Syntax: Sentence Structure

SYCCL 2023

Formal grammars

Grammaticality vs. acceptability

A sentence is **grammatical** iff it is generated by the (human) grammar, and **ungrammatical** if it is not.

A sentence may be **unacceptable** for many reasons:

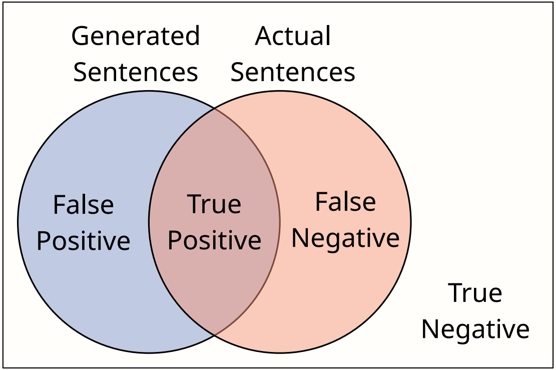
- ungrammatical
- meaningless
- hard to process
- pronunciation is awkward
- socially inappropriate
- ...

When studying syntax we usually **abstract away** from factors other than grammaticality.

Generating sentences

A formal grammar **generates** a set of sentences, which might be infinite.

Linguists use formal grammars to model the grammars of actual languages.



N-grams

N-grams

In the last Python notebook we looked at *n*-gram grammars.

In a **phonotactic** *n*-gram grammar, a word is grammatical if all **sound sequences** of length *n* are in the grammar.

- 2-gram grammar: $\{$pa, ma, pi, mi, pu, mu, ap, am, ip, im, up, um$\}$
- Words that fit the grammar:
  pa, papa, pama, mumu, umu, ipipi, $\ldots$

N-grams (2)

In a **syntactic** *n*-gram grammar, a sentence is grammatical if all **word sequences** of length *n* are in the grammar.

- 2-gram grammar:
  {START the, the man, the old, old old, old man, man saw, saw the, man END}
- Sentences that fit the grammar:
  The man saw the man
  The old old old old man saw the man
  The man
  The man saw the man saw the man

N-grams (3)

N-gram grammars have many nice properties:

- They are very simple
- They can be used to process words/sentences quickly
- They can be learned from positive data

They are a good model for phonotactics, but not for syntax.

- They can’t handle **hierarchical recursion**: phrases inside phrases.

Context-free grammars (CFGs)

Context-free grammars (CFGs)

A CFG uses **rewrite rules**. When rewriting a symbol:

- Look for any rule with that symbol on its left-hand side and replace it with the string of symbols on the right hand side.
- If more than rule applies, you can use any of them.
- Parentheses mean “optional”. Curly braces mean “choose one”.

Example Grammar

S $\longrightarrow$ NP VP
NP $\longrightarrow$ D N
VP $\longrightarrow$ V (NP)
D $\longrightarrow$ the
N $\longrightarrow$ $\{$cat, bird$\}$
V $\longrightarrow$ $\{$sits, chases$\}$

Transitive and intransitive verbs

Intransitive verbs make a VP all on their own.

Transitive verbs need an NP object as part of the VP.

This grammar doesn't distinguish transitive and intransitive verbs.

Grammar
S $\longrightarrow$ NP VP
NP $\longrightarrow$ D N
VP $\longrightarrow$ V (NP)
D $\longrightarrow$ the
N $\longrightarrow$ $\{$cat, bird$\}$
V $\longrightarrow$ $\{$sits, chases$\}$

Question: How can we fix this?

Conjunctions

Let's add the **conjunction** *and* to the grammar.

Now we need to let the verb be singular or plural, so it can **agree** with the subject.

Grammar
S $\longrightarrow$ NP VP
NP $\longrightarrow$ NP Conj NP
NP $\longrightarrow$ D N
VP $\longrightarrow$ V (NP)
D $\longrightarrow$ the
N $\longrightarrow$ $\{$cat, bird$\}$
V $\longrightarrow$ $\{$sits, **sit**, chases, **chase**$\}$
Conj $\longrightarrow$ and

Question: What is wrong with this grammar?

Agreement

Our current grammar doesn't enforce subject-verb agreement.

How can we fix this?

Grammar

S $\longrightarrow$ NP VP
NP $\longrightarrow$ NP Conj NP
NP $\longrightarrow$ D N
VP $\longrightarrow$ V (NP)
D $\longrightarrow$ the
N $\longrightarrow$ $\{$cat, bird$\}$
V $\longrightarrow$ $\{$sits, sit, chases, chase$\}$
Conj $\longrightarrow$ and

Recursion

A **recursive** rule is one that can be applied to its own output.

In a CFG, if you can rewrite a symbol with a string that contains the same symbol, then the original rule can be applied again.

Grammar

S $\longrightarrow$ NP VP
NP $\longrightarrow$ NP Conj NP
NP $\longrightarrow$ D N
VP $\longrightarrow$ VP Conj VP
VP $\longrightarrow$ V (NP)
D $\longrightarrow$ the
N $\longrightarrow$ $\{$cat, bird$\}$
V $\longrightarrow$ $\{$sits, sit, chases, chase$\}$

Recursion (2)

Question: What are some other examples of recursive sentence structures?

Are CFGs a good model for syntax?

Pros

- They are make very precise predictions.
- They are good at modeling phrase structure.
- They can handle recursive structure.

Cons

- Agreement (and movement) are extremely complicated to model with CFGs.
- There are syntactic phenomena that are *impossible* to model with a CFG.