Lecture 4: Unsupervised Word-sense Disambiguation

Lexical Semantics and Discourse Processing
MPhil in Advanced Computer Science

Simone Teufel

Natural Language and Information Processing (NLIP) Group
UNIVERSITY OF CAMBRIDGE
Simone.Teufel@cl.cam.ac.uk

Slides after Frank Keller

February 2, 2011

1. Bootstrapping
   - Heuristics
   - Seed Set
   - Classification
   - Generalization

2. Graph-based WSD
   - Introduction
   - Graph Construction
   - Graph Connectivity
   - Evaluation

Reading: Yarowsky (1995), Navigli and Lapata (2010).

Heuristics

Yarowsky's (1995) algorithm uses two powerful heuristics for WSD:

- **One sense per collocation:** nearby words provide clues to the sense of the target word, conditional on distance, order, syntactic relationship.
- **One sense per discourse:** the sense of a target words is consistent within a given document.

The Yarowsky algorithm is a **bootstrapping** algorithm, i.e., it requires a small amount of annotated data.

Figures and tables in this section from Yarowsky (1995).

Seed Set

Step 1: Extract all instances of a polysemous or homonymous word.

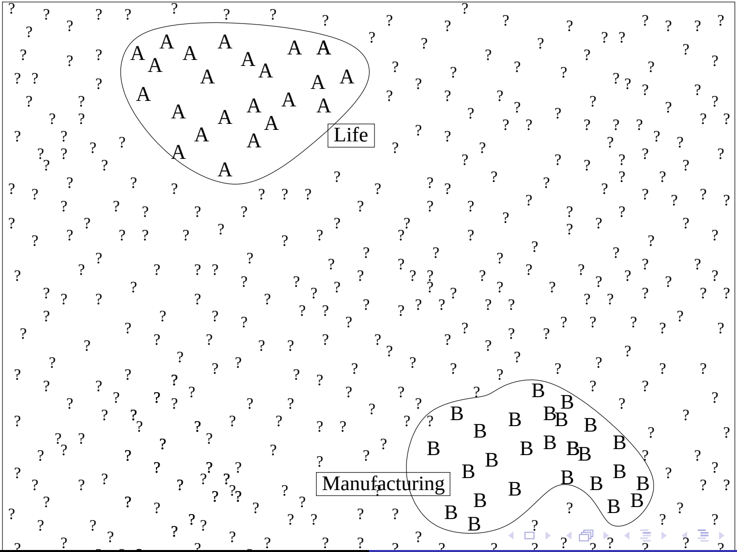
Step 2: Generate a seed set of labeled examples:

- either by manually labeling them;
- or by using a reliable heuristic.

Example: target word *plant*: As seed set take all instances of

- *plant life* (sense A) and
- *manufacturing plant* (sense B).

Seed Set



Classification

Step 3a: Train classifier on the seed set.

Step 3b: Apply classifier to the entire sample set. Add those examples that are classified reliably (probability above a threshold) to the seed set.

Yarowsky uses a **decision list** classifier:

- rules of the form: collocation $\rightarrow$ sense
- rules are ordered by log-likelihood:

$$\log \frac{P(sense_A | collocation_i)}{P(sense_B | collocation_i)}$$

- classification is based on the first rule that applies.

Classification

LogL	Collocation	Sense
8.10	*plant* life	$\rightarrow$ A
7.58	manufacturing *plant*	$\rightarrow$ B
7.39	life (within +-2-10 words)	$\rightarrow$ A
7.20	manufacturing (in +- 2-10 words)	$\rightarrow$ B
6.27	animal (within +-2-10 words)	$\rightarrow$ A
4.70	equipment (within +-2-10 words)	$\rightarrow$ B
4.39	employee (within +-2-10 words)	$\rightarrow$ B
4.30	assembly *plant*	$\rightarrow$ B
4.10	*plant* closure	$\rightarrow$ B
3.52	*plant* species	$\rightarrow$ A
3.48	automate (within +-10 words)	$\rightarrow$ B
3.45	microscopic *plant*	$\rightarrow$ A
	...	

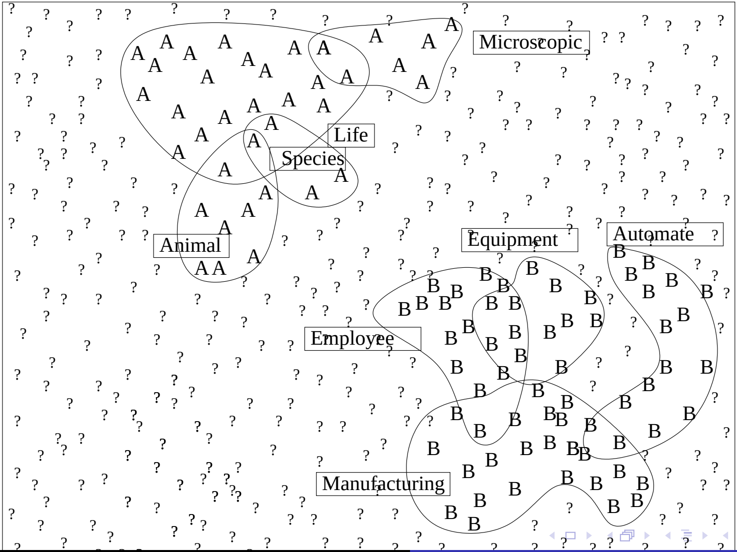
Classification

Step 3c: Use one-sense-per-discourse constraint to filter newly classified examples:

- If several examples have already been annotated as sense A, then extend this to all examples of the word in the discourse.
- This can form a bridge to new collocations, and correct erroneously labeled examples.

Step 3d: repeat Steps 3a–d.

Classification



Generalization

Step 4: Algorithm converges on a stable residual set (remaining unlabeled instances):

- most training examples will now exhibit multiple collocations indicative of the same sense;
- decision list procedure uses only the most reliable rule, not a combination of rules.

Step 5: The final classifier can now be applied to unseen data.

Discussion

Strengths:

- simple algorithm that uses only minimal features (words in the context of the target word);
- minimal effort required to create seed set;
- does not rely on dictionary or other external knowledge.

Weaknesses:

- uses very simple classifier (but could replace it with a more state-of-the-art one);
- not fully unsupervised: requires seed data;
- does not make use of the structure of the sense inventory.

Alternative: **graph-based algorithms** exploit the structure of the sense inventory for WSD.

Introduction

Navigli and Lapata's (2010) algorithm is an example of graph-based WSD.

It exploits the fact that **sense inventories** have internal structure.

Example: synsets (senses) of *drink* in Wordnet:

(1)

a. $\{drink_v^1,\ imbibe_v^3\}$
b. $\{drink_v^2,\ booze_v^1,\ fuddle_v^2\}$
c. $\{toast_v^2,\ drink_v^3,\ pledge_v^2,\ salute_v^1,\ wassail_v^2\}$
d. $\{drink\ in_v^1,\ drink_v^4\}$
e. $\{drink_v^5,\ tope_v^1\}$

Figures and tables in this section from Navigli and Lapata (2010).

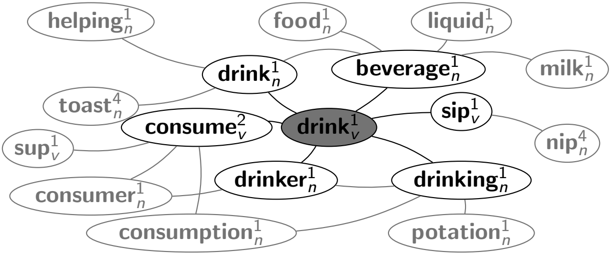
WN as a graph

We can represent Wordnet as a **graph whose nodes are synsets and whose edges are relations between synsets.**

Note that the edges are not labeled, i.e., the type of relation between the nodes is ignored.

Introduction

Example: graph for the first sense of *drink*.



Graph Construction

Disambiguation algorithm:

1. Use the Wordnet graph to construct a graph that incorporates each content word in the sentence to be disambiguated;
2. Rank each node in the sentence graph according to its importance using **graph connectivity measures;**
3. For each content word, pick the highest ranked sense as the correct sense of the word.

Graph Construction

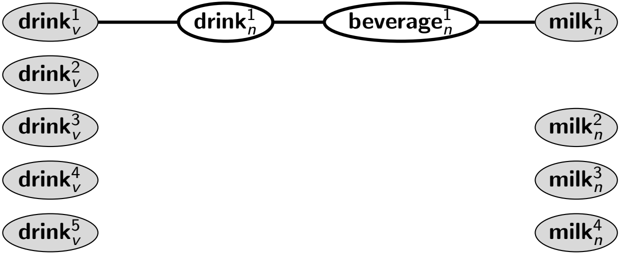
Given a word sequence $\sigma = (w_1, w_2, \ldots, w_n)$, the graph G is constructed as follows:

1. Let $V_\sigma := \bigcup_{i=1}^{n} Senses(w_i)$ denote all possible word senses in σ. We set $V := V_\sigma$ and $E := \emptyset$.
2. For each node $v \in V_\sigma$, we perform a depth-first search (DFS) of the Wordnet graph: every time we encounter a node $v' \in V_\sigma$ $(v' \neq v)$ along a path $v \rightarrow v_1 \rightarrow \cdots \rightarrow v_k \rightarrow v'$ of length L, we add all intermediate nodes and edges on the path from v to v': $V := V \cup \{v_1, \ldots, v_k\}$ and $E := E \cup \{\{v, v_1\}, \ldots, \{v_k, v'\}\}$.

For tractability, we fix the maximum path length at 6.

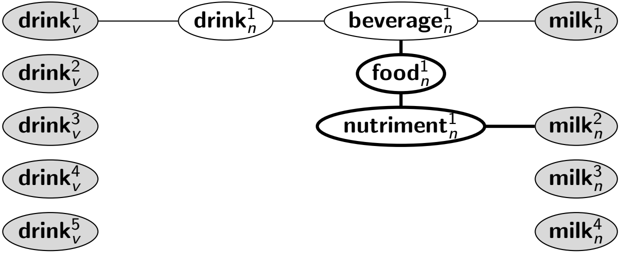
Graph Construction

Example: graph for *drink milk*.



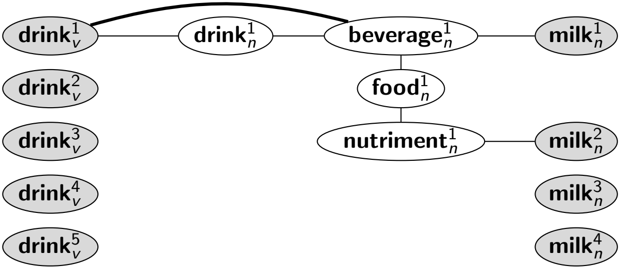
Graph Construction

Example: graph for *drink milk*.



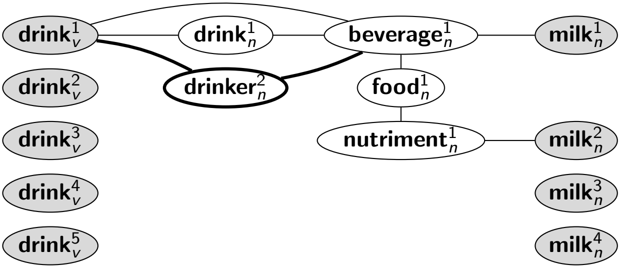
Graph Construction

Example: graph for *drink milk*.



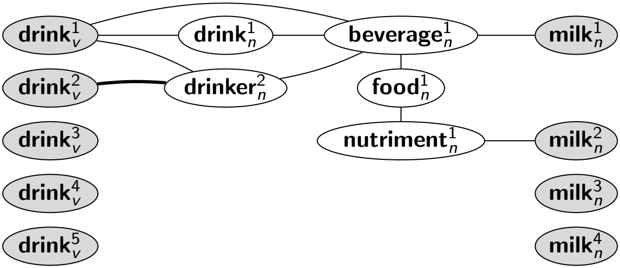
Graph Construction

Example: graph for *drink milk*.



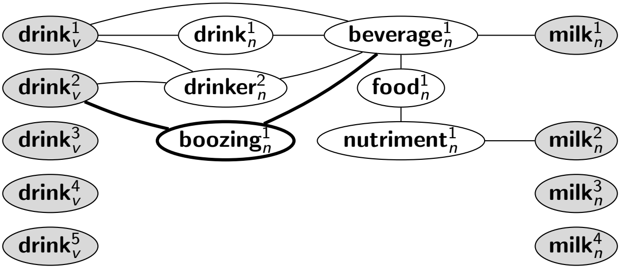
Graph Construction

Example: graph for *drink milk*.



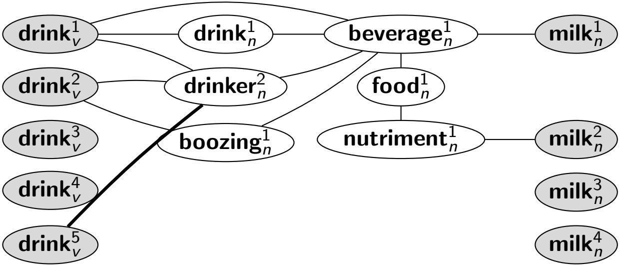
Graph Construction

Example: graph for *drink milk*.



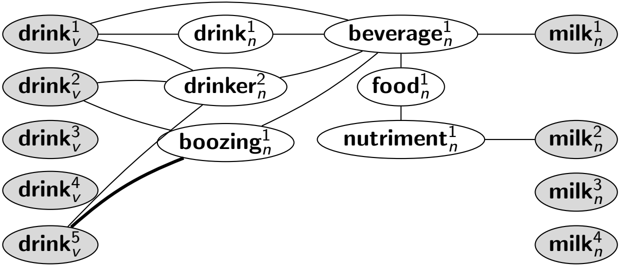
Graph Construction

Example: graph for *drink milk*.



Graph Construction

Example: graph for *drink milk*.



We get $3 \cdot 2 = 6$ interpretations, i.e., subgraphs obtained when only considering one connected sense of *drink* and *milk*.

Graph Connectivity

Once we have the graph, we pick the most connected node for each word as the correct sense. Two types of connectivity measures:

- **Local measures:** gives a connectivity score to an individual node in the graph; use this directly to pick a sense;
- **Global measures:** assigns a connectivity score the to the graph as a whole; apply the measure to each interpretation and select the highest scoring one.

Navigli and Lapata (2010) discuss a large number of graph connectivity measures; we will focus on the most important ones.

Degree Centrality

Assume a graph with nodes V and edges E. Then the **degree** of $v \in V$ is the number of edges terminating in it:

$$deg(v) = |\{\{u, v\} \in E : u \in V\}| \tag{1}$$

Degree centrality is the degree of a node normalized by the maximum degree:

$$C_D(v) = \frac{deg(v)}{|V| - 1} \tag{2}$$

For the previous example, $C_D(drink_v^1) = \frac{3}{14}$, $C_D(drink_v^2) = C_D(drink_v^5) = \frac{2}{14}$, and $C_D(milk_n^1) = C_D(milk_n^2) = \frac{1}{14}$. So we pick $drink_v^1$, while $milk_n$ is tied.

Edge Density

The **edge density** of a graph is the number of edges compared to a complete graph with $|V|$ nodes (given by $\binom{|V|}{2}$):

$$ED(G) = \frac{|E(G)|}{\binom{|V|}{2}} \tag{3}$$

The first interpretation of **drink milk** has $ED(G) = \frac{6}{\binom{5}{2}} = \frac{6}{10} = 0.60$, the second one $ED(G) = \frac{5}{\binom{5}{2}} = \frac{5}{10} = 0.50$.

Evaluation on SemCor

	Measure	WordNet		EnWordNet	
		All	Poly	All	Poly
	Random	39.13	23.42	39.13	23.42
	ExtLesk	47.85	34.05	48.75	35.25
Local	**Degree**	**50.01**	**37.80**	**56.62**	**46.03**
Local	PageRank	49.76	37.49	56.46	45.83
Local	HITS	44.29	30.69	52.40	40.78
Local	KPP	47.89	35.16	55.65	44.82
Local	Betweenness	48.72	36.20	56.48	45.85
Global	Compactness	43.53	29.74	48.31	35.68
Global	Graph Entropy	42.98	29.06	43.06	29.16
Global	Edge Density	43.54	29.76	52.16	40.48
	First Sense	74.17	68.80	74.17	68.80

Evaluation on Semeval All-words Data

System	F
Best Unsupervised (Sussex)	45.8
ExtLesk	43.1
Degree Unsupervised	52.9
Best Semi-supervised (IRST-DDD)	56.7
Degree Semi-Unsupervised	60.7
First Sense	62.4
Best Supervised (GAMBL)	65.2

Discussion

Strengths:

- exploits the structure of the sense inventory/dictionary;
- conceptually simple, doesn't require any training data, not even a seed set;
- achieves good performance for unsupervised system.

Weaknesses:

- performance not good enough for real applications (F-score of 53 on Semeval);
- sense inventories take a lot of effort to create (Wordnet has been under development for more than 15 years).

Summary

- The Yarowsky algorithm uses two key heuristics:
  - one sense per collocation;
  - one sense per discourse;
- It starts with a small seed set, trains a classifier on it, and then applies it to the whole data set (bootstrapping);
- Reliable examples are kept, and the classifier is re-trained.
- **Unsupervised graph-based WSD** is an alternative, where the connectivity of the sense inventory is exploited.
- A graph is constructed that represents the possible interpretations of a sentence; the nodes with the highest connectivity are picked as correct senses;
- A range of connectivity measures exists, simple degree is best.

References

Yarowsky (1995): Unsupervised Word Sense Disambiguation rivaling Supervised Methods. Proceedings of the ACL.

Navigli and Lapata (2010): An Experimental Study of Graph Connectivity for Unsupervised Word Sense Disambiguation. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 32(4), IEEE Press, 2010, pp. 678-692.