Topics in Natural Language Processing

Shay Cohen

Institute for Language, Cognition and Computation

University of Edinburgh

Lecture 5

Administrativia

- Information has been sent over the weekend – anything unclear?
- Class forum is now available (see website)
- I removed anyone who is not enrolled on the class from the mailing list. If you are not enrolled and want to continue receiving emails, please let me know

Last class

Prior conjugacy:

Prior family $P = \{ p(\theta \mid \alpha, \beta) \mid (\alpha, \beta) \in \mathcal{A} \}$. We pick one

Conjugacy: $p(\theta \mid w_1 \ldots w_n) \in P$ $\Leftarrow$ $p(\theta) \in P$

"everything stays in the family"

Structure in NLP:

Sequences
Chunking
segmentation

} B-I-O scheme

applications for all those structures

Parsing

Today's class

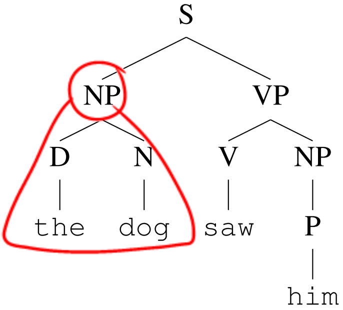
- Structure in NLP
- Grammars in NLP (models for structure)
  - Context-free grammars and their canonical form
  - How to add probabilities?
  - How to do inference with them?

Parsing

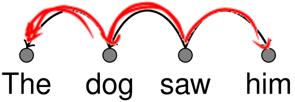
$\Omega = \{(\text{sentences}, \text{parse structures})\}$

Two main types of parsing structures:

- Constituency

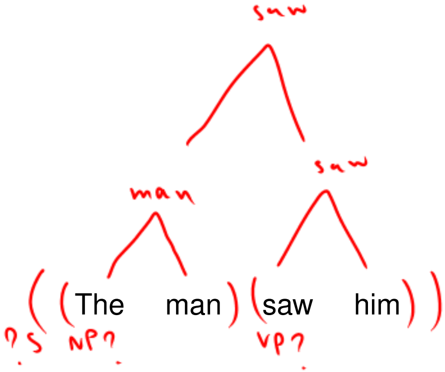


- Dependency



Conversion of dependency to constituency bracketing

Can be done by using head rules

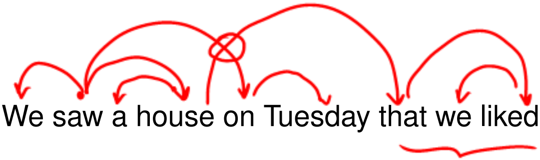


Projective vs. non-projective parsing

Projective trees:

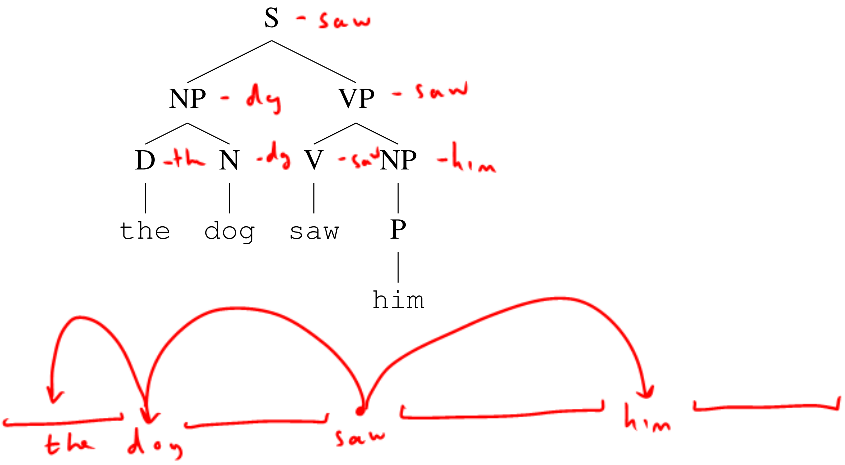
If you draw ~~the~~ edges above the words, they won't cross.

Non-projective trees:



Any constituent tree induces a projective tree

Why do constituent trees induce only projective trees?



Non-projectivity

Not so common in English. Very common in languages such as Dutch and Swiss-German:

mer	em	Hans	s	huus	hälfed	aastriiche
we		Hans	the	house	helped	paint

Sentence in English: We helped Hans paint the house

(Called “cross-serial dependencies”)

Clustering

$\Omega =$ { (collection of entities, ~~partition~~ clustering over these entities) }

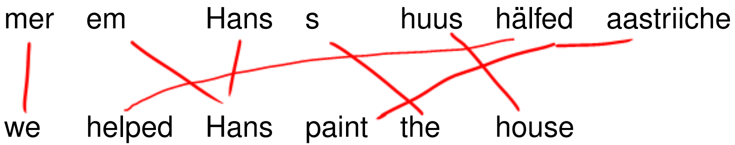
Useful for:

- Word clustering
- Coreference resolution
  - Algorithms for such resolution usually use specialised methods
  - Enforcing transitivity

Alignments

$\Omega = \{$ (pair of objects, alignments between subparts) $\}$
or tuple

Most useful for machine translation



monotonic alignment - no edges cross

Useful for extracting phrases (phrase-based translation)

Most common models for alignment were developed by IBM (“IBM models 1–4”)

Modelling new problems

How do we represent our space of inputs/outputs?

- What is the space used for?
- What are the available inference algorithms?
- Is there a linguistic theory that can help?

Natural language representations

Representations are a moving target

- Linguistic theories develop
- Changes in data and domains
- New algorithms develop, make hard representations feasible

Grammars

What is a grammar? A set of rules that govern language

Grammars

What is a grammar?
A system of rules that govern the production of a language

What is a formal grammar?

Grammars

What is a grammar?
A system of rules that govern the production of a language

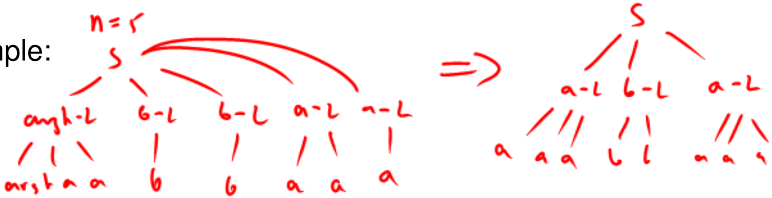
What is a formal grammar?
A formal system of rules that govern the production of a language

Usually describes a step-by-step process

A pre-historic grammar

- Choose a number n
- Generate n labels that can be “argh-phrase” or “blah-phrase”
- Each argh-phrase and blah-phrase generate 1-3 “argh” or “blah”
- Optional: merge identically-labelled consecutive phrases

Example:



Components in a formal grammar

Language: $L(G) = \{ \text{argh}, \text{blah} \}^*$

Structure: $S(G) = \{ \text{argh and blah phrases} \}$

Derivations: Sequence of steps it takes to generate a structure

Context-free grammars

What is a CFG?

A set of nonterminals N

" " " terminals V

Production rules

$A \to \alpha \quad A \in N$

$\alpha \in (N \cup V)^*$

Start symbol $S \in N$

Context-free grammars

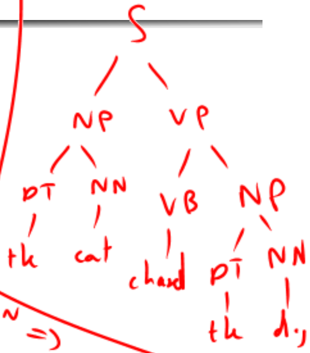
What is a CFG?

- A set of nonterminals N
- A set of vocabulary terminals V
- A start symbol $S \in N$
- A set of production rules R, $a \rightarrow \alpha$ is such that $a \in N$ and $\alpha \in (V \cup N)^*$

CFGs: basic terminology

How to represent a partial derivation?

$S \Rightarrow NP\ VP \Rightarrow DT\ NN\ VP \Rightarrow$ the $NN\ VP \Rightarrow$ the cat $VP \Rightarrow$ the cat $VB\ NP \Rightarrow$ the cat chased $NP \Rightarrow$ the cat chased $DT\ NN \Rightarrow \ldots$ the dog



Two relations: $\Rightarrow$ and $\Rightarrow^*$

$\Rightarrow^*$

$\alpha \Rightarrow^* \beta$

transitive closure of $\Rightarrow$

Canonical forms of grammars

Canonical form: (1) a specific form for writing a grammar; (2) every general CFG can be converted to an "equivalent" canonical form.

Important example: Chomsky normal form (or binarised form)

$A \to B\ C \qquad A, B, C \in N$

$A \to w \qquad A \in N \quad w \in V$

Why is CNF a normal form?

$S \rightarrow NP \;\; VP \;\; NP \;\; PP$

$S \rightarrow NP \;\; X_1$

$X_1 \rightarrow VP \;\; X_2$

$X_2 \rightarrow NP \;\; PP$

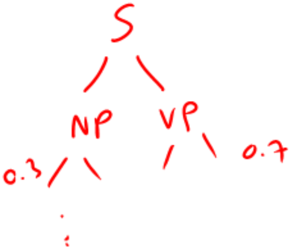
$S \rightarrow Y_1 \;\; PP$

⋮

Probabilistic grammars

$$p(A \to B\ C \mid A) \geq 0$$

$$\sum_{A \to \alpha} p(A \to \alpha \mid A) = 1 \quad \text{for all } A$$



Weighted grammars

Arbitrary positive weights to the rules

$$p(\text{tree}, \text{sentence}) \propto \prod_{\text{rules} \in \text{tree}} w(\text{rule})$$

$$Z(G) = \sum_{\text{tree}} \prod_{\text{rule} \in \text{tree}} w(\text{rule})$$

$$p(\text{tree}) = \frac{\prod w(\text{rule})}{Z(G)}$$

with PCFGs "almost always" $\sum_{\text{tree} \in S(G)} p(\text{tree}) = 1$

Basic inference with grammars

The probability of a derivation:

We estimate a PCFG. How do we parse a sentence?

The CKY algorithm

Succinct representation of the CKY algorithm

$\text{constit}(a, i, j) \vdash \text{constit}(b, i, k), \text{constit}(c, k+1, j), \text{rule}(a \rightarrow b\, c)$

$\text{constit}(a, i, i) \vdash \text{rule}(a \rightarrow w)$

Goal: $\text{constit}(S, 0, n)$

- Weighted logic program
- Can be solved using generic tools

Next class

- Semirings
- Inference in NLP