Cognitive Computation Group

Natural Language Processing Tutorial

May 26 & 27, 2011

http://cogcomp.cs.illinois.edu
So why aren’t words enough?

- Depends on the application… more advanced task may require more sophisticated patterns to separate classes

- **Sparsity of features**
  - Many words/sequences of words may not occur very often
  - This means a learned classifier may not generalize well
  - More abstract representation can help

- **Ambiguity of words** – e.g. “terminal”, “moving”
  - Additional information may help

- Meaning encoded in **structure** – e.g. “Matthew Smith, the Maverick’s last hope…”

- **NLP annotation tools** generally abstract over underlying words so that features generalize better
Outline

- Annotators
  - POS, Chunk, NER, Coreference, SRL

- Comparators
  - Overview
  - Instances: WNSim, NESim

- Curator
  - Overview
  - Installing and running
  - Current services
ANNOTATORS
Available from CCG

- Tokenization/Sentence Splitting
- Part Of Speech
- Chunking
- Named Entity Recognition
- Coreference
- Semantic Role Labeling
Tokenization and Sentence Segmentation

- Given a document, find the sentence and token boundaries

  The police chased Mr. Smith of Pink Forest, Fla. all the way to Bethesda, where he lived. Smith had escaped after a shoot-out at his workplace, Machinery Inc.

- Why?
  - Word counts may be important features
  - Words may themselves be the object you want to classify
  - “lived.” and “lived” should give the same information
  - different analyses need to align if you want to leverage multiple annotators from different sources/tasks
Tokenization and Sentence Segmentation ctd.

- Believe it or not, this is an open problem
- No agreed standard for token-level segmentation
  - e.g. “American-led” vs. “American - led”?
  - e.g. “$ 32 M” vs “$32 M” and “$32M”?
- Different tasks may use different standards
- No wildly successful sentence segmenter exists (see the excerpts in news aggregators for some nice errors)
- Noisier text (e.g. online consumer reviews) => poorer performance (for reasons like inconsistent capitalization)
- LBJ distribution includes the Illinois tokenizer and sentence segmenter
# Part of Speech (POS)

- Allows simple abstraction for pattern detection

<table>
<thead>
<tr>
<th>POS</th>
<th>DT</th>
<th>NN</th>
<th>VBD</th>
<th>PP</th>
<th>DT</th>
<th>JJ</th>
<th>NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word</td>
<td>The</td>
<td>boy</td>
<td>stood</td>
<td>on</td>
<td>the</td>
<td>burning</td>
<td>deck</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>POS</th>
<th>DT</th>
<th>NN</th>
<th>VBD</th>
<th>PP</th>
<th>DT</th>
<th>JJ</th>
<th>NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word</td>
<td>A</td>
<td>boy</td>
<td>rode</td>
<td>on</td>
<td>a</td>
<td>red</td>
<td>bicycle</td>
</tr>
</tbody>
</table>

- **Disambiguate** a target, e.g.
  
  “make (a cake)” vs. “make (of car)”

- **Specify** more abstract patterns,
  
  e.g. Noun Phrase: ( DT JJ* NN )

- **Specify** context in abstract way
  
  - e.g. “DT boy VBX” for “actions boys do”
  - This expression will catch “a boy cried”, “some boy ran”, …
Chunking

- Identifies phrase-level constituents in sentences

  [NP Boris] [ADVP regretfully] [VP told] [NP his wife]
  [SBAR that] [NP their child] [VP could not attend] [NP
  night school] [PP without] [NP permission].

- Useful for filtering: identify e.g. only noun phrases, or only verb phrases
  - Groups modifiers with heads
  - Useful for e.g. Mention Detection

- Used as source of features, e.g. distance (abstracts away determiners, adjectives, for example), sequence,…
  - More efficient to compute than full syntactic parse
  - Applications in e.g. Information Extraction – getting (simple) information about concepts of interest from text documents
Named Entity Recognition

- Identifies and classifies strings of characters representing proper nouns

  [PER Neil A. Armstrong], the 38-year-old civilian commander, radioed to earth and the mission control room here: “[LOC Houston], [ORG Tranquility] Base here; the Eagle has landed.”

- Useful for filtering documents
  
  - “I need to find news articles about organizations in which Bill Gates might be involved…”

- Disambiguate tokens: “Chicago” (team) vs. “Chicago” (city)

- Source of abstract features
  
  - E.g. “Verbs that appear with entities that are Organizations”
  
  - E.g. “Documents that have a high proportion of Organizations”
Coreference

- Identify all phrases that refer to each entity of interest – i.e., group mentions of concepts

  [Neil A. Armstrong] , [the 38-year-old civilian commander], radioed to [earth]. [He] said the famous words, “[the Eagle] has landed”.

- The Named Entity recognizer only gets us part-way…

- …if we ask, “what actions did Neil Armstrong perform?”, we will miss many instances (e.g. “He said…”)

- Coreference resolver **abstracts over different ways of referring to the same person**
  - Useful in feature extraction, information extraction
**Semantic Role Labeler**

**Semantic Role Labeling Output**

**Input Text:**
A car bomb that exploded outside the U.S. military base in Beniji killed 11 Iraqi citizens.

**Result: Complete!**

**General Explanation of Argument Labels**

<table>
<thead>
<tr>
<th>A</th>
<th>bomb [A1]</th>
<th>killer [A0]</th>
</tr>
</thead>
<tbody>
<tr>
<td>car</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bomb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>that</td>
<td></td>
<td></td>
</tr>
<tr>
<td>exploded</td>
<td>V: explode</td>
<td></td>
</tr>
<tr>
<td>outside</td>
<td>location</td>
<td></td>
</tr>
<tr>
<td>the</td>
<td>[AM-LOC]</td>
<td></td>
</tr>
<tr>
<td>U.S.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>military</td>
<td>temporal</td>
<td></td>
</tr>
<tr>
<td>base</td>
<td>[AM-TMP]</td>
<td></td>
</tr>
<tr>
<td>in</td>
<td>location</td>
<td></td>
</tr>
<tr>
<td>Beniji</td>
<td>[AM-LOC]</td>
<td></td>
</tr>
<tr>
<td>killed</td>
<td>V: kill</td>
<td>corpse [A1]</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iraqi</td>
<td></td>
<td></td>
</tr>
<tr>
<td>citizens</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- SRL reveals **relations and arguments** in the sentence (where relations are expressed as verbs)
- Cannot abstract over variability of expressing the relations – e.g. kill vs. murder vs. slay…
COMPARATORS
So you want to compare some text:

- How similar are two words? Two strings? Two paragraphs?
  - Depends on what they are
  - String edit distance is usually a weak measure
  - … think about coreference resolution…

<table>
<thead>
<tr>
<th>String 1</th>
<th>String 2</th>
<th>Norm. edit sim.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shiite</td>
<td>Shi’ ‘ite</td>
<td>0.667</td>
</tr>
<tr>
<td>Mr. Smith</td>
<td>Mrs. Smith</td>
<td>0.900</td>
</tr>
<tr>
<td>Wilbur T. Gobsmack</td>
<td>Mr. Gobsmack</td>
<td>0.611</td>
</tr>
<tr>
<td>Frigid</td>
<td>Cold</td>
<td>0.167</td>
</tr>
<tr>
<td>Wealth</td>
<td>Wreath</td>
<td>0.667</td>
</tr>
<tr>
<td>Paris</td>
<td>France</td>
<td>0.167</td>
</tr>
</tbody>
</table>

- Solution: specialized metrics
WNSim

- Generate table mapping terms linked in WordNet ontology
  - Synonymy, Hypernymy, Meronymy
- Score reflects distance (up to 3 edges, undirected – e.g. via lowest common subsumer)
- Score is symmetric

<table>
<thead>
<tr>
<th>String 1</th>
<th>String 2</th>
<th>WNSim distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shiite</td>
<td>Shi’ ‘ite</td>
<td>0</td>
</tr>
<tr>
<td>Mr. Smith</td>
<td>Mrs. Smith</td>
<td>0</td>
</tr>
<tr>
<td>Wilbur T. Gobsmack</td>
<td>Mr. Gobsmack</td>
<td>0</td>
</tr>
<tr>
<td>Frigid</td>
<td>Cold</td>
<td>1</td>
</tr>
<tr>
<td>Wealth</td>
<td>Wreath</td>
<td>0</td>
</tr>
<tr>
<td>Paris</td>
<td>France</td>
<td>0</td>
</tr>
</tbody>
</table>
Using WNSim (present)

- Install and run the WNSim code (see software page)
  - Sets up an xmlrpc server
  - Expects xmlrpc ‘struct’ data structure (analogous to Dictionary)

  ```
  STRUCT { FIRST_STRING: aString;
          SECOND_STRING anotherString }
  ```

  - Returns another xmlrpc data structure:

  ```
  STRUCT {  SCORE: aDouble; REASON: aString }
  ```

- USE: call and cache (reduce network latency overhead)
NESim

- Set of entity-type-specific measures
  - Acronyms, Prefix/Title rules, distance metric
- Score reflects similarity based on type information
- Score is asymmetric

<table>
<thead>
<tr>
<th>String 1</th>
<th>String 2</th>
<th>Norm. edit distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shiite</td>
<td>Shi’ ‘ite</td>
<td>0.922</td>
</tr>
<tr>
<td>Joan Smith</td>
<td>John Smith</td>
<td>0</td>
</tr>
<tr>
<td>Wilbur T. Gobsmack</td>
<td>Mr. Gobsmack</td>
<td>0.95</td>
</tr>
<tr>
<td>Frigid</td>
<td>Cold</td>
<td>0</td>
</tr>
<tr>
<td>Wealth</td>
<td>Wreath</td>
<td>0.900</td>
</tr>
<tr>
<td>Paris</td>
<td>France</td>
<td>0.411</td>
</tr>
</tbody>
</table>
Using NESim (present)

- Either: Install and run the WNSim code (…, …)
  - Sets up an xmlrpc server
  - Expects xmlrpc ‘struct’ data structure (analogous to Dictionary)

```plaintext
STRUCT { FIRST.STRING: aString;
          SECOND.STRING anotherString }
```

- Returns another xmlrpc data structure:

```plaintext
STRUCT {   SCORE: aDouble; REASON: aString }
```

- USE: call and cache (reduce network latency overhead)
- OR put jar on classpath, call programmatically
  - CompareNames()
Using NESim (cont’d)

- Strings have optional extra information – type, context
  
  `<Type>##<original string>[#<start offset>##<end offset>`

- NESim will use specialized resources depending on the type
  - Rules/gazetteers for People’s names
  - Acronyms for Organizations

- NESim can use context to help determine similarity
Big NLP

- We introduced a lot of tools, some of them quite sophisticated.
- The more complex, the bigger the memory requirement:
  - NER: 1G; Coref: 1G; SRL: 4G ….
- If you use tools from different sources, they may be:
  - In different languages
  - Using different data structures
- If you run a lot of experiments on a single corpus, it would be nice to cache the results:
  - …and for your colleagues, nice if they can access that cache.
- Curator is our solution to these problems.
Curator

- Supports distributed NLP resources
  - Central point of contact
  - Single set of interfaces
  - Code generation in many languages (using Thrift)
- Programmatic interface
  - Defines set of common data structures used for interaction
- Caches processed data
- Enables highly configurable NLP pipeline

Overhead:
- Annotation is all at the level of character offsets: Normalization/mapping to token level required
- Need to wrap tools to provide requisite data structures
Curator

Cache

NER

SRL

POS, Chunker
Using Curator for Flexible NLP Pipeline

- [http://cogcomp.cs.illinois.edu/curator/demo/](http://cogcomp.cs.illinois.edu/curator/demo/)

- Setting up:
  - Install Curator Server instance
  - Install components (Annotators)
  - Update configuration files

- Use:
  - Use libraries provided: curatorClient.provide() method
  - Access Record field indicated by Component documentation/configuration
Record Data Structure

struct Record {
 /** how to identify this record. */
  1: required string identifier,
  2: required string rawText,
  3: required map<string, base.Labeling> labelViews,
  4: required map<string, base.Clustering> clusterViews,
  5: required map<string, base.Forest> parseViews,
  6: required map<string, base.View> views,
  7: required bool whitespaced,
}

- rawText contains original text span
- Annotators populate one of the <abc>Views
  - Key is specified in configuration files
Annotator Example: Parser

- Will populate a View, named ‘charniak’
- Curator will expect a Parser interface from the annotator
- Client will expect prerequisites to be provided in other Record fields
  - Specified via Curator server’s annotator configuration file:

```xml
<annotator>
  <type>parser</type>
  <field>charniak</field>
  <host>mycharniakhost.uiuc.edu:8087</host>
  <requirements>sentences:tokens:pos</requirements>
</annotator>
```