

Automatic Acquisition of Paraphrases and Inference Patterns

Annemarie Friedrich, afried@coli.uni-sb.de

Seminar: Recent Developments in Computational Semantics
(Prof. Dr. M. Pinkal)
Department of Computational Linguistics
Saarland University

May 17th, 2010

Outline

- 1 Introduction
 - Paraphrases
 - Areas of Application
 - Automatic Acquisition of Paraphrases
- 2 DIRT
 - Paths in Dependency Trees
 - Similarity Measure
 - Finding Patterns
 - Experimental Results
- 3 TEASE
 - Overview
 - Anchor Set Extraction (ASE)
 - Template Extraction (TE)
 - Experimental Results
- 4 Conclusion

Outline

- 1 Introduction
 - Paraphrases
 - Areas of Application
 - Automatic Acquisition of Paraphrases
- 2 DIRT
 - Paths in Dependency Trees
 - Similarity Measure
 - Finding Patterns
 - Experimental Results
- 3 TEASE
 - Overview
 - Anchor Set Extraction (ASE)
 - Template Extraction (TE)
 - Experimental Results
- 4 Conclusion

Paraphrases

Are also called:

- **Variants**
- **Inference Rules**
- **Entailment Relations**

Example

Oscar Wilde *wrote* 'The Picture of Dorian Gray'.

Oscar Wilde *is the author of* 'The Picture of Dorian Gray'.

$X \text{ writes } Y \Leftrightarrow X \text{ is the author of } Y$

Dorian Gray *bought* a picture.

Dorian Gray *owns* a picture.

$X \text{ buys } Y \Rightarrow X \text{ owns } Y$

$X \text{ buys } Y \not\Leftarrow X \text{ owns } Y$

Areas of Application of Paraphrase Patterns

- Information Retrieval
 - Information Extraction
 - Question Answering
 - Natural Language Generation
 - Text Summarization
 - Machine Translation
- } Query Expansion
- } Avoidance of Repetitions

Automatic Acquisition of Paraphrases

- Traditionally manually created knowledge bases
 - ☹ Completeness
- General Procedure:
 - Find linguistic structure (= **templates**) that share the same **anchors** (= lexical items describing the context in a sentence).
- Automatic Discovery from Corpus
 - Highly redundant corpora
 - ☺ Accuracy
 - ☹ Limited availability
 - Regular Corpus (⇒ DIRT)
 - Web (⇒ TE/ASE)

Automatic Acquisition of Paraphrases

- DIRT - Discovery of Inference Rules from Text, D. Lin & P. Pantel, 2001
- Scaling Web-based Acquisition of Entailment Relations (TE/ASE), Idan Szpektor et. al., 2004, Tel Aviv University

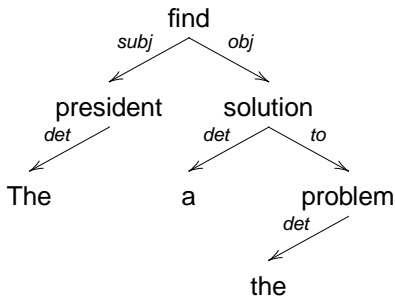
⇒ They don't care about the direction of the relation between two patterns.

Outline

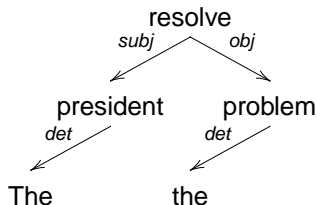
- 1 Introduction
 - Paraphrases
 - Areas of Application
 - Automatic Acquisition of Paraphrases
- 2 DIRT
 - Paths in Dependency Trees
 - Similarity Measure
 - Finding Patterns
 - Experimental Results
- 3 TEASE
 - Overview
 - Anchor Set Extraction (ASE)
 - Template Extraction (TE)
 - Experimental Results
- 4 Conclusion

Discovery of Inference Rules from Text (DIRT)

- Dekang Lin, Patrick Pantel (University of Alberta), 2001
- Discovers correspondence between paths in dependency trees.
- **Dependency trees** generated by MINIPAR:

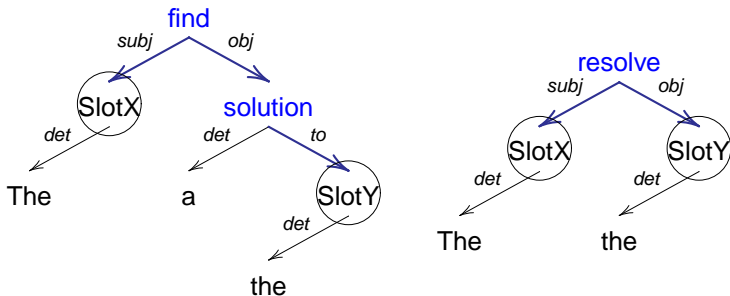


The president found a solution to the problem.



The president resolved the problem.

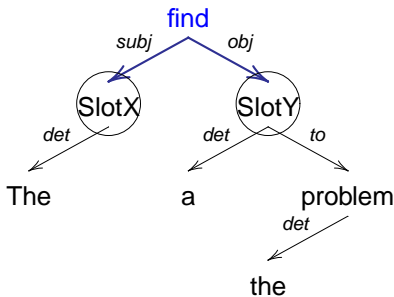
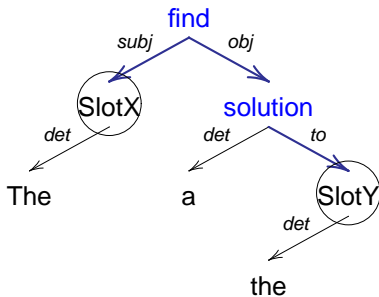
Paths in Dependency Trees - Constraints



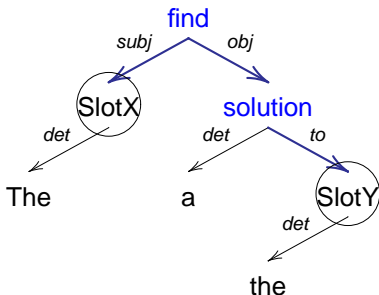
- Substitute slot fillers by 'SlotX' or 'SlotY'
- Slot fillers must be nouns.
- Only dependency relations between two content words (nouns, adjectives, verbs, adverbs) are taken into consideration.

Paths in Dependency Trees - Side Remark

From the sentence *'The president found a solution to the problem.'*,
the following paths can be extracted:



Paths in Dependency Trees - More Constraints



- **Internal Relations** = dependency relations that are not connected to slots. Paths are only extracted if the frequency count of the internal relation (in a corpus) exceeds a threshold.

Assumptions

Distributional Hypothesis

Words that occur in the same contexts tend to have similar meanings.

Extended Distributional Hypothesis

If two paths tend to occur in similar contexts, the meanings of the paths tend to be similar.

⇒ Two paths have similar meaning if their respective **sets of slot fillers** (that occurred in a corpus) are similar.

Database for Paths

- Main Idea of DIRT system: If sets of slot fillers are similar, the paths are similar.
- Collect frequency counts of all paths and the slot fillers in a corpus.

Example

X finds solution to Y		
SLOT	Slot Filler	Count
SlotX	president	1
	committee	2
	employee	3

SlotY	argument	2
	issue	2
	problem	3

X resolves Y		
SLOT	Slot Filler	Count
SlotX	president	2
	committee	1
	detective	3

SlotY	murder	2
	issue	2
	problem	3

Mutual Information between Path, Slot and Slot Filler

- Compute the **Mutual Information** between all pairs of paths and slot fillers.
- Measure strength of the association between a slot (of a specific path) and a filler.

Mutual Information for three events

Three events: $mi(x, y, z) = \log \frac{P(x, y, z)}{P(x)P(y)P(z)}$

Mutual Information for Path, Slot and Slot Filler (1)

$$mi(p, Slot, w) = \log \frac{P(p, Slot, w)}{P(Slot)P(p|Slot)P(w|Slot)}$$

Path and filler are conditionally independent given a slot.

Mutual Information between Path, Slot and Slot Filler

Mutual Information for Path, Slot and Slot Filler (1)

$$mi(p, Slot, w) = \log \frac{P(p, Slot, w)}{P(Slot)P(p|Slot)P(w|Slot)}$$

$|p, Slot, w|$ = frequency count of triple $(p, Slot, w)$

$$|p, Slot, *| = \sum_{w \in W} |p, Slot, w| \quad |*, *, *| = \sum_{p, Slot, w} |p, Slot, w|$$

Mutual Information for Path, Slot and Slot Filler (2)

$$mi(p, Slot, w) = \log \frac{\frac{|p, Slot, w|}{|*, *, *|}}{\frac{|*, Slot, *|}{|*, *, *|} \frac{|p, Slot, *|}{|*, Slot, *|} \frac{|*, Slot, w|}{|*, Slot, *|}}$$

$$= \log \frac{|p, Slot, w| \times |*, Slot, *|}{|p, Slot, *| \times |*, Slot, w|}$$

Database for Paths (with MI)

Example

X finds solution to Y			
SLOT	Slot Filler	Count	Mutual Information
SlotX	president	1	2.45
	committee	2	1.65
	employee	3	4.8

SlotY	argument	2	1.03
	issue	2	2.7
	problem	3	4.8

Similarity between a pair of Slots

Slot Similarity

$$slot_1 = (p_1, s)$$

$$slot_2 = (p_2, s)$$

$T(p_i, s)$ = set of words that fill in the s slot (SlotX or SlotY) of path p_i

$$sim(slot_1, slot_2) = \frac{\sum_{w \in T(p_1, s) \cap T(p_2, s)} [mi(p_1, s, w) + mi(p_2, s, w)]}{\sum_{w \in T(p_1, s)} mi(p_1, s, w) + \sum_{w \in T(p_2, s)} mi(p_2, s, w)}$$

- Add up MI values of common slot fillers.
- Normalize using the sum of all MI values for both slots.

Similarity between a pair of Paths

Path Similarity

Similarity between two paths p_1 and p_2 :

$$S(p_1, p_2) = \sqrt{\text{sim}(\text{Slot}X_1, \text{Slot}X_2) \times \text{sim}(\text{Slot}Y_1, \text{Slot}Y_2)}$$

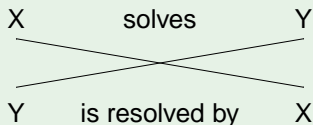
- Geometric average of the SlotX and SlotY similarities of the two paths.

Inverse Paths

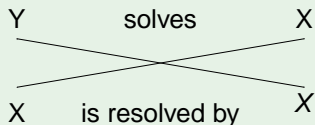
For each path that can be extracted from a dependency tree, an **inverse path** is also added to the set of paths to be compared.

Example

X solves Y
Y is resolved by X



Y solves X
X is resolved by X



This is necessary because the algorithm only compares X with X slots and Y with Y slots. The alignment in the example shows which slots will be similar. So, we get 'double' results.

Finding the most similar Patterns

- Large number of paths in the database
→ How to find most similar patterns in an efficient way?
- Given a path p :
 - 1 Retrieve all paths that share at least one slot filler value with p
→ candidate paths (c).
 - 2 Count the number of slot fillers shared by p and c , filter out c if number of common features is too small.
(Pre-filtering → less costly than computing MI / similarity)
 - 3 Compute similarity between p and c → output is a ranked list.

Example

X solves Y: X resolves Y, X find a solution to Y, X overcomes Y,...

Experimental Results

- Perform DIRT algorithm on 1GB of newspaper text, input: patterns of the first six questions in the TREC-8 Question-Answering Track.
- Compare with a set of human-generated paraphrases.
- Manually inspect the top 40 outputs for each input path (judge correct / incorrect)

Experimental Results (ctd.)

Evaluation of top-40 most similar paths				
	Input Pattern	humans*	DIRT*	correct
Q1	X is author of Y	7	21	52.5%
Q2	X is monetary value of Y	6	0	N/A
Q3	X manufactures Y	13	37	92.5%
Q4	X spend Y	7	16	40.0%
	spend X on Y	8	15	37.5%
Q5	X is managing director of Y	5	14	35.0%
Q6	X asks Y	2	23	57.5%
	asks X for Y	2	14	35.0%
	X asks for Y	3	21	52.5%
		5.9	17.8	50.3%

*number of correct patterns found

DIRT: out of the top-40 list

Experimental Results (ctd.)

Observations:

- Little overlap between manually generated and machine generated phrases. \Rightarrow Paraphrase generation is difficult both for humans and machines.
- Output of DIRT System: Humans can easily identify correct phrases (get a more complete list)
 \Rightarrow DIRT helps humans to build paraphrase knowledge bases.

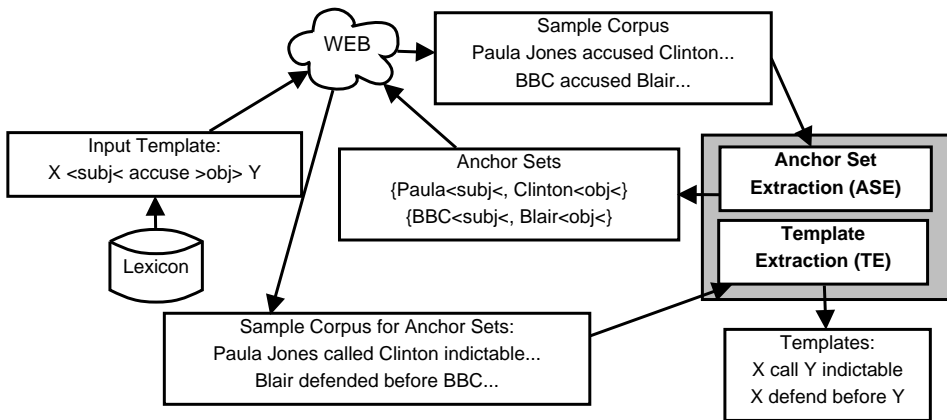
Question to be addressed:

- Polarity in sentences (e.g. 'X worsens Y' has a high similarity to 'X solves Y')

Outline

- 1 Introduction
 - Paraphrases
 - Areas of Application
 - Automatic Acquisition of Paraphrases
- 2 DIRT
 - Paths in Dependency Trees
 - Similarity Measure
 - Finding Patterns
 - Experimental Results
- 3 **TEASE**
 - Overview
 - Anchor Set Extraction (ASE)
 - Template Extraction (TE)
 - Experimental Results
- 4 Conclusion

TE/ASE System Overview



Scaling Web-based Acquisition of Entailment Relations, Szpektor et al., 2004

(Diagram inspired by TE/ASE presentation of Idan Szpektor)

TE/ASE Overview (ctd.)

- ! Unlike in the DIRT approach, here anchors are always used as pairs.

Example

{aspirine (subj), headache (obj)}

'Aspirine prevents headaches.'

'Aspirine reduces headaches.'

{aspirine (subj), heart attack (obj)}

'Aspirine prevents heart attacks.'

Anchor Set Extraction (ASE)

For each pivot P :

- Create a **pivot template** T_P .

For verbs: add subj and obj slots.

Example

Pivot: $P = \text{arrest}$

$$T_P = 'X \xleftarrow{\text{subj}} \text{arrest} \xrightarrow{\text{obj}} Y'$$

Anchor Set Extraction (ctd.)

- Construct a sample corpus S for T_P :
 - Retrieve an initial sample from the web (= sentences containing P)

Example

Police arrests John Lennon for drug charges.
Whitney Houston was arrested at the airport.

...

- Only keep phrases where all slots in the template structure are instantiated:

Example

Police arrests John Lennon for drug charges.
~~Whitney Houston was arrested at the airport.~~

...

Anchor Set Extraction (ctd.)

- Construct a sample corpus S for T_P :
 - Retrieve an initial sample from the web (= sentences containing P)

Example

Police arrests John Lennon for drug charges.
Whitney Houston was arrested at the airport.

...

- Only keep phrases where all slots in the template structure are instantiated:

Example

Police arrests John Lennon for drug charges.
~~Whitney Houston was arrested at the airport.~~

...

Anchor Set Extraction (ctd.)

- Identify **associated phrases** for the pivot (measure tf.idf for NP in the sample sentences).
- Extend sample corpus *S*: Query web with pivot and each of the associated phrases.

Example

Assume 'drug charges' is an associated NP for 'arrest'.

Query web with 'arrest drug charges'.

Police arrests John Lennon for drug charges.

Officers arrest Whitney Houston for drug charges.

...

- Retrieve and **filter anchor sets**.
 - Don't keep anchor sets which are too frequent on the web.
 - Only keep anchor sets that are associated with the pivot to a certain degree.

Anchor Set Extraction (ctd.)

- Identify **associated phrases** for the pivot (measure tf.idf for NP in the sample sentences).
- Extend sample corpus *S*: Query web with pivot and each of the associated phrases.

Example

Assume 'drug charges' is an associated NP for 'arrest'.

Query web with 'arrest drug charges'.

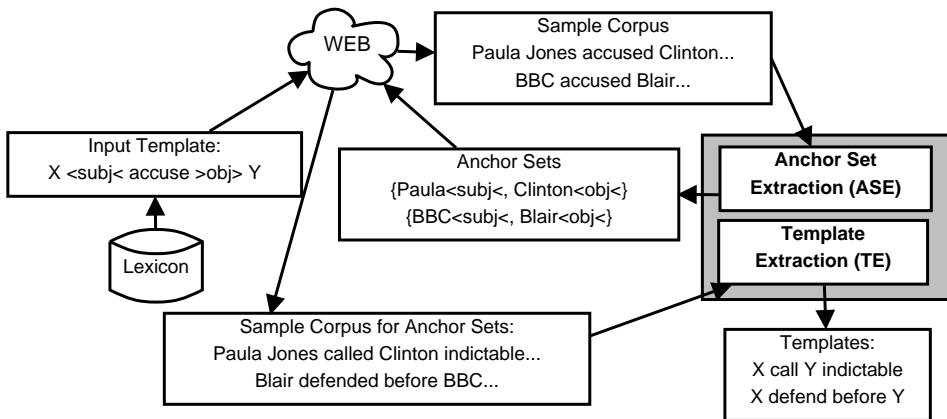
Police arrests John Lennon for drug charges.

Officers arrest Whitney Houston for drug charges.

...

- Retrieve and **filter anchor sets**.
 - Don't keep anchor sets which are too frequent on the web.
 - Only keep anchor sets that are associated with the pivot to a certain degree.

TE/ASE System Overview



Scaling Web-based Acquisition of Entailment Relations, Szpektor et al., 2004

(Diagram inspired by TE/ASE presentation of Idan Szpektor)

Template Extraction (TE)

Example

Anchor Sets for 'arrest':

{Police \xleftarrow{subj} ; John Lennon \xleftarrow{obj} }

{Officers \xleftarrow{subj} ; Whitney Houston \xleftarrow{obj} }

- 1 Acquire sample corpus from Web for each input anchor set.

Example

(1) Police takes John Lennon into custody.

(2) Police busts John Lennon for drug charges.

...

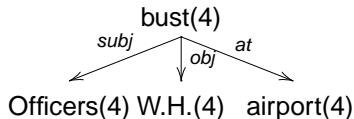
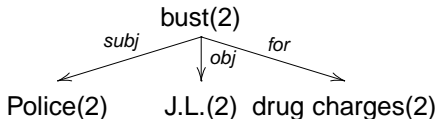
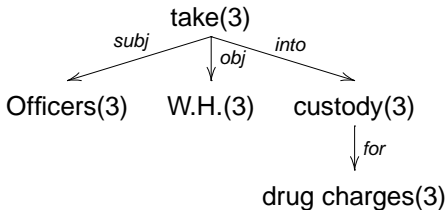
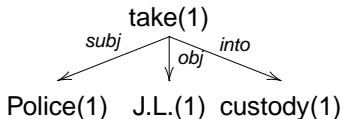
(3) Officers take Whitney Houston into custody for drug charges.

(4) Officers bust Whitney Houston at the airport.

...

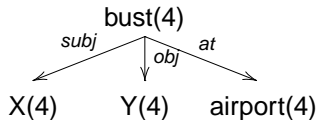
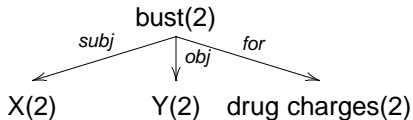
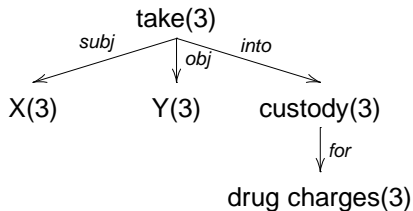
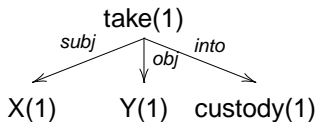
Template Extraction (TE)

- Dependency Parsing
- Associate sentence set (= **support set**) with each node.



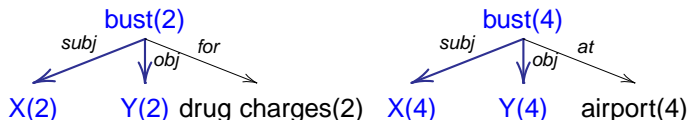
Template Extraction (TE)

- Substitute anchor slots with slot variable (X or Y)

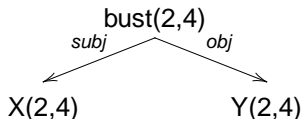


Extraction of maximal most general templates

- **Spanning Template:** smallest subtree spanning over X and Y



Intuitively, the parse tree:



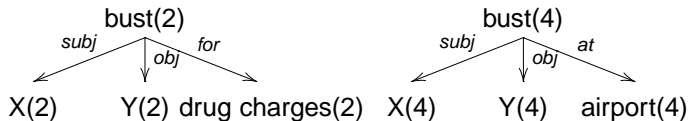
is included in both parse trees (for (2) and (4)).

⇒ How do we find it?

Extraction of maximal most general templates (1)

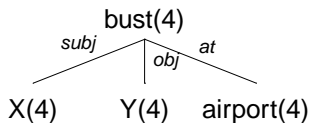
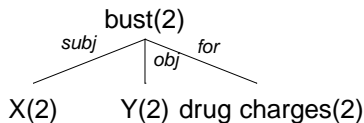
We start with an empty graph and add all available spanning templates to it one by one.

Step 1: Union of the two input graphs (one of them is the new spanning template).



Extraction of maximal most general templates (2)

Step 2: For every two nodes having the same label, add a new **generalized node** to the graph. Same label, union of sentence sets.



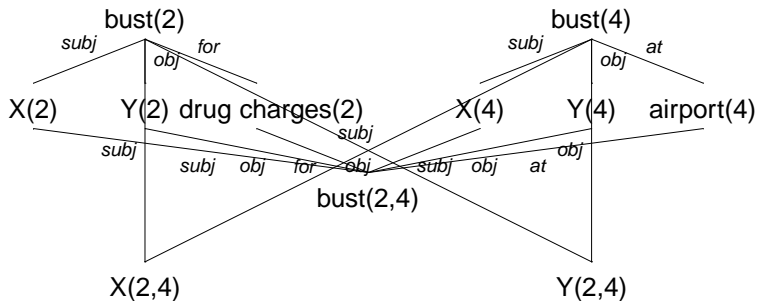
bust(2,4)

X(2,4)

Y(2,4)

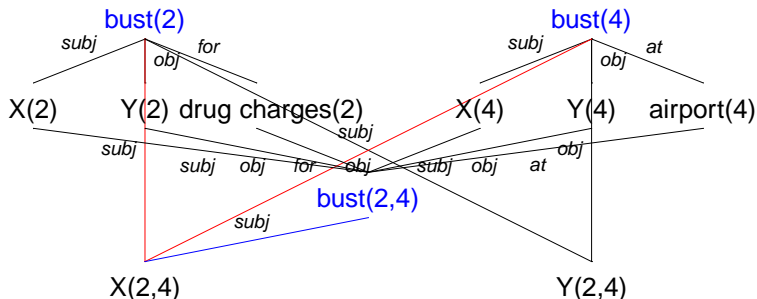
Extraction of maximal most general templates (3)

Step 3: Connect the new node with all nodes that are connected with the merged nodes.



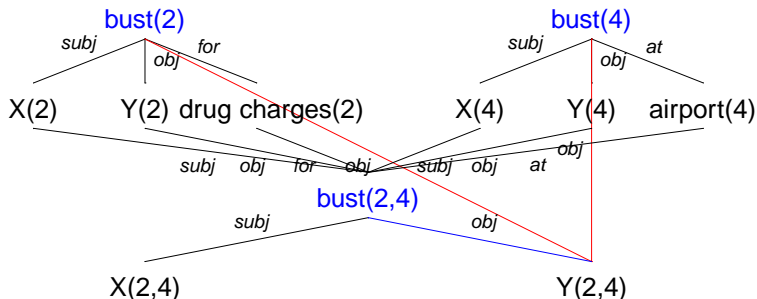
Extraction of maximal most general templates (4)

Step 4: Merge all arcs that have the same label and whose corresponding nodes have been merged. Connect the new arc to the merged node.



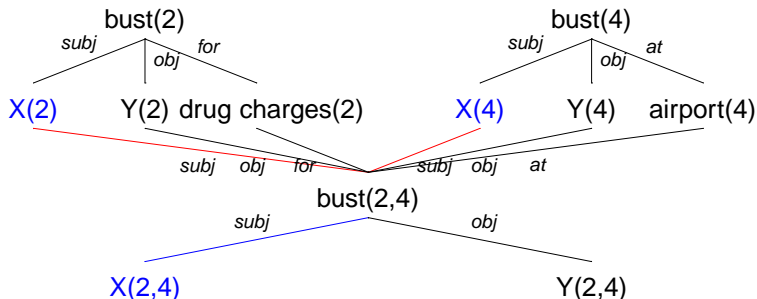
Extraction of maximal most general templates (4)

Step 4: Merge all arcs that have the same label and whose corresponding nodes have been merged. Connect the new arc to the merged node.



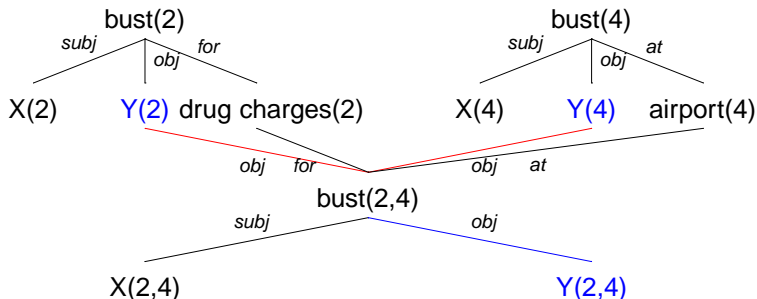
Extraction of maximal most general templates (4)

Step 4: Merge all arcs that have the same label and whose corresponding nodes have been merged. Connect the new arc to the merged node.



Extraction of maximal most general templates (4)

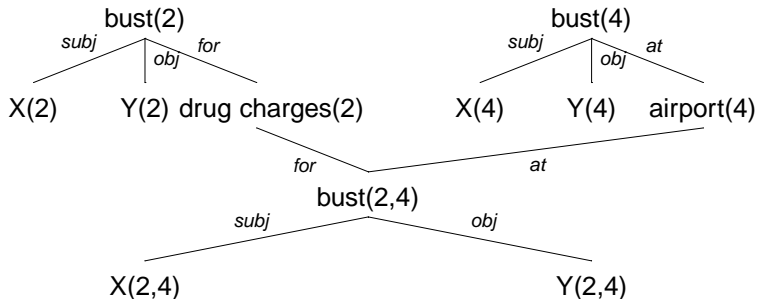
Step 4: Merge all arcs that have the same label and whose corresponding nodes have been merged. Connect the new arc to the merged node.



Extraction of maximal most general templates (4)

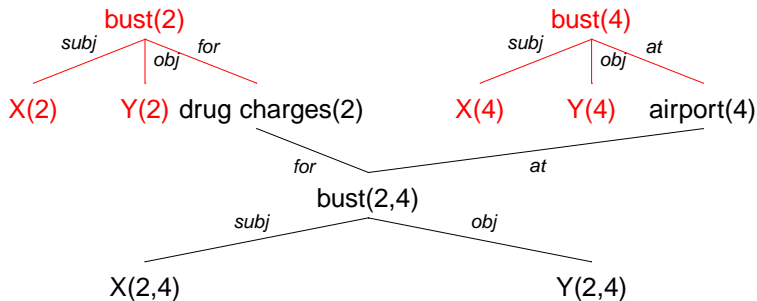
Step 4: Merge all arcs that have the same label and whose corresponding nodes have been merged. Connect the new arc to the merged node.

Result of Step 4:



Extraction of maximal most general templates (5)

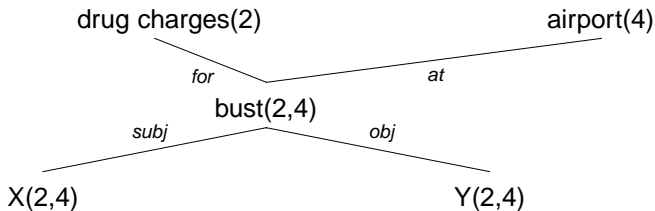
Step 5: Delete nodes which have been merged (and the arcs connected to them).



Extraction of maximal most general templates (5)

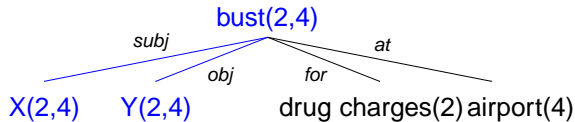
Step 5: Delete nodes which have been merged (and the arcs connected to them).

Result of Step 5:

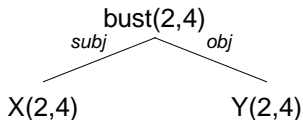


Extraction of maximal most general templates (6)

Step 6: Look at slots (X and Y) and find the **minimal spanning template** containing them.

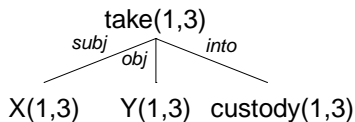
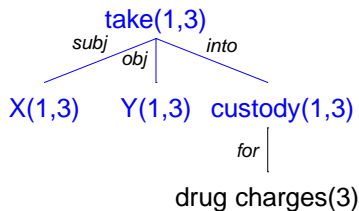


⇒ Extracted template:



Extraction of maximal most general templates (7)

Step 7: Expand the minimal spanning tree to its largest sub-graph that has the same sentence set. *Extracted Template:*

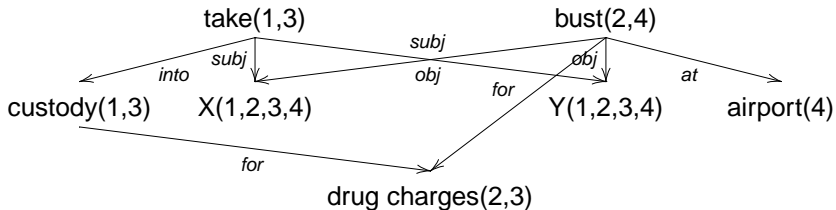


Maximal most general template

- may not contain more general templates (i.e. sub-trees that have a larger sentence set)
- should not be part of a larger template that has the same sentence set

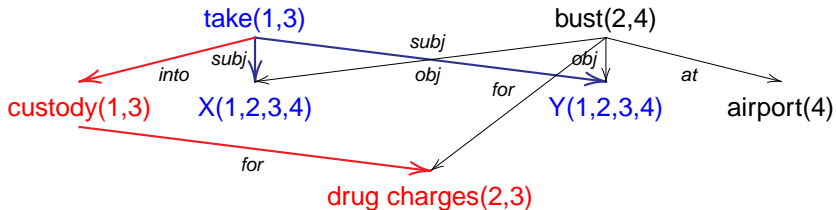
Compact Graph Representation

What we actually get when merging all templates are directed graphs like the following:



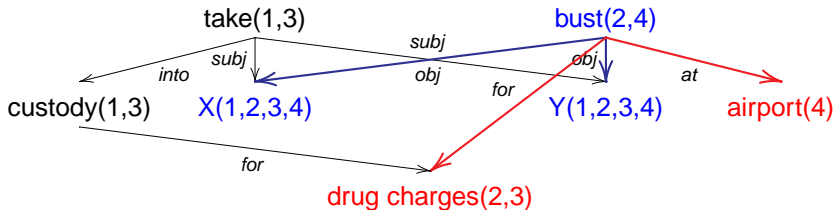
Compact Graph Representation

So we need to extract all minimal spanning templates for X and Y, and then go on as described.



Extraction of maximal most general templates (9)

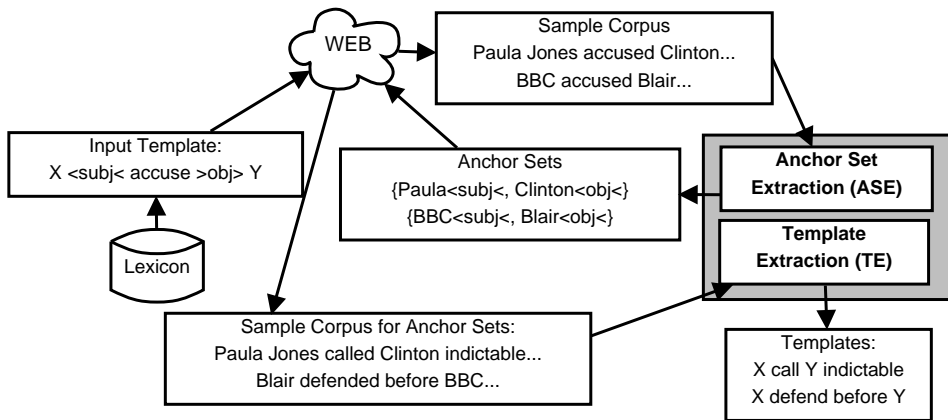
So we need to extract all minimal spanning templates for X and Y, and then go on as described.



TE: Ranking of the extracted Templates

- 1 Sort by number of different anchor sets supporting a template.
- 2 Sort by number of sentences supporting a template.

TE/ASE System Overview



Scaling Web-based Acquisition of Entailment Relations, Szpektor et al., 2004

(Diagram inspired by TE/ASE presentation of Idan Szpektor)

Experimental Results

- Human judges evaluate 752 templates retrieved for 53 pivot lexicon entries → Correct / Incorrect / No evaluation possible
- Average: 5.5 (correct) templates per verb found
- Average precision: 44.15% (percentage of good templates out of all extracted templates (per verb))
- Correct templates found for 86% of the verbs (46 out of 53)

Outline

- 1 Introduction
 - Paraphrases
 - Areas of Application
 - Automatic Acquisition of Paraphrases
- 2 DIRT
 - Paths in Dependency Trees
 - Similarity Measure
 - Finding Patterns
 - Experimental Results
- 3 TEASE
 - Overview
 - Anchor Set Extraction (ASE)
 - Template Extraction (TE)
 - Experimental Results
- 4 Conclusion

Conclusion

- We have seen two approaches for learning Paraphrase Patterns:
 - DIRT: Computing similarity between slots of dependency paths (By which sets of words are they filled?).
 - TEASE: Retrieve anchor sets from web and then find templates that occur together with these anchor sets.
- Performance is about the same (experimental results cannot be compared directly):
 - find about 5 / 17 templates per verb
 - average precision 45-50%
- It is easy for humans to judge the extracted templates, but hard to come up with a complete list.

Outlook

- DIRT and TEASE approaches only learn *binary* templates with exactly one subject and one object slot. → Methods to learn templates with an arbitrary number of slots?
- What about the **direction** of the entailment relation?
→ Next session!

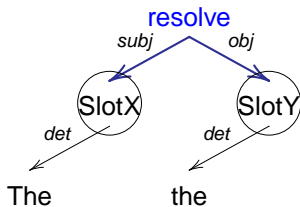
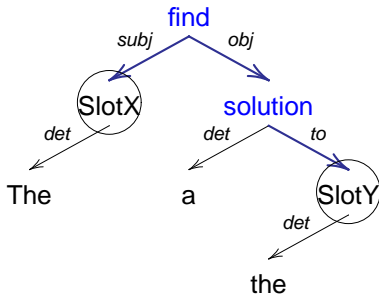
Thanks for your attention!

References

- DIRT - Discovery of Inference Rules from Text, D. Lin & P. Pantel, University of Alberta, 2001
- Scaling Web-based Acquisition of Entailment Relations (TE/ASE), Idan Szpektor et. al., Tel Aviv University, 2004
- Discovery of Inference Rules for Question Answering, D.Lin & P. Pantel, University of Alberta, 2001

BACKUP SLIDES

Paths in Dependency Trees - Naming



$N:\text{subj}:V \leftarrow \text{resolve} \rightarrow V:\text{obj}:N$

$N:\text{subj}:V \leftarrow \text{find} \rightarrow V:\text{obj}:N \leftarrow \text{solution} \rightarrow N:\text{to}:N$

- Needed when counting occurrences of paths in corpus
 → hash table keys → efficient access

Anchor Set Extraction (ctd.)

- Identify associated phrases for the pivot.
 - Discard noun phrases which are too common on the web:
 $freq_W(X) > MAXPHRASEF$
 - Compute *tf.idf* for all remaining noun phrases.

tf.idf

$$tf.idf = freq_S(X) * \log\left(\frac{N}{freq_W(X)}\right)$$

$freq_S(X)$ = number of items in sample corpus S containing
noun phrase X

$freq_W(X)$ = number of web documents containing X

N = number of web documents (estimation)