

Semantic Class Learning from the Web

- The Web can be viewed as an *enormous* text collection and source for knowledge acquisition.
- Some research focuses on extracting knowledge from structured lists and tables.
- NLP techniques can be used to extract knowledge from natural language text on the Web.
- The enormity of the Web requires shallow text processing, typically with pattern matching, to identify and analyze relevant text snippets.

Hyponym pattern mining

[Hearst 1992] proposed the idea of applying **hyponym patterns** to text to find category members:

*The **bow lute**, such as the **Bambara ndang**, is plucked...*

Several hyponym patterns were suggested:

Hypernym such as *
Hypernym including *
Hypernym especially *
Hyponym and/or other *

Examples:

Works by **authors** such as **Shakespeare** ...
*Scent hounds, including **beagles**, are good at ...*
*Many **European countries**, especially **Spain**, ...*
***Bruises, broken bones**, and other **injuries** ...*

Semantic Taxonomies

- **Long-term goal:** automatically create and populate a large-scale semantic network by mining Web text.
- Ideally, we'd like a rich semantic ontology with many different types of semantic relationships.
- The most studied type of categorical knowledge is hierarchical **hypernym/hyponym** relations.

Hypernym = superordinate semantic category
Hyponym = subordinate semantic category

Examples: *mammal* is a hypernym of *dog*
dog is a hyponym of *mammal*
dog is a hypernym of *beagle*

Extracting Phrases

The * position frequently reveals multi-word phrases that must be extracted. For example:

*for **artists** such as **Picasso***
*for **artists** such as **Pablo Picasso***
*for **artists** such as **Pablo Ruiz Picasso***
*for **artists** such as **painter Pablo Picasso***
*for **artists** such as **20th century painter Pablo Picasso***

The entire text snippet that matches a hyponym pattern is saved and then a phrase is extracted.

Ideally, parsing would be helpful, but web text can be challenging to parse.

Doubly-anchored hyponym pattern

[Kozareva et al., ACL 2008]

Kozareva proposed the idea of using a **doubly-anchored hyponym pattern (DAP)** that includes both a class name and one class member that begins a conjunction:

*ClassName such as ClassMember and **

Examples:

artists such as Picasso and *

dogs such as terriers and *

countries such as France and *

The Power of the DAP

By including a class member in the pattern, ambiguities are usually resolved.

For example:

*languages such as English and **
*languages such as Java and **

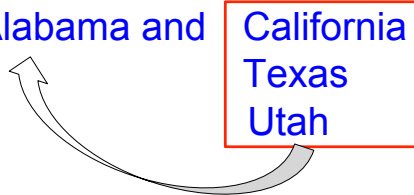
*presidents such as Ford and **
*companies such as Ford and **

*presidents such as Bill Clinton and **
*presidents such as Bill Gates and **

Reckless bootstrapping

Naive Approach: instantiate a DAP with one ClassName and one Member, extract new class members, and bootstrap via breadth-first search.

states such as Alabama and California
Texas
Utah



For proper name classes, all adjacent capitalized words are extracted. Otherwise, just one word is extracted (if it's not capitalized).

Evaluation

- Four semantic classes:
 - closed
 - countries (194 elements)
 - U.S. states (50 elements)
 - open
 - fishes (gold standard is Wikipedia)
 - singers (manually reviewed)
- Evaluated the performance of each class with five randomly selected seeds and reported the average performance.

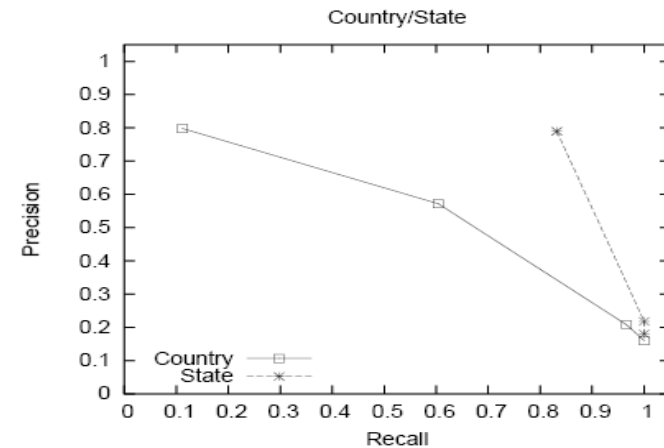
Precision of reckless bootstrapping

Iter.	countries	states	singers	fish
1	.80	.79	.91	.76
2	.57	.21	.87	.64
3	.21	.18	.86	.54
4	.16	-	.83	.54

Problem: search needs guidance

Solution: evaluate and rank the learned instances

Performance of Reckless Bootstrapping



Challenges in Extracting Correct Phrases

Adjacent Phrases

for many *artists* such as *Picasso Europe* is ...

Conjunctions

companies such as *Abercrombie* and *Fitch*...
 some birds and *reptiles*, such as *parrots* and *iguanas* ...

Lexicalized Phrases

some hot *dogs* such as *Oscar Mayer* are made...

Prepositional Phrases

many diseases in *dogs* including *parvovirus* ...

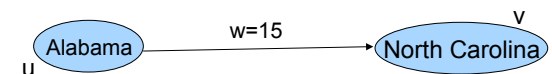
Web Issues

broken words (*Merce -dez*)
 incomplete snippets

Hyponym pattern linkage graphs

HPLG=(V,E) where vertex $v \in V$ is an instance, and $e \in E$ is an edge between two instances

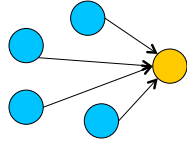
Some *states*, such as *Alabama* and *North Carolina*, offer a list of approved health care providers...



The weight w of an edge is the frequency with which u generated v .

Popularity

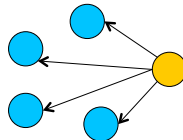
- Measure the *popularity* of a term as the ability of a class member to be discovered by other class members



- The highest scoring unexplored node is learned during each iteration.
- The graph can grow dynamically during the bootstrapping process

Productivity

- Measure the *productivity* of a term as the ability of a class member to discover other class members.
- Intuition: if a term is truly a class member, then it should co-occur with other class members in the pattern.



- Requires a precompiled graph:
 - Perform reckless bootstrapping (exhaustively)
 - Re-rank the learned terms based on graph properties.

Popularity ranking measures

- in-Degree:** $\text{inD}(v)$ is the sum of the weights of all incoming edges (u,v) , where u is a trusted member.
- Best edge:** $\text{BE}(v)$ is the maximum edge weight among the incoming edges (u,v) , where u is a trusted member.

- Key Player Problem:**

$$KPP(v) = \frac{\sum_{u \in \mathcal{V}} \frac{1}{d(u,v)}}{|\mathcal{V}| - 1}$$

$d(u,v)$ is the shortest path between u and v

High KPP indicates strong connectivity and proximity to other nodes

Productivity ranking measures

- OutDegree:** $\text{outD}(v)$ is the sum of all outgoing edges from v normalized by $|\mathcal{V}| - 1$
- TotalDegree:** $\text{totD}(v)$ is the sum of inDegree and outDegree edges of v , normalized by $|\mathcal{V}| - 1$

- Betweenness:**

$$BE(v) = \sum_{\substack{s \neq v \neq t \in \mathcal{V} \\ s \neq t}} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

σ_{st} is the number of shortest paths from s to t , and $\sigma_{st}(v)$ is the number of shortest paths from s to t that pass through v

- PageRank:**

$$PR(v) = \frac{(1-\alpha)}{|\mathcal{V}|} + \alpha \sum_{u,v \in \mathcal{E}} \frac{PR(u)}{\text{outD}(u)}$$

Performance

number of learned instances →	States						
	N	Popularity			Pop&Prd		
	BE	KPP	inD	totD	BT	PR	
25	1.0	1.0	1.0	1.0	.88	.88	
50	.96	.98	.98	1.0	.86	.82	
64	.77	.78	.77	.78	.77	.67	

dynamic graph →
precompiled graph →

BE – best edge
KPP – key player problem
inD – in-Degree
totD – total degree
BT – betweenness
PR – Page Rank

Performance

	States						
	Popularity			Pop&Prd			Prd
N	BE	KPP	inD	totD	BT	PR	outD
25	1.0	1.0	1.0	1.0	.88	.88	1.0
50	.96	.98	.98	1.0	.86	.82	1.0
64	.77	.78	.77	.78	.77	.67	.78

- HPLGs perform much better than reckless bootstrapping!
- outD and totD discovered all 50 U.S. states.

But there are only 50 states, so why does the algorithm learn 64?

Investigating the Extra States

The additional 14 learned "states" were:

Russia, Ukraine, Uzbekistan, Azerbaijan, Moldova, Tajikistan, Armenia, Moldavia

Chicago, Boston, Atlanta, Detroit, Philadelphia, Tampa

Authoritarian former Soviet **states** such as *Georgia* and *Ukraine* ...

Findlay now has over 20 restaurants in **states** such as *Florida* and *Chicago* ...

Full Results

Countries		
	Pop	Prd
N	inD	outD
50	.98	1.0
100	.94	1.0
150	.91	1.0
200	.83	.90
300	.61	.61
323	.57	.57

Singers		
	Pop	Prd
N	inD	outD
10	.92	1.0
25	.91	1.0
50	.92	.97
75	.91	.96
100	.89	.96
150	.88	.95
180	.87	.91

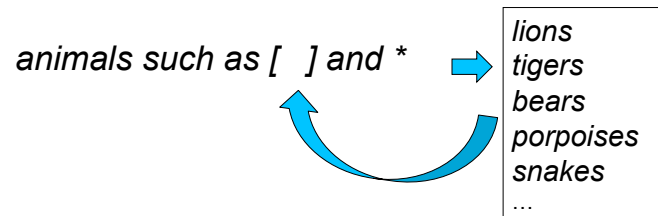
Fish		
	Pop	Prd
N	KPP	outD
10	.90	1.0
25	.88	1.0
50	.80	1.0
75	.69	.93
100	.68	.84
116	.65	.80

Error analysis

- Type 1: incorrect proper name extraction
- Type 2: instances that formerly belonged to the semantic class
- Type 3: spelling variants
- Type 4: sentences with wrong factual assertions
- Type 5: broken expressions

Step 1: Hyponym Acquisition

- The first step is the original bootstrapping process for hyponym learning.
- The learned instances are cycled back into the pattern to generate more instances:



Learning both Hypernyms and Hyponyms

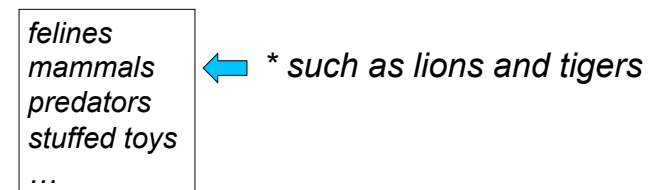
[Hovy et al., EMNLP 2009]

- The ultimate goal is to create a semantic taxonomy that is richly organized and represents a “structure justified by evidence drawn from text”.
- Also, learning from a single hypernym will always be limited, so how can we learn more?
- Idea:** the doubly-anchored hyponym pattern can also be used to extract new hypernym terms.
- The bootstrapping process alternates between learning a set of hyponyms and then learning a new hypernym.

Step 2: Hypernym Acquisition

Next, we use DAP⁻¹ to acquire conceptual terms that are superordinate to the hyponyms:

* such as Member1 and Member2



Ranking Hypernyms

- All pairs of class members found in the DAP are saved (e.g., <lions, tigers>).
- DAP⁻¹ is instantiated with all member pairs, and candidate hypernyms are extracted.
- A bipartite graph is constructed with category vertices (V_c) for the candidate hypernyms, and member pair vertices (V_{mp}) for the hyponym pairs.
- An edge is created between each hypernym that extracted a hyponym pair, with the frequency as the edge weight.
- The InDegree popularity measure is used to rank the hypernyms.

Step 3: Concept Positioning Test

A **Concept Positioning Test** is applied to determine whether a learned hypernym is more or less general than the original semantic category.

- (a) <Hypernym> such as <RootConcept> and *
- (b) <RootConcept> such as <Hypernym> and *

The candidate hypernym is selected only if:

- (b) produces at least 50 hits, and
- (b) returns at least 4 times as many hits as (a)

Hypernym selection: we apply the CPT to the ranked list of hypernyms. The first hypernym that satisfies this test is chosen for expansion in the next bootstrapping cycle.

Problem: Overly General Hypernyms

Problem: some learned hypernyms are more general than the original semantic category (e.g., *species*), so bootstrapping wanders into a broader conceptual space.

Idea: we can use the DAP to determine whether one conceptual term is more general than another.

- (1) X such as Y and *
- (2) Y such as X and *

If (1) produces more hits than (2), then X is more general than Y.

Animals such as *mammals* and *
Mammals such as *animals* and *

Data Collection

Two semantic categories: Animals & People

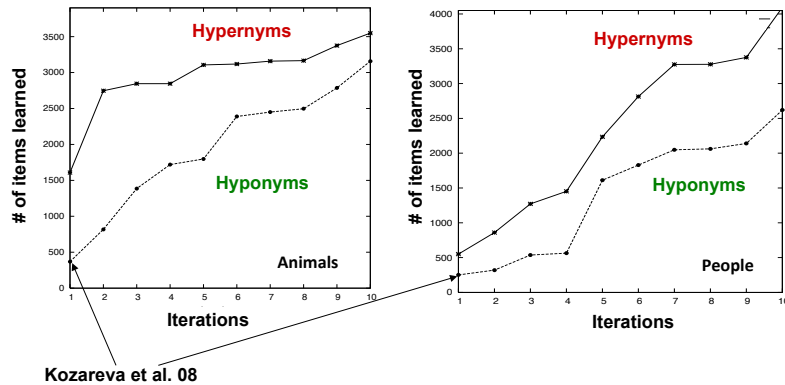
Animal Seed: lion

People Seed: Madonna

Procedure:

- Sent DAP and DAP⁻¹ queries to Google
- Collected 1000 snippets per query, kept only unique answers (counting freqs)
- Algorithm ran for 10 iterations:
- Produced 1.1 GB of snippets for Animals and 1.5 GB for People

Learning Curves



Evaluation of Hyponyms

Animals (evaluated against lists compiled from websites)

Iteration	1	2	3	4	5	6	7	8	9	10
Accuracy	0.79	0.79	0.78	0.70	0.68	0.68	0.67	0.67	0.68	0.71
# Inst.	396	448	453	592	663	708	745	755	770	913

People (human judges)

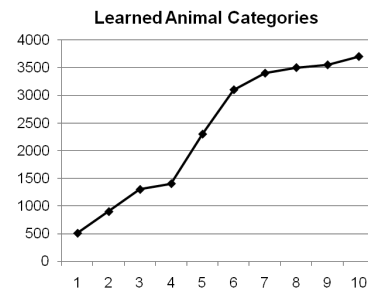
	Judge 1	Judge 2	Judge 3
Person	190	192	189
NotPerson	10	8	11
Accuracy	0.95	0.96	0.95

Examples of Learned Categories

- Animal categories:

accessories, activities, agents, amphibians, animal groups, animal life, amphibians, apes, arachnids, area, ..., felines, fish, fishes, food, fowl, game, game animals, grazers, grazing animals, grazing mammals, herbivores, herd animals, household pests, household pets, house pets, humans, hunters, insectivores, insects, invertebrates, laboratory animals, ..., water animals, wetlands, zoo animals

- Growth doesn't top out!
- Collection growth curve:



How to Evaluate Categories?

- Produced a staggering variety of concept terms!

Examples of Learned Intermediate Concepts for

Animals: *amphibians, arachnids, area, felines, fishes, food, fowl, grazers, herbivores, herd animals, hunters, insectivores, invertebrates, laboratory animals, water animals, wetlands, zoo animals, ...*

- Much more diverse than expected.
 - Probably useful: *laboratory animals, forest dwellers, endangered species*
 - Maybe useful: *bait, allergens, seafood, vectors, protein, pests, vermin*
 - Relative concepts: *native animals, large mammals*

Conclusions

- All experiments were conducted with DAP and DAP⁻¹ starting with only with one *RootConcept* and one *Seed Instance*
- The DAP is simple, yet very powerful.
- The bootstrapping algorithm serves multiple purposes:
 - generates highly accurate, rich and diverse lists of concepts
 - finds instances and intermediate concepts that are missing from WordNet
 - learns partial taxonomic structures
- **Concept evaluation and organization is challenging even for humans.**