Genifer 5.3 theoretical notes

YKY (甄景贤)

August 10, 2015

1 Top-level architecture

在最高层面上，RL 控制 RNN：

可以想像，RNN 是 RL 的 “mental model”，即这 RL 比普通 RL 有更复杂的内部模型。

馀下的工作是，定义 RL 的四个元素：states, actions, rewards, policy。

2 Unification of RL and RNN

在统一 RL 和 RNN 的时候，我发现它们之间的关系非常简单：

从**强化学习**的角度来看，我们需要的是这个函数：

$$\text{policy}: \quad \text{state} \xmapsto{\text{action}} \text{state'}$$

而 K 可以看成是我们的 mental state，那么 RL 的 action 就是将 K 变成 K' 的作用，而那正是 D！

$$K \xmapsto{D} K'$$

换句话说，D 就是 policy。

从强化学习的角度看，某些推导过程的结果，可以给予奖励：

$$K_0 \xmapsto{D^n} K_\perp \quad \updownarrow \bigstar$$

$\updownarrow \bigstar$ 的意思是「给予正或负奖励」。我们要学习的是 D 也就是 policy。学习算法的基础是著名的 Bellman optimality condition。

> 回顾一下 Bellman equation 是：
>
> $$U(S) = \max_a \{R(S,a) + U(S')\}$$
>
> a 是对所有 actions。
>
> 这个方程有微分版本，叫 Hamilton-Jacobi equation：
>
> $$\begin{aligned} U(t,x) &:= \min_u \{J(t,x,u)\} \\ J(t,x,u) &:= \int_t^{t_1} L(s,x(s),u(s))ds + K(x(t_1)) \end{aligned}$$

但 D 本身是 RNN，它还可以透过 back-prop 进行学习，两者似乎是不同的。Back-prop 是透过 $\frac{\partial \mathcal{E}}{\partial D}$ 的梯度来学习。

在思维过程中，除了结果以外，其他步骤是没有奖励的，所以 reward 只可以是一个微小的负常数（代表时间损耗）。在这情况下 Bellman 算法可能变到和 back-prop 一样，但我还未详细验算。

以下我们分别讲述 4 种学习模式：

- 学习 听 / 讲
- RL-based learning
- inductive learning

3 学习 听 / 讲

我发现 听 / 讲 是可以靠 RL 学习的。

Learning modes:

1. Listening: K contains understanding of linguistic input, test K
2. Speaking: K is known, output $\approx_L K$

$$K_0 \xmapsto{D^n} \text{输出句子} \quad \updownarrow \bigstar$$

4 RL-based learning

5 Inductive learning

Actions:

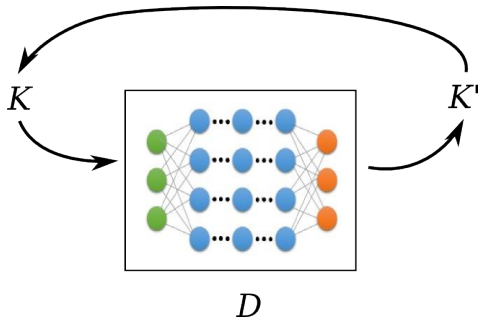
(A) deduction (apply D)

(B) write / add to K partially: $K := K'$ or $K += K'$

(C) learn = change D

(D) learn = neurogenesis

上次的模型：



K 可以直接输出一些 words，words 的序列构成句子。

这做法太简单，因为我假设自然语言是一些 sequences，但自然语言的结构不是序列那么简单，例如「未吃过饭」包含「吃过饭」的序列，但语义是相反的。所以自然语言必须要在语义 (semantic) 层面上处理，而不是语法 (syntactic) 层面上。自然语言的复杂性是不会轻易消失的。

6 Language map

在 Memory Networks [1] 里，他们用了一个 language map，将自然语言句子转换成 internal representation of knowledge in $\mathbf{K}$，$\mathbf{K} \ni K$ 是知识状态的空间。可以叫这个 map 做 language map：

$$\begin{aligned} \mathcal{L}: \quad \mathbf{S} \quad &\to \mathbf{K} \\ \mathcal{L}: \quad \text{句子} \quad &\mapsto K \end{aligned}$$

（但其实句子只是知识状态 K 的一部分。）

於是我们的 RNN model 变成这样：

$$\text{句子输入} \xrightarrow{\mathcal{L}} \mathbf{K} \xrightarrow{\mathcal{L}^{-1}} \text{句子输出}, \quad \mathbf{K} \circlearrowright D$$

有了这个 language map 是很方便的，但它有几个问题：

- 需要用到 NLP parsing，很麻烦，如果能省略比较好
- 这个 map 基本上 fix 了 internal knowledge representation 的形式。但我的直观（也可能错误）觉得 knowledge representation 应该是「不知道的」比较好，它应该是由学习 D 的过程「诱导」出来的，换句话说：the knowledge representation format should be *induced* from the learning of D. 因为以前在经典逻辑系统中，K = KB (knowledge base) 是一些命题的集合，换句话说 $K = \bigsqcup S_i$，S_i 是句子或「命题」。当时的做法是将 KB 组织成 hierarchical 结构，方便搜寻。但我觉得如果 $\mathbf{K}$ 的结构是这样的话，跟原问题的情况太相似，一切都太「有秩序」，可能不是神经网络最有效的用法。
- 自然语言是需要慢慢「吸收」或「理解」(comprehend) 的，但这过程在 Memory Networks 的模型里忽略了。将自然句子转换成 logical 形式（即 internal representation），几乎是不需时间的 transliteration 过程。如果输入一本《世界历史》的原文，不消一秒便可以转换成 internal representation，但不能说 Genifer 已经「理解」了全书的内容吧？

7 理解 / 慢吸收

所以 $\mathcal{L}$ 不是一个普通 map 而是一个很复杂的过程。

对新输入知识的「慢吸收」包括这些运作：

- consequences （找出新命题的所有可能推论结果，至少在 n 步之内）
- consistency （新信念不和旧信念抵触）
- explanations （新信念可以被已有知识解释）

「理解」的过程可以看成是：经过 n 次的推导 $\mathcal{D}$，K 变成 K'：

$$K \xrightarrow{D^n} K'$$

而 K' 有我们想要的特性（即「后果、和谐、解释」）。问题就是怎样测试 K' 有这些特性？特别困难的原因是 $K' \in \mathbf{K}$ 的 representation 是不透明 的。

K' 的特性可以怎样测试？

似乎唯一的方法是透过这样的查询：

$$K \xrightarrow{D^n} Q\,?$$

但 Q 也是 $\in \mathbf{K}$，而 $\mathbf{K}$ 的结构是不透明的。

如果有 $\mathcal{L}$ 的话，直接将问题用自然语言问，变成 $Q = \mathcal{L}$(句子)，就可以查询 K。换句话说，$\mathcal{L}$ 的好处是容许 direct access to K，将知识状态的内容，直接用自然语言读出来（或写入去）。但现实中不存在一个简单的 $\mathcal{L}$ map。

$\mathcal{L}$ 不存在，找不到 Q 用来查询 K。

8 Speech generation （发言）

如果 Genifer 学会「说话」，就等於查看 K 的内容。但说话不等於将所有知识状态的内容「和盘托出」，而是有两种模式：

- 查询某句子 S_{query} 是不是真的
- 查询关於某题目 $S_{interest}$ 的内容，eg: ”Tell me about your mother”

但这两个动作都要求有某种办法将 $query$ 或 $interest$ 映射到 $\mathbf{K}$ 的某个「位置」，而这正是我们没有的。

「说话」本身是一个复杂的算法，它不是一个简单的映射，而是需要用 D 思考的运作。

无论是「理解」或「说话」，都是复杂算法，而且 $\mathbf{K}$ 是不透明的。表面上似乎没有希望，但神经网络的好处是：可以同时学习两个过程，即使这两个过程互相依赖。

9 学习

为了方便思考，现假设有 $\mathcal{L}$，容易做到以下两种学习方式：

9.1 Inductive learning

举例：

小明是香港人 $\wedge$ 小明戴眼镜
小强是香港人 $\wedge$ 小强戴眼镜
小雄是香港人 $\wedge$ 小雄戴眼镜
小娟是香港人 $\wedge$ 小娟戴眼镜
结论：所有香港人都戴眼镜

用逻辑表示这法则就是：

$$\forall X.\ \text{Hong-Kong}(X) \Rightarrow \text{wear-glasses}(X)$$

但在 RNN 里，这法则是暗含在 D 之中。例如如果 $k =$「大强是香港人」，那么 D 作用在 K 上最终会得出 $K' =$「大强戴眼镜」。

换句话说，已知 K_0，K^*，学习 D：

$$K_0 \xrightarrow{D} .... K^*$$

因为我们可以用 $K = \mathcal{L}$(自然语言句子) 来计算 K_0 和 K^*，所以这个算法是可行的。

References

[1] Weston, Chopra, and Bordes. Memory networks. *ICLR (also arXiv)*, 2015.