Less error propagation

Chunk Parsing: Domain Specificity

Chunk parsing is less domain specific:
- Dependencies on lexical/semantic information tend to occur at levels "higher" than chunks:
  - Attachment
  - Argument selection
  - Movement
- Fewer stylistic differences within chunks

Chunk Parsing: Efficiency

Chunk parsing is more efficient
- Smaller solution space
- Relevant context is small and local
- Chunks are non-recursive
- Chunk parsing can be implemented with a finite state machine

Psycholinguistic Motivations

Chunk parsing is psycholinguistically motivated:
- Chunks as processing units
  - Humans tend to read texts one chunk at a time
  - Eye-movement tracking studies
- Chunks are phonologically marked
  - Pauses, Stress patterns
- Chunking might be a first step in full parsing

Chunk Parsing Techniques

Chunk parsers usually ignore lexical content
- Only need to look at part-of-speech tags

Techniques for implementing chunk parsing:
- Regular expression matching / Finite State Machines (see next)
- Transformation Based Learning
- Memory Based Learning
- Other tagging-style methods
Regular Expression Matching

Define a regular expression that matches the sequences of tags in a chunk

- A simple noun phrase chunk regexp:
  \(<DT>\)? <JJ>* <NN>* <NN>

- Chunk all matching subsequences:
  the/DT little/JJ cat/NN sat/VBD on/IN the/DT mat/NN
  [the/DT little/JJ cat/NN] sat/VBD on/IN [the/DT mat/NN]

- If matching subsequences overlap, the first one or longest one gets priority

 Chunking as Tagging

Map Part of Speech tag sequences to \{I,O,B\}*

- I – tag is part of an NP chunk
- O – tag is not part of
- B – the first tag of an NP chunk which immediately follows another NP chunk

Alternative tags: Begin, End, Outside

Example:
- Input: The teacher gave Sara the book
- Output: I I O I B I

Chunking State of the Art

When addressed as tagging – methods similar to POS tagging can be used

- HMM – combining POS and IOB tags
- TBL – rules based on POS and IOB tags

Depending on task specification and test set: 90-95% for NP chunks

Chunking with Machine learning

Chunking performance on Penn Treebank

<table>
<thead>
<tr>
<th></th>
<th>Recall</th>
<th>Precision</th>
<th>F-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Winnow (with basic features) (Zhang, 2002)</td>
<td>90.60</td>
<td>89.54</td>
<td>90.57</td>
</tr>
<tr>
<td>Perceptron (Carreras, 2003)</td>
<td>93.29</td>
<td>94.19</td>
<td>93.74</td>
</tr>
<tr>
<td>SVM + voting (Kudo, 2003)</td>
<td>93.92</td>
<td>93.89</td>
<td>93.91</td>
</tr>
<tr>
<td>SVM (Kudo, 2000)</td>
<td>93.51</td>
<td>93.45</td>
<td>93.48</td>
</tr>
<tr>
<td>Bidirectional MEMM (Tsarfaty, 2005)</td>
<td>91.70</td>
<td>83.70</td>
<td>93.70</td>
</tr>
</tbody>
</table>

Named-Entity Recognition

We have shown that interleukin-1 (IL-1) and IL-2 control protein protein protein
IL-2 receptor alpha (IL-2R alpha) gene transcription in DNA
CD4-CD8-murine T lymphocyte precursors.
cell_line

Recognize named-entities in a sentence.

- Gene/protein names
- Protein, DNA, RNA, cell_line, cell_type

Syntactic Parsing

Estimated volume was a light 2.4 million ounces.
Estimated volume was a light 2.4 million ounces.

Phrase Structure + Head Information

Dependency relations

Parse Tree

Semantic Structure

Feature-Based Parsing

HPSG
- A few schema
- Many lexical entries
- Deep syntactic analysis

Grammar
- Corpus-based grammar construction (Miyao et al. 2004)

Parser
- Beam search (Tsuruoka et al.)