# Computational Semantics and Pragmatics

Autumn 2012

Raquel Fernández
Institute for Logic, Language & Computation
University of Amsterdam

# Today: WSD

WSD – the task of assigning a sense to a token word in a given context – is a classic task in NLP ("AI-complete problem")

Its history is parallel to the history of NLP:

- research on WSD began in the 40's and 50's in connection to Machine Translation – it was a bottleneck for MT in the 60's
- the 70's were dominated by rule-based approaches
- the creation of digital lexical resources in th 80's (i.e WordNet) was a turning point for WSD
- since the 90's there has been a massive use of statistical / machine learning methods
- in the second half of the 90's evaluation methods became very important – the Senseval campaign was launched in 1998

Term used in psycholinguistics: lexical ambiguity resolution

What sense of a word is being activated by the use of the word in a given context?

From Weaver (1955) in the context of machine translation:

> *If one examines the words in a book, one at a time as through an opaque mask with a hole in it one word wide, then it is obviously impossible to determine, one at a time, the meaning of the words [...] But if one lengthens the slit in the opaque mask, until one can see not only the central word in question but also say $N$ words on either side, then if $N$ is large enough one can unambiguously decide the meaning of the central word [...] The practical question is: "What minimum value of $N$ will, at least in a tolerable fraction of cases, lead to the correct choice of meaning for the central word?"*

# Key elements of WSD

- Word senses
  - enumerative vs. generative approach
  - most work on WSD adopts an enumerative approach
- Context
  - local, global, shallow, syntactic, ...
- Extra knowledge sources
  - dictionaries, ontologies, ...

Existing methods can be classified according to two dimensions:

- Knowledge:
  - knowledge-rich: dictionaries, ontologies, ...
  - knowledge-poor or corpus-based
- Supervision
  - supervised: learning from sense-tagged training data
  - unsupervised: unlabeled data

# Supervised Corpus-based Approaches

Most approaches see WSD as a classification task, where

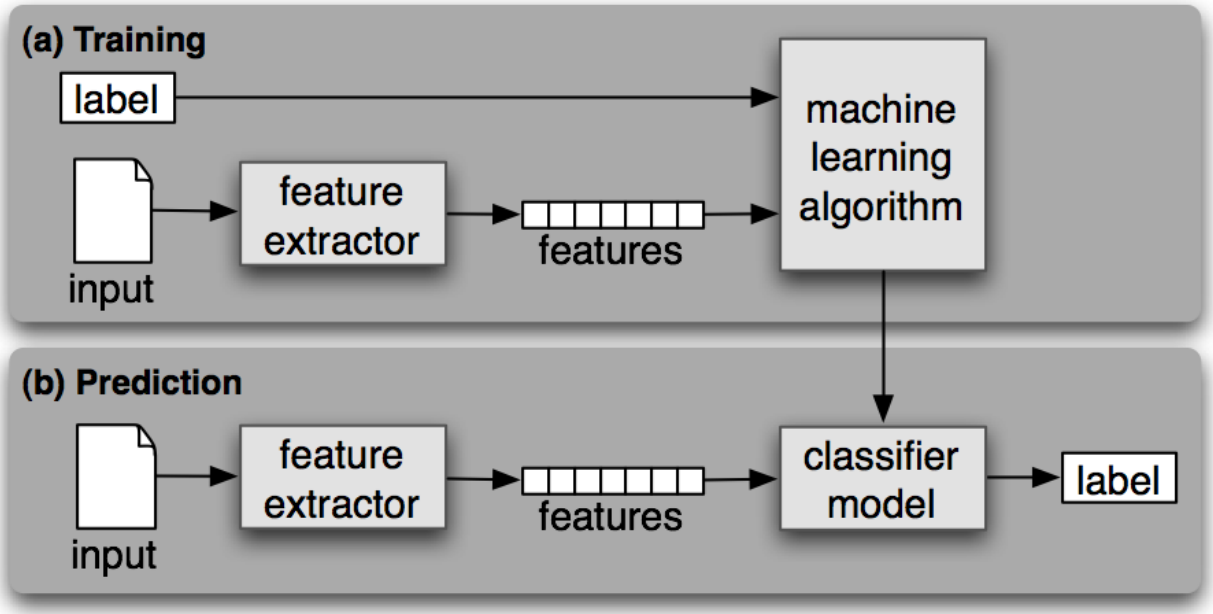
- word occurrences are the items to be classified
- word senses are the classes
- each item is represented as feature vector encoding evidence from the context or external knowledge sources
- an automatic classification algorithm is used to assign one or more classes to each item based on information provided by the features

Note that unlike other NL classification tasks, in WSD the set of classes typically changes for each item.

A classifier is called supervised if it is built based on training corpora containing the correct label for each item.
Sense-tagged corpora:

- SemCor: 234k words from Brown Corpus tagged with WordNet senses
- SensEval data sets

# Supervised Approaches




Figures from the NLTK Book. Chapter 6 *Learning to Classify Text* provides a very clear and gentle introduction to supervised machine learning for natural language tasks.

More advanced but still accessible sources of information:
Manning & Schütze (1999) *Foundations of Statistical Natural Language Processing*, MIT Press.
Witten, Frank & Hall (2011) *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann.

# Features for Supervised WSD

Two common types of features that aim at capturing aspects of the context of a target word occurrence:

- Collocational features: information about words in specific positions with respect to the target word
- Co-occurrence features or bag-of-words: information about the frequency of co-occurrence of the target word with other pre-selected words within a context window ignoring position

# Features for Supervised WSD: Example

For instance, consider the following example sentence with target word $w_i$ = **bass**:

> An electric guitar and **bass** player stand off to one side, not really part of the scene, just as a sort of nod to gringo expectations perhaps.

- Example of possible collocational features:

| $w_{i-2}$, POS$w_{i-2}$, | $w_{i-1}$, POS$w_{i-1}$, | $w_{i+1}$, POS$w_{i+1}$, | $w_{i+2}$, POS$w_{i+2}$ |
|---|---|---|---|
| ⟨ guitar, N, | and, C, | player, N, | stand, V ⟩ |

- Example of possible bag-of-words features:

| fishing, | big, | sound, | player, | fly, | rod, | pound, | double, | guitar, | band |
|---|---|---|---|---|---|---|---|---|---|
| ⟨ 0, | 0, | 0, | 1, | 0, | 0, | 0, | 0, | 1, | 0 ⟩ |

Most approaches use both types of features combined in one long vector.

# Learning Methods

Pretty much any supervised machine learning method has been used for WSD: Naive Bayes, Maximum Entropy, Decision Trees, Support Vector Machines, Neural Networks, etc.

Manning & Schütze (1999) *Foundations of Statistical Natural Language Processing*, MIT Press.
Witten, Frank & Hall (2011) *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann.

# Evaluation

Two types of evaluation:

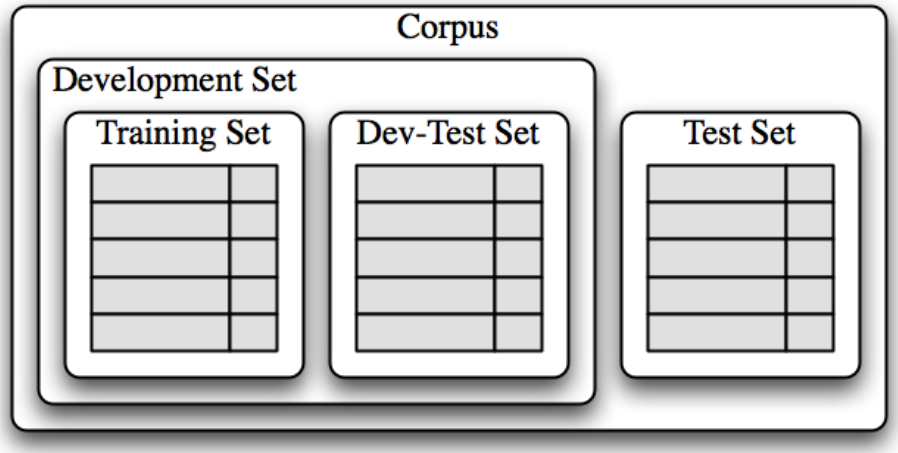
- intrinsic / in vitro / stand-alone:
  evaluation as an independent task.
- extrinsic / in vivo / task-based:
  how much does WSD contribute to improving performance of some real task?

To date, evaluation of WSD has been in vitro. This has been standardised by the SENSEVAL project: a shared task framework that has produced a number of freely available hand-labelled datasets `http://www.senseval.org/`

# In vitro Evaluation of Supervised Approaches

The development and evaluation of an automated learning system involves partitioning the data into the following **disjoint** subsets:

- Training data: data used for developing the system's capabilities
- Development data: possibly some data is held out for use in formative evaluation for developing and improving the system
- Test data: data used to evaluate the system's performance after development (what you report on your paper).



# Evaluation: Cross-Validation

If only a small quantity of annotated data is available, it is common to use cross-validation for training and evaluation.

- the data is partitioned into $k$ sets or *folds* (often $k = 10$)
- training and testing are done $k$ times, each time using a different fold for evaluation and the remaining $k - 1$ folds for training
- the mean of the $k$ tests is taken as final results

To use the data even more efficiently, we can set $k$ to the total number $N$ of items in the data set so that each fold involves $N - 1$ items for training and 1 for testing.

- this form of cross-validation is known as leave-one-out.

In cross-validation, every items gets used for both training and testing. This avoids arbitrary splits that by chance may lead to biased results.

# Evaluation Measures

Measures for reporting the system's performance on the test data:

- Accuracy: percentage of instance where the class hypothesised by the system matches the gold standard label.
- Error rate: the inverse of accuracy $1 - A$

[*precision, recall and F-measure are not typically used in WSD*]

# Lower and Upper Bounds

The system's performance needs to be compared to some baseline or lower bound. The results of your system will be more convincing the more it improves over a more challenging baseline.

A baseline can be the accuracy achieved by e.g.:

- a random classifier
- a majority class classifier: always choose the most frequent class
- a basic algorithm

Human inter-annotator agreement can be taken to define an upper bound for the performance of an automatic system:

- we can expect that an automatic system will agree with the gold standard only as much as other humans are able to agree with it.

# Manual Annotation

Supervised learning requires humans annotating corpora by hand. Can we rely on the judgements of one single individual?

- an annotation is considered reliable if several annotators agree sufficiently – they consistently make the same decisions.

Several measures of inter-annotator agreement have been proposed. One of the most commonly used is Cohen's *kappa* ($\kappa$). $\kappa$ measures how much coders agree correcting for chance agreement

$$\kappa = \frac{A_o - A_e}{1 - A_e}$$

$A_o$: observed agreement
$A_e$: expected agreement by chance

$\kappa = 1$ : perfect agreement
$\kappa = 0$ : no agreement beyond chance

There are several ways to compute $A_e$. For further details, see:

Arstein & Poesio (2008) Survey Article: Inter-Coder Agreement for Computational Linguistics, *Computational Linguistics*, 34(4):555–596.

For classification experiments, only a particular version of an annotation is considered – the so-called gold standard.

# Feature Analysis

A very important part of developing automatic classifiers is the selection of a predictive set of features — theoretical and linguistic insights can help us to come up with interesting features.

Once we have our set of features, we want to investigate which features have the most predictive power. Two possible methods:

- Feature ablation: remove one single feature at a time and re-train and re-test the classifier to compare results with and without that feature.
- Information gain: if we know the value of feature $F$, how much does that reduce our uncertainty regarding the correct class $X$?
  - the difference between the prior probability of $X$ (it's frequency) and the conditional probability $p(X \mid F)$ of $X$ given $F$ gives us the info gain of $F$ for $X$
  - also called *Kullback-Leibler divergence* or *relative entropy*

$\Rightarrow$ coming up with well-motivated features and analysing their relative predictive power in a categorisation task is what makes supervised machine learning approaches interesting theoretically.

# Knowledge-based Approaches

If a sense-labeled corpus is not available, electronic dictionaries such as WordNet can be used as a source of indirect supervision.

The most common approaches exploit the following sources of information:

- overlap of sense definitions
- selectional preferences of predicates

# The Simplified Lesk Algorithm

```
function SIMPLIFIED LESK(word, sentence) returns best sense of word

  best-sense ← most frequent sense for word
  max-overlap ← 0
  context ← set of words in sentence
  for each sense in senses of word do
    signature ← set of words in the gloss and examples of sense
    overlap ← COMPUTEOVERLAP(signature, context)
    if overlap > max-overlap then
        max-overlap ← overlap
        best-sense ← sense
  end
  return(best-sense)
```

It chooses the sense whose signature shares most words with the context of the input word. Or if there is none, because there is no overlap or there is a tie, it takes the most frequent sense.

When calculating overlap, only content words are taken into account (nouns, verbs, adjectives, and adverbs).

# The Simplified Lesk Algorithm: Example

**function** SIMPLIFIED LESK(*word*, *sentence*) **returns** best sense of *word*

*best-sense* ← most frequent sense for *word*
*max-overlap* ← 0
*context* ← set of words in *sentence*
**for each** *sense* **in** senses of *word* **do**
  *signature* ← set of words in the gloss and examples of *sense*
  *overlap* ← COMPUTEOVERLAP(*signature*, *context*)
  **if** *overlap* > *max-overlap* **then**
    *max-overlap* ← *overlap*
    *best-sense* ← *sense*
**end**
**return**(*best-sense*)

Target sentence: *the port they served us was deliciously sweet*

- S: (n) **port** (a place (seaport or airport) where people and merchandise can enter or leave a country)
- S: (n) **port**, port wine (sweet dark-red dessert wine originally from Portugal)
- S: (n) **port**, embrasure, porthole (an opening (in a wall or ship or armored vehicle) for firing through)
- S: (n) larboard, **port** (the left side of a ship or aircraft to someone who is aboard and facing the bow or nose)
- S: (n) interface, **port** ((computer science) computer circuit consisting of the hardware and associated circuitry that links one device with another (especially a computer and a hard disk drive or other peripherals))

# Lesk Algorithms

There are several variants of the Lesk algorithm. For instance:

- Original Lesk (Lesk 1986): it compares the target word's signature with the signatures of each of the context words.
- Simplified Lesk, due to Kilgarriff and Rosenzweig (2000)
- Corpus Lesk (Vasislescu et al. 2004):
  - it uses a sense-labeled corpus to extract the context of all the instances of a particular sense
  - it applies a weight to each overlapping word (inverse document frequency) which weights higher those words that are less frequent in a corpus.

Corpus Lesk is often used as a baseline system.

# Selectional Restrictions

(1) In our house everybody has a career and none of them includes washing **dishes**

(2) Ms Chen works efficiently, stir-frying simple **dishes**, including braised pig's ears.

Presumably we don't perceive an ambiguity due to the constraints imposed by the verbs *wash* and *stir-fry*.

- in the 80's these intuitions were used in rule-based systems, which discarded senses that did not meet selectional restrictions
- in the 90's probabilistic approaches were developed that define selectional *preferences* rather than restrictions
  - one of the most well-known models is due to Resnik (1997)

Resnik (1997) Selectional preferences and sense disambiguation, in *Proceedings of the ACK Workshop Tagging Text with Lexical Semantics: Why, What and How?*

# Resnik's selectional association: main ideas

Selectional preference strength: how much information a verb gives about the semantic class of its arguments

- the difference between $P(c)$ the probability of finding nouns with semantic class $c$ and $P(c|v)$ the probability that class $c$ occurs as an argument of verb $v$.
- the bigger the difference (calculated by relative entropy), the more informative the verb is.

Selectional association between a particular $v$ and $c$ measures how much $c$ contributes to the selectional preference strength of $v$

- in a parsed corpus count how often a word occurs as argument of $v$
- use WordNet to extract the frequency of a semantic class instantiated by an argument words
- how much does each $c$ contribute to $v$'s preference strength?

$\Rightarrow$ select the sense with the highest selectional association

# Open problems

A few years back, there was a feeling in the community that a change was needed:

> We believe that there are two complementary routes forward. The first is to become more theoretical, to return to computational linguistics, to work on WSD embracing more realistic models of word sense (including non-discreteness, vagueness, and analogy), thus drawing on and feeding theories of word meaning and context from (computational) lexical semantics and lexicography. While not obviously immediately applicable, this research has defensible goals. Can we look to WSD research to provide a practical computational lexical semantics?
>
> The second route is to focus on making WSD applicable whatever it takes. Can any of the results to date be applied in real applications? Why doesn't explicit WSD work in applications when other generic NLP components do? Does WSD have to be more accurate? Are homographs the best level of granularity? Is domain-based WSD the answer?
>
> Both routes could lead to better applications and a better understanding of meaning and language – surely the two main goals of NLP and computational linguistics.

Eneko Agirre & Philip Edmonds (eds.) *Word Sense Disambiguation: Algorithms and Applications*, Springer, 2007.

# Open problems

Ide & Véronis (1998) mentioned the following open problems:

- the role of context: which feature types are best predictors? different for different word classes?
- sense division: what representation, what granularity
- evaluation: in vitro, in vivo?

Agirre & Edmons (2007) mention the following open directions:

- domain- and application-based WSD
- unsupervised cross-lingual approaches
- WSD as an optimization problem rather than classification where there is interdependency amongst senses
- applying deeper linguistic knowledge
- sense discovery

# Resources

- ACL Wiki: http://aclweb.org/aclwiki/index.php?title=Word_sense_disambiguation
- SemEval wikipedia entry with links to Senseval / SemEval datasets http://en.wikipedia.org/wiki/SemEval
- Main survey papers, including summaries of Senseval / SemEval results:
  - Eneko Agirre& Philip Edmonds (eds.) *Word Sense Disambiguation: Algorithms and Applications*, Springer, 2007.
  - Navigli (2009) Word Sense Disambiguation: A Survey, *ACM Computing Surveys*, 41(2).
- A game-based data collection experiment where you are asked to annotate words with the right sense: http://www.wordrobe.org/

# Readings for Wednesday

Two classic non-technical papers by computational lexicographers. Choose one of them.

Adam Kilgarriff (1997) I don't believe in word senses.
*Computers and the Humanities*, 31:91-113.

Patrick Hanks (2000) Do Word meanings exist?
*Computers and the Humanities*, 34:205–215.

See COSP website for HW#3, due on Monday (or Wednesday if you choose the new NLTK exercise I will add).