Computational Semantics and Pragmatics

Autumn 2013

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

Summary of DSMs Features and Parameters

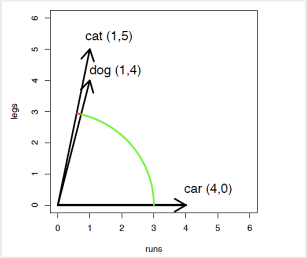
Usage-based perspective on meaning: word meaning depends, at least in part, on the contexts in which words are used.

How do we build contextual (i.e. distributional) meaning representations? Essentially, by quantifying what kind of expressions words occur with.

A distributional semantic model (DSM) is a co-occurrence matrix where rows correspond to vectors for our target terms and columns to contexts where the target terms appear (the vector dimensions).

	run	legs
dog	1	4
cat	1	5
car	4	0

A DSM allows us to measure the semantic similarity between words by comparing their vector representations.



Summary of DSMs Features and Parameters

Main parameters and steps in constructing a DSM:

- Target terms and contexts: raw text, lemmas, POS-tagged lemmas, only content words, dependency relations, patterns,. . .
- Corpus to be used and required pre-processing
- Context where to look for co-occurrence events: window of k words from the target, within sentence boundaries,. . .
- Build the matrix by extracting counts of co-occurrence events
- Possible logarithmic scaling of features
- Possible weighting of features to give more weight to less expected events
- Possible dimensionality reduction to compress the matrix
- Similarity measure [if model is evaluated on how well it captures semantic similarity]: cosine of angle between two vectors, . . .

DSMs and Word Senses

- DSMs we have seen until now:
  - ∗ one vector (meaning representation) per word form.
- What about ambiguity?
  - ∗ Kilgarriff's "*I don't believe in word senses*" was a proposal to characterise word senses as abstractions over clusters of word usages.
- We can use a version of a DSM where we compute context vectors for each use of a given word form.

Distributional Word Senses

How to extract the senses of a given word form w:

- Given a corpus, select the n most frequent lemmas within a given context window from any token of w.
- For each observation w_j of w in the corpus, compute a vector $\vec{w_j} = (f_1, f_2, f_3, \ldots, f_n)$, where f_i is the frequency of co-occurrenave with context word i.
- Alternatively, compute a *second-order co-occurrence vector*: compute a vector $\vec{x_i}$ for each context word i co-occurring with w_j and take the centroid (average) of $\vec{x_1} \ldots \vec{x_n}$ as $\vec{w_j}$.
- Use a clustering algorithm to cluster the vectors w_j into groups or clusters; each cluster defines a sense of w.

Hinrich Schütze. *Ambiguity Resolution in Language Learning*. CSLI Publications, Stanford, 1997.

Hinrich Schütze, Automatic Word Sense Discrimination, *Computational Linguistics*, 24 (1): 97–124, 1998

Word Sense Disambiguation

Each cluster of vectors defines a sense for a given word form w.

- prototype representation of a sense: centroid
- exemplar-based representation: all the vectors in the cluster

How can we disambiguate a particular instance w_k of w?

- compute a context vector for w_k as before
- measure distance between $\vec{w_k}$ and each sense: either distance to centroid (prototype) or re-clustering taking all examplars
- assign the closest sense

This method is fully unsupervised. How can we evaluate it?

- if a sense tagged corpus is available (e.g. SemCor):
  - $*$ map each induced sense to the predefined sense with which there is the most overlap; or alternatively
  - $*$ for all pairs of instances, test whether the instances in the pair would have the same sense according to the induce and predifined senses.

Extra-Brief Intro to Clustering

Clustering is a general term referring to the task of classifying a set of objects into groups (clusters) so that the objects in the same cluster are more similar to each other than to those in other clusters.

Several clustering algorithms exist. Two common techniques are:

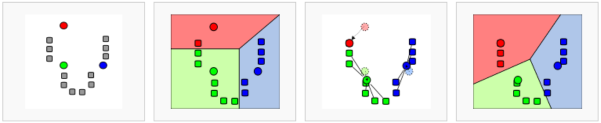
- k-means clustering
- Agglomerative hierarchical clustering

The next two slides contain an extra brief review of the basic steps involved in these two types of algorithms. For further details, you can consult these references:

Manning & Schütze (1999) Foundations of Statistical Natural Language Processing, ch. 14: *Clustering*, MIT Press.
Jain, Murty & Flynn (1999) Data Clustering: A Review, *ACM Computing Surveys*, 31:264-323.

k-means Clustering: Basics

1. assume a certain number k of clusters;
2. select k objects that are as distant as possible from each other; these are the starting centroids of the clusters;
3. assign each remaining object to the cluster whose centroid is the closest;
4. when all objects have been assigned, recalculate the positions of the k centroids.
5. Repeat Steps 3 and 4 until the centroids are stable.

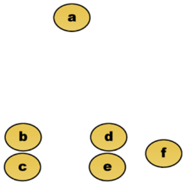


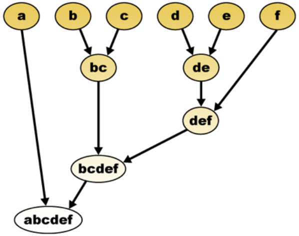
Picture from Wikipedia `http://en.wikipedia.org/wiki/K-means_algorithm`
There seems to be a mistake with red cluster, but good enough for illustration

Agglomerative Clustering: Basics

1. assign each training instance to its own cluster
2. compute the distance between the clusters and merge the most similar pair of clusters
   * similarity between clusters can be computer by taking the shortest, the longest, or the average distance
3. repeat step 2 until either a specified number of clusters is reached or the clusters have some desired property.

By repeating step 2 until all items belong to the same cluster we end up with a tree that can be cut at the desired level of specificity.





Other WSD Approaches

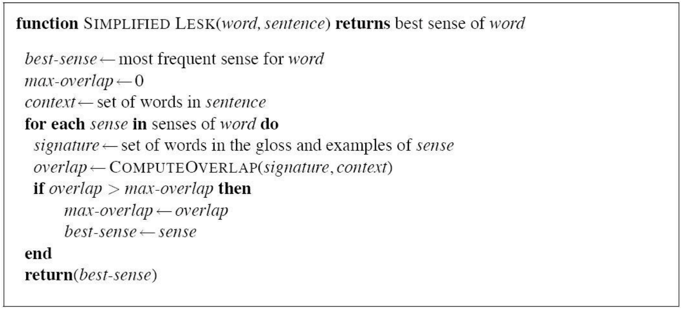
The most common approaches to WSD in NLP are supervised or knowledge-based, in contrast to the distributional approach we have seen earlier.

They assume the WordNet inventory of senses and use

- supervised machine learning methods (classification algorithms trained on a manually disambiguated corpus), or
- WordNet as a source of indirect supervision

They exploit context and distributional information (obviously) but assume a predefined set of senses $\rightsquigarrow$ less interesting from a theoretic point of view.

The Simplified Lesk Algorithm



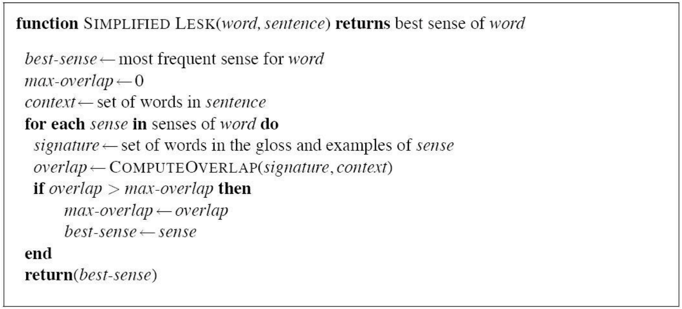
```
function SIMPLIFIED LESK(word, sentence) returns best sense of word

 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).

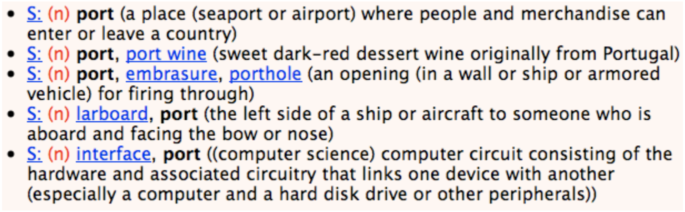
The Simplified Lesk Algorithm: Example



```
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 for supervised approaches.

Some Resources

- SemCor http://www.cse.unt.edu/~rada/downloads.html#semcor
- SemEval wikipedia entry with links to Senseval / SemEval datasets http://en.wikipedia.org/wiki/SemEval
- A game-based data collection experiment where you are asked to annotate words with the right sense: http://www.wordrobe.org/

Back to Distributional Semantics

Essence: (aspects of) word meaning can be characterised by abstracting over the (linguistic) contexts in which words are used.

Main features (pros?) of distributional meaning representations:

- inherently quantitative and gradual
  - ∗ no definitions; ambiguity accounted for by clusters of uses; continuum between polysemy-homonomy.
  - ∗ compatible with theories of concepts: vagueness, typicality.
- inherently context-dependent
  - ∗ the linguistic contexts in which words are used enter into their semantic constitution (blurred boundary semantics / pragmatics);
- inherently dynamic
  - ∗ meaning derives from the global distributional history of a word, which is constantly evolving – compatible with meaning change;

Main Challenges for DSMs

Initially unsatisfactory aspects of DSMs currently being addressed:

- Beyond semantic similarity: DSMs are good at modeling similarity, but words may be similar in very different respects.

  Turney (2006) Similarity of Semantic Relations, *Computational Linguistics*, 33(3).
  Baroni & Lenci (2010) Distributional Memory: A general framework for corpus-based semantics. *Computational Linguistics* 36(4).

- Compositionality: can DSMs explain how the meaning of a complex expression can be build up from the meaning of its parts?

  Mitchell & Lapata (2010) Composition in distributional models of semantics. *Cognitive Science* 34(8).
  Baroni & Zamparelli (2010) Nouns are vectors, adjectives are matrices. *Proc. EMNLP*
  Grefenstette & Sadrzadeh (2011) Experimental support for a categorical compositional distributional model of meaning. *Proc. EMNLP*

- Embodiment: meanings/concepts are not purely linguistic, they are *embodied*, grounded in perception.

  Feng & Lapata (2010) Visual Information in Semantic Representation, *Proc. NAACL*
  Bruni, Boleda, Baroni & Tran (2012) Distributional semantics in technicolor, *Proc. *SEM*.

Examples of research topics where the methodologies we have seen are relevant:

Katja Abramova, Raquel Fernández, and Federico Sangati (2013) Automatic Labeling of Phonesthemic Senses. In *Proceedings of the 35th Annual Conference of the Cognitive Science Society*, pp. 1696-1701. Berlin, Germany.

Next Week

We move on to the topic of referring expressions.

Dale & Reiter (1995) Computational Interpretations of Gricean Maxims in the Generation of Referring Expressions, *Cognitive Science*, 18:233-266.

Krahmer & van Deemter (2012) Computational Generation of Referring Expressions: A Survey, *Computational Linguistics* 38(1):173-218.

Make sure you read Dale & Reiter (1995) by Tuesday.