Information extraction:
Conceptual hierarchies and relations

Vivi Nastase

with material from Marius Pasca’s CIKM-2011 tutorial on IE
Summer semester 2012, ICL, University of Heidelberg
Bootstrapping in general – reminder

Seokhwan Kim et al., 2011 Semi-supervised Information Extraction
Bootstrapping for relation extraction

Start either with a non-empty set $S = (n_{i1}, n_{i2})$ of seed pair examples or a non-empty set $P$ of patterns (let’s assume examples):

1. find all occurrences of the examples $(n_{i1}, n_{i2})$ in the text collection
2. extract [and rank] patterns joining the terms in each pair: $n_{i1}w_1...w_kn_{i2}$
3. add the [highest ranking] extracted patterns to $P$
4. use the patterns in $P$ to find additional pairs
5. add the [highest ranking] extracted pairs to $S$, go to step 1
### Extracting taxonomical relations

Hearst, 1992: *Automatic acquisition of hyponyms from large text corpora*

<table>
<thead>
<tr>
<th>NP such as NP, NP, ...</th>
<th>The bow lute, <em>such as</em> the Bambara ndang ...</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>such NP as</em> {NP,}* {(or,</td>
<td>,and)} NP</td>
</tr>
<tr>
<td>NP{, NP}*{, (or,</td>
<td>,and) other NP</td>
</tr>
<tr>
<td>NP{, (including,</td>
<td>,especially) NP</td>
</tr>
</tbody>
</table>
Adding pattern evaluation

Brin, 1998 *Extracting patterns and relations from the World Wide Web*

\[
\text{specificity}(p) \approx -\log(P(X \in M_D(p)))
\]

\(M_D(p)\) is the set of tuples that match the pattern \(p\) in the document set \(D\), and \(X\) is a random variable uniformly distributed over the domain of tuples for the mined relation \(R\). (In practice, the specificity of the pattern is measured based on the length of the pattern.)

And different style patterns:

<table>
<thead>
<tr>
<th>URL Pattern</th>
<th>Text Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.sff.net/locus/c">www.sff.net/locus/c</a>.*</td>
<td>(&lt;\text{LI}&gt;\text{title}&lt;\text{/B}&gt;\ \text{by author})</td>
</tr>
<tr>
<td>dns.city-net.com/ lmann/awards/hugos/1984.html</td>
<td>(&lt;\text{i}&gt;\text{title}&lt;\text{/i}&gt;\ \text{by author})</td>
</tr>
</tbody>
</table>
Kozareva & Hovy, 2010 *A semi-supervised method to learn and construct taxonomies using the Web*

1. A semi-supervised algorithm that learns hyponym-hypernym pairs subordinated to a root concept
2. Web-based concept positioning procedure used to validate extracted relations
3. A graph algorithm that derives the taxonomy
Extracting hyponym-hypernym relations

**Input** root concept for the target hierarchy, specified as one-seed instance: lions for animals, cucumbers for plants, ...

**Data source** Web documents
## Extracting hyponym-hypernym relations – steps

### gather hyponyms

1. fill in extraction pattern **C such as I and * from known pairs**
2. convert patterns to queries, fetch Web documents
3. gather all terms that instantiate *
4. if new terms have been extracted, go to step 1

### gather hypernyms

1. filter concepts based on \( outDegree(v) = \frac{\sum_{(v, x)} w(v, x)}{|V| - 1} \)
2. fill in pattern **such as \( t_1 \) and \( t_2 \)**
3. convert pattern to query, fetch documents
4. gather all terms that instantiate *
5. rank terms by \( inDegree = \sum_{(t_1 - t_2, h)} w(t_1 - t_2, h) \)
Organize extracted pairs into a hierarchy

1. for each pair, determine the most specific concept – based on instantiated pattern counts $X$ such as $Y$, $X$ including $Y$

2. eliminate edge cycles and transitive closures
Issues in relation extraction

concepts: what terms to link

relation types: what types of relations to target

- *is-a* (taxonomical relations)
- *part-of*
- other relations
Learning from infoboxes

- they provide examples of relations of interest
- the associated articles provide (free and annotated!) training for these relations
- (reused) infobox templates
Creating missing infoboxes

Wu & Weld, 2007 *Autonomously semantifying Wikipedia*
Creating missing infoboxes

- **Preprocessor**
  - identify relevant attributes from articles with the same infobox template
  - generate training data for classification and extraction

- **Document classifier**
  - determine whether an article belongs to a certain class
  - one classifier per class of articles

- **Sentence classifier**
  - determine whether a sentence contains the value of an attribute
  - one classifier per attribute per infobox template

- **Extractors**
  - extract a value from a (marked) sentence
  - aggregate across sentences, return values for attributes
Sentence classifier

**Training data**

For each article with an infobox:

1. split document in sentences
2. for each attribute value find a (unique) corresponding sentence in the article (positive training example)
3. take other sentences as negative training examples

**Features**

- sentence tokens
- tokens’ POS tags

Multi-class classification – Maximum Entropy model
# Learning extractors

<table>
<thead>
<tr>
<th>Feature Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>First token of sentence</td>
<td><em>Hello world</em></td>
</tr>
<tr>
<td>In first half of sentence</td>
<td><em>Hello world</em></td>
</tr>
<tr>
<td>In second half of sentence</td>
<td><em>Hello world</em></td>
</tr>
<tr>
<td>Start with capital</td>
<td><em>Hawaii</em></td>
</tr>
<tr>
<td>Start with capital, end with period</td>
<td><em>Mr.</em></td>
</tr>
<tr>
<td>Single capital</td>
<td><em>A</em></td>
</tr>
<tr>
<td>All capital, end with period</td>
<td><em>CORP.</em></td>
</tr>
<tr>
<td>Contains at least one digit</td>
<td><em>AB3</em></td>
</tr>
<tr>
<td>Made up of two digits</td>
<td><em>99</em></td>
</tr>
<tr>
<td>Made up of four digits</td>
<td><em>1999</em></td>
</tr>
<tr>
<td>Contains a dollar sign</td>
<td><em>$20</em></td>
</tr>
<tr>
<td>Contains an underline symbol</td>
<td><em>km_square</em></td>
</tr>
<tr>
<td>Contains an percentage symbol</td>
<td><em>20%</em></td>
</tr>
<tr>
<td>Stop word</td>
<td><em>the; a; of</em></td>
</tr>
<tr>
<td>Purely numeric</td>
<td><em>1929</em></td>
</tr>
<tr>
<td>Number type</td>
<td><em>1932; 1,234; 5.6</em></td>
</tr>
<tr>
<td>Part of Speech tag</td>
<td></td>
</tr>
<tr>
<td>Token itself</td>
<td></td>
</tr>
<tr>
<td>NP chunking tag</td>
<td></td>
</tr>
<tr>
<td>String normalization: capital to “A”, lowercase to “a”, digit to “1”, others to “0”</td>
<td>(TF - 1 \rightarrow AA01)</td>
</tr>
<tr>
<td>Part of anchor text</td>
<td><em>Machine Learning</em></td>
</tr>
<tr>
<td>Beginning of anchor text</td>
<td><em>Machine Learning</em></td>
</tr>
<tr>
<td>Previous tokens (window size 5)</td>
<td></td>
</tr>
<tr>
<td>Following tokens (window size 5)</td>
<td></td>
</tr>
<tr>
<td>Previous token anchored</td>
<td><em>Machine Learning</em></td>
</tr>
<tr>
<td>Next token anchored</td>
<td><em>Machine Learning</em></td>
</tr>
</tbody>
</table>
Moving to the Web through Wikipedia

Wu & Weld, 2010 *Open information extraction using Wikipedia*

- **Data**
  - Wikipedia articles for acquiring positive examples
  - Web document for finding new relation instances

- **Output**: relational tuples (Arg1-relation-Arg2)
Extraction components

Preprocessing Wikipedia articles
- sentence splitting
- POS tagging
- syntactic parsing

Infobox entries matcher
- find sentences that contains the article title (\textbf{Arg1}) and the value of the infobox attribute (\textbf{Arg2})
- apply filters and heuristics to improve matching accuracy
Extraction components – continued

Learner

- **deep**
  - extract the syntactic path that connects Arg1 and Arg2 from each matching sentence
  - collect and generalize unlexicalized patterns

- **shallow**
  - collect and generalize POS and lexical context

exploit deep (with parsing) and shallow (no parsing) patterns to extract tuples from Web documents
Learning patterns from Wikipedia

Infobox of Wikipedia article

- Founded: March 26, 1804
- Seat: Clearfield
- Largest city: DuBois
- Area: Total 1,154 sq mi (2,989 km²)

Text of Wikipedia article

Clearfield County was created in 1804, from parts of Huntingdon and Lycoming Counties but was administered as part of Centre County until 1812 [..]

Clearfield County was created in 1804

Arg1: NNP
Arg2: NNP
Rel: VBD

- prep-in

Clearfield County was created in 1804

Arg1: NNP
Arg2: NNP
Rel: VBN

- prep-in

Arg1: N
Arg2: N
Rel: IN

Frequency of pattern

Weight of pattern

Across all patterns

\[ w(p) = \frac{\max(\log(f_p) - \log(f_{\text{min}}), 0)}{\log(f_{\text{max}}) - \log(f_{\text{min}})} \]
Relation extraction from the Web

Pantel & Pennacchiotti, 2006 Espresso: Leveraging generic patterns for automatically harvesting semantic relations

- **Input**: target relation, as small sets of seed pairs
  - (nitrogen, element), (wheat, crop) for IsA
  - (city, region), (hand, body) for PartOf

- **Data sources**: corpora / Web documents

- **Output**: ranked lists of relations

- **Approach**: bootstrapping
Relation extraction from the Web

(Courtesy Pantel & Pennacchiotti)
Pattern Induction

Sentence retrieval
- match input seed relations to sentences

Sentence generalization
- “Because/IN HF/NNP is/VBZ a/DT weak/JJ acid/NN and/CC ...”
- “Because/IN <TR> is/VBZ a/DT <TR> and/CC ...”

Frequency count
- count frequency of occurrence of each pattern
Pattern ranking

Rank patterns according to reliability

\[ r_\pi(p) = \frac{\sum_{i \in I'} \left( \frac{pmi(i, p) \times r_i(i)}{\max_{pmi} \left| \frac{p}{i} \right|} \right)}{|I'|} \]

Strength of association between pattern p and input relation i

Reliability of input relation i

Cardinality of set of input relations

\[ pmi(i, p) = \log \frac{|x, p, y|}{|x, *, y| |*, p, *|} \]
Relation extraction

- match patterns to sentences in the document collection
- if low-redundancy matches, expand:
  - convert patterns to queries:
    - \[(\text{italy,country}) \cup C \text{ such as I} \] \rightarrow \text{country such as } *
    - \[(\text{european country, location}) \rightarrow (\text{country, location})\]
Rank extracted relations

Strength of association between instance $i$ and input pattern $p$

Reliability of input pattern $p$

Cardinality of set of input patterns

$$ r_t(i) = \frac{\sum_{p \in P'} \frac{pmi(i, p) \times r_\pi(p)}{\max_{pmi} |P'|}}{\max_{pmi} |P'|} $$

Select top relations
Fact extraction from queries

Pasca, 2007 *Organizing and searching the World Wide Web of facts - Step two: harnessing the wisdom of the crowds*

- **Input**
  - target classes, as sets of seeds: e.g. for *Company* – Honda, Oracle, Reuters, ...
  - seed attributes: e.g. for *Company* – headquarters, stock price, ceo, ...

- **Data** – anonymized search queries and their frequencies

- **Output** – ranked list of attributes, one per class
Extraction from queries

1. select candidate attributes from queries containing an instance
2. create internal representation of candidate attributes, from queries containing an instance and a candidate attribute
3. rank candidate attributes, from similarity between internal representation of a candidate attribute and combined internal representation of all seed attributes

Example attributes

<table>
<thead>
<tr>
<th></th>
<th>actor</th>
<th>aircraft model</th>
<th>award</th>
<th>basic food</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>awards, height, age, date of birth, weight, ...</td>
<td>weight, length, history, fuel consumption, ...</td>
<td>recipients, date, winners list, result, gossip, printable ballot ...</td>
<td>calories, color, size, allergies, taste, carbs, nutritional information, ...</td>
</tr>
</tbody>
</table>
**Idea**

- \( CW \): content words – frequency < \( F_C \)
- \( HFW \): high frequency words – frequency > \( F_H \)
  
  [Prefix] \( CW_1 \) [Infix] \( CW_2 \) [Postfix]

- Prefix, Infix, PostFix \( \sim HFW + \)

**Focusing the extraction**

- one of \( CW_i \) is a hook (seed) word, the other is the target
- filter documents to those that contain the hook word (hook corpus)
- sort targets by PMI relative to the hook
- use various hook words
Pattern clustering

1. cluster patterns that share both $CW_i$s
2. merge clusters that share $x\%$ of their patterns
3. remove patterns generated from a single hook corpus (force generality)
4. iteratively merge clusters by looking at shared patterns ($P_{core}$)
5. remove clusters that don’t share patterns (contain only $P_{unconf}$)

Cluster labels – top 5 pairs according to:

$$\text{Hits}(C, (w_1, w_2)) =$$

$$\frac{|\{p; (w_1, w_2) \text{ appears in } p \in P_{core}\}|}{|P_{core}|} + \alpha \frac{|\{p; (w_1, w_2) \text{ appears in } p \in P_{unconf}\}|}{|P_{unconf}|}$$
Clusters and labels

- such X as Y
- X such as Y
- Y and other X

---

- buy Y accessory for X!
- shipping Y for X

(pets, dogs)

(phone, charger)
Next week: large scale knowledge acquisition from the web

- Never Ending Language Learning (NELL) (CMU)
- KnowItAll by Machine Reading the World Wide Web (UofW)