Paraphrasing

- In natural language, there are many different ways to say the same thing.
- Some NLP systems aim to learn **paraphrases** for a given phrase, usually to identify relation expressions.

  X manufactures Y *X makes Y*

  X produces Y *Y is a product of X*

- In addition, there are often phrases that suggest a specific relation even though it is not explicit and it may not always be true.

  X's Y factory → *X manufactures Y*

Discovering Inference Rules

- The DIRT (Discovery of Inference Rules from Text) algorithm was developed by Lin and Pantel in 2001.
- DIRT's goal is to learn expressions that link two nouns and for which a relationship can be inferred. This includes both paraphrases and expressions that *suggest* a relation.
- INPUT: an expression representing a relation, such as *"X manufactures Y"*
- OUTPUT: a set of expressions from which the same relationship can be inferred.

Key Idea

- The **Distributional Hypothesis**: words that occur in the same contexts tend to have similar meanings.
- Previously, **distributional similarity** methods had been used to identify semantically similar words based on the contexts around those words.
- The DIRT algorithm flips this approach by applying distributional similarity measures to identify semantically related paths in dependency trees.
- Each parse tree path links two nouns, which are treated as the contexts for measuring similarity of the paths.

Extended Distributional Hypothesis

Extended Distributional Hypothesis: If two paths occur in similar contexts, the meanings of the paths tends to be similar.

Intuitively, similar paths should have similar slot fillers.

"X finds a solution to Y"		*"X solves Y"*	
SLOTX	*SLOTY*	*SLOTX*	*SLOTY*
commission	strike	committee	problem
committee	civil war	clout	crisis
committee	crisis	government	problem
government	crisis	he	mystery
government	problem	she	problem
he	problem	petition	woe
legislator	budget deficit	researcher	mystery
sheriff	dispute	sheriff	murder

DIRT's Algorithm

1. Generate a set of paths through dependency trees that satisfy a few constraints. In particular, they must link two nouns.
2. For each path, generate triples that represent the "slot fillers" at the ends of the path.
3. Create a function that computes the similarity of two paths based on the set of triples (slot fillers) associated with them.
4. Given an input phrase, find the paths that are most similar to the input phrase.

Paths in Dependency Trees

- A dependency parser is applied to a large text corpus and every path that links two nouns is extracted.
- The noun on the left is called SlotX, and the noun on the right is called SlotY. So the position of each slot filler matters!
- Only the dependency relations linking two content words (*nouns, verbs, adjectives, adverbs*) are used.
- Each path is named by concatenating the dependency relations and words along the path.

Dependency Path Example

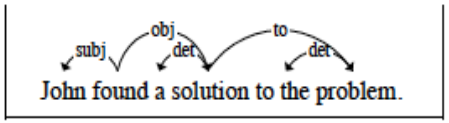


Figure 1. Example dependency tree.

The path between **SlotX**=*John* and **SlotY**=*problem* would be:

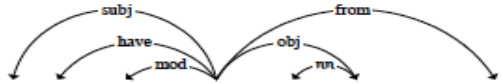
(**SlotX**) N:*subj*:V←find→V:*obj*:N→solution→N:*to*:N (**SlotY**)

The path between **SlotX**=*problem* and **SlotY**=*John* would be:

(**SlotX**) N:*to*:N←solution←N:*obj*:V←find→V:*subj*:N (**SlotY**)

All Possible Paths are Extracted

Consider the following sentence:



The paths extracted from this sentence and their meanings are:

(a) N:subj:V←buy→V:from:N
≡ *X* buys something from *Y*

(b) N:subj:V←buy→V:obj:N
≡ *X* buys *Y*

(c) N:subj:V←buy→V:obj:N→sheep→N:nn:N
≡ *X* buys *Y* sheep

(d) N:nn:N←sheep←N:obj:V←buy→V:from:N
≡ *X* sheep is bought from *Y*

(e) N:obj:V←buy→V:from:N
≡ *X* is bought from *Y*

An inverse path is also added for each one above.

Triple Database

All of the extracted paths are stored in a **Triple Database**, paired with the nouns in the SlotX and SlotY positions.

For example, the previous example sentence would yield these triples for the triple database:

SlotX	Path	SlotY
They	N:*subj*:V←buy→V:*from*:N	*Comstock*
They	N:*subj*:V←buy→V:*obj*:N	*sheep*
They	N:*subj*:V←buy→V:*obj*:N→sheep→N:nn:N	*bighorn*
bighorn	N:nn:N←sheep←N:*obj*:V←buy→V:*from*:N	*Comstock*
sheep	N:*obj*:V←buy→V:*from*:N	*Comstock*

Path Similarity

- Given a phrase as input, DIRT measures the similarity of the phrase with each path in the Triple Database.
- The most paths are ranked based on their similarity scores and the Top *k* most similar paths are returned.
- The similarity between a pair of paths p_1 and p_2 is the geometric mean of their SlotX and SlotY similarity scores.

$$\text{PathSim}(p1,p2) = \sqrt{\text{sim}(\text{SlotX}_1,\text{SlotX}_2) * \text{sim}(\text{SlotY}_1,\text{SlotY}_2)}$$

Slot Similarity

Given a pair of slots: $slot_1 = (p_1, s)$ and $slot_2 = (p_2, s)$ their similarity is defined as the ratio of the mutual information of their shared slot fillers over the mutual information of all their slot fillers.

$T(p_i,s)$ is the set of words that fill slot s in path p.

$$\text{sim}(slot_1, slot_2) = \frac{\sum_{w \in T(p_1,s) \cap T(p_2,s)} \text{MI}(p_1, s, w) + \text{M1}(p_2, s, w)}{\sum_{w \in T(p_1,s)} \text{MI}(p_1, s, w) + \sum_{w \in T(p_2,s)} \text{M1}(p_2, s, w)}$$

Pointwise Mutual Information

Pointwise mutual information (PMI) measures the degree to which two words are statistically dependent.

$$\text{PMI}(w_1, w_2) = \log_2 \left[\frac{P(w_1 \,\&\, w_2)}{P(w_1) * P(w_2)} \right]$$

DIRT uses mutual information to measure the strength of association between a path and its slot filler:

$$\text{MI}(p, Slot, w) = \log_2 \left[\frac{|\, p, Slot, w \,|\ \text{X}\ |\, *, Slot, * \,|}{|\, p, Slot, * \,|\ \text{X}\ |\, *, Slot, w \,|} \right]$$

Mutual Information Example

MI (*will buy*, SlotX, *Trump*) =

$$\log_2 \left[\frac{|\ will\ buy,\ SlotX,\ Trump\ |\ \times\ |\ *,\ SlotX,\ *\ |}{|\ will\ buy,\ SlotX,\ *\ |\ \times\ |\ *,\ SlotX,\ Trump\ |} \right]$$

where:

|*will buy*, *SlotX*, *Trump*| = # times *Trump* fills SlotX for path *will buy*

|*, *SlotX*, *| = total # fillers for SlotX in all paths

|*will buy*, *SlotX*, *| = # fillers in SlotX for path *will buy*

|*, *SlotX*, *Trump*|= # times *Trump* fills SlotX in all paths

Example Entries in Triple Database

The mutual information scores between a path and each slot filler are stored in the triple database.

X pulls body from Y:			
SlotX:			
	diver	1	2.45
	equipment	1	1.65
	police	2	2.24
	rescuer	3	4.84
	resident	1	1.60
	who	2	1.32
	worker	1	1.37
SlotY:			
	bus	2	3.09
	coach	1	2.05
	debris	1	2.36
	feet	1	1.75
	hut	1	2.73
	landslide	1	2.39
	metal	1	2.09
	wreckage	3	4.81

Heuristics for Efficiency

- Computing the similarity with every path is expensive! Lin & Pantel parsed 1Gb of news texts, which produced 7 million path instances and 231,000 distinct paths.
- They used several heuristics to only consider promising paths. Given a path *p*:
  - Retrieve all paths that share at least one filler with *p*
  - Discard candidate paths that share < 1% of the same fillers as *p*
- Compute the similarity of *p* with all remaining candidate paths and output the Top *k* most similar paths.

Examples of Similar Paths

Table 3. The top-20 most similar paths to "X solves Y".

Y is solved by *X*	*Y* is resolved in *X*
X resolves *Y*	*Y* is solved through *X*
X finds a solution to *Y*	*X* rectifies *Y*
X tries to solve *Y*	*X* copes with *Y*
X deals with *Y*	*X* overcomes *Y*
Y is resolved by *X*	*X* eases *Y*
X addresses *Y*	*X* tackles *Y*
X seeks a solution to *Y*	*X* alleviates *Y*
X do something about *Y*	*X* corrects *Y*
X solution to *Y*	*X* is a solution to *Y*

Experimental Results

They evaluated DIRT by comparing the inference rules that it produced with human-generated paraphrases for 6 questions in the TREC-8 question answering track.

Q_1	Who is the author of the book, "The Iron Lady: A Biography of Margaret Thatcher"?
Q_2	What was the monetary value of the Nobel Peace Prize in 1989?
Q_3	What does the Peugeot company manufacture?
Q_4	How much did Mercury spend on advertising in 1993?
Q_5	What is the name of the managing director of Apricot Computer?
Q_6	Why did David Koresh ask the FBI for a word processor?

Results for Q/A Evaluation

For each question, a dependency path was produced and the Top 40 most similar paths were generated by DIRT. These paths were manually labeled as correct or incorrect.

They were also compared with manually generated paraphrases to see if additional rules were learned (INT = intersection).

Q#	PATHS	MAN.	DIRT	INT.	ACC.
Q_1	X is author of Y	7	21	2	52.5%
Q_2	X is monetary value of Y	6	0	0	N/A
Q_3	X manufactures Y	13	37	4	92.5%
Q_4	X spend Y	7	16	2	40.0%
	spend X on Y	8	15	3	37.5%
Q_5	X is managing director of Y	5	14	1	35.0%
Q_6	X asks Y	2	23	0	57.5%
	asks X for Y	2	14	0	35.0%
	X asks for Y	3	21	3	52.5%

Examples of Learned Rules

PATHS	*X* is author of *Y*	*X* manufactures *Y*
MANUAL VARIATIONS	*Y* is the work of *X*; *X* is the writer of *Y*; *X* penned *Y*; *X* produced *Y*; *X* authored *Y*; *X* chronicled *Y*; *X* wrote *Y*	*X* makes *Y*; *X* produce *Y*; *X* is in *Y* business; *Y* is manufactured by *X*; *Y* is provided by *X*; *Y* is *X*'s product; *Y* is product from *X*; *Y* is *X* product; *Y* is product made by *X*; *Y* is example of *X* product; *X* is manufacturer of *Y*; find *Y* in *X*'s product line; find *Y* in *X* catalog
DIRT VARIATIONS	X co-authors Y; X is co-author of Y; X writes Y; X edits Y; Y is co-authored by X; Y is authored by X; X tells story in Y; X translates Y; X writes in Y; X notes in Y; …	X produces Y; X markets Y; X develops Y; X is supplier of Y; X ships Y; X supplies Y; Y is manufactured by X; X is maker of Y; X introduces Y; X exports Y; X makes Y; X builds Y; X's production of Y; X unveils Y; Y is bought from X; X's line of Y; X assembles Y; X is Y maker; X's Y factory; X's Y production; X is manufacturer of Y; X's Y division; X meets demand for Y; …

Summary

- DIRT applies distributional similarity measures in a new way: to identify similar relationship phrases by considering their slot fillers (arguments) as the contexts.
- It is difficult for people to manually generate long lists of paraphrases, so automatically generating them is important.
- DIRT worked best for paths representing verb arguments.
- This approach can work well in many cases, but is not sufficient to distinguish paths that can take the same sets of fillers.