To change

- More textrunner,
- more pattern learning
- Reorder:
  - Kia start

What is Open Information Extraction?

<table>
<thead>
<tr>
<th>Traditional I.E.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input</strong></td>
</tr>
<tr>
<td><strong>Relations</strong></td>
</tr>
<tr>
<td><strong>Complexity</strong></td>
</tr>
</tbody>
</table>

Example:
- Microsoft is the largest software company.
- Boeing moved its headquarters to Chicago in 2003.
- Hank Levy was named chair of Computer Science & Engr.
- HeadquarterOf(<company>,<city>)
What is Open Information Extraction?

<table>
<thead>
<tr>
<th>Traditional IE</th>
<th>Open IE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>Corpus + Domain-Independent Methods</td>
</tr>
<tr>
<td>Relations</td>
<td>Discovered Automatically</td>
</tr>
<tr>
<td>Complexity</td>
<td>$O(1) + O(1)$ documents, $O(1)$ relations</td>
</tr>
</tbody>
</table>

Methods for Open IE

- **Self Supervision**
- Kylin (Wikipedia)
- Shrinkage & Retraining
- Temporal Extraction
- **Hearst Patterns**
- PMI Validation
- Subclass Extraction
- **Pattern Learning**
- Structural Extraction
- List Extraction & WebTables
- TextRunner

---

The Intelligence in Wikipedia Project

Daniel S. Weld  
Department of Computer Science & Engineering  
University of Washington  
Seattle, WA, USA  

Joint Work with  
Fei Wu, Raphael Hoffmann, Stef Schoenmackers,  
Eytan Adar, Saleema Amershi, Oren Etzioni,  
James Fogarty, Chloe Kiddon,  
Shawn Ling & Kayur Patel

Motivating Vision

Next-Generation Search = Information Extraction + Ontology + Inference

Which German Scientists Taught at US Universities?

- Albert Einstein was a German-born theoretical physicist...
- New Jersey is a state in the northeastern region of the United States...

Next-Generation Search

- Information Extraction
- <Einstein, Born-In, Germany>
- <Einstein, ISA, Physicist>
- <Einstein, Lectured-At, IAS>
- <IAS, In, New-Jersey>
- <New-Jersey, In, United-States>
- Ontology
  - Physicist (x)  Scientist(x)
- Inference
  - Einstein = Einstein

Why Wikipedia?

- **Comprehensive**
- High Quality  
  [Giles Nature 05]
- **Useful Structure**
  Unique IDs & Links  
  Infoboxes  
  Categories & Lists  
  First Sentence  
  Disambiguation pages  
  Revision History  
  Multilingual

Cons

- Natural-Language
- Missing Data
- Inconsistent
- Low Redundancy
Clearfield County was created in 1804 from parts of Huntingdon and Lycoming Counties but was administered as part of Centre County until 1812. Its county seat is Clearfield.  

2,972 km² (1,147 mi²) of it is land and 17 km² (7 mi²) of it (0.56%) is water. As of 2005, the population density was 28.2/km².

Kylin: Self-Supervised Information Extraction from Wikipedia

Kylin Architecture

The Precision / Recall Tradeoff
- **Precision**
  \[
  \frac{tp}{tp + fp}
  \]
  Proportion of selected items that are correct

- **Recall**
  \[
  \frac{tp}{tp + fn}
  \]
  Proportion of target items that were selected

- **Precision-Recall curve**
  - Shows tradeoff

- **Correct Tuples**
- **Tuples returned by System**

- **AuC**

**Preliminary Evaluation**
- Kylin Performed Well on Popular Classes:
  - Precision: mid 70% ~ high 90%
  - Recall: low 50% ~ mid 90%
- ... Floundered on Sparse Classes – Little Training Data

Long-Tail 2: Incomplete Articles
- Desired Information Missing from Wikipedia
  800,000/1,800,000 (44.2%) stub pages

Shrinkage?
- performer (44)
- actor (8738)
- comedian (106)

stub [July 2007 of Wikipedia]
Subsumption Detection

- Binary Classification Problem
- Nine Complex Features
  - E.g., String Features
  - IR Measures
  - Mapping to Wordnet
  - Hearst Pattern Matches
  - Class Transitions in Revision History
- Learning Algorithm
  - SVM & MLN Joint Inference

KOG Architecture

- Heuristics
  - Edit History
  - String Similarity
- Experiments
  - Precision: 94%    Recall: 87%
- Future
  - Integrated Joint Inference

Improving Recall on Sparse Classes

- Shrinkage
  - Extra Training Examples from Related Classes
- How Weight New Examples?
Improvement due to Shrinkage

Improving Recall on Sparse Classes

Retraining
- Compare Kylin Extractions with Tuples from Textrunner
- Additional Positive Examples
- Eliminate False Negatives

TextRunner [Banko et al. IJCAI-07, ACL-08]
- Relation-Independent Extraction
- Exploits Grammatical Structure
- CRF Extractor with POS Tag Features

Recall after Shrinkage / Retraining...

Shrinkage
- Retraining

Extract from Broader Web
- 44% of Wikipedia Pages = "stub"
  - Extractor quality irrelevant
- Query Google & Extract
  - How maintain high precision?
  - Many Web pages noisy, describe multiple objects
  - How integrate with Wikipedia extractions?

Bootstrapping to the Web

Main Lesson: Self Supervision
- Find structured data source
- Use heuristics to generate training data
  - E.g. Infobox attributes & matching sentences
Self-supervised Temporal Extraction

- Goal Extract:
  - happened(recognizes(UK, China), 1/6/1950)

Other Sources

- Google News Archives

Methods for Open IE

- Self Supervision (Wikipedia)
- Kylin & Shrinkage & Retraining
- Temporal Extraction
- Hearst Patterns
- PMI Validation
- Subclass Extraction
- Pattern Learning
- Structural Extraction
- List Extraction & WebTables
- TextRunner

The KnowItAll System

Unary predicates: instances of a class

- Unaries predicates:
  - instanceOf(City), instanceOf(Film), instanceOf(Company), ...

- Good recall and precision from generic patterns:
  - <class> "such as" X
  - X "and other" <class>

- Instantiated rules:
  - “cities such as” X X “and other cities”
  - “films such as” X X “and other films”
  - “companies such as” X X “and other companies”

Recall – Precision Tradeoff

High precision rules apply to only a small percentage of sentences on Web

<table>
<thead>
<tr>
<th></th>
<th>hits for “X”</th>
<th>“cities such as X”</th>
<th>“X and other cities”</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boston</td>
<td>365,000,000</td>
<td>15,600,000</td>
<td>12,000</td>
</tr>
<tr>
<td>Tukwila</td>
<td>1,300,000</td>
<td>73,000</td>
<td>44</td>
</tr>
<tr>
<td>Gjatsk</td>
<td>88</td>
<td>34</td>
<td>0</td>
</tr>
<tr>
<td>Hadaslav</td>
<td>51</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

“Redundancy-based extraction” ignores all but the unambiguous references.
**Limited Recall with Binary Rules**

Relatively high recall for unary rules:
- “companies such as” X: 2,800,000 Web hits
- X “and other companies”: 500,000 Web hits

Low recall for binary rules:
- X “is the CEO of Microsoft” → 160 Web hits
- X “is the CEO of Wal-mart” → 19 Web hits
- X “is the CEO of Continental Grain” → 0 Web hits
- X “, CEO of Microsoft” → 6,700 Web hits
- X “, CEO of Wal-mart” → 700 Web hits
- X “, CEO of Continental Grain” → 2 Web hits

**Examples of Extraction Errors**

Rule: countries such as X ⇒ instanceOf(Country, X)
- “We have 31 offices in 15 countries such as London and France.”
  ⇒ instanceOf(Country, London) instanceOf(Country, France)

Rule: X and other cities ⇒ instanceOf(City, X)
- “A comparative breakdown of the cost of living in Klamath County and other cities follows.”
  ⇒ instanceOf(City, Klamath County)

**“Generate and Test” Paradigm**

1. Find extractions from generic rules
2. Validate each extraction
   - Assign probability that extraction is correct
   - Use search engine hit counts to compute PMI
   - PMI (pointwise mutual information) between
     - extraction
     - “discriminator” phrases for target concept

**Computing PMI Scores**

Measures mutual information between the extraction and target concept.

\[
PMI(D, I) = \frac{\text{hits}(D + I)}{\text{hits}(I)}
\]

**Example of PMI**

Discriminator: “countries such as X”
Instance: “France” vs. “London”
PMI for France ⇒ PMI for London (2 orders of mag.)
Need features for probability update that distinguish
- “high” PMI from “low” PMI for a discriminator

- “countries such as France”: 27,800 hits
- “France”: 14,300,000 hits
- “countries such as London”: 71 hits
- “London”: 12,600,000 hits

\[
PMI = \frac{27,800}{14,300,000} = 1.94E^{-4}
\]

**PMI for Binary Predicates**

\[
PMI(D, I_1, I_2) = \frac{\text{hits}(D + I_1 + I_2)}{\text{hits}(I_1, I_2)}
\]

**PmI**

Extraction: CeoOf(“Jeff Bezos”, “Amazon”)
Discriminator: <arg1> ceo of <arg2>
PMI = 0.017
620 hits for “Jeff Bezos ceo of Amazon”
39,000 hits for “Jeff Bezos”, “Amazon”
Bootstrap Training

1. Only input is set of predicates with class labels.
   instanceOf(Country), class labels “country”, “nation”
2. Combine predicates with domain-independent templates
   <class> such as NP => instanceOf(class, NP)
   to create extraction rules and discriminator phrases
   rule: “countries such as” NP => instanceOf(Country, NP)
   discrim: “country X”
3. Use extraction rules to find set of candidate seeds
4. Select best seeds by average PMI score
5. Use seeds to train discriminators and select best discriminators
6. Use discriminators to rerank candidate seeds, select new seeds
7. Use new seeds to retrain discriminators, ….

Bootstrap Parameters

- Select candidate seeds with minimum support
  - Over 1,000 hit counts for the instance
  - Otherwise unreliable PMI scores
- Parameter settings:
  - 100 candidate seeds
  - Pick best 20 as seeds
  - Iteration 1, rank candidate seeds by average PMI
  - Iteration 2, use trained discriminators to rank candidate seeds
  - Select best 5 discriminators after training
    - Favor best ratio of \( \frac{P(PMI > \text{thresh} | \phi)}{P(PMI > \text{thresh} | -\phi)} \)
    - Slight preference for higher thresholds
  - Produced seeds without errors in all classes tested

Discriminator Phrases from Class Labels

From the class labels “country” and “nation”

- country X nation X
- countries X nations X
- X country X nation
- X countries X nations

Equivalent to weak extraction rules
- no syntactic analysis in search engine queries
- ignores punctuation between terms in phrase

PMI counts how often the weak rule fires on entire Web
- low hit count for random errors
- higher hit count for true positives

Discriminator Phrases from Rule Keywords

From extraction rules for instanceOf(Country)

- countries such as X nations such as X
- such countries as X such nations as X
- countries including X nations including X
- countries especially X nations especially X
- X and other countries X and other nations
- X or other countries X or other nations
- X is a country X is a nation
- X is the country X is the nation

Higher precision but lower coverage than discriminators from class labels

Using PMI to Compute Probability

Standard formula for Naïve Bayes probability update

\[
P(\phi | f_1, f_2, ..., f_n) = \frac{P(\phi) \cdot P(f_1 | \phi) \cdot P(f_2 | \phi) \cdot ... \cdot P(f_n | \phi)}{P(\phi) \cdot P(f_1) \cdot P(f_2) \cdot ... \cdot P(f_n)}
\]

Probability that fact \( \phi \) is a correct, given features \( f_1, f_2, ..., f_n \)

Need to turn PMI-scores into features \( f_1, f_2, ..., f_n \)

Need to estimate conditional probabilities \( P(f_i | \phi) \) and \( P(f_i | -\phi) \)

Features from PMI: Method #1

Thresholded PMI scores

Learn a PMI threshold from training
Learn conditional probabilities for PMI > threshold,
  given that \( \phi \) is in the target class, or not

\[
P(PMI > \text{thresh} | \text{class}) \cdot P(PMI <= \text{thresh} | \text{class})
\]

\[
P(PMI > \text{thresh} | \neg \text{class}) \cdot P(PMI <= \text{thresh} | \neg \text{class})
\]
One Threshold or Two?

Wide gap between positive and negative training.
Often two orders of magnitude.

With two thresholds, learn conditional probabilities:

- \( P(\text{PMI} > \text{thresh}_A | \text{class}) \)
- \( P(\text{PMI} > \text{thresh}_A | \text{not class}) \)
- \( P(\text{PMI} < \text{thresh}_B | \text{class}) \)
- \( P(\text{PMI} < \text{thresh}_B | \text{not class}) \)
- \( P(\text{PMI} \text{ between } A,B | \text{class}) \)
- \( P(\text{PMI} \text{ between } A,B | \text{not class}) \)

Polysemy

- Problems with Polysemy
  - Low PMI if instance has multiple word senses
  - False negative if target concept is not the dominant word sense.
- “Amazon” as an instance of River
  - Most references are to the company, not the river
- “Shaft” as an instance of Film
  - 2,000,000 Web hits for the term “shaft”
  - Only a tiny fraction are about the movie

Chicago

\[
\text{PMI} (I, D, C) = \frac{|\text{Hits} (I + D \mid C)|}{|\text{Hits} (I \mid C)|} - |\text{Hits} (\text{Chicago} + \text{Movie} \mid \text{City})| - |\text{Hits} (\text{Chicago} \mid \text{City})|
\]

Impact of Unmasking on PMI

<table>
<thead>
<tr>
<th>Name</th>
<th>Recessive</th>
<th>Original</th>
<th>Unmask</th>
<th>Boost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Washington city</td>
<td>0.50</td>
<td>0.99</td>
<td>96%</td>
<td></td>
</tr>
<tr>
<td>Casablanca city</td>
<td>0.41</td>
<td>0.93</td>
<td>127%</td>
<td></td>
</tr>
<tr>
<td>Chevy Chase actor</td>
<td>0.09</td>
<td>0.58</td>
<td>512%</td>
<td></td>
</tr>
<tr>
<td>Chicago movie</td>
<td>0.02</td>
<td>0.21</td>
<td>972%</td>
<td></td>
</tr>
</tbody>
</table>

Methods for Open IE

- Self Supervision
  - Kylin (Wikipedia)
- Shrinkage & Retraining
- Temporal Extraction
- Hearst Patterns
  - PMI Validation
  - Subclass Extraction
- Pattern Learning
- Structural Extraction
  - List Extraction & WebTables
  - TextRunner
How to Increase Recall?

RL: learn class-specific patterns.
   “Headquarted in <city>”

SE: Recursively extract subclasses.
   “Scientists such as physicists and chemists”

LE: extract lists of items
   (~ Google Sets).

List Extraction (LE)

1. Query Engine with known items.
2. Learn a wrapper for each result page.
3. Collect large number of lists.
4. Sort items by number of list “votes”.

LE+A=sort list according to Assessor.

Evaluation: Web recall, at precision= 0.9.

Results for City

Results for Scientist

Found 10,300 cities missing from Tipster Gazetteer.

Methods for Open IE

- Self Supervision
- Kylin (Wikipedia)
- Shrinkage & Retraining
- Temporal Extraction
- Hearst Patterns
- PMI Validation
- Subclass Extraction
- Pattern Learning
- Structural Extraction
- List Extraction & WebTables
- TextRunner
(Formerly) Open Question #3

- Sparse data (even with entire Web)
- PMI thresholds are typically small (1/10,000)
- False negatives for instances with low hit count
- City of Duvall
  - 312,000 Web hits
  - Under threshold on 4 out of 5 discriminators
- City of Mossul
  - 9,020 Web hits
  - Under threshold on all 5 discriminators

See next talk…

Future Work III
Creating Better CRF Features?

But where do we get the lists from?

Mining Lists from the Web

But where do we get the lists from?
Tom was born in Seattle.