FEATURE ENGINEERING

NLP Tutorial
Lab Session, Thursday August 26th

http://cogcomp.cs.illinois.edu/page/tutorial-201008

Setting Up…

- Download/untar the day 2 tarball

  http://cogcomp.cs.illinois.edu/page/tutorial.201008

- Download LBJPOS.jar (and other LBJ jars if needed)

  http://cogcomp.cs.illinois.edu/software

- Set the CLASSPATH for LBJ2.jar, LBJ2Library.jar, and LBJPOS.jar

  ```
  myprompt > export
      CLASSPATH=/home/myname/lib/LBJ2.jar:$CL
  ASSPATH
  ```

  etc.

Feature Engineering

- Two principal files for generating features:

```
Fame/fame.lbj
```

```
Fame/src/edu/illinois/cs/cogcomp/tutorial/Entity.
  java
```

- Entity.java defines feature generating methods (makes sense, as it holds the entity data)
- .lbj file is a good place to take advantage of LBJ's syntactic sugar (e.g. combine features)
- Third file: EntityParser.java: Change the representative sentences for each entity

Information Sources

- Entity data structure has:
  - The canonical name of the entity
  - The type of the entity
  - A set of Instances, each corresponding to a sentence in which this entity appeared
- Each Instance is:
  - A vector of Token -- see LBJLibrary javadoc: http://l2r.cs.uiuc.edu/~cogcomp/software/LBJ2/library/
  - Each token corresponds to a word in the original sentence
  - The tokens corresponding to the owning Entity are marked (in their 'label' field) as 'TARGET'
  - Tokens corresponding to other Entities tagged in the same sentence are marked with the NE Type (also in their 'label' field)
  - The 'label' field in all other tokens is set to the empty string

Entity.java

- **`BagOfWordsCondition`** interface
  - Implement method “boolean accept(Token t)”
  - Several implementations to specify verbs, nouns, adjectives, combinations
- **`bagOfXWindow()`** methods
  - Automatically extract counts, for an entity, of word occurrences within a window of the entity
- **`ClosestWords()`** method
  - Searches for nearest occurrences of words (instead of within window)
  - Allows filtering by BagOfWordsCondition
- **`incrementMap()`** helper method
  - Easily generate histograms

Implemented Feature Generators

- BagOfWordsWindow(i , j)

- BagofVNAsWindow()

IDEAS FOR IMPROVEMENTS (YOU FIRST!)

Hardest (?) first: changing the Parser

- Right now, we use a very crude heuristic to select relevant mentions for a given entity
  - Substring AND tagged NE only
  - Misses pronominal mentions etc.
- Possible change 1: better entity matching
  - Use NESim as a measure to determine similarity
  - Need to choose a threshold – experiment on output (0.80 is a minimum)
- Possible change 2: better entity coverage
  - Use a Coreference annotator on the data
    - Extra work as the files are already tagged with NEs
  - Expand a) mentions of entities within sentences, and b) across sentences (add new sentences)

Ideas for Features

- Not very imaginative…
  - ClosestWord bigrams, trigrams
    - Any potential problems with these features?
  - POS bigrams, trigrams in window of +/- k
    - May want to add POS to tokens in EntityParser's updateEntity() method, instead of in static feature generator method(s)
  - Shallow Parse (Chunker) patterns near entity
- More imaginative:
  - Other entity types in entity's sentences (types, counts, proximity)

Odds and Ends

- Try out the `cache:` and/or `cachedin:` keywords
  - Though you need to think of features that require caching…
- We're missing an essential component of a meaningful evaluation… what is it, and how might we get it?
- If SRL annotator was available, what kinds of features based on SRL might help?