Outline of today’s lecture

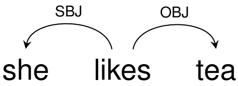
Introduction to dependency structures for syntax

Word order across languages

Dependency parsing

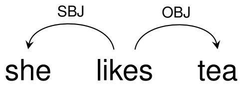
Universal dependencies

Dependency structures

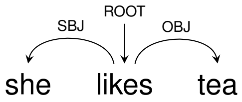


- Relate words to each other via labelled directed arcs (dependencies).
- Lots of variants: in NLP, usually weakly-equivalent to a CFG, with ROOT node.

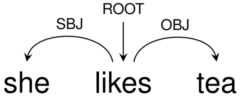
Dependency structures

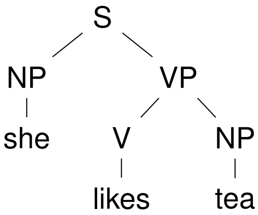


- Relate words to each other via labelled directed arcs (dependencies).
- Lots of variants: in NLP, usually weakly-equivalent to a CFG, with ROOT node.



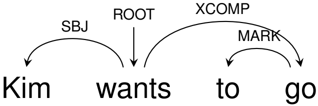
Dependency structures vs trees





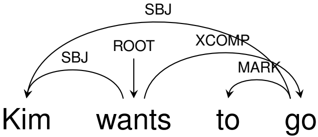
- No direct notion of constituency in dependency structures:
  - + constituency varies a lot between different approaches.
  - - can't model some phenomena so directly/easily.
- Dependency structures intuitively closer to meaning.
- Dependencies are more neutral to word order variations.

Non-tree dependency structures



XCOMP: clausal complement, MARK: marker (semantically empty)

But *Kim* is also the agent of *go*.



But this is not a tree . . .

English word order

subject verb object (SVO)
'who did what to whom' indicated by order

The dog bites that man
That man bites the dog

Also, in right context, topicalization:
That man, the dog bites

Passive has different structure:
The man was bitten by the dog

Word order variability

Many languages mark case and allow freer word order:

Der Hund beißt den Mann
Den Mann beißt der Hund
both mean ‘the dog bites the man’

BUT only masc gender changes between nom/acc in German:
Die Kuh hasst eine Frau — only means ‘the cow hates a woman’

Case and word order in English

Even when English marks case, word order is fixed:
* him likes she

But weird order is comprehensible:
found someone, you have

* (unless +YODA — linguist's joke . . .)

More about Yodaspeak:
https://www.theatlantic.com/entertainment/archive/2015/12/hmmmmm/420798/

Case and word order in English

Even when English marks case, word order is fixed:
* him likes she

But weird order is comprehensible:
found someone, you have

* (unless +YODA — linguist's joke . . .)

More about Yodaspeak:
`https://www.theatlantic.com/entertainment/archive/2015/12/hmmmmm/420798/`

Free word order languages

Russian example (from Bender, 2013):

Chelovek	ukusil	sobaku
man.NOM.SG.M	bite.PAST.PFV.SG.M	dog-ACC.SG.F

the man bit the dog

All word orders possible with same meaning (in different discourse contexts):

Chelovek ukusil sobaku
Chelovek sobaku ukusil
Ukusil chelovek sobaku
Ukusil sobaku chelovek
Sobaku chelovek ukusil
Sobaku ukusil chelovek

Word order and CFG

Because of word order variability, rules like:

```
S -> NP VP
```

do not work in all languages.

Options:

- ignore the order of the rule's daughters, and allow discontinuous constituency e.g., VP is split for sobaku chelovek ukusil ('dog man bit') etc. Parsing is difficult.
- Use richer frameworks than CFG (e.g., feature-structure grammars — see Bender (ACL 2008) on Wambaya)
- dependencies

Dependency parsing

- For NLP purposes, we assume structures which are weakly-equivalent to CFGs.
- Some work on adding arcs for non-tree cases like want to go in a second phase.
- Different algorithms: here transition-based dependency parsing, a variant of shift-reduce parsing.
- Trained on dependency-banks (possibly acquired by converting treebanks).

Transition-based dependency parsing (without labels)

- Deterministic: at each step either SHIFT a word onto the stack, or link the top two items on the stack (LeftArc or RightArc).
- Retain the head word only after a relation added.
- Finish when nothing in the word list and only ROOT on the stack.
- Oracle chooses the correct action each time (LeftArc, RightArc or SHIFT).

Transition-based dependency parsing example

stack	word list	action	relation added
ROOT	she, likes, tea	SHIFT	
ROOT, she	likes tea	SHIFT	
ROOT, she, likes	tea	LeftArc	she $\leftarrow$ likes
ROOT, likes	tea	SHIFT	
ROOT, likes, tea		RightArc	likes $\rightarrow$ tea
ROOT, likes		RightArc	ROOT $\rightarrow$ likes
ROOT		Done	

Transition-based dependency parsing example

stack	word list	action	relation added
ROOT	she, likes, tea	SHIFT	
ROOT, she	likes tea	SHIFT	
ROOT, she, likes	tea	LeftArc	she $\leftarrow$ likes
ROOT, likes	tea	SHIFT	
ROOT, likes, tea		RightArc	likes $\rightarrow$ tea
ROOT, likes		RightArc	ROOT $\rightarrow$ likes
ROOT		Done	

Transition-based dependency parsing example

stack	word list	action	relation added
ROOT	she, likes, tea	SHIFT	
ROOT, she	likes tea	SHIFT	
ROOT, she, likes	tea	LeftArc	she $\leftarrow$ likes
ROOT, likes	tea	SHIFT	
ROOT, likes, tea		RightArc	likes $\rightarrow$ tea
ROOT, likes		RightArc	ROOT $\rightarrow$ likes
ROOT		Done	

Transition-based dependency parsing example

stack	word list	action	relation added
ROOT	she, likes, tea	SHIFT	
ROOT, she	likes tea	SHIFT	
ROOT, she, likes	tea	LeftArc	she $\leftarrow$ likes
ROOT, likes	tea	SHIFT	
ROOT, likes, tea		RightArc	likes $\rightarrow$ tea
ROOT, likes		RightArc	ROOT $\rightarrow$ likes
ROOT		Done	

Transition-based dependency parsing example

stack	word list	action	relation added
ROOT	she, likes, tea	SHIFT	
ROOT, she	likes tea	SHIFT	
ROOT, she, likes	tea	LeftArc	she $\leftarrow$ likes
ROOT, likes	tea	SHIFT	
ROOT, likes, tea		RightArc	likes $\rightarrow$ tea
ROOT, likes		RightArc	ROOT $\rightarrow$ likes
ROOT		Done	

Transition-based dependency parsing example

stack	word list	action	relation added
ROOT	she, likes, tea	SHIFT	
ROOT, she	likes tea	SHIFT	
ROOT, she, likes	tea	LeftArc	she $\leftarrow$ likes
ROOT, likes	tea	SHIFT	
ROOT, likes, tea		RightArc	likes $\rightarrow$ tea
ROOT, likes		RightArc	ROOT $\rightarrow$ likes
ROOT		Done	

Transition-based dependency parsing example

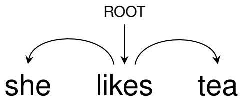
stack	word list	action	relation added
ROOT	she, likes, tea	SHIFT	
ROOT, she	likes tea	SHIFT	
ROOT, she, likes	tea	LeftArc	she $\leftarrow$ likes
ROOT, likes	tea	SHIFT	
ROOT, likes, tea		RightArc	likes $\rightarrow$ tea
ROOT, likes		RightArc	ROOT $\rightarrow$ likes
ROOT		Done	

Transition-based dependency parsing example

stack	word list	action	relation added
ROOT	she, likes, tea	SHIFT	
ROOT, she	likes tea	SHIFT	
ROOT, she, likes	tea	LeftArc	she $\leftarrow$ likes
ROOT, likes	tea	SHIFT	
ROOT, likes, tea		RightArc	likes $\rightarrow$ tea
ROOT, likes		RightArc	ROOT $\rightarrow$ likes
ROOT		Done	

Transition-based dependency parsing example

Output: she $\leftarrow$ likes, likes $\rightarrow$ tea, ROOT $\rightarrow$ likes

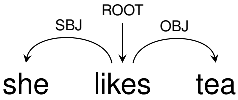


Creating the oracle

- The oracle's decisions are a type of classification: given the stack and the word list, choose an action.
- Supervised machine learnng: trained by extracting parsing actions from correctly annotated data.
- MaxEnt, SVMs, deep learning etc.
- features extracted from the training instances (word forms, morphology, parts of speech etc).
- feature templates: automatically instantiated to give huge number of actual features.
- Labels on arcs increase the number of classes.

Transition-based dependency parsing with labels

R	she_PNP, likes_VVZ, tea_NN1	SHIFT	
R,she_PNP	likes_VVZ, tea_NN1	SHIFT	
R,she_PNP, likes_VVZ	tea_NN1	LASUBJ	she ← likes SUBJ
R,likes_VVZ	tea_NN1	SHIFT	
R,likes_VVZ, tea_NN1		RAObj	likes → tea OBJ
R,likes_VVZ		RightA	ROOT → likes
R		Done	



Dependency parsing

- Dependency parsing can be very fast.
- Greedy algorithm can go wrong, but usually reasonable accuracy (Note that humans process language incrementally and (mostly) deterministically.)
- No notion of grammaticality (so robust to typos and Yodaspeak).
- Decisions sensitive to case, agreement etc via features
  Den Mann beißt der Hund
  choice between LeftArcSubj and LeftArcObj conditioned on case of noun as well as position.

Universal dependencies (UD)

- Ongoing attempt to define a set of dependencies which will work cross-linguistically (e.g., Nivre et al 2016).
- `http://universaldependencies.org`
- Also ‘universal’ set of POS tags.
- UD dependency treebanks for over 50 languages (though most small).
- No single set of dependencies is useful cross-linguistically: tension between universality and meaningful dependencies.

Dependency annotation

- Balance between linguistically-motivated scheme, ease of human annotation, parsing efficiency and so on.
- Some vague ‘catch all’ classes in UD: e.g., MARK.
- Words like English infinitival *to* resist clean classification.
- Many linguistic generalizations can’t be captured by dependencies.
- Semantic dependencies next time (briefly).