Lecture 13 Component technologies: Word Sense Disambiguation, and Reference resolution

stephen.pulman@comlab.ox.ac.uk

Task: decide which sense a word has in a given context:

> The man sat fishing on the **bank**.
>
> I went to the **bank** to get some money.
>
> You'd better **bank** that quickly.
>
> Don't **bank** on it.
>
> The plane had to **bank** suddenly to avoid the helicopter

For: parsing, logical form construction / automatic translation (banque vs. bord vs. virage) / information retrieval / extraction etc.

Questions: what knowledge sources are needed? What is a 'sense'? What is a context? How well can humans do it?

Google's Adsense:

Gmail text: The Alan Tayler Lecture: Mathematical Modelling in Medicine, Sports and Technology ...

Sidebar:

> Want to be a Model? Kick-start your modelling career! 14 - 21 yrs, get expert advice
> www.dieselmodels.co.uk

Gmail text: I just saw a seal in the Thames! ...

Sidebar:

> ShowerSeal Proven To Work. Fully Seals Your Shower Tray. 10 Years Mould Free. Free UK Delivery
> www.byretech.com

> Navy Seals Watches. It's Time you Had this Watch Great Value - Order Now!
> www.SpecialOpsWatch.co.uk

> Vulcan Mechanical Seal. Manufacturer; Superb Range Quality, Service And Price
> www.vulcan-eng.com

Dictionary based methods

(Lesk 1986). Assume you have a machine readable dictionary (LDOCE etc.) with definitions: e.g.

cone: (noun)
(i) a mass of ovule bearing or pollen bearing scales or bracts in trees of the pine family or in cyads that are arranged usually on a somewhat elongated axis
(ii) something that resembles a cone in shape: as ... a crisp cone-shaped wafer for holding ice cream
(iii) etc.

The intuition is that if words occurring in the definition occur in the context, or in the definitions of words in the context, then that is the appropriate sense:

> Given word W in context C,
> let E be the union of the words in C and their definitions
>   for each sense S_i and its definition D_i of W
>   score(S_i) = similarity(D_i,E)
> choose the sense with the highest score

There are a variety of measures of similarity that have been tried:

similarity(A,B) $=$

$|A \cap B|$, (the 'matching coeffient') or

$\frac{2*|A \cap B|}{|A \cup B|}$ (the Dice coefficient), or

$\frac{|A \cap B|}{|A \cup B|}$ (the Jaccard coefficient), or

$\frac{|A \cap B|}{\min(|A|,|B|)}$ (the overlap coefficient)

The matching coefficient may reward lengthy definitions. Dice normalises for length; Jaccard penalises small numbers of shared words more heavily. The overlap coefficient rewards inclusion.

Example (found on Google)

Bank erosion and **stream** widening may also occur here.

One way of **raising** this **finance** is to go to a **bank**.

Definitions from `http://dictionary.cambridge.org`:
bank 1: sloping **raised** land, especially along the sides of a **river**.
bank 2: an organization where people and businesses can invest or borrow **money**...
erosion: soil/coastal erosion.
stream (small **river**) water that flows naturally along a fixed route formed by a channel cut into rock or earth, usually at ground level
finance: (the management of) a supply of **money**... **raise**: to cause to exist: ... the **money**/cash/capital/funds.

D1 sloping raise land side river
D2 organization people business invest borrow money
S1 erosion stream widen river water flow ....
S2 raise finance management supply money cash....

D1∩S1 = {river}; D1∩S2 = {raise}; D2∩S1 = {}; D2∩S2 = {money}

Typically this approach gets c.50% right. Using a thesaurus doesn't improve things much.

Corpus-based methods

Need a sense-tagged corpus (small one distributed with Wordnet).

Gale et al. used a Naive Bayes classifier trained on a corpus. We want to find the most likely sense given the context (the bag of words surrounding the ambiguous word) i.e.

$\max_i$ P(sense$_i$ | context).

$$\text{P(sense}_i \mid \text{context)} = \frac{\text{P(context} \mid \text{sense}_i\text{)*P(sense}_i\text{)}}{\text{P(context)}}$$

P(context) will be constant for all senses, so we can ignore that.

P(context | sense$_i$) = $\prod_j$ word$_j$ in context P(word$_j$ | sense$_i$)

Example: drug
sense 1: medication, contexts= prices, prescription, patent, increase, consumer, pharmaceutical
sense 2: illegal substance, contexts = abuse, paraphernalia,illicit, alcohol, cocaine,traffickers

System gets 90% accuracy when tested on 6 ambiguous nouns in Hansard.

What is a sense?

To apply the previous method, you need a sense-tagged corpus. But it has been found that it is difficult to get good agreement between annotators. This seems to be because senses are vague, and sometimes the use of the word does not clearly fit any sense, or is even genuinely ambiguous:

For better or worse, this would bring competition to the licensed trade.

Competition: competitors, or act of competing?

Kilgarriff - there are no determinate senses.

Veronis: no agreement for: correct, historique, économie, comprendre...

Yarowsky: one sense per text - i.e. an ambiguous word will tend to be used in only one sense on a given text.

Automatic Word Sense Disambiguation: Schütze 1998

Use vector space models:

1. Construct 'word space'. Select some words as 'features' and construct vectors for the target words representing how many times the feature words occurred within a 50 word window of the target words:

target	river	stream	money	raise	finance
bank	10	15	25	20	13
water	28	25	2	15	0
cheque	0	0	30	20	25
etc.					

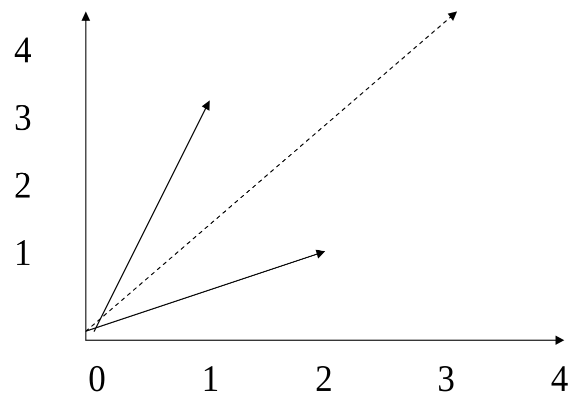
Use χ^2 test to make sure the co-occurrences are meaningful. Features can be 'local' i.e. those that occur in the contexts, or 'global' i.e. those that are most frequent in the corpus.

Words with similar meanings should have vectors pointing in a similar direction...

Vectors

We regard any sequence of numbers like those as a 'vector' in a multi-dimensional space (here 5). (We can visualise vectors in 2 or 3 dimensions: more than that is difficult, but their behaviour is analogous.) Vectors have both a length (defined for a vector like (a_1,a_2,...,a_n) as $\sqrt{a_1^2 + a_2^2 + ... + a_n^2}$) and a direction.

Addition of vectors averages their direction: easy to see this for 2D case:



(3,1) + (1,2) = (4,3)

We can ‘normalise’ vectors so that they have length 1: then only their direction is important.

$$\text{v'} = \frac{\text{v}}{\sqrt{v_1^2 + v_2^2 + ...v_n^2}}$$

We can find how close two vectors are by measuring their cosine distance:

$$\text{cd(A,B)} = \cos(\theta) = \frac{\Sigma_i A_i \times B_i}{\sqrt{\Sigma i A_i^2 \times \Sigma_i B_i^2}}$$

This measures the extent to which they are pointing in the same direction. Completely ‘contradictory’ vectors will be ‘orthogonal’, i.e. at 90° to each other.

2. Construct ‘context space’. Go back to the corpus, and for each context that a target word occurs in, construct the centroid (sum) of the vectors for each of the feature words in that context. The centroid ‘averages’ the direction of the set of vectors.

E.g in context 1 for **bank**, you may sum river+water+edge, but in context 2, it may be money+loan+interest.

3. Construct ‘senses’ by clustering the context vectors. There are many clustering algorithms. Schütze used a form of ‘agglomerative clustering’, where each vector initially forms its own cluster, and clusters are repeatedly merged based on some criterion until the target number of clusters (2-10 here) is arrived at. Each cluster should correspond to a distinct sense, which can be represented by the centroid of the vectors in the cluster.

4. To disambiguate a word in a context:

i. construct the vector for that context, as above
ii. compare the sense vectors for the word to the context vector
iii. choose the sense whose vector is closest to the context vector

Evaluation: use ambiguous words if you have the data, otherwise use ‘pseudo-words’. Choose two distinct words from the corpus, for example ‘computer’ and ‘banana’, and replace all occurrences of ‘computer’ and ‘banana’ by the pseudo-word ‘bananacomputer’. We can evaluate how well the algorithm does on disambiguating ‘bananacomputer’ by looking at the original form of the corpus. In fact, since single words are often ambiguous, it is better to create pseudo-words from pairs: e.g. wide range + consulting firm.

Results:

For naturally ambiguous words: **interest, space, plant, ...**

and pseudo-words: **'wide range'+'consulting firm', 'league baseball'+'square feet',...**

average accuracy is:

Natural:	2 clusters	10 clusters
local	76.0%	84.4%
global	80.8%	83.1%
Artificial:		
local	89.9%	92.2%
global	98.6%	98.0%

Different types of pronoun interpretation

Personal pronouns:

a) John likes him/her (*he/she)

b) John thinks he likes him

c) John thinks Bill likes him

d) He thinks that Bill likes him

Reflexives:

a) John likes himself (*herself)

b) John thinks Bill likes himself

c) John thinks Mary likes herself/him (*himself/her)

‘Bound variable’-like readings:

Every politician thinks he is a future Prime Minster

‘Donkey’ Sentences:

a) If any man owns a donkey he beats it

b) Every farmer who owns a donkey beats it

c) No-one will be admitted to the examination unless he has registered 4 weeks in advance

Definite and Indefinite NPs:

a) There was once a handsome prince. The prince lived in a castle.

b) On every car the radio aerial had been broken.

Constraints: number, gender. Preferences: grammatical role, distance, depth of embedding etc.

Most pronoun antecedents are in earlier part of current sentence, or immediately preceding one. But definites (‘the...’ can refer much further back in a discourse or text.

Kennedy and Boguraev's pronoun resolution algorithm

1. list all NPs + offsets (position relative to start of text)

2. locate all NPs within adverbial adjuncts, or complex NPs (NP+PP, NP+S, possessives) This is done by patterns looking for adverbs, rel pron, N_P sequences, N+complementiser sequences.

3. locate occurrences of pleonastic i.e. non-referring 'it' (and existential 'there').

4. represent each DR with features:

```
text = string
type = ref, pron, reflexive (def/indef)
agr = pers/num/gen
gfun = subj, obj, iobj, obl
adjunct = t/f
embed = t/f
offset = number
```

Interpret DRs left to right, sentence by sentence. A DR either introduces a new entity or is coreferential, joining a COREF class. A class has a canonical form (typically that of the first membet), a list of members, and a salience value. The value of a class is the sum of the values satisfied by some member of the class (only 1 value for each feature)

Salience factors:

```
SENT-S = 100 iff in current sentence
CNTX-S = 50 iff in current context (as defined by Hearst topic segmentation)
SUBJ-S = 80 ifff gfun=subj
exst-s = 70 iff in an existential
poss-s = 65 iff in a possessive
acc-s  = 50 iff gfun=obj
dat-s  = 40 iff gfun= iobj
oblq-s = 30 iff complement of P
head-s = 80 if embed=f
arg-s  = 50 if adjunct=f
```

When a DR is assigned to a COREF class, the salience value of the class is recalculated.

```
For each new sentence
  for each existing COREF class, decrease salience by 50%
  for each non-anaphor, generate a new COREF and calculate salience value
  for each reflexive or reciprocal
      if gfun=obj, antecedent is closest preceding gfun=subj
      if gfun = iobj or obl, antecedent is closest preceding subject,
                             or obj with no intervening subj
                             whichever has highest salience
  for each anaphor
     eliminate COREFs that disagree in person, number or gender
     eliminate COREFs with disjoint reference:
         (1)  {subj,obj} followed by {obj,iobj,obl} with no
              intervening subj are disjoint
         (2) for any anaphor which is adjunct=f or embed=f,
             any following non-anaphor is disjoint
         (3) a DR is disjoint from any following anaphor with embed=t
             up to the next DR with embed=f.
     for intra-sentence candidates:
      - decrease salience of candidate COREFs which follow the anaphor
      - if an anaphor-COREF candidate are both embed=t and/or adjunct=t,
        treat the COREF salience temporarily as if the values were f.
      - increase salience of anaphor-COREF if  <gfun(ana),gfun(coref)>
        is identical to a pair previously resolved.
     choose candidate COREF with highest salience, using closeness as
     a tie-breaker.
```

Machine Learning approaches

Many subsequent approaches have treated reference resolution as a classification problem: classify a pair ⟨NPAntecedent, NPPronoun⟩ as co-referring or not. Typically the classifier uses features similar to those in Kennedy and Boguraev.

Creating training data is an issue. If we take a pair of sentences where the first contains an antecedent NPa for an anaphor in the second, NPp, a simple strategy would be to take ⟨NPa,NPp⟩ as the positive example, and ⟨NPx,NPp⟩, for all NPx other than NPa in the sentences as the negative examples. But this gives many more negative than positive examples. An alternative is to just make negative examples from NPs **between** NPa and NPp.

Now train your favourite machine learning classifier to distinguish coreferring pairs of NPs. But still need to group together sets of NPs:

Hadson Corp. said it(1) expects to report a third quarter net loss of $17 million to $19 million because of special reserves and continued low natural gas prices. The Oklahoma City energy and defense concern said it(2) will record a $7. 5 million reserve for its(3) defense group...

```
{HC, it(1)},{HC, a third quarter net loss}, {HC, $17m}, {HC, $19m},
{HC, special reserves},{HC, continued low natural gas prices},
{HC, The Oklahoma city energy and defense concern},
{The Oklahoma cedf,it(2)}, {it(2),{its(3)}
etc.
```

References

Improving Machine Learning Approaches to Coreference Resolution Vincent Ng and Claire Cardie, http://www.cs.cornell.edu/home/cardie/papers/acl2002.pdf

Vincent Ng (2010) Supervised Noun Phrase Coreference Research: The First Fifteen Years 2010 ACL Proceedings http://www.aclweb.org/anthology/P/P10/P10-1142.pdf

Christopher Kennedy; Branimir Boguraev, 1996, Anaphora for Everyone: Pronominal Anaphora Resolution without a Parser http://www.aclweb.org/anthology/C/C96/C96-1021.pdf

Gale, W., Church, K., and Yarowsky, D. (1992). One sense per discourse. In Proceedings of the Fourth DARPA Speech and Natural Language Workshop, pages 233–237.

William Gale, Kenneth Church and David Yarowsky. A Method for Disambiguating Word Senses in a Large Corpus. Computers and the Humanities, 26, 415-439, 1992.

Kilgarriff, A. (1993) Dictionary word-sense distinctions: an enquiry into their nature, Computers and the Humanities, vol 26.

Michael Lesk, 1986, Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone, SIGDOC '86: Proceedings of the 5th annual international conference on Systems documentation, pp24–26.

Manning, C. and Schütze, H. 1999 Foundations of Statistical Natural Language Processing, Cambridge Mass: MIT Press, esp. Chapter 7.

Hinrich Schütze: Automatic Word Sense Discrimination. Computational Linguistics 24(1): 97-123 (1998) *NB: This is a special issue of CL devoted to word sense disambiguation, and contains a useful intro by Ide and Veronis*

Wilks, Y. and Stevenson, M. (1998a) The Grammar of Sense: Using part-of-speech tags as a first step in semantic disambiguation, Journal of Natural Language Engineering, 4(1), pp. 1-9.

Wilks, Y. and Stevenson, M. (1998b) Optimising Combinations of Knowledge Sources for Word Sense Disambiguation, Proceedings of the 36th Meeting of the Association for Computational Linguistics (COLING-ACL-98), Montreal, Canada.

Yarowsky, D. (1995) Unsupervised Word-Sense Disambiguation Rivaling Supervised Methods, Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL-95), pp. 189-196, Cambridge, MA.

There is an excellent on-line tutorial at:

`http://www.d.umn.edu/~tpederse/WSDTutorial.html`

Eneko Agirre and Philip Edmonds (eds) 2006, Word Sense Disambiguation:Algorithms and Applications, Springer Verlag.

R. Navigli, 2009, Word sense disambiguation: A survey. ACM Computing Surveys (CSUR) Volume 41 Issue 2, February 2009.