

神经网络中直接注入知识

甄景贤 (King-Yin Yan)

General.Intelligence@Gmail.com

August 12, 2017

Abstract. 在人工智能历史上，迄今为止仍未有一个快速的学习算法，可以同时「像教孩子那样」直接注入知识。经典逻辑 AI 可以直接写入知识，但其学习算法太慢。深度神经网络的学习算法很快，但它是「黑盒」。本文提出一个解决方案：让神经网络直接作用在它自身的 weights 上。

0 Introduction

智能系统需要有能力读 / 写它内部的知识。例如说，一个比较蠢的智能系统可以用 sequence-to-sequence 的方式将中文翻译成英文：

$$\text{“中文句子”} \xrightarrow{F} \text{“英文句子”} \quad (1)$$

F 代表系统的函数。但系统并不真的明白句子的意义，句子只是「水过鸭背」地流过系统。一个更聪明的系统是：句子可以进入到 F 里。

1 Applications

DKI (direct knowledge injection) is useful in:

- learning by instructions, or “learn by being told”
(a technique crucial to accelerating the learning of human knowledge)
- belief revision / truth maintenance
(the most challenging and highest-level task in logic-based AI)

举例来说，小孩子的行为是由他内部的知识决定的，「知识决定行为」。

- 当小孩子看到一个成人做的动作，他会模仿那动作。



(2)

- 或者小孩子听到一句说话：「不要吃污糟食物」，他明白了那句说话的意思而改变行为。
- 或者「今天高考放榜了，这本教科书可以丢进垃圾桶」，这句话应该由 working memory x 进入到 $\boxed{\text{KB}}$ 里面，影响日后的行为（eg，以后不会再见到那本书）。

这些例子都涉及到将「感觉资料」放进 F 里面：

$$\boxed{\text{sensory data}} \hookrightarrow F \quad (3)$$

2 Cartesian closure

DKI requires the functional closure $\mathbb{X} \simeq \mathbb{X}^{\mathbb{X}}$ which yields a **Cartesian-closed category** (CCC).

举例来说，「吃了污糟的食物会肚痛」是一个句子，它经由 eye 进入 mental state x ，变成 proposition。但我们希望这逻辑命题变成 $\boxed{\text{KB}}$ 的一部分。 F is the state-transition function:

$$x_{n+1} = F(x_n) \quad (4)$$

where

$$F = \boxed{\text{KB}} = \text{state}$$

An individual logic rule is a restriction of F to a specific input; Perhaps I could call such elements “micro-functions”.

$F \equiv \boxed{\text{KB}}$ is the “union” of micro-functions:

$$\boxed{\text{KB}} = \bigcup f_i \quad (5)$$

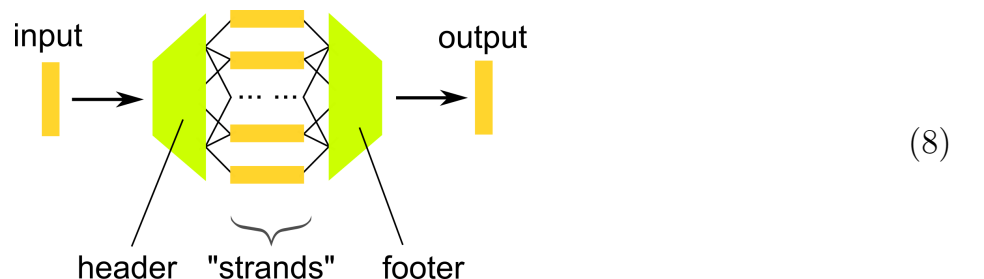
At this point the meaning of \bigcup is unspecified yet. F is the sum total of objects like x :

$$F = \bigcup x_i \quad (6)$$

但 F 是一个神经网络，它的一般形式是：

$$\boxed{\text{output}} \quad x_{n+1} = F(x_n) = \bigcirc^1 \bar{W} \bigcirc^2 \bar{W} \dots \bigcirc^L \bar{W} x_n \quad (7)$$

L = total number of layers. 由於各层的非线性「纠缠在一起」，表面上无法将神经网络「分解」。直到笔者受了 David Ha *et al* 提出的 PathNet [1] 理论所启发，PathNet 是由一些较小的神经网络 modules 组成，所以或许可以建构如下形式的「丝状神经网络」：



这些「丝条」可以是简单的神经网络，例如每个的宽度或深度很小，因而可以用较短的 weights vector 描述。正是因为这原因，一个 本身可以作为神经网络的输入。但整个神经网络 F 无法输入自己，因为根据 Cantor's theorem, $\mathbb{X} = \mathbb{X}^{\mathbb{X}}$ 是不可能的。

Let \overline{F} = header, \underline{F} = footer, f_i = strands, then (abusing the \biguplus notation):

$$F = \overline{F} \circ \biguplus f_i \circ \underline{F} \tag{9}$$

每个 大约对应於逻辑上的一个命题（proposition, 可以是条件命题或普通命题）。

读者或许会质疑，这个「条状」结构为什么一定要设计成这样？其实我也觉得这个设计不够 elegant，甚至不太肯定它会不会 work。在 §4 - §6 我们会介绍一个数学上更优美的做法。

3 Overall architecture

For reference, the architecture for **visual recognition** is:

$$\text{eye} \rightarrow \text{RNN} \rightarrow \text{mouth} \tag{10}$$

Our basic AGI architecture is:

$$\tag{11}$$

RNN = [deep] neural network, trained via **reinforcement learning**

The overall **recurrent** setup operates like this:

$$\tag{12}$$

注意： 的原始输入不可以直接写入 ，因为 会变成 = weights，而直接写 weights 的后果当然是灾难性的。换句话说， 的结构是要用 emergent（涌现）的方法 learn 出来，不能被外界的输入干扰。 和 的输入 / 输出要透过某些神经网络的 mapping 间接地做。

Viewing the “information flow” in a simplified way, we notice a “second” pass through the network’s internal weights:

$$\tag{13}$$

这种操作上的结构在经典逻辑 AI 是「免费赠品」, 但似乎还未有人提出过神经网络的做法。

对应於经典逻辑 AI:

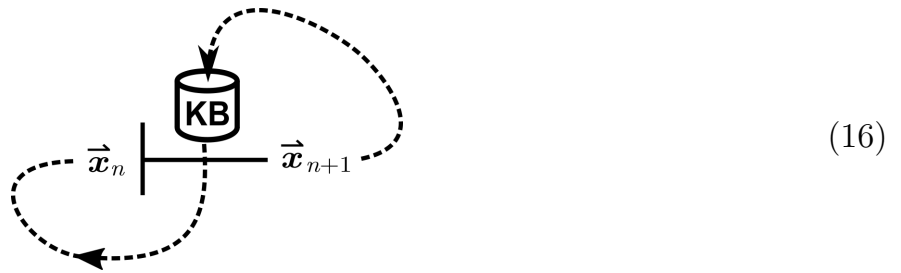
$$\text{KB} = \text{KB} \quad (14)$$

- The **horizontal pass** represents using the KB for logical inference (thinking), ie:

$$x_n \cup \text{KB} \vdash x_{n+1} \quad (15)$$

- The **vertical pass** represents reading/writing information to/from KB :
 - **Update:** x 是 KB 的一部分, 所以 x_{n+1} 改变了, KB 也要 update。
 - **Read:** x = working memory 会因为 注意力 (attention) 而改变, 所以 x_{n+1} 并不直接进入下一轮的 iteration, 而是先经过 KB 的 **attentional change**。

In logic-based AI this workflow has always been standard (but not made explicit):



4 What is required of F ?

We now try to explain the meaning of

$$F = \bigcup x_i \quad (17)$$

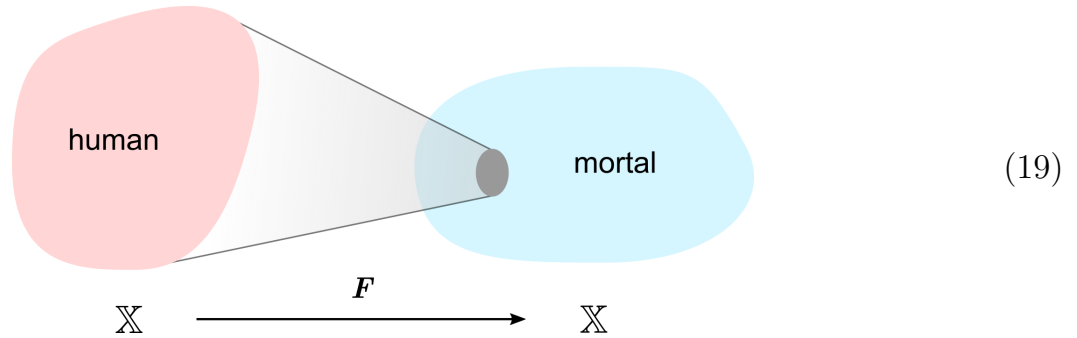
Our goal is to organize the 's into a deep network. What are the most general *desiderata* for such a function F ?

1. $F(x; \theta)$ is a function of x parametrized by θ .
2. the parameters θ is organized hierarchically with the “deep” property, ie, high-level θ_i 's have higher “degree”.
3. $F(x; \theta)$ is capable of universal function approximation.
4. x can be put into θ , ie, θ is a collection of x 's.
5. F encodes **logical consequence**.

The last condition (F5) is hardest to satisfy, but there is an informal argument that may justify it. For example, suppose:

$$\begin{aligned}
 x &= \text{"all men are mortal"} && \text{is put into } F = \boxed{\text{KB}}. \\
 x_0 &= \text{"Socrates is a man"} && \text{is the new input.} \\
 \text{Then the expected output should be:} &&& (18) \\
 F(x_0) &= \text{"Socrates is mortal"}
 \end{aligned}$$

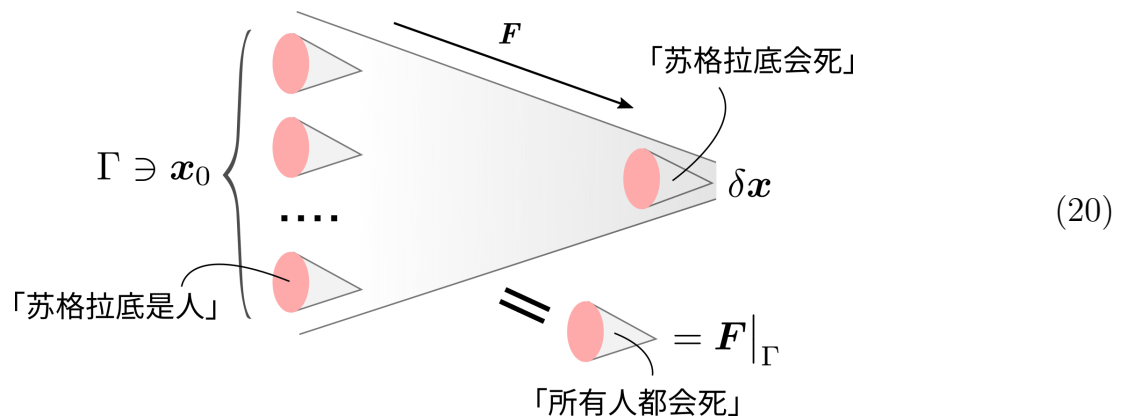
凭什么认为 F 能满足类似上面的要求？可以将一个 conditional proposition 看成是一个 mapping，它将 source domain \mathbb{X} 的某个区域映射到 target domain（也是 \mathbb{X} ）的某个区域，例如：



而我们有信心 F 能够表达这个 mapping 的原因，正如在机器视觉中，类似的 F 里面有些神经元可以辨认「眼、耳、口、鼻」等 features，原理是一样的。换句话说，「X 是人 \rightarrow X 会死」这句条件命题，其实和负责辨认「眼」这个 feature 的那些神经元，本质上是沒有分別的。

5 Decomposition of F

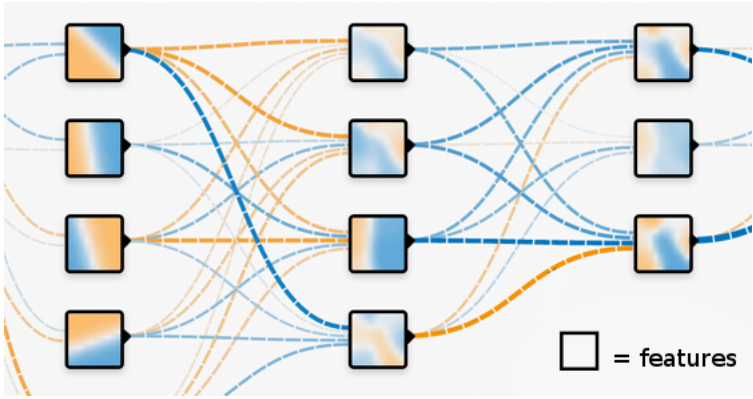
换句话说，我们需要的是将 F 分拆成一些小部分，亦即是 F 对於某个输入 x （及其邻域）的 **截面** (section, or restriction)。假设获得了这些截面函数之后，我们再需要一个方法，将一个截面「放进」 F 里面。



x_0 = current state, 它由若干个 $\text{red circle} \triangleright$ = 命题构成。 $F: x_0 \mapsto x'_0$ 但实际上 F 的输出是 δx , 因为 \vdash 的特性是它 每次只改变一个命题, 亦即 $x'_0 = x_0 + \delta x$, 而 x'_0 也会「忘记」它里面的一个 / 几个命题。更重要的是: F 的 restriction 也是一个命题 = $\text{red circle} \triangleright$ = $F|_\Gamma$ 其中 Γ 是 x_0 的邻域。注意: 下一步会将 δx 放进 F 里, 但 δx 这个函数的 source 邻域并不是 Γ 。

(要透彻理解上面这幅图，需要熟悉命题逻辑和谓词逻辑的几何化)

但是，当 \mathbf{F} 是一个神经网络时，怎样获得「截面」？我的想法是受到 TensorFlow Playground 的这图片启发的：



(21)

可以看到 weights 的大小不同，它们对函数 \mathbf{F} 在输入 \mathbf{x} 的贡献也不同。What we need is the set of **input-biased weights** of the neural network, in other words, they are the set of **outputs** (or “activities”) of all neurons. Let’s denote it as $\mathbf{W}(\mathbf{x})$.

6 Spectral compression

Now we need a way to **compress** $\mathbf{W}(\mathbf{x})$ to prune out the low-contribution (ie, small) weights. The Fourier (or wavelet) transform is a good candidate because it can compress $\mathbf{W}(\mathbf{x})$ to a fixed-length vector, which can then be used as inputs to the neural network \mathbf{F} .

The compression (which is a projection, P) is performed via:

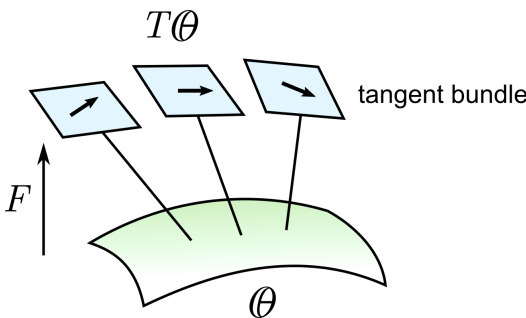
$$P \mathbf{F} = \sum_{i=1}^N \langle \mathbf{F}, \Psi_i \rangle \Psi_i \tag{22}$$

where $\{\Psi_i\}_1^N$ is the basis set.

7 几何结构

[此段对熟悉微分几何的人或许有帮助，否则可以略过。]

首先我们有一个很 standard 的 Hamiltonian 力学系统 / 控制系统的结构：



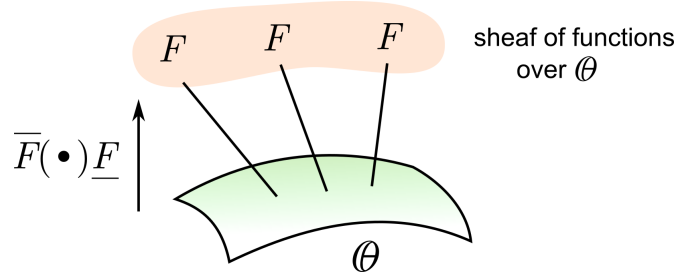
(23)

$\mathbf{x} = \text{working memory} \subset \boldsymbol{\theta}$, $\boldsymbol{\theta}$ 代表整个 $\boxed{\text{KB}}$ 的状态, 而 $\boldsymbol{\theta} \in \Theta$, 后者是所有可能 $\boxed{\text{KB}}$ 的空间。

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}) \quad (24)$$

是系统的状态方程。换句话说, 在思维空间 Θ 中的一个点就是思维状态 $\mathbf{x} \subset \boldsymbol{\theta} \in \Theta$, 而 \mathbf{F} 给出的是这个点在思考过程中的「运动速度」= $\dot{\mathbf{x}}$ 。换句话说, \mathbf{F} 定义了一个 vector field, 它是思维空间中思维的“flow”, 或者可以叫作「理性流」。每个点的速度属于流形 Θ 上的 tangent space, 他们的总和就是 tangent bundle。而 tangent bundle + base manifold (亦即「位置 & 动量」) 构成系统的「相空间」(phase space)。

另外, 特别地, 有这个 sheaf of functions 的结构:



(25)

换句话说, 给定 $\mathbf{x} \in \Theta$, 我们可以透过

$$\mathbf{F} = \overline{\mathbf{F}} \circ (\mathbf{x} \subset \boldsymbol{\theta}) \circ \underline{\mathbf{F}} \quad (26)$$

得出 \mathbf{F} , 而这个 \mathbf{F} 再给出对应於这点的 $\dot{\mathbf{x}}$ 。

注意 (23) 和 (25) 是两个不同的结构, 只是它们的 base manifold 相同。

特别之处在於 \mathbf{F} 是由参数 $\mathbf{x} \subset \boldsymbol{\theta} \in \Theta$ 确定的 (因为 \mathbf{x} 是 $\boldsymbol{\theta} = \boxed{\text{KB}}$ 的一部分, 而所有可能的 $\boxed{\text{KB}}$ 属于思维空间 Θ), 换句话说:

$$\mathbf{F}(\mathbf{x}) \equiv \mathbf{F}_{\boldsymbol{\theta}}(\mathbf{x}) \equiv \mathbf{F}(\mathbf{x}; \boldsymbol{\theta}) \quad (27)$$

这和经典控制并没有抵触, 因为经典理论中, \mathbf{F} 也是位置 \mathbf{x} 的函数。更确切地说, 位置空间其实是由 $\boldsymbol{\theta} \in \Theta$ 决定的, \mathbf{x} 只是 $\boldsymbol{\theta}$ 的一部分。

In general, the differential version seems to be a bad idea, as the increments of \mathbf{x} is small, and the $\boxed{\text{KB}}$ may be overwhelmed by recent memories of \mathbf{x} , which is a wasteful usage of weights.

8 Conclusion

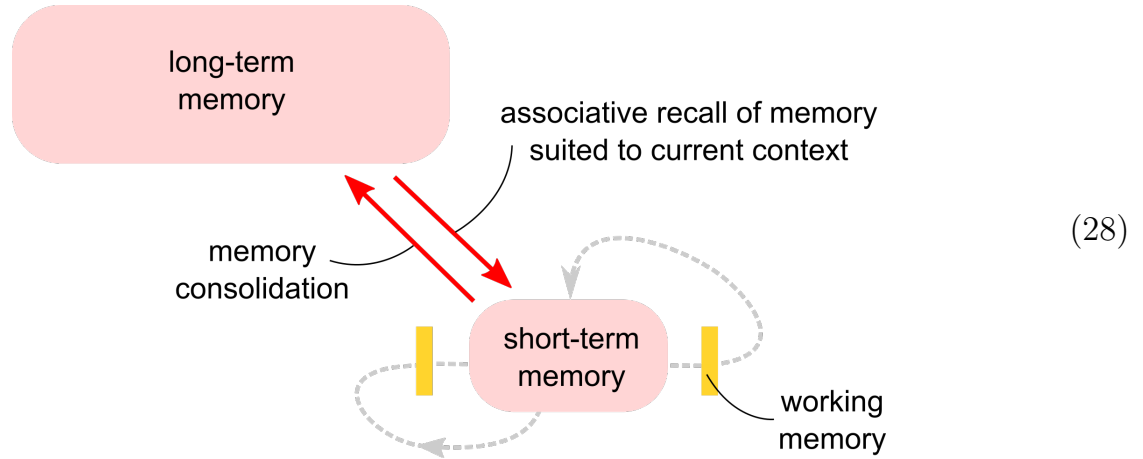
其实我自己也怀疑, 这个 architecture 是否太复杂了, 究竟值不值得做? 而暂时我只能说, 它符合了两项非常难达到的要求:

- $\boxed{\text{KB}}$ is represented as a neural network (that admits a fast learning algorithm);
- Symbolic knowledge can be directly **injected** into $\boxed{\text{KB}}$.

9 Future direction

There may exist some variations and improvements under this general architecture.

Another problem is the design of the **memory system**, in particular **episodic memory**. This is the general idea, but details are still undecided:



Acknowledgements

Thanks to David Ha and his co-authors for their PathNet idea.

Bibliography

- [1] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A. Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *CoRR*, abs/1701.08734, 2017. URL <http://arxiv.org/abs/1701.08734>.