NLP toolkits and hands-on

Kan Min-Yen

Day 1 / Afternoon

Clearing Houses – not just for toolkits

- **ACL Wiki**
http://aclweb.org/aclwiki/index.php?title=Main_Page

- **NLP Software Registry**
http://registry.dfki.de/

- **Local NUS NLP / IR Repository**
  http://www.comp.nus.edu.sg/~rpnlpir
  – Ok, not a repository, just a listing of tools that we use internally at NUS (especially at WING)

Frameworks – UIMA

Unstructured Information Management Architecture

http://domino.research.ibm.com/comm/research_projects.nsf/pages/uima.architectureHighlights.html

- Created by IBM, open sourced to Apache
- Actively supported by universities
  - –Common Data Representation
  - –Plug-n-Play Analysis Engines
  - –Multiple Views and Multi-Modal Support
  - –Java and C++ Interoperability
  - –Component Packaging and Reuse
  - –Collection Processing and Scalability

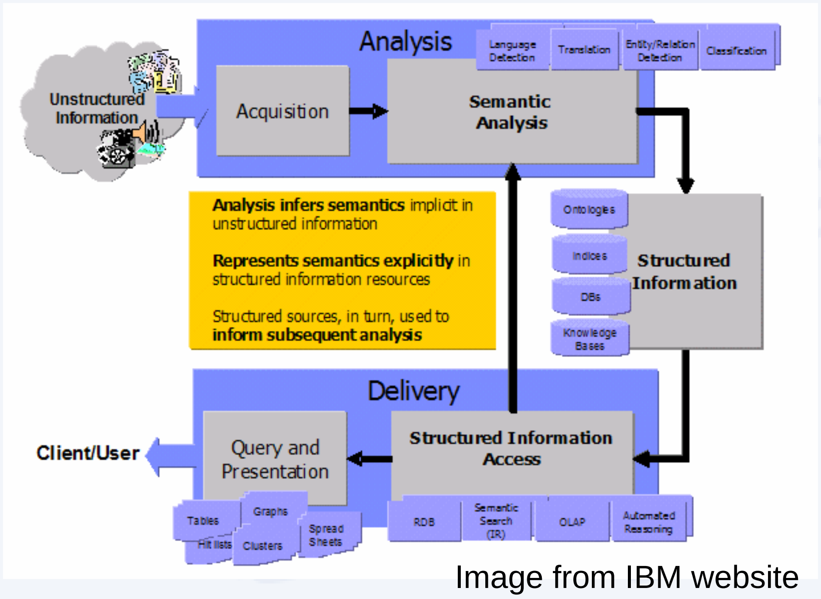


Image from IBM website

Alias-I LingPipe

- Commercial kit developed by researchers

http://alias-i.com/lingpipe/web/demo-coref.html

- Geared to information extraction
- English-centric

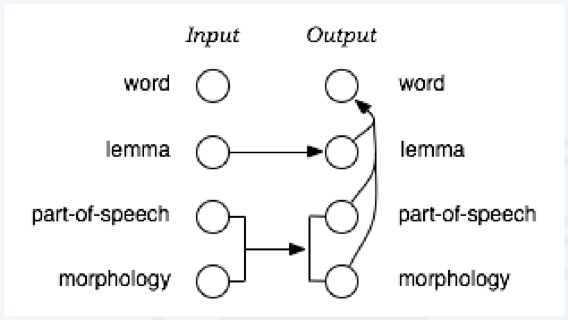
MEAD – Generic Summarization Engine

- **Created by a team, primarily U Michigan**
- **In public domain**
- **Coded in perl**

- **Implementation**
  - – Extract based
  - – Several basic algorithms and baselines
  - – Client-server extensions
  - – Extensible, interface for evaluation (ROUGE)

Moses – Machine Translation Model

- **Funded by EuroMatrix, licensed under LGPL**
- **Features state-of-the-art statistical MT model**
  - Beam search – efficiently explore many hypotheses
  - Factored model – represent different levels of information



  - Phrase based – improves over earlier word based models
    - Work being done to incorporate syntax based methods

GATE

GATE — General Architecture for Text Engineering

- U Sheffield developed, LPGL licensed
- Java (Swing) based, UIMA compliant

- Annotation framework to help gather and store annotations for ML
- Comes with Information Extraction Engine, ANNIE
- Good integration
  - Language plugins
  - Search Engine plugins
  - Tutorials and movies developed by team

openNLP

- **Java based with opennlp.* hierarchy**
- **Development federation, not really unified**

- **Mature packages:**
  - Maximum Entropy framework
  - Open NLP tools – applying MaxEnt to specific NLP problems
  - OpenCCG – parsing based on CCG
  - WordFreak

    Its own annotation tool (but may not work in Java 1.6)

NLTK

- **Team effort of several universities, GPL license**
- **Python based, UIMA compliant**

- **Like GATE, has lots of support**
- **Currently undergoing significant revisions**
  - Book coming out soon
  - nltk-lite to become nltk 2.0

Crash Course in Python

NLTK book draft

NLTK documentation

NLTK example page

Example

```
>>> import sys                          # load the system library
>>> for line in sys.stdin:              # for each line of input text
...     for word in line.split():       # for each word in the line
...         if word.endswith('ing'):    # does the word end in 'ing'?
...             print word              # if so, print the word
```

- Whitespace
- OO
- Methods and Arguments

Two forms of python

- **Command line**
  - `python`
  - works with shebang line

    ```
    #! /usr/bin/env python
    #
    ```
  - use in programming

- **Interactive Shell**
  - `idle`
  - exploratory evaluator
  - tab completion and pop-up help for arguments

Basics

- Strings

```
s = '123456'
print s
print s[0]
print s[2:4]
print s[0:-1]
print  s + s
print t * 3
print s[::2]
s[0] = "2"
```

- Numbers

```
1+2
1/3
1/3.0
```

- Lists

```
a = [ 1, 2, "x" ]
print a[2]
b = a + [ "y", 3 ]
b[0:2] = [1, 12]
b[1:1] = ['c', 'd']
```

Conditionals, looping and functions

```
if x == y:
    print "x has the same value as y"
elif x is y:
    print "x is identical to y"

for w in vocabulary:

for (w,t) in tagged_text:

def my_proc(arg, opt_arg=1, opt_arg_2=2):
```

Others

- **Strings are immutable**
- **Lists are mutable**
  – Some methods change the list directly
- **Tuples are immutable lists**
  – Used by NLTK’s taggers, more efficient?

`type()` – get the type of an object

`help()` – get information on some object

`Import` vs. `from…import`

`Import`

- Keeps module functions separate from user functions.
- Requires the use of dotted names.
- Works with `reload`.

`from…import`

- Puts module functions and user functions together.
- More convenient names.
- Does not work with `reload`.

Hands on with NLTK

Build a Named Entity Recognizer from NIST IE:ER 1999 corpus

Seven steps

1. **Explore IDLE / Python**
2. **Explore IEER corpus**
3. **Change IEER into a tagged corpus**
    1. Understand the Tree data representation
    2. Change to a tagged Tuple representation
4. **Using the default tagger**
    1. Using the evaluation procedures
5. **Regular expression tagging**
6. **Non-trivial unigram tagging**
7. **Bigram tagger with train and test portions**

Summary

- **Myriad of processing pipelines out there**
  - Most open source, but may be able to license (not LGPL)
- **Learned python in the context of NLTK**
  - Stable, interpreted language well-suited for NLP
  - Applied to Named Entity Recognition

  - Regular expressions
  - N gram models
  - Evaluation

Looking Ahead

Day 1

AM
- Applications' Input / Output
- Resources

PM
- Selected Toolkits
- Python Intro
- NLTK Hands-on

>> Day 2

AM
- Evaluation
- Annotation
- Information Retrieval
- ML Intro

PM
- Machine Learning
- SVM Hands-on

Day 3

AM
- Sequence Labeling
- CRF++ Hands-on

PM
- Dimensionality Reduction
- Clustering
- Trends & Issues