TTIC 31230, Fundamentals of Deep Learning

David McAllester, Winter 2020

Representing Functions and Knowledge with Logic

The Procedural vs. Declarative Debate

Any Discussion today of the "knowledge representation problem" is likely to entail a debate between proponents of **declarative** and **procedural** representations of knowledge.

Terry Winograd, 1974

Programming Languages (Procedural)

vs. Logic or Natural Language (Declarative)

Is Human Common Sense Based on Logic?

Certain facts are obvious.

A king on empty chess board can reach every square (obvious).

A knight on an empty chess board can reach every square (true but not obvious).

Obviousness

Consider a graph with colored nodes.

If every edge is between nodes of the same color, then any path connects nodes of the same color.

Consider a swiss chocolate bar of 3×5 little squares.

How many breaks does it take to reduce this to fifteen unconnected squares?

What inference happens when one observes that each break increases the number of pieces by one?

Logical Representations of Events

Let e range over "events".

$$e : \text{give}(a_1, x, a_2) \Rightarrow \text{had}(a_1, x, \text{before}(e)) \wedge \text{had}(a_2, x, \text{after}(e))$$

This is related to Davidsonian semantics for natural language (1969) and the situation calculus of McCarthy and Hayes (1968).

Bottom-up Logic Programming

Bottom-up logic programming is distinguished by its relationship to dynamic programming algorithms.

$$\text{At}(x) \Rightarrow \text{Reachable}(x)$$

$$\text{Reachable}(x) \wedge \text{CanGo}(x, y) \Rightarrow \text{Reachable}(y)$$

This defines a linear time algorithm for reachability.

Datalog

A set of inference rules each of which has antecedents and conclusions that are just predicates applied to variables is called a **datalog** program.

It can be shown that datalog "captures the complexity class P" — they can express **all and only** polynomial time decidable relations (provided the entities are assigned a total order).

General bottom-up logic programs, including expressions (terms), are Turing complete.

$$N(x) \Rightarrow N(s(x))$$

Type Theoretic Foundations of Mathematics

variables, pairs	x	(e_1, e_2)	$\pi_i(e)$
functions	$\lambda x\!:\!\sigma\ e[x]$	$f(e)$	
atoms	$P(e)$	$e_1 \doteq e_2$	$e_1 =_\sigma e_2$
formulas	$\neg\Phi$	$\Phi_1 \vee \Phi_2$	$\forall x\!:\!\sigma\ \Phi[x]$
types	$\Sigma_{x:\sigma}\ \tau[x]$	$\Pi_{x:\sigma}\ \tau[x]$	$S_{x:\sigma}\ \Phi[x]$
type constants	Bool	Set	Type

The Substitution of Isomorphics

The isomorphism relation $u =_\sigma v$ is challenging to define in complete generality.

But we know we want the following substitution rule.

$$\frac{\Sigma \vdash \sigma : \mathbf{Type} \quad \Sigma;\ x:\tau \vdash e[x]:\sigma \quad \Sigma \vdash a =_\tau b}{\Sigma \vdash e[a] =_\sigma e[b]}$$

The Foundations of Mathematics

Independent of the utility of mathematical foundations in the quest for AGI, mathematical truth seems important.

The philosophy of the mathematics is difficult.

How do we have access to mathematical truth?

Are mathematical theorems really true?

Is our access to mathematical truth based on innate inference principles — a particular architecture of thought?

END