Distributional approaches to semantic analysis

Diarmuid Ó Séaghdha

Natural Language and Information Processing Group
Computer Laboratory
University of Cambridge
do242@cam.ac.uk

HIT-MSRA Summer Workshop on Human Language Technology
August 2011
http://www.cl.cam.ac.uk/~do242/Teaching/HIT-MSRA-2011/


UNIVERSITY OF CAMBRIDGE

Learning about entities and relations

Learning entity sets

Taxonomy learning

Relation discovery

Relation classification

Compound noun interpretation

Learning about entities and relations

- At a fundamental level, the content of a text talks about a set of entities and the relations between them.
- Humans also have a large body of "background knowledge" that we use whenever we interpret a text, e.g., from the sentence

  I spent two days in Beijing.

  we can understand that the speaker was in China.
- Therefore two important issues in computational semantics are:
  - how to identify the relations between entites that exist in the world or in some domain.
  - how to identify the relations between entities expressed in a given text.

Entity set expansion

- Given a set of "seed" entities representing a semantic class, the entity set expansion task involves finding other members of the class.
- For example, the seeds

$$\{ACL, EMNLP, COLING, NAACL\}$$

  belong to the class of *Conferences in NLP* and can be expanded with further members $\{CoNLL, EACL, IJCNLP, LREC \ldots\}$.
- More examples:

$\{Harbin, Beijing, Shanghai, Chengdu\}$	*Cities in China*
$\{KFC, McDonalds, Burger King\}$	*Fast food chains*

Entity set expansion

- One solution: look up Wikipedia or WordNet! E.g., `http://en.wikipedia.org/wiki/List_of_cities_in_China`
- But not everything is there; we may be interested in:
  - A category in a specialist domain, such as *Statistical parsers in NLP*.
  - An "ad hoc" category (Barsalou, 1983), such as *Things to take on a camping holiday*.
  - Finding an exhaustive list of category members, including those not listed in manually compiled resources.

Entity set expansion

- One popular method for entity set expansion is to use distributional similarity to find which entities are semantically closest to the seed entities.
- Given a seed set $S \subset V$ and a feature mapping from V to $\mathcal{R}^k$, a simple algorithm for expansion is as follows (Sarmento et al., 2007):

Step 1: Calculate the seeds' centroid $\bar{S} = \frac{1}{|S|} \sum_{w \in S} \mathbf{w}$.

Step 2: For each candidate word $w' \in V$, calculate $sim(w', \bar{S})$ using, e.g., cosine similarity.

Step 3: Output a list of candidates ranked by similarity.

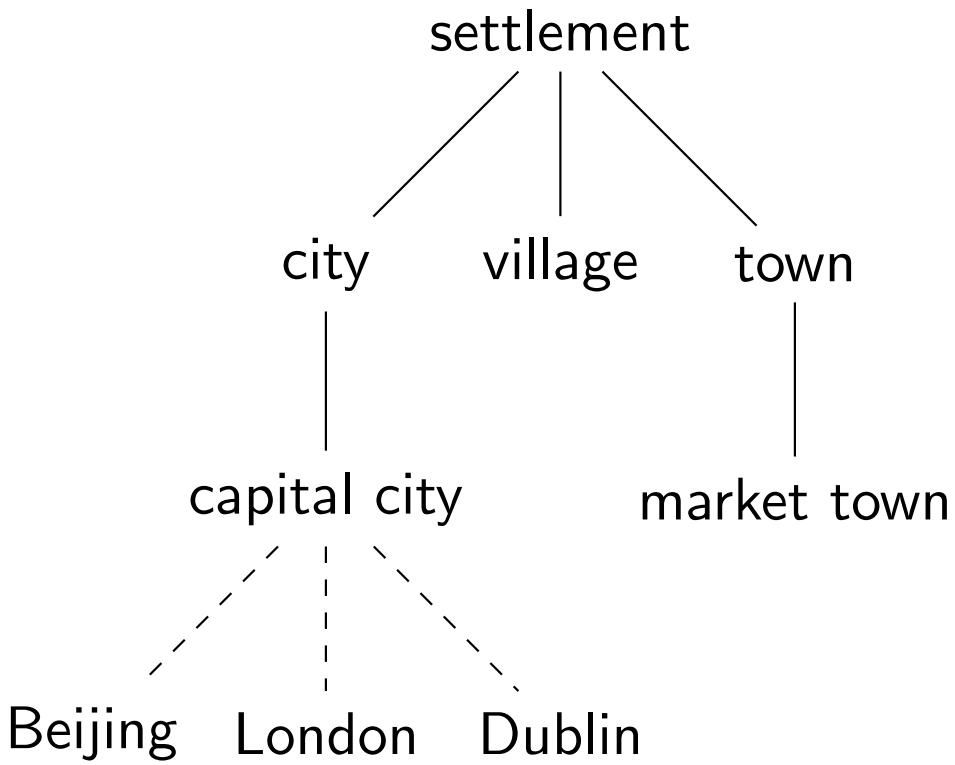
Entity set expansion

- As is usual with distributional similarity, there is flexibility in how the feature mapping is defined. Sarmento et al. (2007) only count co-occurrences in coordinative patterns such as w_1, w_2 *and* w_3 to build distributional vectors, while Pantel et al. (2009) use the leftmost and rightmost NP chunks.
- This approach seems to work well in practice (though we also need a way to "cut off" the ranking at a certain point).

Taxonomy learning

- In our discussion of entity set expansion, we assumed that we had a predefined category in mind and that we were only interested in extracting a flat set of entities.
- A more structured kind of knowledge can be captured in an *inheritance hierarchy* or *taxonomy* consisting of IS-A relation links between terms or concepts.
- If the relation IS-A(A,B) holds (e.g., IS-A(*eagle*,*bird*)), we also say that A is a *hypernym* or B or B is a *hyponym* of A.
- Some well-known taxonomies exist (WordNet, HowNet, UMLS) but they are not comprehensive. The *taxonomy learning* task is to induce IS-A relations from text data.
- It can sometimes be useful to distinguish between IS-A relations between categories and INSTANCE-OF relations between categories and instances, but in general we can use the same methods for both.

An example taxonomy fragment



Patterns for taxonomy learning

- Most work on taxonomy learning is based on the principle that hypernymy relations are expressed by specific text patterns; by searching for these patterns in a large corpus we can discover new instances of hypernymy.
- Hearst (1992) lists a number of such patterns:

  1. such NP_0 as NP_1 (and/or NP_2)
  2. NP_1 and/or other NP_0
  3. NP_0, including NP_1 and/or NP_2
  4. NP_0, especially NP_1 (and/or NP_2)

  where NP_0 is a plural noun phrase and NP_1 and NP_2 are extracted as hypernyms of NP_0.

Patterns for taxonomy learning

- Some Google hits for "** and other **":
  - Fly, hop, or walk through this guide to insects, with photographs and descriptions of all kinds of *insects and other arthropods*.
  - Emma Goldman, *Anarchism and Other Essays* (1927).
  - Had *unions and other groups* lobbied hard for alternatives to current policy we also might not be in this mess.
  - *Influenza and Other Respiratory Viruses*
  - Virus records Android phone *conversations and other tech news*

Patterns for taxonomy learning

- Some Google hits for "** and other **":
  - Fly, hop, or walk through this guide to insects, with photographs and descriptions of all kinds of *insects and other arthropods*. ✓
  - Emma Goldman, *Anarchism and Other Essays* (1927). ✗
  - Had *unions and other groups* lobbied hard for alternatives to current policy we also might not be in this mess. ✓
  - *Influenza and Other Respiratory Viruses* ✓
  - Virus records Android phone *conversations and other tech news* ✗
- Even assuming we can identify NP chunks, we are extracting both good and bad hypernymy links. Ideally we would like to combine multiple strands of evidence to predict whether or not a link is good.

Doubly-anchored patterns

- Kozareva et al. (2008) introduce the concept of *doubly-anchored patterns (DAP)*, which they claim are less error-prone than traditional singly-anchored patterns:

$$DAP: Seed_1 \textit{ such as } Seed_2 \textit{ and } X$$

  where the slot X is filled the noun phrase or list of noun phrases to be harvested.

- For taxonomy learning, Hovy et al. (2009) consider a related pattern:

$$DAP^{-1}: X \textit{ such as } Seed_1 \textit{ and } Seed_2$$

- While a singly-anchored pattern such as X *such as Ford* might be confused between the former US President Harold Ford, the actor Harrison Ford and the car manufacturer Ford, a DAP^{-1} pattern X *such as Ford and Toyota* should find the correct hypernym class.

Bootstrapping entity sets

- One way to generalise the learning of IS-A relations for a particular hypernym class is to initialise the set of class members with a *seed set* of hand-picked members and progressively add more members to the set through an iterative process.
- This is known as *bootstrapping*.
- This is useful when the members of the entity set have an influence on the harvesting process, e.g., by weighting patterns (Etzioni et al., 2005) or by generating new patterns (Kozareva et al., 2008).
- If an incorrect class member is added to the set, it can have a "polluting" effect by causing more bad members to be added. Therefore it is often useful to test new candidates before adding them by checking:
  - Extra Web queries (Etzioni et al., 2005).
  - Query logs (Marius Pa and Van Durme, 2008).
  - Graph centrality (Kozareva et al., 2008).

Bootstrapping pseudocode

Given a set of patterns P, starting set of seed entities E and stopping condition $STOP$:

while Test($STOP$) = **false do**
  New = HarvestEntities(E,P)
  $Filtered$ = FilterEntities(New)
  $E = E \cup Filtered$
end while
return E

An alternative approach to bootstrapping is to expand the pattern set P in parallel with the expansion of E (Brin, 1998).

A graph-based filtering approach I

- Kozareva et al. (2008) propose using graph algorithms to identify true class membership relations in an automatically harvested collections.
- Given a target class C they search the Web for instances matching the doubly anchored pattern *C such as A and X*, where A is an already collected class member (initially a seed member) and the instantiation of X is added to the set of class members.
- They call their approach "reckless bootstrapping": during bootstrapping they do not check whether they are collecting good class members.
- Unsurprisingly, this leads to a very noisy set of entities and requires postprocessing.

A graph-based filtering approach II

- Kozareva et al.'s solution is to build a directed graph whose vertices are the collected entities and edges between vertices are weighted by the number of times they appeared together in a pattern instantiation.
- Candidate class members can then be ranked by the *centrality* of the corresponding vertices. Candidates with weak connections to most other candidates will not be ranked highly.

Graph centrality measures

Let V be the set of vertices, $E \subset V \times V$ be the set of directed edges and $w(v, v')$ be a function giving the weight of the edge from v to v'.

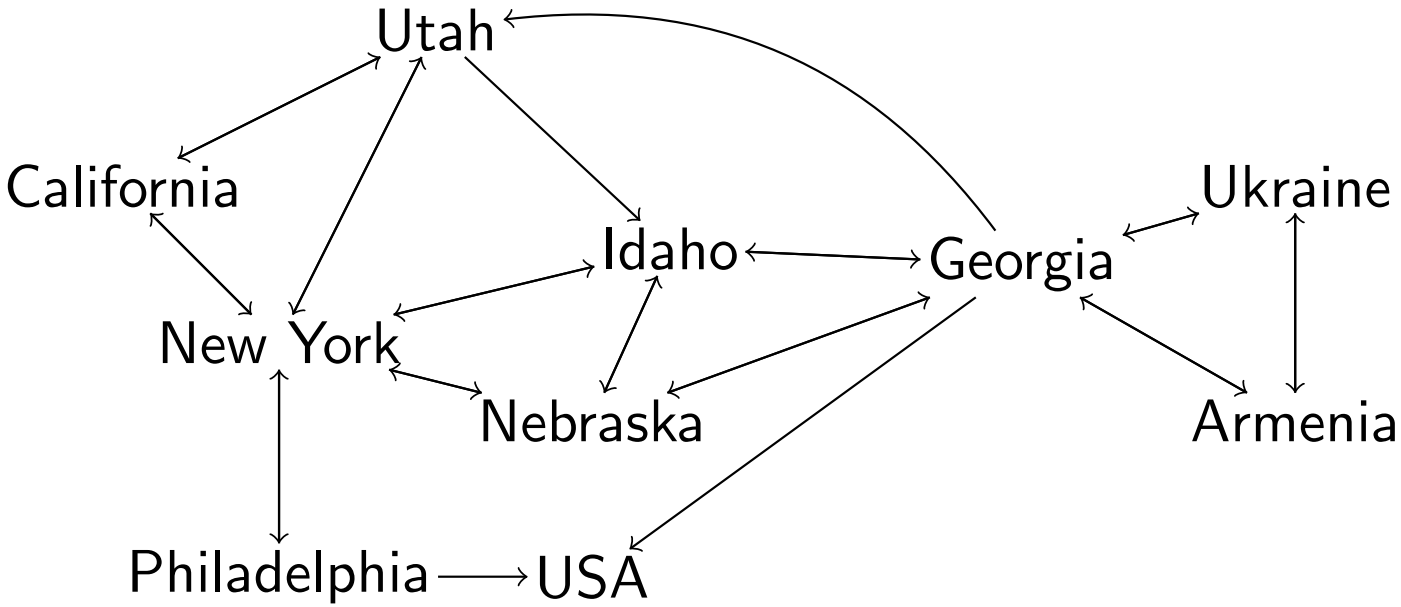
$$\text{OutDegree}(v) = \frac{1}{|V|} \sum_{v' \in V : w(v,v') > 0} w(v, v')$$

$$\text{PageRank}(v) = \frac{1-\alpha}{|V|} + \alpha \sum_{v' \in V : w(v,v') > 0} \frac{\text{PageRank}(v')}{\text{OutDegree}(v')}$$

where α is a damping coefficient between 0 and 1.

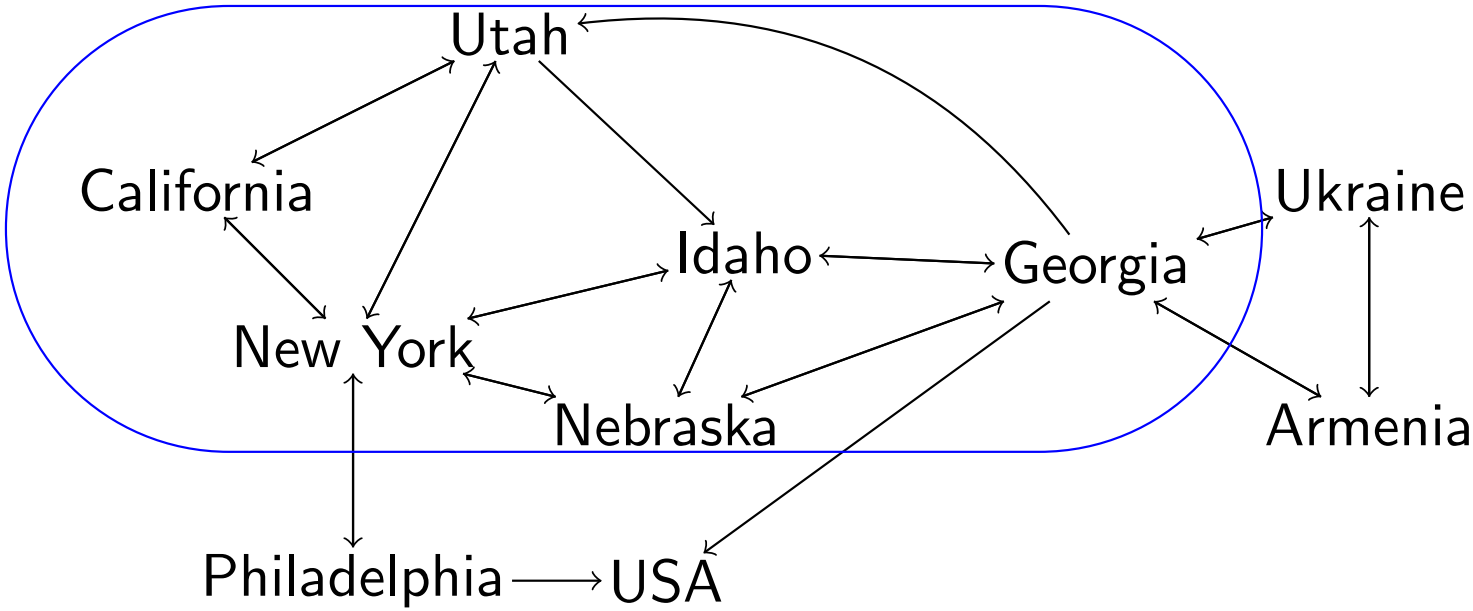
Hyponym graph example

We are trying to learn members of the class of US states, so we start off with a seed query "*states such as Idaho*" and bootstrap from there.



Hyponym graph example

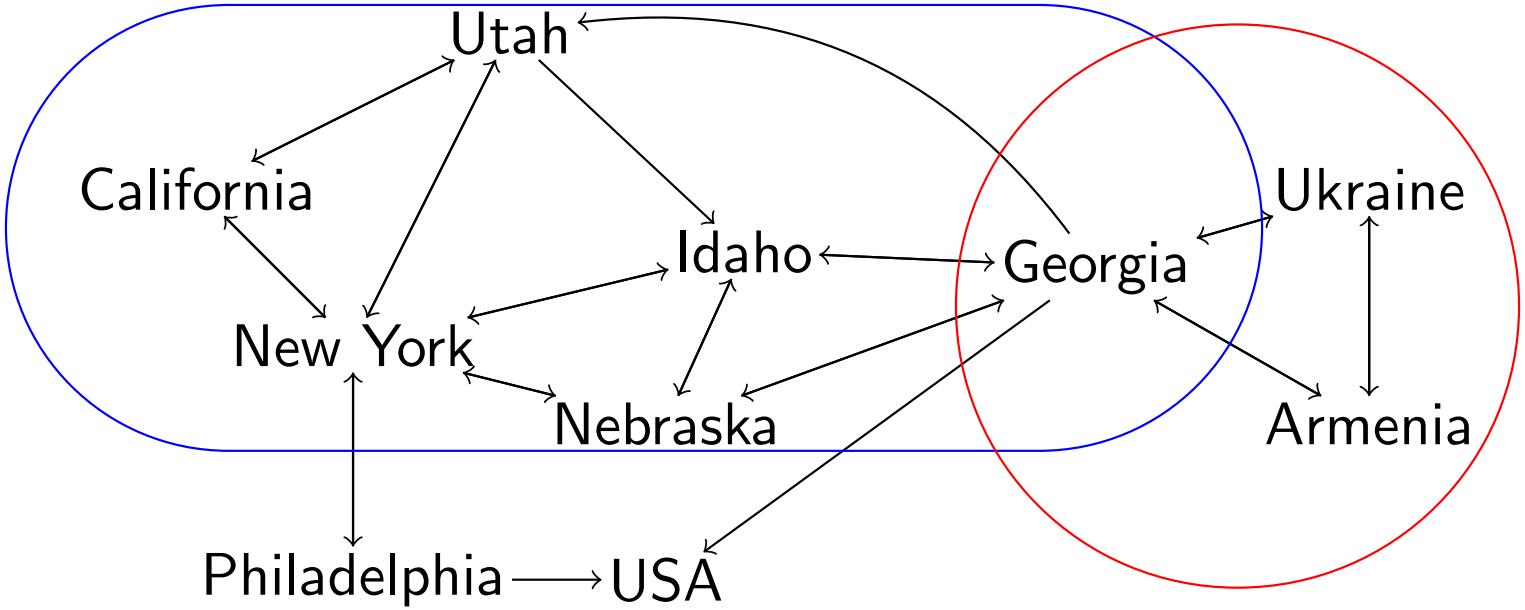
We are trying to learn members of the class of US states, so we start off with a seed query "*states such as Idaho*" and bootstrap from there.



We hope that the correct set members have the highest centrality.

Hyponym graph example

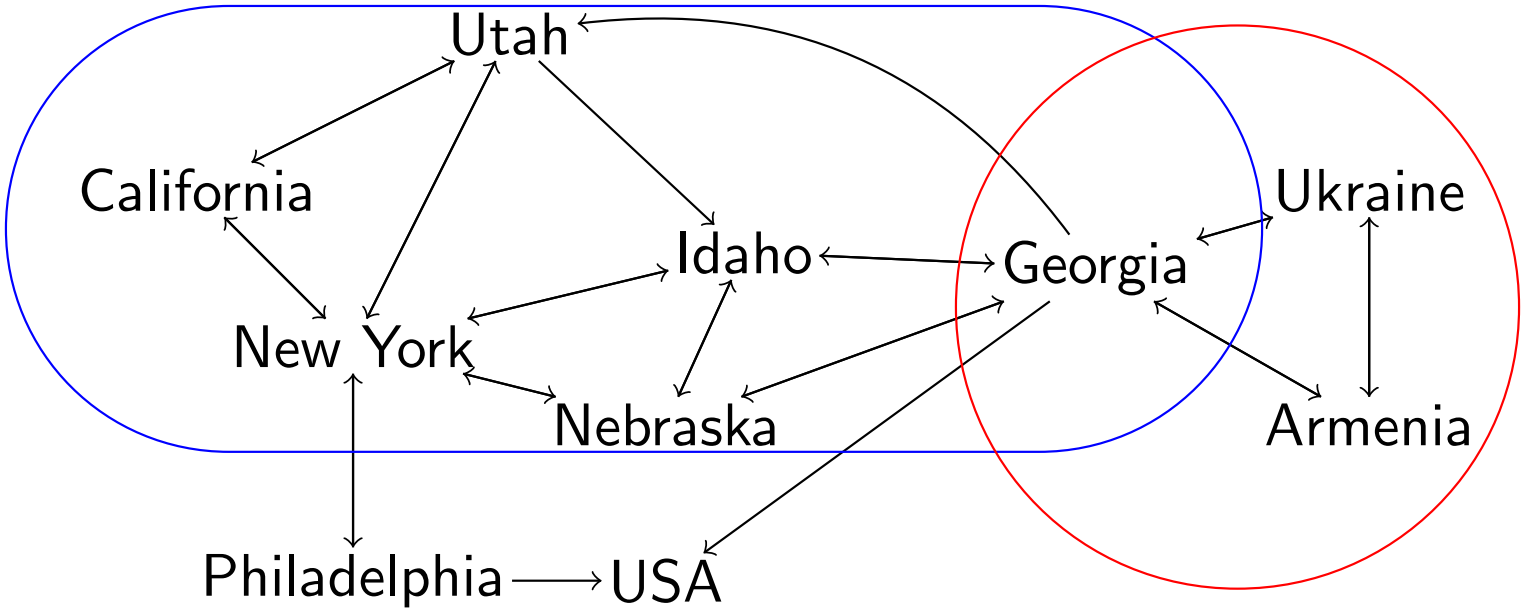
We are trying to learn members of the class of US states, so we start off with a seed query "*states such as Idaho*" and bootstrap from there.



What's going on here?

Hyponym graph example

We are trying to learn members of the class of US states, so we start off with a seed query "*states such as Idaho*" and bootstrap from there.



What's going on here? Georgia is both a US state and an ex-USSR state. Danger of semantic drift? Or maybe an opportunity to detect ambiguous names (Widdows and Dorow, 2002)?

A supervised approach

- Snow et al. (2005) train a supervised classifier to predict hypernymy relations without prespecifying extraction patterns.
- Given a parsed text corpus, they extract all dependency paths connecting a word pair w_1, w_2. These paths and their frequency are used to construct high-dimensional feature vectors for that pair.
- For training, positive and negative examples of hypernymous pairs are extracted from WordNet. Once trained, the classifier can predict whether two unseen words stand in a hypernymy relation even when one or both have not been seen before.
- Snow et al. report that they automatically discover high-precision extraction patterns not previously used in the literature and that their system finds many valid hypernymy links not in WordNet.

Relation discovery

- We want to learn facts of the sort:

  $\textsc{City\_In}(\textit{Harbin}, \textit{China})$
  $\textsc{City\_In}(\textit{Beijing}, \textit{China})$
  $\textsc{Capital\_City}(\textit{Beijing}, \textit{China})$
  $\textsc{Capital\_City}(\textit{Cairo}, \textit{Egypt})$
  $\textsc{University\_In}(\textit{HIT}, \textit{Harbin})$
  $\textsc{Works\_At}(\textit{Prof. Zhao}, \textit{HIT})$
  $\textsc{Works\_At}(\textit{Dr. Ó Séaghdha}, \textit{Uni. Cambridge})$

  where $\textsc{City\_In}$ and $\textsc{Capital\_City}$ are binary relations taking cities and countries as first and second arguments, respectively.
- I call this "relation discovery", others call it "relational information extraction".

Methods for relation discovery

- Note that hypernymy relations are a subset of binary relations; in general, many of the same (or similar) techniques we use for IS-A learning can be applied to general relation learning:
  - Handcrafted patterns (Girju et al., 2003; Pustejovsky et al., 2002)
  - Bootstrapping pattern sets (Brin, 1998; Agichtein and Gravano, 2000; Stevenson and Greenwood, 2005)
  - Supervised classification (**?**)

Patterns for relation discovery

- Just as we saw for hypernymy discovery, it is possible to define handcrafted patterns for identifying semantic relations in a corpus.

PART-WHOLE	*X is made of Y*
(Girju et al., 2003)	*X part of Y*
INHIBITION	*X inhibits Y*
(Pustejovsky et al., 2002)	*X is an inhibitor of Y*

- "Open" relation discovery (Banko et al., 2007) automatically generates patterns for arbitrary relations, e.g. *X works at Y*.

Open relation discovery with relation clustering I

- Davidov et al. (2007) propose an algorithm that discovers a range of semantic relations that apply to an entity class C as well as the lexical realisations of those relations.
- The goal is to discover that e.g., *X is the capital of Y* and *Y's capital, X* express the same relation, without prespecifying the set of relations of interest.

Step 1: Starting from a set of seed entities, expand the seed set through bootstrapping.

Step 2: For each entity e in the bootstrapped set, extract all contexts S containing e from a Web corpus, where each context matches $s_1 x s_2 e x_3$ or $s_1 e s_2 x s_3$ such that h_1, h_2, h_3 are strings of high-frequency words and x is a lower-frequency "content word" with $PMI(x, e)$ above a certain threshold.

Open relation discovery with relation clustering II

Step 3: For each e, group all extracted contexts that have the same x and more than two-thirds word overlap into an *S-group*.

Step 4: Cluster the S-groups for all entities in the entity set by grouping all S-groups with more than two-thirds identical contexts (apart from e and x) and adding other S-groups to those *core clusters* via word overlap.

Step 5: For each e, rank all its extracted relations $R(e, x)$ by PMI and discard the lowest-ranking third.

Open relation discovery with relation clustering

Examples of discovered relations reported by Davidov et al. (2007) for the seed set {*France*,*Angola*}:

Relation	Pattern	Instance
PRESIDENT_OF	*president (x) of (y) has*	(Bush, USA)
POLITICAL_PARTY	*the (x) party of (y) ,*	(Labour,England)
ISLAND_IN	*, (x) island , (y) ,*	(Bathurst,Canada)
CAPITAL_OF	*in (x) , capital of (y) ,*	(Luanda,Angola)
INDUSTRY_OF	*the (x) industry in (y) ,*	(oil,Russia)

One issue in open relation discovery: relation cluster learning is unsupervised. We have to apply relation labels such as ISLAND_IN manually.

Classifying semantic relations in text

- Relation discovery systems discover general facts about the world. A separate task in computational semantics is to identify the relations between entities expressed in a particular sentence. For example, these two sentences express a causal relationship between smoking and cancer:

  Smoking causes cancer.
  Cancer due to smoking is on the increase.

- Identifying sentence-level relations is sometimes called *relation extraction*, especially in the context of the MUC and ACE competitions, but I prefer the term *relation classification* to avoid confusion with information extraction.

The SemEval-07 relation classification dataset

- The SemEval-07 competition on semantic evaluations included a shared task on *Classification of Semantic Relations between Nominals* (Girju et al., 2007).
- Unlike previous tasks such as ACE, this task involves identifying generic semantic relations between arbitrary nouns, rather than relations between named entities.
- The material for this task consists of 7 datasets, each containing > 200 sentences annotated as either positive or negative for one of 7 relations: Cause-Effect, Instrument-Agency, Theme-Tool, Origin-Entity, Part-Whole, Content-Container and Product-Producer. The negative examples are selected to be "near-misses" that are not easy to detect.
- Each sentence contains two marked-up target entities and the task is to predict whether the specified relation holds between those entities.

beginframeThe SemEval-07 relation classification dataset

- Some example sentences from the CONTENT-CONTAINER dataset:

Put $<$ *e*1 $>$tea$<$ /*e*1 $>$ in a $<$ *e*2 $>$heat-resistant jug$<$ /*e*2 $>$ and
CONTENT-CONTAINER(e1,e2) = True
Among the contents of the $<$ *e*1 $>$vessel$<$ /*e*1 $>$ were a set of car
CONTENT-CONTAINER(e2,e1) = True
$<$ *e*1 $>$Batteries$<$ /*e*1 $>$ stored in $<$ *e*2 $>$contact$<$ /*e*2 $>$ with on
CONTENT-CONTAINER(e1,e2) = False
The $<$ *e*1 $>$kitchen$<$ /*e*1 $>$ holds a $<$ *e*2 $>$cooker$<$ /*e*2 $>$, fridge,
CONTENT-CONTAINER(e2,e1) = False (the correct relation would

Features for relation classification

- It is natural to approach this task as a supervised learning problem and train a classifier such as a Support Vector Machine or logistic regression on the training data for each relation. The trained classifier will then predict either a positive or negative label for each sentence in the test set for the appropriate relation.
- Classifiers generally learn a function that views each data instance as a set of "features" that describes properties of the instance that are relevant for the task.
- While some classification algorithms have better general performance than others, a classifier will only perform well if presented with high-quality features for the task.

Features for relation classification

- A variety of feature types have been used for the SemEval-07 relation classification task; most fall into one of three categories.
- **Context features** are extracted from the context of the target entities in the sentence. These features may simply correspond to the words in the context, or they may be structured features such as subsequences or dependency graph fragments.
- **Lexical features** describe the target entities $< e1 >$ and $< e2 >$. They may just correspond to the entities' lexical content or they may also encode distributional information about the entities (Ó Séaghdha and Copestake, 2008).
- **Relational features** use the relations discovered by a relation discovery system to encode context-independent information about how $< e1 >$ and $< e2 >$ interact. **?**) use relation clusters as features, while Nakov and Hearst (2008) derive features from Web queries such as *e1 that * e2*.

Compound noun interpretation

- A *compound noun* (also called a *noun-noun compound*) is a sequence of two or more nouns that function as a single lexical unit:

  water bottle
  tin opener
  tuna fish
  tuna fish tin
  tuna fish tin opener

- Compound nouns are very common in English text: close to 3% of all words in the British National Corpus are members of a compound (Ó Séaghdha, 2008).
- Compound nouns are also very productive: about half the compounds in the BNC occur just once. We will never have seen all possible compounds!

Compound noun interpretation

- Beside their frequency and productivity, the main interest in compounds lies in their semantic diversity.
- In general, the *head noun* of a two-noun compound (in English, the right-most component) determines its semantic class (a *water bottle* is a *bottle*) and the *modifier noun* specifies a subset of that class by evoking an implicit semantic relation (a *water bottle* is a kind of *bottle* that contains water or typically contains water).
- Almost any meaningful semantic relation can be used to generate a compound noun!
- In order to do comprehensive semantic interpretation of a text (or information extraction) we need to identify the implicit relation contained in its noun compounds.

Compound noun examples

olive oil	*oil extracted from olives*
baby oil	*oil for applying to babies*
lamp oil	*oil for burning in lamps*
fish soup	*soup containing fish*
fish food	*food eaten by fish*
fish head	*head belonging to a fish*

There is no direct mapping from lexical content to semantic relation.

Defining the task

- One popular approach to compound noun interpretation is to use an inventory of coarse-grained semantic relations (e.g., Nastase and Szpakowicz (2003), Tratz and Hovy (2010)). Here's one example from Ó Séaghdha (2008):

BE	*steel knife*, *elm tree*
HAVE	*street name*, *car door*
IN	*forest hut*, *lunch time*
INSTRUMENT	*rice cooker*, *bread knife*
AGENT	*honey bee*, *bus driver*
ABOUT	*fairy tale*, *history book*

- This allows us to treat compound interpretation as a classification task.
- Other approaches treat compound interpretation as a paraphrasing task (Butnariu et al., 2010).

Features for interpretation

- Most methods for semantic analysis of compounds use one of two sources of information: *lexical information* and *relational information*.
- Lexical information encodes knowledge about a compound's constituent words. For example, we might expect *tea cup* to express a similar relation to *coffee mug* because *tea* is similar to *coffee* and *cup* is similar to *mug*.
- Relational information encodes knowledge about how the relation between a compound's constituents is expressed elsewhere in a text corpus. For example, to interpret *tea cup* we extract strings such as "*drink tea from a cup*" and "*cup of tea*".

Features for interpretation

- Details on the features used for classification by Ó Séaghdha and Copestake (2009):
- Lexical features: for a compound $N_1 N_2$, we extract a distributional vector of corpus co-occurrences for N_1 and N_2 individually. The co-occurrence types we use are syntactic coordination relations, i.e., instances of N_1 *and/or* N or N *and/or* N_1.
- Relational features: we extract all pattern strings matching * N_1 * N_2 * or * N_2 * N_1 * from a large corpus, where N_1 and N_2 are no more than 10 words apart and we only include up to 5 words to the left and right of the target nouns. Similarity between patterns is based on subsequence matching.

Some technical details

- We used a Support Vector Machine classifier with a combination of a vector kernel to compare lexical features and a kernel on sets of strings to compare relational features.
- The results presented here for relational features are based on discontinuous subsequence matching with a subsequence length of 2–3 tokens.
- You could get good results with your favourite standard feature-based classifier, though the relational feature space might need pruning to avoid computational issues.

Results

Features	Accuracy	F-score
Lexical	59.9	57.8
Relational	52.1	49.9
Combined	**63.1**	**61.6**

Take-home story: combining different information sources gives improved performance (at a statistically significant level).

Summary

- Today we have taken a tour of various NLP tasks involving the discovery of semantic information about entities and relations:
  - Entity set expansion
  - Taxonomy learning
  - Relation discovery
  - Relation classification
  - Compound noun interpretation
- While these tasks all have distinct characteristics, there are connections between them in terms of underlying intuitions and the methods that can be applied to them.