COSC343: Artificial Intelligence

Lecture 21: Natural Language Syntax I

Alistair Knott

Dept. of Computer Science, University of Otago

Structures in human language

Sentences in a natural language are syntactic objects, just like expressions in predicate logic.

- We can speak about well-formed and ill-formed sentences, just as we can for logical expressions.
- The meaning of a sentence is determined by its form, just like the meaning of a logical expression.

However:

- We *know* the syntax of predicate logic, because we invented it.
- The syntax of a natural language is something we have to *discover*.

Building a syntactic theory

We have decided that words combine together to produce the meaning of a sentence. Now we need to work out *how* they do this.

Since there are an infinite number of possible sentences, we will clearly need to invoke *general principles* in our explanation. I'll introduce two kinds of general principle:

- Principles which group words into *general categories*.
- Principles which define *hierarchical structure* in sentences.

Any language also encodes a lot of *specific* knowledge:

- Knowledge of individual word meanings
- Knowledge of *idioms*: word combinations that occur particularly frequently.

Outline of the next few lectures

In today's lecture and Lecture 22 I'll discuss phrase-structure grammars, which are good at capturing the general principles.

In Lecture 23 I'll look at probabilistic language models ('*n*-gram models'), which are good at capturing knowledge of idioms.

In Lecture 24 I'll look at how these two types of model can be combined.

1. Word classes

Let's begin by considering a simple grammatical sentence:

THE DOG BARKED IN THE PARK

Interesting: we can replace 'dog' with any word denoting an object, and preserve well-formedness:

```
THE  CAT     BARKED IN THE PARK
THE  HOUSE   BARKED IN THE PARK
THE  PUDDLE  BARKED IN THE PARK
```

(Note that the new sentences might not *make any sense*... but they certainly count as correctly formed English sentences.)

Some examples of word classes

By this reasoning, we can establish sets of words which can be substituted for one another. For instance:

- Count nouns: DOG, CAT, HOUSE, PUDDLE...
- Determiners:

```
THE  CAT  BARKED IN THE PARK
A    CAT  BARKED IN THE PARK
NO   CAT  BARKED IN THE PARK
```

- Intransitive verbs:

```
THE  CAT  BARKED
THE  CAT  DANCED
THE  CAT  SANG
```

- Prepositions:

```
THE CAT BARKED  IN/AT/THROUGH  THE PARK
```


2. Hierarchical structure in sentences

If we consider a sentence, certain words in it seem to hang together more tightly than others.

For instance:

- Sometimes, a sequence of words can be replaced by a single word or phrase.

```
THE  DOG  BARKED  IN THE PARK
IT        BARKED  IN THE PARK
```

- Sometimes, a sequence of words can be moved to a different part of a sentence.

```
              THE  DOG  BARKED  IN THE PARK
IN THE PARK   THE  DOG  BARKED
```


2. Hierarchical structure in sentences

The words *the* and *dog* seem to combine together to form a sub-unit in the sentence.

- The words then interact with the rest of the sentence as a unit, rather than individually.

THE DOG BARKED IN THE PARK

What other sub-units do you think there are in this sentence?

Hierarchical groupings

Frequently, groupings of words can *themselves* be grouped.

```
(JOHN PUT THE BIG RED CUP ON THE TABLE)
```

What groupings are there in this sentence?

Representing sentences using trees

We can describe a hierarchical structure of word groupings by representing a sentence as a tree, in which

- the leaf nodes are words;
- the non-leaf nodes represent local groupings of words.

For instance:

Grammars

A phrase-structure grammar is a formal specification of what counts as a well-formed sentence.

- You can define a grammar for a formal language (e.g. predicate logic, or a programming language).
- The same kind of techniques can be used to define a grammar for natural language.

The primitive symbols in the grammar are words.

- Words can be put into word classes:
  E.g. 'DOG is in the class NOUN'
  E.g. 'THE is in the class DETERMINER'
- Rules about word groups can now be stated efficiently using word classes:
  E.g. 'DETERMINER followed by NOUN makes a word group'

Context-free rules

A convenient formalism for defining a grammar is as a set of context-free rules.

- Each rule has the form $lhs \rightarrow rhs_1, (\ldots), rhs_n$
  (Read: 'The sequence $rhs_1, (\ldots), rhs_n$ is an instance of category *lhs*'.)

We can use these rules to put words into classes:

N	$\rightarrow$	dog
Det	$\rightarrow$	the

And also to specify ways of grouping words:

NP	$\rightarrow$	Det, N

Note: in this last case, we have introduced a new class, NP which is the name of a *group* of words (or phrase). ('NP' stands for 'noun phrase'.)

Some more context-free rules

Rules about grouping words:

S	→	NP, VP
NP	→	PN
NP	→	Det, N
VP	→	V0
VP	→	V1, NP
S	→	S, Conj, S

Rules about word classes:

Det	→	“the” | “a”
N	→	“dog” | “cat”
PN	→	“Fred” | “Jip”
V0	→	“slept” | “ran” | “snorted”
V1	→	“bit” | “chased” | “caught”
Conj	→	“and” | “so”

Exercise: write down some sentences that this grammar allows.

Parsing

Context-free rules are *declarative*: they just provide a definition of sentences/phrases.

We also need a *procedure* for determining whether a given string of words can be represented as a phrase of type S.

- Such a procedure is called a parsing algorithm.

the dog chased Fred

Parsing as search

There are many different ways of defining a parsing algorithm. In the abstract, the goal state is to find a way of creating an S node, and the initial condition is a list of words.

- We can work backwards from the goal, and ‘grow’ the parse tree from the S node.
  This is a top-down parsing strategy.
- We can work forwards from the initial state, and ‘grow’ the parse tree up from the words in the list (left-to-right or right-to-left).
  This is a bottom-up parsing strategy.
- There are also lots of mixed strategies.

Parsing as search

If parsing is a search process, then it should be possible to draw a tree that indicates the search space.

Assume we’re implementing a top-down search algorithm:

- What would be at the root of the search tree?
- What will be at the goal state of the search tree?
- What would be at each node of the search tree?
- What would be at the leaves of the search tree?

Parsing and recursion

The possibility of recursion in grammar rules makes top-down search a dangerous option. For instance, consider this rule:

$$s \rightarrow s, \text{conj}, s$$

What problem will this introduce?

Summary

- Natural language sentences have hierarchical structure: certain sub-sequences of words 'hang together' particularly tightly.
- To describe this structure, we can use context-free grammar rules.
- The process of searching for syntactic structures in a sequence of words is called parsing.

Readings

For this lecture (and next lecture): AIMA Section 23.1