Named Entity Recognition and the Stanford NER Software

Jenny Rose Finkel
Stanford University
March 9, 2007

Why NER?
- Question Answering
- Textual Entailment
- Coreference Resolution
- Computational Semantics
- ...

Hidden Markov Models (HMMs)
- Generative
  - Find parameters to maximize $P(X,Y)$
  - Assumes features are independent
  - When labeling $X$, future observations are taken into account (forward-backward)

MaxEnt Markov Models (MEMMs)
- Discriminative
  - Find parameters to maximize $P(Y|X)$
  - No longer assume that features are independent
  - Do not take future observations into account (no forward-backward)

Named Entity Recognition

Germany’s representative to the European Union’s veterinary committee Werner Zwingman said on Wednesday consumers should …

IL-2 gene expression and NF-kappa B activation through CD28 requires reactive oxygen production by 5-lipoxygenase.

NER Data/Bake-Offs
- CoNLL-2002 and CoNLL-2003 (British newswire)
  - Multiple languages: Spanish, Dutch, English, German
  - 4 entities: Person, Location, Organization, Misc
- MUC-6 and MUC-7 (American newswire)
  - 7 entities: Person, Location, Organization, Time, Date, Percent, Money
- ACE
  - 5 entities: Location, Organization, Person, FAC, GPE
- BBN (Penn Treebank)
  - 22 entities: Animal, Cardinal, Date, Disease, …
Conditional Random Fields (CRFs)

- Discriminative
- Doesn’t assume that features are independent
- When labeling \( Y_i \), future observations are taken into account

\[ \Rightarrow \] The best of both worlds!

Model Trade-offs

<table>
<thead>
<tr>
<th></th>
<th>Speed</th>
<th>Discrim vs. Generative</th>
<th>Normalization</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM</td>
<td>very fast</td>
<td>generative</td>
<td>local</td>
</tr>
<tr>
<td>MEMM</td>
<td>mid-range</td>
<td>discriminative</td>
<td>local</td>
</tr>
<tr>
<td>CRF</td>
<td>kinda slow</td>
<td>discriminative</td>
<td>global</td>
</tr>
</tbody>
</table>

Stanford NER

- CRF
- Features are more important than model
- How to train a new model

Our Features

- Word features: current word, previous word, next word, all words within a window
- Orthographic features:
  - Jenny \( \rightarrow \) Xxxx
  - IL-2 \( \rightarrow \) XX-#
- Prefixes and Suffixes:
  - Jenny \( \rightarrow \) <J, <Je, <Jen, …, nny>, ny>, y>
- Label sequences
- Lots of feature conjunctions

Distributional Similarity Features

- Large, unannotated corpus
- Each word will appear in contexts - induce a distribution over contexts
- Cluster words based on how similar their distributions are
- Use cluster IDs as features
- Great way to combat sparsity
- We used Alexander Clark’s distributional similarity code (easy to use, works great!)
- 200 clusters, used 100 million words from English gigaword corpus

Training New Models

Reading data:
- edu.stanford.nlp.sequences.DocumentReaderAndWriter
- Interface for specifying input/output format
- edu.stanford.nlp.sequences.ColumnDocumentReaderAndWriter:

<table>
<thead>
<tr>
<th>German</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘s</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td></td>
</tr>
<tr>
<td>( \rightarrow )</td>
<td></td>
</tr>
<tr>
<td>to</td>
<td></td>
</tr>
<tr>
<td>The</td>
<td></td>
</tr>
<tr>
<td>European</td>
<td>ORGANIZATION</td>
</tr>
<tr>
<td>Union</td>
<td>ORGANIZATION</td>
</tr>
</tbody>
</table>
Training New Models

- Creating features
  - edu.stanford.nlp.sequences.FeatureFactory
    - Interface for extracting features from data
    - Makes sense if doing something very different (e.g., Chinese NER)
  - edu.stanford.nlp.sequences.NERFeatureFactory
    - Easiest option: just add new features here
    - Lots of built-in stuff: computes orthographic features on-the-fly

- Specifying features
  - edu.stanford.nlp.sequences.SeqClassifierFlags
    - Stores global flags
    - Initialized from Properties file

- Other useful stuff
  - useObservedSequencesOnly
    - Speeds up training/testing
    - Makes sense in some applications, but not all
  - window
    - How many previous tags do you want to be able to condition on?
  - feature pruning
    - Remove rare features
  - Optimizer: LBFGS

Distributed Models

- Trained on CoNLL, MUC and ACE
- Entities: Person, Location, Organization
- Trained on both British and American newswire, so robust across both domains
- Models with and without the distributional similarity features

Incorporating NER into Systems

- NER is a component technology
- Common approach:
  - Label data
  - Pipe output to next stage
- Better approach:
  - Sample output at each stage
  - Pipe sampled output to next stage
  - Repeat several times
  - Vote for final output
- Sampling NER outputs is fast

Textual Entailment Pipeline

- Topological sort of annotators
  - NE Recognizer -> Parser -> Coreference -> SR Labeler -> RTE

Sampling Example

- NE Recognizer -> Parser -> Coreference -> SR Labeler -> RTE

<NER, Parser, SRL, Coreference, RTE>
Conclusions

- NER is a useful technology
- Stanford NER Software
  - Has pretrained models for English newswire
  - Easy to train new models
  - http://nlp.stanford.edu/software
- Questions?