Combining deep learning with logical structure

YKY 甄景贤

Independent researcher, Hong Kong

generic.intelligence@gmail.com

February 1, 2018

Talk summary

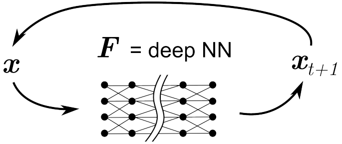
1. Deep reinforcement learning
2. Logical structure
   - Propositional logic
   - Predicate logic

Section 1

Deep reinforcement learning

Problem setup

- Consider a (deep) neural network connected end-to-end to form a loop:

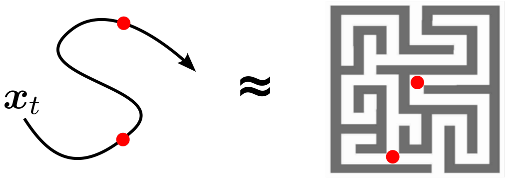


(1)

where "deep" means "many layers".

- This is referred to as a **recurrent** neural network (RNN).

- The state vector $\boldsymbol{x}_t$ of the neural network traces out a **trajectory** in configuration space, which is analogous to a "maze" with **rewards** (•) inside it:



(2)

- We regard the state $\boldsymbol{x}_t$ as the **mental state** of an intelligent agent, the rewards are given externally by a teacher to reward intelligent behavior.

Neural network

- A neural network is a generic function with a large number of **parameters** called **weights**:

weight matrix for each layer total $\#$ of layers

$$\boldsymbol{x}_{t+1} = \boldsymbol{F}(\boldsymbol{x}) = \sigma(W_1 \sigma(W_2 ... \sigma(W_L\ \boldsymbol{x}))) \tag{3}$$

- σ is the **sigmoid** function applied *component-wise* to the vector $\boldsymbol{x}$:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{4}$$

- Neural networks are **universal approximators** of vector-valued functions.

Dynamical system

- So we have a dynamical system given by the **transition function** $\boldsymbol{F}$:

$$\boldsymbol{x}_{t+1} = \boldsymbol{F}(\boldsymbol{x}_t, \boldsymbol{u}_t) \tag{5}$$

  where $\boldsymbol{u}$ is the **control variable**.

- The reward at state $\boldsymbol{x}_t$ is $L(\boldsymbol{x}_t)$ which corresponds to the **Lagrangian** in the Hamiltonian formulation. Our goal is to maximize the total reward over time:

$$\boxed{\text{action}} \quad A = \int L(\boldsymbol{x}_t) dt \tag{6}$$

  which coincides with the notion of **action** in Hamiltonian mechanics.

background: AIXI theory

Theorem (Hutter 2000)

*The action chosen by this formula defines an **optimal** intelligent agent:*

$$a_k := \arg\max_{a_k} \sum_{o_k r_k} \ldots \max_{a_\infty} \sum_{o_\infty r_\infty} [r_k + \ldots + r_\infty] \sum_{\substack{q:U(q,a_1,\ldots,a_\infty) \\ =o_1 r_1 \ldots o_\infty r_\infty}} 2^{-\ell(q)} \tag{7}$$

a*ction*, r*eward*, o*bservation*, U*niversal Turing Machine*, q*rogram*, k=*now*

- similar to my setup
- widely regarded by AI practitioners as “useless” because it involves **algorithmic** / Kolmogorov complexity.

background: Google's DeepMind

Marcus Hutter

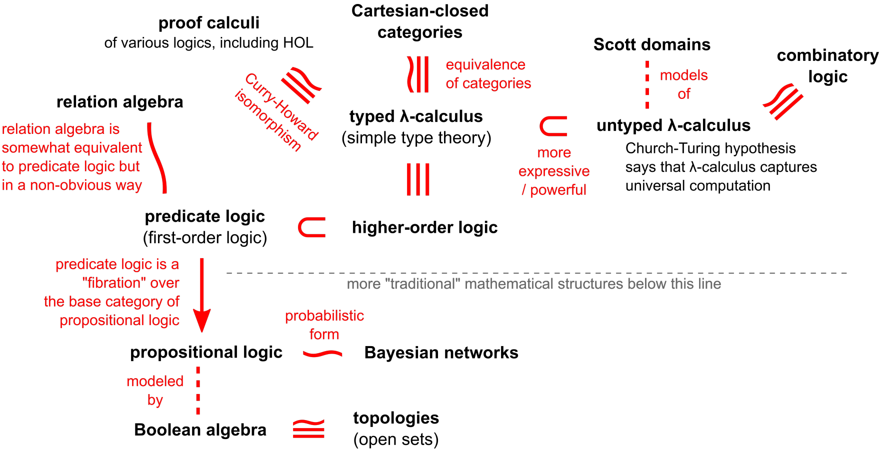
Jüergen Schmidhuber

Some of their students became founding members of **DeepMind**.

Section 2

Logical structure

The world of logical structures

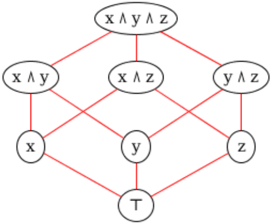


Subsection 1

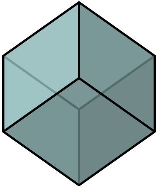
Propositional logic

Boolean lattice

The Boolean lattice generated from 3 propositions, with 2^3 elements:



$\cong$



(8)

which is isomorphic to the **hypercube**.

Lattice $\leftrightharpoons$ neural network

- The **output vector** of a neural network can be mapped to the **hypercube** if each output value is **binarized** to $\{0, 1\}$.
- Each output corresponds to the **truth value** $(\top, \bot)$ of a **proposition**.
- Thus the **Boolean lattice** is mapped to the neural network's **state vector**.
- Performing **logical deduction** would be same as traversing **vertices** of the hypercube, which is same as determining which propostion(s) become true during a deductive step.
- We may call this the "canonical embedding" of a Boolean lattice into a neural network's state space.

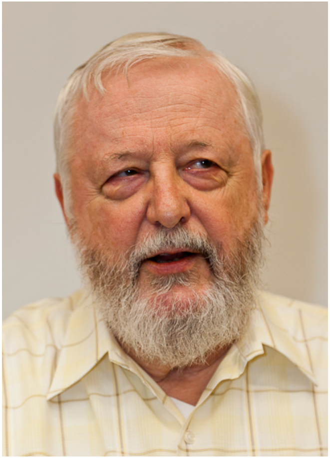
My problem #1

- Logic deduction is a **monotone** function going down the Boolean lattice
- The neural network $\boldsymbol{F}$ is "free" (unconstrained)
- Use the monotonicity to constrain the **learning** algorithm of $\boldsymbol{F}$ to make it faster?

Subsection 2

Predicate logic

Categorical logic

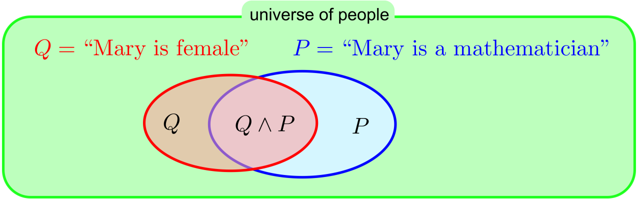


William Lawvere (1937-)

- Originated the **categorical** formulation of **logic** in the 1970's.
- Established category theory as a **foundation** of mathematics similar to **set theory**.

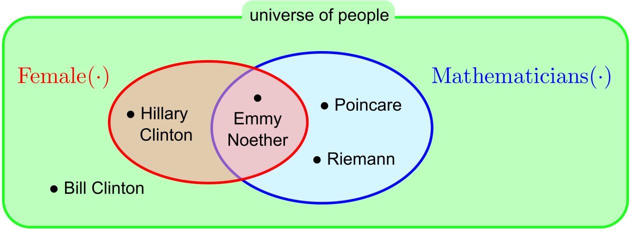
Predicates represented as topological open sets

Propositional logic:



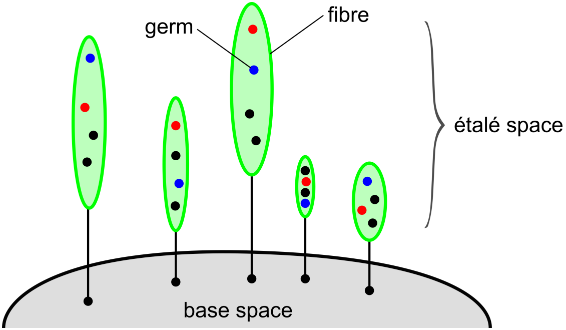
(9)

Predicate logic:



(10)

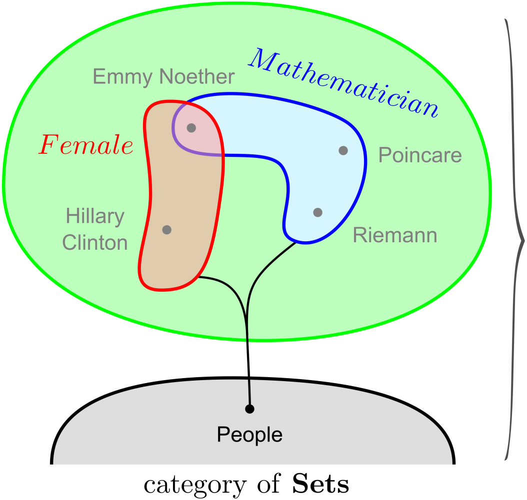
Fibration



$$(11)$$

Predicates as fibration

The fibration $\begin{array}{c}\mathbf{Pred}\\ \downarrow\\ \mathbf{Sets}\end{array}$ is a **forgetful functor**:



$$\text{category of } \mathbf{Pred} \tag{12}$$

Predicates as fibration

- The objects in $\mathbf{Pred}$ are *predicates*.
- The objects in $\mathbf{Sets}$ are *sets*, in our case there is only 1 set which is the universe. If multiple sets, they correspond to **types** in computer science.
- The forgetful functor sends each predicate to its **underlying set**.
- Figure (12) shows 1 **fibre**. Each fibre is a **Boolean algebra**.

My problem #2: the big challenge

- Use the structure of predicate logic to formulate the **learning algorithm** of neural network $\boldsymbol{F}$ so that it can perform predicate-logic deduction?

References

Bart Jacobs (1999)
Categorical logic and type theory

Robert Goldblatt (2006)
Topoi – the categorical analysis of logic

The End