COGNITIVE COMPUTATION GROUP
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Cognitive Computation Group

Natural Language Processing Tools Overview
October 28, 2014

http://cogcomp.cs.illinois.edu

MULTIMODAL INFORMATION ACCESS & SYNTHESIS mias

Why might you need Natural Language Processing (NLP)?

- Consider the following task:

  A law firm wants to go through a large set of emails and other electronic documents from a company involved in a legal dispute. They want to understand who has been working with whom, so they want to get all the names of people in the documents and an estimate of the relative importance of each.

- Keyword search won't solve this problem.
- Gazetteers, DBPedia, Wikipedia etc. are not sufficient either.
- You need deeper, more open-ended analysis offered by machine-learned NLP tools.

NLP helps users to…

- …Reduce sparsity of features
  - Many words/sequences of words may not occur very often
  - This means even a learned classifier may not generalize well
  - More abstract representation can help
- …Work around problems due to ambiguity of words – e.g. “terminal”, “moving”
  - Additional information/higher level of abstraction may help
- …Recover meaning encoded in structure – e.g.

  “Matthew Smith, the Maverick’s center back…”
- For machine learning applications, NLP annotation tools abstract over underlying words so that features generalize better

Outline

- CCG NLP Tools for enriching text
- Illinois NLP Curator: managing Annotators
- IllinoisCloudNLP: text analytics in the cloud
- Comparators: computing text similarity
- Learning Based Java: integrating machine learning directly into applications

CCG NLP TOOLS

Available from CCG

- Tokenization/Sentence Splitting
- Part Of Speech
- Chunking
- Named Entity Recognition
- Coreference
- Semantic Role Labeling
- Wikifier
- Hierarchical Dataless Classifier

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 single 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)
- LBJava distribution includes the Illinois tokenizer and sentence segmenter

Part of Speech (POS)

- Allows simple abstraction for pattern detection

POS	DT	NN	VBD	PP	DT	JJ	NN
Word	The	boy	stood	on	the	burning	deck

POS	DT	NN	VBD	PP	DT	JJ	NN
Word	A	boy	rode	on	a	red	bicycle

- 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
- 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 Information Extraction, e.g. Term Extraction

Named Entity Recognition

- Identifies and classifies strings of characters representing proper nouns:

In **[LOC South Ossetia]** , **[ORG Human Rights Watch]** confirmed that a cluster strike in the center of the city of **[LOC Gori]** killed at least eight civilians, including **[MISC Dutch]** journalist **[PER Stan Storimans]**. **[MISC Israeli]** journalist **[PER Zadok Yehezkeli]** was among the injured.

NER (cont'd)

- Disambiguate tokens: "Chicago" (team) vs. "Chicago" (city)
- Useful for filtering documents
  - "I need to find news articles about organizations referred to as "Chicago" in which Bill Gates was also mentioned…" (ORG Chicago + PER Bill Gates)
- 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

  After checking in with pilot **[Buzz Aldrin]**, **[Neil]** radioed to **[earth]**. With a serious look on **[his]** face, **[the 38-year-old civilian commander]** 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

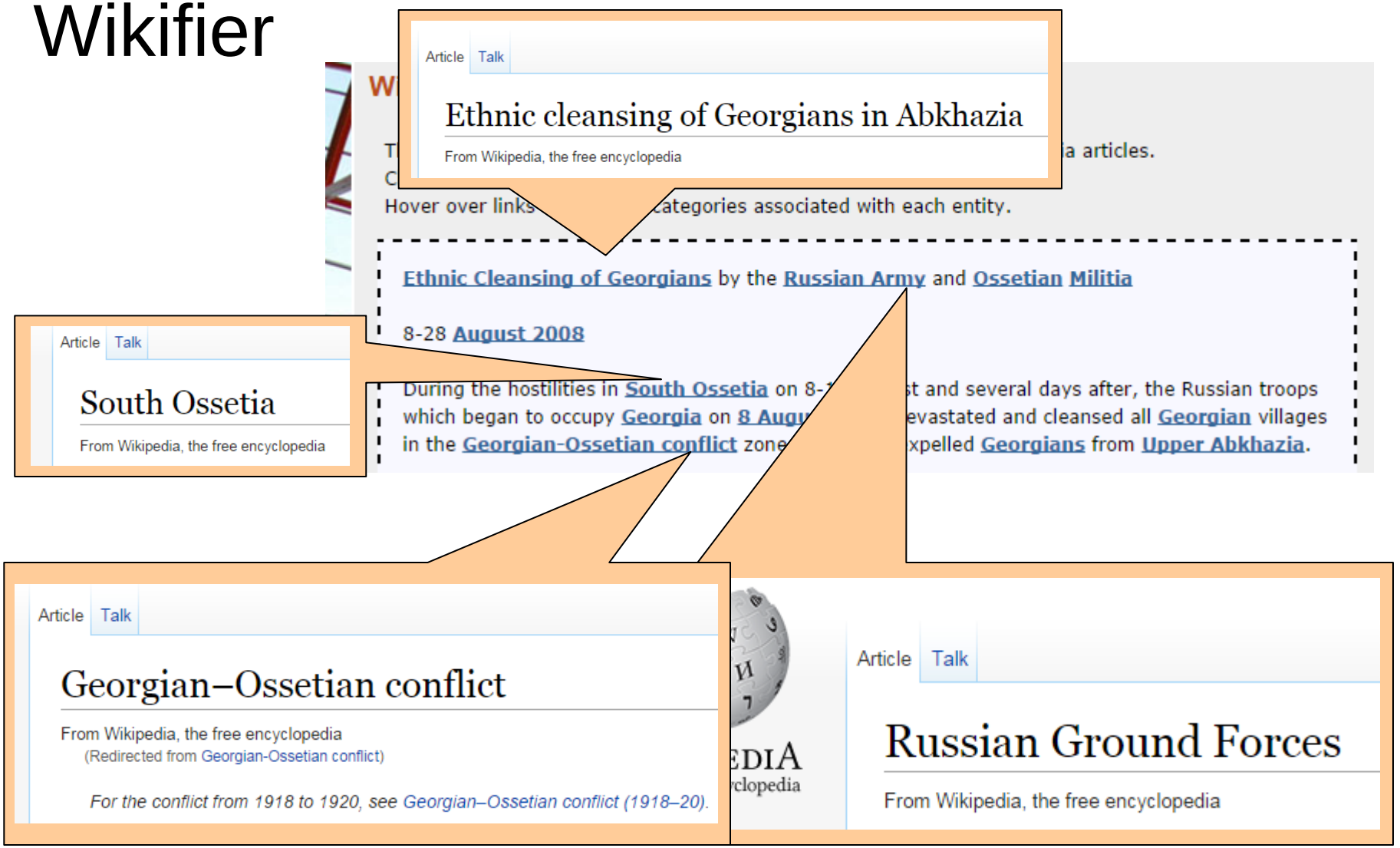
Semantic Role Labeler

Output:

	NE	SRL	Nom	trip	Preposition	Preposition	Preposition
Bart	Person	rider [A0]					
rode		V: ride.01			Governor		Governor
his		steed [A1]					
bike						Governor	
from		direction [AM-DIR]				Source (from)	
Springfield	Geo-political Entity					Object	
to		direction [AM-DIR]			Destination (to)		
Shelbyville	Geo-political Entity				Object		
on		temporal [AM-TMP]					Temporal (on)
Tuesday	Date						Object
.							
Bart	Person			traveller [A0]			
's							
trip			trip 01				
to				destination or path [A1]			
Shelbyville	Geo-political Entity						
was							
on							
Tuesday	Date						
.							

- 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…

Wikifier


Article
Talk
Ethnic cleansing of Georgians in Abkhazia
From Wikipedia, the free encyclopedia
Hover over links
categories associated with each entity.
Ethnic Cleansing of Georgians by the Russian Army and Ossetian Militia
8-28 August 2008
During the hostilities in South Ossetia on 8-
st and several days after, the Russian troops
which began to occupy Georgia on 8 Augu
evastated and cleansed all Georgian villages
in the Georgian-Ossetian conflict zone
xpelled Georgians from Upper Abkhazia.
Article
Talk
South Ossetia
From Wikipedia, the free encyclopedia
Article
Talk
Georgian–Ossetian conflict
From Wikipedia, the free encyclopedia
(Redirected from Georgian-Ossetian conflict)
For the conflict from 1918 to 1920, see Georgian–Ossetian conflict (1918–20).
Article
Talk
Russian Ground Forces
From Wikipedia, the free encyclopedia

Dataless Classifier

- Hierarchical classification of text using a single universal model
- Relies on semantics of **labels** to allow unsupervised/semi-supervised training of hierarchical and multi-label text classification models
  - Build representation of category using Explicit Semantic Analysis (ESA) – wikipedia-derived term-based representation
  - Use a nearest-neighbor model to map examples to labels
  - Bootstrapping process over target corpus improves performance

Performance

Tool	Publictn	Dataset	CCG	Best Other
Co-reference	EMNLP '13	OntoNotes 5	63.30 (avg. of MUC, B^3, CEAF)	63.37*
Named Entity	ACL '11	CoNLL '03	90.36	90.90*
Wikifier	EMNLP '13	Custom	87.12 (avg. over 4 data sets)	76.30
SRL (Verb)	CoNLL '05	WSJ+Brown	**77.92**	77.30
SRL (Prep)	EMNLP '11	WSJ 23	67.82	-
Dataless	AAAI '14	20NG (unsup)	68.2/83.7**	59.5

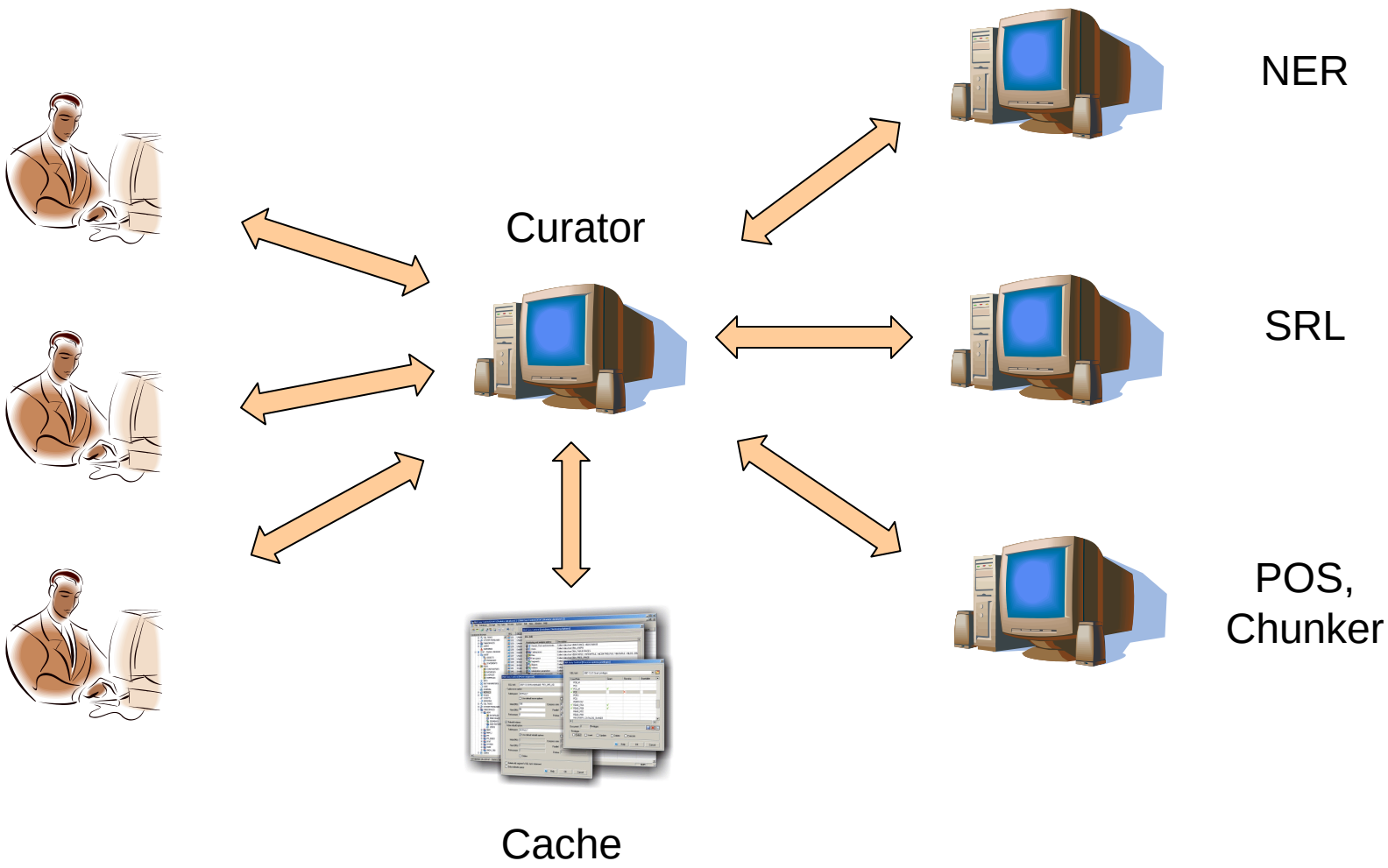
*could not find online release of software
** second result uses bootstrapping

ILLINOIS NLP CURATOR

CCG NLP and Curator

- CCG NLP emphasizes performance in terms of accuracy. Our tools are state-of-the-art.
- While most are also fast, some are not as fast as lighter-weight counterparts that sacrifice some accuracy for speed.
- Many CCG tools also have higher memory-requirements than their counterparts.
- In our use cases, many people run multiple experiments on overlapping or identical data sets. Caching can help to speed things up.
- We want to use some non-Java NLP components seamlessly with Java

Curator



Illinois NLP Curator

- Supports distributed NLP resources using Software as Service model
  - 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: Need to wrap tools to provide requisite data structures (if you want something beyond what ships with Curator)

Getting Started With the Curator

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

- The default installation comes with the following annotators (Illinois, unless mentioned):
  - Sentence splitter and tokenizer
  - POS tagger
  - Shallow Parser
  - Named Entity Recognizer (4-type and 18-type)
  - Coreference resolution system
  - Charniak Syntactic Parser
  - Verb and Noun Semantic Role Labeler
  - Wikifier

Basic Concept

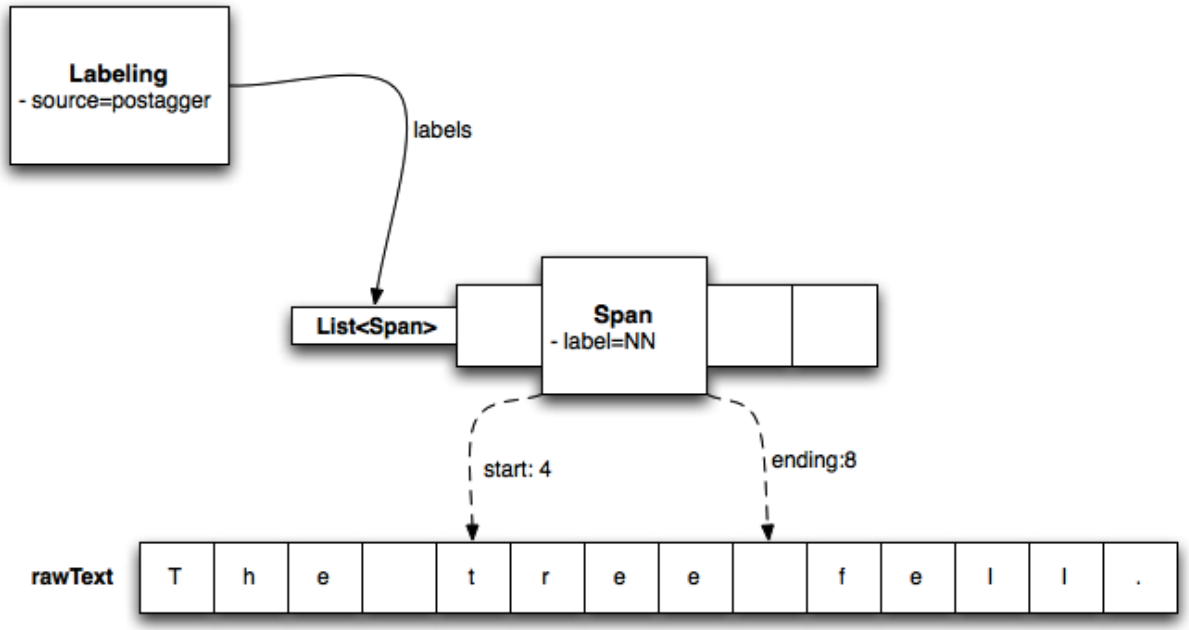
- Different NLP annotations can be defined in terms of a few simple data structures:
  1. Record: A big container to store all annotations of a text
  2. Span: A span of text (defined in terms of character offsets) along with a label (A single token, a POS tag, a Named Entity)
  3. Labeling: A collection of Spans (e.g. all POS tags for the text)
  4. Trees and Forests (Parse trees, predicate-argument structures)
  5. Clustering: A collection of Labelings (Co-reference)
  6. View: A layer of annotation consisting of a Labeling, Tree, Forest, or Clustering

For more information, see:

http://cogcomp.cs.illinois.edu/trac/curator.php

Example of a Labeling

The tree fell.



Using Curator for Flexible NLP Pipeline

- 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

Native 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

Using Curator

- Low-level socket-like API: numerous languages
- Curator-utils (Java library)
  - Simple curator client, serialization and deserialization
  - Uses native Curator data structures
  - Character-offset-based annotations
- Edison (Java library)
  - Simple curator client, richer data structures
  - More support for e.g. feature extraction over multiple annotations
  - Token-based and character-offset-based annotations
  - Out-of-the-box alignment of different annotations with each other

Curator snippet (Edison)

```
import edu.cs.illinois.cogcomp.edison.data.curator.CuratorClient;

String host = “somewhere.cs.illinois.edu”;
int port = 10010;
CuratorClient client = new CuratorClient( host, port );
boolean forceUpdate = false; // if ‘true’, flush cache for this input
String corpusId = “test”;
String textId = “test-01”;
String text = “The car cost $800. Noone wanted to buy it.”;
TextAnnotation ta = client.getTextAnnotation(corpusId, textId,
                                                    text, forceUpdate);
client.addSRLVerbView(ta, forceUpdate);

....
```

IllinoisCloudNLP

http://cogcomp.cs.illinois.edu/page/software_view/IllinoisCloudNLP

- Uses Amazon Web Services to run multiple Curator instances on demand
- Self-contained workflow for processing large document collections with a range of CCG NLP tools:
  - Part of Speech tagger
  - Chunker
  - Basic Named Entity Recognizer
  - Extended Named Entity Recognizer (18 types)
  - Wikifier
- Release is imminent; more annotators to be added soon…

COMPARATORS

So you want to compare some text….

- **How similar are two words? Two strings? Two paragraphs?**
  - Depends on what they are and what your task is
  - String edit distance is usually a weak measure
  - … think about co-reference resolution…

String 1	String 2	Norm. edit sim.
Shiite	Shi' 'ite	0.667
Mr. Smith	Mrs. Smith	0.900
Wilbur T. Gobsmack	Mr. Gobsmack	0.611
Frigid	Cold	0.167
Wealth	Wreath	0.667
Paris	France	0.167

- **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**

String 1	String 2	WNSim distance
Shiite	Shi' 'ite	0
Mr. Smith	Mrs. Smith	0
Wilbur T. Gobsmack	Mr. Gobsmack	0
Frigid	Cold	1
Wealth	Wreath	0
Paris	France	0

Using WNSim

- 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)
- OR: there is a "limited" version in Java: use programmatically

NESim

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

String 1	String 2	Norm. edit distance
Shiite	Shi' 'ite	1
Joan Smith	John Smith	0
Wilbur T. Gobsmack	Mr. Gobsmack	1
Frigid	Cold	0
Wealth	Wreath	1
Paris	France	0

Using NESim

- Non-Java: Install and run the WNSim code
  - 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)

Using NESim (2)

- Programmatic use:

```
EntityComparison ec = new EntityComparison();
ec.compare(name1, name2);
ec.getScore();  //1 if the names could refer to same entity
ec.getConfidence(); //A confidence level between 0 and 1.
```

Argument Format: name1 and name2 must be of the following two forms.

Type#Name //

Name

Type can be PER, LOC, ORG, DEG, MISC. Any other type will be treated as MISC.

NESim argument format -- examples

String name1 = "PER#Clint Eastwood";
String name2 = "PER#Clint";

String name1 = "Eastwood";
String name2 = "Mr. Eastwood";

String name1 = "PER#Clint Eastwood";
String name2 = "Mr. Eastwood";

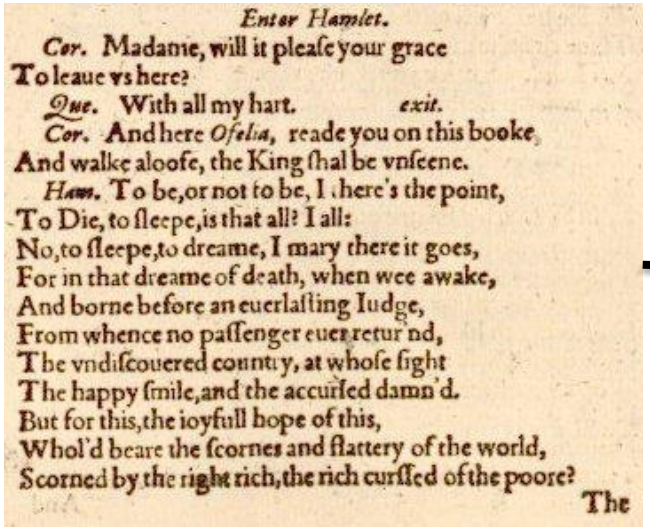
String name1 = "ORG#Mitsubishi Inc.";
String name2 = "ORG#Mitsubishi";

LEARNING-BASED JAVA

LBJava: Learning-Based Java

- http://cogcomp.cs.illinois.edu/
- A modeling language for supervised learning
- Supports:
  - Programming using learned models
  - High level specification of features and constraints between classifiers
  - Inference with constraints
- Key features:
  - Classifiers are functions defined in terms of data
  - Learning happens at *compile time*

Sample application: text classification

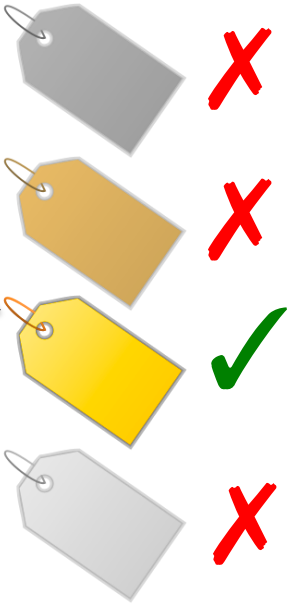


Enter Hamlet.

Cor. Madame, will it pleaſe your grace
To leaue vs here?
Que. With all my hart. *exit.*
Cor. And here *Ofelia*, reade you on this booke,
And walke aloofe, the King ſhal be vnſeene.
Ham. To be, or not to be, I there's the point,
To Die, to ſleepe, is that all? I all:
No, to ſleepe, to dreame, I mary there it goes,
For in that dreame of death, when wee awake,
And borne before an euerlaſting Iudge,
From whence no paſſenger euer retur nd,
The vndiſcouered country, at whoſe ſight
The happy ſmile, and the accurſed damn'd.
But for this, the ioyfull hope of this,
Whol'd beare the ſcornes and flattery of the world,
Scorned by the right rich, the rich curſſed of the poore?
The

A document

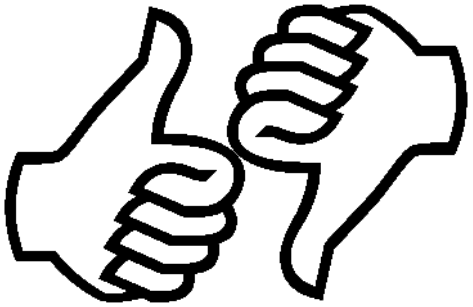
A classifier
(black box)



Some labels

Several applications fit this framework

Spam detection



Sentiment classification

What does LBJava do for you?

- Abstracts away the feature representation, learning and inference
- Allows you to write *learning based programs*
- Application developers can reason about the application at hand, and don't have to worry about the implementation of the learning components

Programming a classifier with LBJava

You will need:

- A Data Parser
  - Read the labeled/unlabeled data into a data structure
- Feature Extractors
  - Java code to extract relevant patterns from the input data structure
- A Classifier Definition
  - Using LBJava's grammar, specify the learning algorithm and its parameters, including the features it will use

A simple LBJava program

```
/** A learned text classifier; its definition comes from data. */
discrete TextClassifier(Document d) <-
learn TextLabel
  using WordFeatures
  from new DocumentReader("data/spam/train")

  with SparseAveragedPerceptron {
    learningRate = 0.1 ;
    thickness = 3.5;
  }
  5 rounds

  testFrom new DocumentReader("data/spam/test")
end
```

Defines a classifier

The object being classified

The function being learned

The feature representation

The source of the training data

The learning algorithm

See http://cogcomp.cs.illinois.edu/page/tutorial.201310 for more details

Using LBJava Classifiers

- Once trained, LBJava classifiers are just another class you can use in your Java code
- We use these in many of our NLP applications…

```
private TextClassifier tc = new TextClassifier();
String label = tc.discreteValue(w); // best prediction
Score[] scores= tc.scores(w).toArray();
// list of scores corresponding to possible labels for w
```

- …and they can be used to generate features in new LBJ applications.

Reusing LBJava Code

- Your LBJava code itself can be adapted to other, similar tasks very simply:
  - Generate a new data set with the same format, but whatever labels you want to use
  - Train the existing LBJava classifier
- For example: you could generate a new data set with your own entity types (e.g. “Politician”, “Tycoon”, “Sportsperson”), then take our existing Named Entity Recognizer, retrain it without changing anything but the data source, and use it to classify with the new types

CCG Use of LBJava

- LBJava has a range of learning algorithms, including Averaged Perceptron and SVM
- LBJava also directly supports cross-validation and confidence interval to evaluate performance
- We have built a number of sophisticated systems using LBJava, including our Named Entity Recognizer, Semantic Role Labeler, Relation Recognizer, Event Timeline extractor
- You can build a close-to-SOA NER tagger in half a day!

SUMMARY

Recap

- CCG has developed a range of state-of-the-art NLP tools and a suite of supporting applications geared toward programmatic use
- You can use these tools to support analysis of target documents for e.g. data mining
- Curator reduces local system requirements, caches annotations for reuse
- IllinoisCloudNLP lets your process documents on Amazon's EC2 infrastructure
- LBJava simplifies the development and use of machine-learned classifiers in Java applications

QUESTIONS?