Mathematics of knowledge representation in AI

YKY 甄景贤

Independent researcher, Hong Kong

generic.intelligence@gmail.com

September 12, 2018

Talk summary

1. Brief review of neural networks
2. AI as a dynamical system
3. The structure of logic in AI
4. 4 candidate solutions
   - Plan A: co-operative co-evolution (COCO)
   - Plan B: hybrid neural + graph
   - Plan C: geometric models
   - Plan D: "quantum" Hilbert-space operators

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

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

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

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

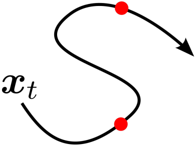
- Neural networks are **parametrized** (vector-valued) **functions**, and they are **universal function approximators**.

“Unreasonable” effectiveness of neural networks

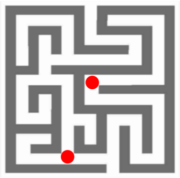
- If  is replaced by polynomial, degree of the composite function increases **exponentially** as # layers increase

Intelligent agent

- The state vector x_t of the neural network traces out a **trajectory** in configuration space, which is analogous to a "maze" with **rewards** (•) inside it:



$\approx$



(3)

- We regard the state x_t as the **mental state** of an intelligent agent, the rewards are given externally by a teacher to reward intelligent behavior.

Hamiltonian control

- **Lagrangian** $L(\vec{x}) =$ *instantaneous* reward at state x:

$$J = \int L(\vec{x}) dt \tag{4}$$

- The **Hamiltonian** is defined as:

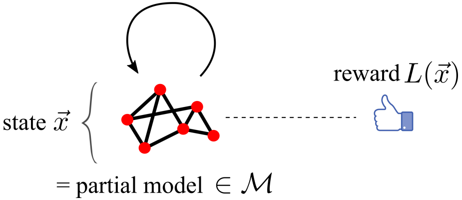
$$H = L + \frac{\partial J}{\partial \vec{x}} \vec{f} \tag{5}$$

- **Pontryagin maximum principle**:

$$H^* = \inf_u H \quad \text{or} \quad \nabla_{\vec{u}} H^* := \frac{\partial H^*}{\partial \vec{u}} = 0 \tag{6}$$

- The operation of the system is as follows:

$$\text{model rewriting} \quad \left.\begin{array}{l} \vec{u} : \mathcal{M} \rightarrow \mathcal{M} \\ \vec{u} : \quad \vec{x} \mapsto \vec{x}' \end{array}\right\} = \Theta \tag{7}$$

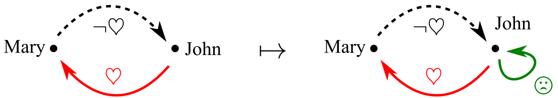


- $\vec{u}$ coincides with $\vec{f}$, its purpose is to **rewrite** $\vec{x}$:

$$\vec{f}(\vec{x}, \vec{u}) \equiv \vec{u}(\vec{x}) \tag{8}$$

Optimization over logic formulas (2)

- For example, the logic rule "'love and not loved back $\Rightarrow$ unhappy" performs the rewriting of the following sub-graph:



(9)

- This is the **state transition** $\vec{u} : \vec{x} \mapsto \vec{x}'$, which can also be regarded as the **logical inference** $\vec{u} : \vec{v} \vdash \vec{x}'$, where $\vec{u}$ is the rewriting function or logic rule.

The problem with predicate logic

$$\forall x, y, z.\ \text{father}(x, y) \wedge \text{father}(y, z) \rightarrow \text{grandfather}(x, z) \tag{10}$$

- This involves **variable substitutions** which are troublesome to handle with neural networks.
  (The difficulty seems to come from the cylindric-algebraic structure of predicate logic: if a formula have variables $x_1, x_2, x_3, \ldots$, we would need to consider the domain $D \times D \times D \times \ldots$ where $D \ni x_i$)

Relation algebra

Given that:

$$\text{Father} \circ \text{Father} = \text{Grandfather} \tag{11}$$

we can deduce:

$$\text{john Father paul} \tag{12}$$
$$\text{paul Father pete} \tag{13}$$
$$\Rightarrow \text{john Father} \circ \text{Father pete} \tag{14}$$
$$\Rightarrow \text{john Grandfather pete} \tag{15}$$

via *direct* substitution of equal terms.

- Relation algebra appears very *natural* and similar to human thinking

We're looking for Tensorflow developers to implement a prototype.

Thank you