

Natural Language Understanding

- We want to communicate with computers using natural language (spoken and written).
 - ▶ unstructured natural language — allow any statements, but make mistakes or failure.
 - ▶ controlled natural language — only allow unambiguous statements that can be interpreted (e.g., in supermarkets or for doctors).
- There is a vast amount of information in natural language.
- Understanding language to extract information or answering questions is more difficult than getting extracting gestalt properties such as topic, or choosing a help page.
- Many of the problems of AI are explicit in natural language understanding. “AI complete”.

- **Syntax** describes the form of language (using a grammar).
- **Semantics** provides the meaning of language.
- **Pragmatics** explains the purpose or the use of language (how utterances relate to the world).

Examples:

- *This lecture is about natural language.*
- *The green frogs sleep soundly.*
- *Colorless green ideas sleep furiously.*
- *Furiously sleep ideas green colorless.*

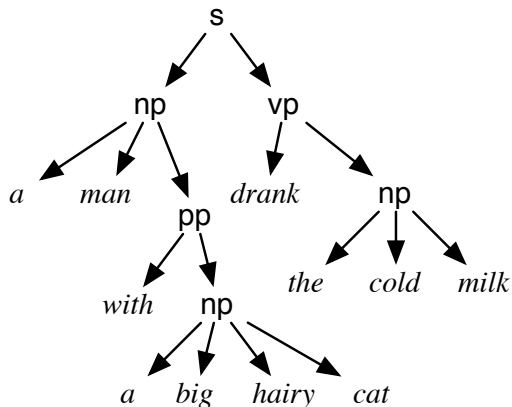
Beyond N-grams

- *A man with a big hairy cat drank the cold milk.*
- Who or what drank the milk?

Beyond N-grams

- *A man with a big hairy cat drank the cold milk.*
- Who or what drank the milk?

Simple syntax diagram:



Context-free grammar

- A **terminal symbol** is a word (perhaps including punctuation).
- A **non-terminal symbol** can be rewritten as a sequence of terminal and non-terminal symbols, e.g.,

$sentence \mapsto noun_phrase, verb_phrase$

$verb_phrase \mapsto verb, noun_phrase$

$verb \mapsto [drank]$

- Can be written as a logic program, where a sentence is a sequence of words:

$sentence(S) \leftarrow noun_phrase(N), verb_phrase(V), append(N, V, S).$

To say word “drank” is a verb:

$verb([drank]).$

Difference Lists

- Non-terminal symbol s becomes a predicate with two arguments, $s(T_1, T_2)$, meaning:
 - ▶ T_2 is an ending of the list T_1
 - ▶ all of the words in T_1 before T_2 form a sequence of words of the category s .
- Lists T_1 and T_2 together form a **difference list**.
- “the student” is a noun phrase:

noun_phrase([*the, student, passed, the, course*],
[*passed, the, course*])

Difference Lists

- Non-terminal symbol s becomes a predicate with two arguments, $s(T_1, T_2)$, meaning:
 - ▶ T_2 is an ending of the list T_1
 - ▶ all of the words in T_1 before T_2 form a sequence of words of the category s .
- Lists T_1 and T_2 together form a **difference list**.
- “the student” is a noun phrase:

$noun_phrase([the, student, passed, the, course],$
 $[passed, the, course])$

- The word “drank” is a verb:

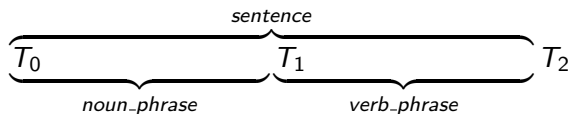
$verb([drank|W], W)$.

Definite clause grammar

The grammar rule

$$\textit{sentence} \mapsto \textit{noun_phrase}, \textit{verb_phrase}$$

means that there is a sentence between T_0 and T_2 if there is a noun phrase between T_0 and T_1 and a verb phrase between T_1 and T_2 :

$$\begin{aligned} \textit{sentence}(T_0, T_2) \leftarrow \\ \textit{noun_phrase}(T_0, T_1) \wedge \\ \textit{verb_phrase}(T_1, T_2). \end{aligned}$$


Definite clause grammar rules

The rewriting rule

$$h \mapsto b_1, b_2, \dots, b_n$$

says that h is b_1 then b_2, \dots , then b_n :

$$\begin{aligned} h(T_0, T_n) \leftarrow \\ & b_1(T_0, T_1) \wedge \\ & b_2(T_1, T_2) \wedge \\ & \vdots \\ & b_n(T_{n-1}, T_n). \end{aligned}$$

using the interpretation

$$\underbrace{\underbrace{T_0}_{b_1} \underbrace{T_1}_{b_2} T_2 \cdots T_{n-1}}_h T_n$$

Terminal Symbols

Non-terminal h gets mapped to the terminal symbols, t_1, \dots, t_n :

$$h([t_1, \dots, t_n | T], T)$$

using the interpretation

$$\overbrace{t_1, \dots, t_n}^h T$$

Thus, $h(T_1, T_2)$ is true if $T_1 = [t_1, \dots, t_n | T_2]$.

Complete Context Free Grammar Example

see

http://artint.info/code/Prolog/ch12/cfg_simple.pl

What will the following query return?

noun_phrase([the, student, passed, the, course, with, a, computer], R).

Complete Context Free Grammar Example

see

http://artint.info/code/Prolog/ch12/cfg_simple.pl

What will the following query return?

noun_phrase([the, student, passed, the, course, with, a, computer], R).

How many answers does the following query have?

sentence([the, student, passed, the, course, with, a, computer], R).

Augmenting the Grammar

Two mechanisms can make the grammar more expressive:

- extra arguments to the non-terminal symbols
- arbitrary conditions on the rules.

We have a Turing-complete programming language at our disposal!

Add an extra argument representing a parse tree:

$$\begin{aligned} \textit{sentence}(T_0, T_2, s(NP, VP)) \leftarrow \\ \textit{noun_phrase}(T_0, T_1, NP) \wedge \\ \textit{verb_phrase}(T_1, T_2, VP). \end{aligned}$$

Enforcing Constraints

Add an argument representing the number (singular or plural), as well as the parse tree:

$$\begin{aligned} \textit{sentence}(T_0, T_2, \textit{Num}, s(\textit{NP}, \textit{VP})) \leftarrow \\ \textit{noun_phrase}(T_0, T_1, \textit{Num}, \textit{NP}) \wedge \\ \textit{verb_phrase}(T_1, T_2, \textit{Num}, \textit{VP}). \end{aligned}$$

The parse tree can return the determiner (definite or indefinite), number, modifiers (adjectives) and any prepositional phrase:

$$\begin{aligned} \textit{noun_phrase}(T, T, \textit{Num}, \textit{no_np}). \\ \textit{noun_phrase}(T_0, T_4, \textit{Num}, \textit{np}(\textit{Det}, \textit{Num}, \textit{Mods}, \textit{Noun}, \textit{PP})) \leftarrow \\ \textit{det}(T_0, T_1, \textit{Num}, \textit{Det}) \wedge \\ \textit{modifiers}(T_1, T_2, \textit{Mods}) \wedge \\ \textit{noun}(T_2, T_3, \textit{Num}, \textit{Noun}) \wedge \\ \textit{pp}(T_3, T_4, \textit{PP}). \end{aligned}$$

Complete Example

see

http://artint.info/code/Prolog/ch12/nl_numbera.pl

- How can we get from natural language to a query or to logical statements?
- Goal: map natural language to a query that can be asked of a knowledge base.
- Add arguments representing the individual and the relations about that individual. E.g.,

noun_phrase(T_0, T_1, O, C_0, C_1)

means

- ▶ $T_0 - T_1$ is a difference list forming a noun phrase.
- ▶ The noun phrase refers to the individual O .
- ▶ C_0 is list of previous relations.
- ▶ C_1 is C_0 together with the relations on individual O given by the noun phrase.

Example natural language to query

see

http://artint.info/code/Prolog/ch12/nl_interface.pl

The student took many courses. Two computer science courses and one mathematics course were particularly difficult. The mathematics course. . .

The student took many courses. Two computer science courses and one mathematics course were particularly difficult. The mathematics course. . .

*Who was the captain of the Titanic?
Was she tall?*