Minds & Machines
SymSys 1/Psych 35/Phil 99/Linguist 35

Class 4: Machines for natural language grammar

October 4, 2016
Recap/upcoming

1. motivating the computation metaphor
2. embodied and abstract machines
3. machines for natural language
4. computation in the brain / objections
5. levels of analysis, language of thought, uncertainty, learning, choice, neural networks, social cognition
What’s so interesting about language?

1. it’s something that human minds do
2. it’s uniquely human (unusual!)
3. it has provided a key model for analysis of other cognitive domains (more later)
4. it has a complex hierarchical recursive structure

Say what?
Hierarchical recursive structure

**Hierarchical**: phrases can contain other phrases

**Recursive**: phrases can contain other phrases of the same kind

- the small cute funny dog

- Bill thinks Mary thinks Bill thinks Mary doesn’t like him
Hierarchical recursive structure

the small cute funny dog

NP
  the
  AP
    small
    AP
      cute
      AP
        funny
        N
          dog
OK, so what?

1. Historically important in cognitive science:
   - Key observation motivating behaviorism \(\Rightarrow\) cognitivism
   - Rise of complex models with hidden structure

2. Evidence for hierarchical recursive structure in many areas of cognition
   - e.g., motivates “Language of thought” hypothesis

3. Rules out finite-state models for (at least) language
Why not finite-state machines?

This lecture

1. why not finite-state machines?
2. context-free grammars
Why not finite-state machines?

Why does the finite-state issue matter?

“Some psychologists have suggested that human language is based on a huge word chain stored in the brain. This idea is congenial to stimulus-response theories: a stimulus elicits a spoken word as a response, then the speaker perceives his or her own response, which serves as the next stimulus, ... and so on. ...

“The modern study of grammar began when Chomsky showed that word-chain devices are not just a bit suspicious: they are deeply, fundamentally, the wrong way to think about how human language works.”

(Pinker, The Language Instinct, p.92-3)
Why not finite-state machines?

Understand language with these two weird tricks

1. arbitrariness of the sign
2. “infinite use of finite means”

There is no longest sentence in English!
  - or even word

Can a FSA cope with infinite sequences?
Why not finite-state machines?

Can a FSA cope with infinite sequences?
Why not finite-state machines?

Can a FSA cope with infinite sequences?

Class 4: Minds & Machines, SymSys 1/Psych 35/Phil 99/Linguist 35
Why not finite-state machines?

But it gets ugly

- the funny cute dog
- the cute funny dog
- the funny funny cute dog
Why not finite-state machines?

But it gets ugly

- the funny cute dog
- the cute funny dog
- the funny funny cute dog
Why not finite-state machines?

But it gets ugly

- the funny cute dog
- the cute funny dog
- the funny funny cute dog
- the small cute funny dog
Why not finite-state machines?

But it gets ugly

- the funny cute dog
- the cute funny dog
- the funny funny cute dog
- the small cute funny dog
Why not finite-state machines?

But it gets ugly

- the funny cute dog
- the cute funny dog
- the funny funny cute dog
- the small cute funny dog
- the cute small dog
Why not finite-state machines?

N-gram models

Predict the next word on the basis of the previous $N$ words.

Estimate transition probabilities from a (hopefully large) corpus.

1-gram example:
Why not finite-state machines?

Long-distance dependencies

- If you don’t shut up then I’ll scream
- If you don’t shut up right now then I’ll scream
- If you don’t shut up right now and leave then I’ll scream
- If you don’t shut up right now and leave and go get me a Coke then I’ll scream

Schema: If S then S, where S = any sentence
Why not finite-state machines?

Long-distance dependencies

- If you don’t shut up then I’ll scream
- If you don’t shut up right now then I’ll scream
- If you don’t shut up right now and leave then I’ll scream
- If you don’t shut up right now and leave and go get me a Coke then I’ll scream

Schema: If $S$ then $S$, where $S =$ any sentence

- Either Mary will come or Bill will come
- Either Mary will come tomorrow or Bill will come
- Either Mary will come tomorrow at noon or Bill will come

Schema: Either $S$ or $S$, where $S =$ any sentence
Why not finite-state machines?

Long-distance dependencies

Any sentence at all! Including each other, selves:

- If [either Mary will come or Bill will come], then I’ll scream

- If [either Mary will come or Bill will come], then [if Jim comes then I’ll scream]

- ...
Why not finite-state machines?

Long-distance dependencies

Same goes for just about any programming language:

```javascript
$('#continue').click(function() {
  var response = $('#yesnoresponse').serialize();
  if (response.length == 0) {
    $('#error').show();
  } else { // record data, then debrief
    data.rt = (new Date()).getTime() - startTime;
    data.response = response.slice(10, response.length);
    $('.slide').hide();
    if (e.which == 13) {
      return false;
    }
  }
});
```
Why not finite-state machines?

Pinker’s example

Either the girl eats ice cream, or the girl eats candy.
If the girl eats ice cream, then the boy eats hot dogs.

At first glance it seems easy to accommodate these sentences:
Why not finite-state machines?

Pinker’s example (p.95-6)

If either the girl eats ice cream or the girl eats candy, then the boy eats hot dogs.
Either if the girl eats ice cream then the boy eats ice cream, or if the girl eats ice cream then the boy eats candy.

something immediately disturbing about this solution: there are three identical subnetworks.”
“For the first sentence, the device has to remember if and either so that it can continue later with or and then, in that order. ... Since there’s no limit in principle to the number of if’s and either’s that can begin a sentence, each requiring its own order of then’s and or’s to complete, it does no good to spell out each memory sequence as its own chain of lists: you’d need an infinite number of chains, which won’t fit inside a finite brain.”
Why not finite-state machines?

Another kind of L-D dependency: semantics/discourse

Posted yesterday on Linguist List:

“We are happy to announce the release of the test portion of the LAMBADA dataset (LAnguage Modeling Broadened to Account for Discourse Aspects). ...”

“Some psychologists have suggested that human language is based on a huge word chain stored in the brain. This idea is congenial to stimulus-response theories: a stimulus elicits a spoken word as a response, then the speaker perceives his or her own response, which serves as the next stimulus, ... and so on. ...

“The modern study of grammar began when Chomsky showed that word-chain devices are not just a bit suspicious: they are deeply, fundamentally, the wrong way to think about how human language works.”

(Pinker, *The Language Instinct*, p.92-3)
Why not finite-state machines?

Outline

1. why not finite-state machines?
2. context-free grammars
Formal grammars

- Taken from Post’s work on production systems (1940s)
- applied by Chomsky to natural language in late 1950s
- close equivalencies to the machine classes we’ve discussed

Consider again

- the dog
- the small cute funny dog
- the funny funny dog
- the cute funny funny dog

Recursive context-free rules:

- $NP \rightarrow \text{the noun}$
- $NP \rightarrow \text{the AP}$
- $AP \rightarrow \text{Adj AP}$
- $AP \rightarrow \text{Adj noun}$
- $Adj \rightarrow \text{small OR cute OR...}$
- $noun \rightarrow \text{dog}$
Small cute funny dogs

- $NP \rightarrow \text{the noun}$
- $NP \rightarrow \text{the AP}$
- $AP \rightarrow \text{Adj AP}$
- $AP \rightarrow \text{Adj noun}$
- $Adj \rightarrow \text{small OR cute OR...}$
- $noun \rightarrow \text{dog}$

Diagram:
```
NP
  the
   AP
     Adj
      small
     AP
       Adj
         cute
       AP
         Adj
           funny
         noun
           dog
```
Pinker’s example

- $S \rightarrow \text{either } S \text{ or } S$
- $S \rightarrow \text{if } S \text{ then } S$

Diagram:

```
          S
         /\   \\
        S   S
       /     \   /
      /       S \
     /         /
    if        S
   /   \       /
  /     \     S
 /       \     /
 either   the girl eats ice cream or the girl eats candy
```

```
          S
         /   \     \\
        S     S       S
       /     /\     /     /
      /     /   \   /     /
    then the boy eats hot dogs
```

Grammars and machines

Each kind of grammar has a corresponding kind of machine. Key differences involve memory.

- **Regular grammars** ⇔ finite-state machines
  - no memory

- **Context-free grammars** ⇔ pushdown automata
  - memory = pushdown stack (first in/first out)

- **Context-sensitive grammars** ⇔ linear bounded automata
  - like TM, but with bounded tape

- **Unrestricted grammars** ⇔ Turing machines
  - unlimited tape

(So why not just use Turing machines to model language?)
Implications beyond language

FSM-like models have been widely used in cognitive science
  ▶ notably, implicit in behaviorist stimulus-response theory
  ▶ arguably present in some neural network models

Arguments that have been raised against:
1. Language can't be modeled in this way
2. Systematicity, productivity of other cognitive faculties suggest they can't either ...
Next time

1. computation in the brain?

2. Some objections to the whole enterprise

Chinese Room Argument

The Chinese room argument is a refutation of strong artificial intelligence. “Strong AI” is defined as the view that an appropriately programmed digital computer with the right inputs and outputs, one that satisfies the Turing test, would necessarily have a mind. The idea of Strong AI is that the implemented program by itself is constitutive of having a mind. “Weak AI” is defined as the view that the computer plays the same role in studying cognition as it does in any other discipline. It is a useful device for simulating and therefore studying mental processes, but the programmed computer does not automatically guarantee the presence of mental states in the computer. Weak AI is not criticized by the Chinese room argument.

The argument proceeds by the following thought experiment. Imagine a native English speaker, let’s say a man, who knows no Chinese locked in a room full of boxes of Chinese symbols (a data base) together with a book of instructions for manipulating the symbols (the program).