



# 现代智能方法：从预测、生成到决策

己亥年己巳月辛酉日之研究生部讨论班材料

作者：罗敏中

组织：中国原子能科学研究院信息中心

时间：May 24, 2019

版本：1.07

 中国原子能科学研究院  
CNNC CHINA INSTITUTE OF ATOMIC ENERGY



人有人的用处；— 控制论之父 罗伯特·维纳

# 目 录

<b>1 本讨论班材料的定位</b>	<b>1</b>
1.1 人工智能是一个新兴交叉学科	1
<b>2 预测方法</b>	<b>2</b>
2.1 预测学习方法概述	2
2.2 损失函数	2
2.3 样本	3
2.4 优化方法	3
2.4.1 梯度下降	3
2.4.2 最大似然	4
<b>3 生成模型</b>	<b>5</b>
3.1 概率密度估计	5
3.2 变分自编码器和生成对抗网络	5
3.2.1 变分自编码器	6
3.2.2 对抗生成网络	6
3.3 生成模型的论文实例	7
<b>4 决策模型</b>	<b>9</b>
4.1 Markov 决策过程	9
4.2 强化学习	10
4.2.1 Model-based 强化学习	11
4.2.2 Model-free 强化学习和 Monte-Carlo 策略估计	11
4.3 决策模型的论文实例	13
<b>A 课后习题</b>	<b>16</b>
A.1 练习题	16
A.2 补充习题	16

# 第1章 本讨论班材料的定位

---

本讨论班材料默认读者有对统计学习方法和深度学习有经验或者直观认识，或者至少对之核心思想有了解。本材料和讨论班的定位是入门，通过介绍一些基础原理，结合当前一些研究实例，使得参与者至少能够对人工智能的研究领域问题、主流解决思路有清晰的认识，进而可以自己甄别学习材料来深入进阶。如有建议或者问题可以邮件<sup>1</sup>联系我沟通解决。本人亦只是硕士三年级生一名，处于学习阶段，如有纰漏错误之处，欢迎指出，必当虚心接受改正。

## 1.1 人工智能是一个新兴交叉学科

在过去的几年间，人工智能因为其广阔的应用前景和令人印象深刻的阶段性成果 [Krizhevsky et al. \(2012\)](#) [Goodfellow et al. \(2014\)](#)，涌入了大量的新研究人员，如何构造一个难度和广度都适宜的教程来引导这些研究者，成为了一个持续性难题（这可归因于人工智能的快速更新迭代、对数学可深可浅的要求、其交叉学科的本质）。在这里，我们首先需要知道人工智能学科至少由哪些交叉学科构成，或者说至少和哪些问题有关系：

1. 认知心理学；
2. 计算神经科学；
3. 计算数学尤其是计算方法、数值分析等等；
4. 统计学尤其是统计推断；
5. 概率论尤其是随机过程论；
6. 分析学尤其是实分析理论；
7. 哲学里的逻辑学；
8. 符号计算 (数学的机械化)；
9. 机器学习；
10. 统计热力学 ([Hinton](#) 就是这个方向出身)；
11. 语言学尤其是计算语言学；
12. 计算机视觉 (不要小看传统压缩/滤波模型)；
13. 离散数学尤其是图论 (有一种越来越火的趋势，如文献 [Battaglia et al. \(2018\)](#) 的工作)；
14. 现代控制理论 (强化学习某种程度上是它玩剩下的)；



**注意** 入门不久的同学不要沉溺于所谓的深度学习中，而要打下坚实的理论基础，从上述方向中选择两到三个了解，至少扎实地学习精通其中一个，才能在人工智能的学界大潮中站稳。

---

<sup>1</sup>[luomincentos@ciae.ac.cn](mailto:luomincentos@ciae.ac.cn).

## 第2章 预测方法

### 2.1 预测学习方法概述

哪些问题属于预测学习的范畴？不严格地说，分类模型、识别模型（如行人探测/人脸识别）、推荐系统、主题抽提、语言实体识别这些问题都属于预测学习。预测学习可以概括为根据输入  $x$  给出标签  $y$  的学习模型  $y = f_{\theta}(x)$ 。在这一章中我主要给出一些预测学习里一些关键元素概念：损失函数、样本、优化方法的入门教学。

### 2.2 损失函数

首先是损失函数的概念：对于任意假设类集合  $\mathcal{H}$  和定义域  $Z$ ，令  $\ell$  为域  $\mathcal{H} \times Z$  上到非负实数域的一个映射： $\ell: \mathcal{H} \times Z \rightarrow \mathbb{R}_+$ ，称之为**损失函数**。

**注** 假设类集合  $\mathcal{H}$  是指可能的预测器的集合，比如神经网络多个权重状态的集合；定义域  $Z$  则是指输入的空间，比如对图片作分类时， $Z$  就是图片的集合。

损失函数可以分为经验损失和期望损失。经验损失即学习已有数据集得到的损失，通过对数据集  $S = (z_1, \dots, z_m) \in Z^m$  进行采样学习，可以得到经验损失：

$$L_S(h) \stackrel{\text{def}}{=} \frac{1}{m} \sum_{i=1}^m \ell(h, z_i)$$

期望损失即预测器在真实任务上会遭遇的损失，通过对真实样本分布采样  $z \sim \mathcal{D}$ ，即可得到期望损失：

$$L_{\mathcal{D}}(h) \stackrel{\text{def}}{=} \mathbb{E}_{z \sim \mathcal{D}} [\ell(h, z)]$$

一个自然的想法是，有时候不太能去优化期望损失，那么只能依赖已有数据集争取得到最小经验损失，这即为最小经验损失准则 (Empirical Risk Minimization, ERM)。下面这道习题是贝叶斯估计和最小经验损失准则的一个巧妙联系：

#### 思考题 2.1. 贝叶斯参数估计

证明：如果损失函数为二次函数，即  $\lambda(\hat{\theta}, \theta) = (\hat{\theta} - \theta)^2$ ，则当  $\theta$  的贝叶斯估计量  $\hat{\theta}$  是为如下条件期望形式时，贝叶斯风险  $R = \int_Z R(\hat{\theta}|\mathbf{x})p(\mathbf{x})d\mathbf{x}$  最小。

$$\hat{\theta} = E[\theta|\mathbf{x}] = \int_{\Theta} \theta p(\theta|\mathbf{x})d\theta$$





## 2.3 样本

什么样的样本集的质量更优，下面我们将给出度量样本优劣的量化定义。一个自然的想法是，如果一个样本集使得损失函数能够在真实分布工作时和训练集上差异不大，那就是良好的。如下定义基于上述想法：

### 定义 2.1. $\epsilon$ -代表性样本

如果样本集  $S$  满足如下不等式，那么则称样本集  $S$  是  $\epsilon$ -代表性样本：

$$\forall h \in \mathcal{H}, \quad |L_S(h) - L_{\mathcal{D}}(h)| \leq \epsilon$$



下面的练习将表明：基于  $\epsilon$ -代表性样本集构建的预测器和理论最优的预测器的差距有多大。

### 思考题 2.2. 样本对预测器效果的影响

假设一个训练集  $S$  是  $\epsilon/2$ -代表性样本，那么对于任何一个  $\text{ERM}_{\mathcal{H}}(S)$  的输出，即对任意  $h_S \in \arg\min_{h \in \mathcal{H}} L_S(h)$  都满足：

$$L_{\mathcal{D}}(h_S) \leq \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon$$



## 2.4 优化方法

有了样本和损失函数，接下来就差怎么优化了。这个环节也是各个学派和主义的学者争论的激烈地段，在这里我们只介绍两种最基础的优化方法，搞懂这些基础方法，有助于今后对任何其他改进方法的理解。

### 2.4.1 梯度下降

最小化损失函数  $L(\hat{y}, y)$  可以用梯度下降解决 (gradient descent)，但是实际中使用梯度下降算法会使学习变得相当缓慢。这是因为：对于每个训练实例  $x$ ，都要计算梯度向量  $\Delta L$ 。如果训练数据集过大，会花费很长时间，学习过程太慢。所以实际中使用随机梯度下降算法 (stochastic gradient descent)。其基本思想是，从所有训练实例中随机或者依固定概率分布取一个小的采样 (sample):  $X_1, X_2, \dots, X_m$  (mini-batch)，来估计  $\Delta L$ ，大大提高学习速度。如果样本够大，即可使随机梯度收敛至最佳梯度：

$$\frac{\sum_{j=1}^m \nabla C_{X_j}}{m} \approx \frac{\sum_x \nabla C_x}{n} = \nabla C$$

将随机梯度下降代入参数更新方程即可得（在这里可以选择一个 mini-batch 用来训练，直到用完所有的训练实例，一轮迭代 (epoch) 即可完成。）：



$$\theta_k \rightarrow \theta'_k = \theta_k - \frac{\eta}{m} \sum_j \frac{\partial C_{x_j}}{\partial \theta_k}$$

因此基于梯度下降的学习方法的核心思维可以总结为，代价函数  $L(\hat{y}, y)$  就是以权值  $\theta$  为参数的一个函数，要做的就是去寻找最小的误差点  $L^*$ ，求最小就是要去找权值点  $\theta^*$ ，因为这是一个以权值为参数的函数，因此求导，导数的含义就是变化率，当求出导数的值为正数时，则说明这是一个上升的点，上升说明这个函数的误差率在变大，因此要往其反方向走（在这里定义以增大的方向为正方向），故切线的方向即为使误差下降的方向。而当导数为负数时，则相反。

### 2.4.2 最大似然

最大似然估计 (Maximum likelihood estimation) 是利用条件概率的估计方法， $\theta$  是前置条件，理解为在  $\theta$  的前提下，事件  $x$  发生的概率，相对应的似然可以表示为： $\mathcal{L}(\theta|x) = P(x|\theta)$ ，也就是说的是依赖样本  $x$  估计  $\theta$ ， $P(x|\theta)$  最大化蕴含着“发生即为大概率”的思想（利用已知的样本结果，反推最有可能（最大概率）导致这样结果的参数值）。

最大似然估计求解参数的一般步骤是先写出似然函数  $L(\theta) = \prod_{i=1}^n p(x_i|\theta)$ ，再对似然函数两边取对数  $\ln L(\theta) = \sum_{i=1}^n \ln p(x_i; \theta)$ ，最后求导数并令之为 0 即可： $\frac{d \ln L(\theta)}{d\theta} = 0$ ；

下面的习题将证明 KL 散度 (Kullback-Leibler divergence 即  $KL(P||Q) = \int P(x) \log \frac{P(x)}{Q(x)} dx$ ，度量了两个分布的差异) 和最大似然模型的等价性。

#### 思考题 2.3. 最大似然即 KL 散度最小化

取最大似然的损失函数：

$$\mathcal{L}(\theta, x) = -\log(\mathcal{P}_{\theta}(x))$$

优化目标即为  $\theta = \min_{\theta} -\log(\mathcal{P}_{\theta}(x))$ ，假设真实数据分布服从  $x \sim Q(x)$ ，那么证明：最大似然即 KL 散度最小化等价。



## 第3章 生成模型

如果我不知如何创造一样物体,就不能算是真的懂它;— 理查德·费曼

### 3.1 概率密度估计

概率生成模型的目的,就是找出给定观测数据内部的统计规律,并且能够基于所得到的概率分布模型,产生全新的,与观测数据类似的数据 JS (2017)。

在开始这一章的内容之前,谈一个我对生成模型的理解。生成模型在于**创造**,但不是无根据的随意创造,而是根据已有的数据集  $X$ , 尝试去创造产生数据集  $X$  的源头  $\mathbb{P}(x)$ 。那么就是对  $\mathbb{P}(x)$  的一个概率密度估计,假如已经知道了分布  $\mathbb{P}_\theta(x)$  的形式,比如正态分布,那么只需要估计  $\mathbb{P}_\theta(x)$  的参数  $\theta$  即可,这就是参数估计。如果一个估计问题所涉及到的分布未知或不能用有穷参数来刻画,称这种估计为非参数估计,需要越过参数去直接估计  $\mathbb{P}(x)$  在每一点的概率密度。

这里介绍一种简单常用的非参数估计方法,窗函数估计法:取窗函数  $\varphi(x) \sim N(0, 1)$ , 则对于样本集合  $\{x_i\}$  的窗函数估计可以表达为:

$$\hat{p}_N(x) = \frac{1}{Nh_N} \sum_{i=1}^N \varphi\left(\frac{x - x_i}{h_N}\right)$$

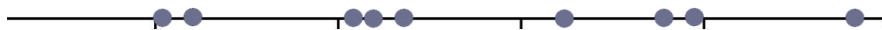
其中  $N$  为样本集容量,参数  $h_N$  是某常数。



**注意** 学过数值分析的同学可以看看,窗函数估计法是不是很像某种插值拟合方法的形式?

#### 思考题 3.1. 简单非参数估计

请以定性非定量的画法画出下面分布的概率密度:



### 3.2 变分自编码器和生成对抗网络

在这一节中我们将结合深度生成模型来讲解生成模型的应用和价值。变分自编码器 VAE 跟生成对抗网络 GAN 的目标基本是一致的——希望构建一个从隐变量  $Z$  生成目标数据  $X$  的模型,但是实现上有所不同。更准确地讲,它们是假设了  $Z$  服从某些常见的分布(比如正态分布或均匀分布),然后希望训练一个模型  $X = g(Z)$ ,这个模型能够将

原来的概率分布映射到训练集的概率分布，也就是说，它们的目的都是进行分布之间的变换。

### 3.2.1 变分自编码器

在这里我们一上来即开始讲变分自编码器其实有点操之过急，因此在此还是需要提及，变分自编码器 VAE 只是自编码器的一种，所谓的自编码器，则是由编码器  $p(\mathbf{z}|\mathbf{x})$  和解码器  $q(\mathbf{x}|\mathbf{z})$  构成，其核心思想是想得到一组可以表达和重构变量  $\mathbf{x}$  的压缩变量  $\mathbf{z}$ （也叫潜变量）。

在这里举一个实例辅助理解，曹操利用人对酸味的感官刺激来骗士兵有梅林的故事是众所周知的，可以用成语“望梅止渴”来指代（相当于编码过程），我们其他人见到“望梅止渴”这个成语，即可还原出整个故事（相当于解码过程）。

VAE 是自编码器的一种高效实现。第一步，是编码：即使用变量  $\mathbf{x}$  来获取压缩变量  $\mathbf{z}$ ：

$$\log p_{\phi}(\mathbf{z}|\mathbf{x}^{(i)}) = \log \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}^{(i)}, \boldsymbol{\sigma}^{2(i)}\mathbf{I})$$

第二步，是解码，这一步的关键是最小化重构误差  $\mathcal{L}(\hat{\mathbf{X}}, \mathbf{X})$ ，即可将误差也传递回编码层，使得编码层编码出能有利于解码的压缩编码。除此之外，VAE 也尽量让编码器  $p(\mathbf{z}|\mathbf{x})$  都以正态分布为目标（这样可以防止压缩编码的噪声为零，同时保证了模型具有生成能力，还有一个原因，请联系回忆上一节中关于概率密度估计的问题来思考），这需要引入编码器与标准正态分布的 KL 散度  $KL(N(\boldsymbol{\mu}, \boldsymbol{\sigma}^2) \| N(0, \mathbf{I}))$  来约束：

$$\mathcal{L}_{\boldsymbol{\mu}, \boldsymbol{\sigma}^2} = \frac{1}{2} \sum_{i=1}^d \left( \mu_{(i)}^2 + \sigma_{(i)}^2 - \log \sigma_{(i)}^2 - 1 \right)$$

最终 VAE 的损失函数可以为  $\mathcal{L}(\hat{\mathbf{X}}_k, \mathbf{X}_k) + \mathcal{L}_{\boldsymbol{\mu}, \boldsymbol{\sigma}^2}$ ；最后一个问题，为何会叫做“变分”自编码器 **JL (2018)**：因为  $KL(p(x) \| q(x))$  实际上是一个泛函，要对泛函求极值就要用到变分法，当然，这里的变分法只是普通微积分的平行推广，还没涉及到真正复杂的变分法。而 VAE 的变分下界，是直接基于 KL 散度就得到的。所以直接承认了 KL 散度的话，就没有变分的什么事了。

### 3.2.2 对抗生成网络

GAN 模型包括了一个生成模型  $G$  即  $\mathbf{x} = G(\mathbf{z}; \theta^{(G)})$  和一个判别模型  $D$  即  $D(\mathbf{x}) \in \{0, 1\}$ ，GAN 的目标函数是关于  $D$  与  $G$  的一个零和游戏。也是一个最小-最大化问题。

这里判别模型  $D$  实际上是对数据的来源进行一个判别：究竟这个数据是来自真实的数据分布  $P_{data}$ ，还是来自于一个生成模型  $G$  所产生的一个数据分布  $P_g$ 。

判别模型  $D$  的训练目的就是要尽量最大化自己的判别准确率。当这个数据被判别为来自于真实数据时，标注 1，自于生成数据时，标注 0。

而与这个目的相反的是，生成模型  $G$  的训练目标，就是要最小化判别模型  $D$  的判别



准确率。在训练过程中，GAN 采用了一种非常直接的交替优化方式，它可以分为两个阶段，第一个阶段是固定判别模型  $D$ ，然后优化生成模型  $G$ ，使得判别模型的准确率尽量降低。而另一个阶段是固定生成模型  $G$ ，来提高判别模型的准确率。

### 思考题 3.2. GAN 的优化函数

假设 GAN 的优化项如下：

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} \log D(\mathbf{x}) - \frac{1}{2}\mathbb{E}_{\mathbf{z}} \log(1 - D(G(\mathbf{z})))$$

$$J^{(G)} = -J^{(D)}$$

请问， $D(\mathbf{x})$  的解是什么，得到这个解需要何种假设？



在训练过程中，GAN 采用了一种非常直接的交替优化方式，它可以分为两个阶段，第一个阶段是固定判别模型  $D$ ，然后优化生成模型  $G$ ，使得判别模型的准确率尽量降低。而另一个阶段是固定生成模型  $G$ ，来提高判别模型  $D$  的准确率，最终的优化函数是：

$$V(G, D) = \mathbb{E}_{p_{\text{data}}} \log D(\mathbf{x}) + \mathbb{E}_{p_{\text{generator}}} (\log(1 - D(\mathbf{x})))$$

## 3.3 生成模型的论文实例

在这里我们借一篇高质量的计算机论文 [Tao et al. \(2017\)](#) 来深入对生成模型的理解。

在本文中，作者提出了一种注意生成对抗网络 (AttnGAN)，它实现了注意力驱动的多阶段细化，以实现文本到图像生成的过程。通过注意力生成网络，AttnGAN 可以通过利用自然语言描述中的相关单词来合成图像的不同子区域的细粒度细节。此外，提出了一种深度注意多模态相似度模型来计算用于训练生成器模型  $G$  的图像文本匹配损失。AttnGAN 明显优于先前的最佳水平，在 CUB 数据集上领先了历史最佳得分 14.14%，在更具挑战性的 COCO 数据集上提升了 170.25%。还通过可视化 AttnGAN 的注意层给出了详细分析。本文首次表明分层注意力 GAN 能够自动选择单词级别的信息以生成图像的不同部分。

这篇文章也蕴含了计算机 AI 论文的一些必要元素：方法描述、关键技术提要、在数据集效果是否超越 state-of-the-art、效果分析。总体来说本文理论性很浅，应用效果很佳，适合入门者熟读。

这篇文章的一个亮点是，实现了编码解码的跨元素，将文字编码为潜变量再解码为文字并不困难，但是本文是将文字编码再解码为图像，这个实现难度可想而知。本文的方法架构如下：

本文的优化项也分为生成和识别两部分，第一部分是生成模型的优化函数：

$$\mathcal{L}_{G_i} = \underbrace{-\frac{1}{2}\mathbb{E}_{\hat{x}_i \sim p_{G_i}} [\log(D_i(\hat{x}_i))]}_{\text{unconditional loss}} - \underbrace{\frac{1}{2}\mathbb{E}_{\hat{x}_i \sim p_{G_i}} [\log(D_i(\hat{x}_i, \bar{e}))]}_{\text{conditional loss}}$$

第二部分是识别模型的优化函数：



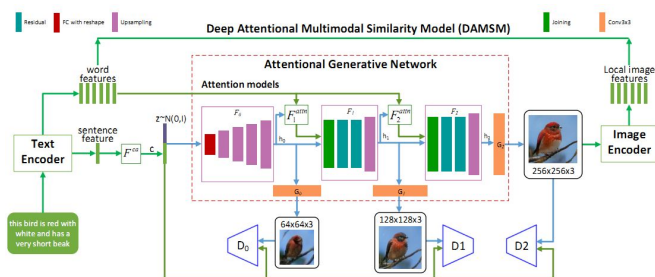


图 3.1: 文章提出的 AttnGAN 的架构。每个注意模型自动关注用于生成图像的不同子区域的信息 (即最相关的词向量), DAMSM 为生成网络  $G$  计算图像文本内容匹配损失。

$$\mathcal{L}_{D_i} = \underbrace{-\frac{1}{2}\mathbb{E}_{x_i \sim p_{\text{data}}}[\log D_i(x_i)] - \frac{1}{2}\mathbb{E}_{\hat{x}_i \sim p_G}[\log(1 - D_i(\hat{x}_i))]}_{\text{unconditional loss}} - \underbrace{\frac{1}{2}\mathbb{E}_{x_i \sim p_{\text{data}_i}}[\log D_i(x_i, \bar{e})] - \frac{1}{2}\mathbb{E}_{\hat{x}_i \sim p_{G_i}}[\log(1 - D_i(\hat{x}_i, \bar{e}))]}_{\text{conditional loss}}$$



图 3.2: 实现效果, 生成器能够针对文本中单词级别的语义信息进行图像解码。

## 第4章 决策模型

The only stupid question is the one you were afraid to ask but never did.—Rich Sutton

### 4.1 Markov 决策过程

马尔可夫决策过程 (Markov Decision Processes, MDPs) 是一个智能体 (Agent) 采取行动 (Action) 从而改变自己的状态 (State) 获得奖励 (Reward) 与环境 (Environment) 发生交互的循环过程。MDP 的策略完全取决于当前状态，这也是它马尔可夫性质的体现，根据当前的状态来决定动作。因此第一个需要引入的定义是马尔科夫性：

#### 定义 4.1. 马尔科夫性

状态  $S_t$  具备马尔科夫性当且仅当：

$$\mathbb{P}[S_{t+1}|S_t] = \mathbb{P}[S_{t+1}|S_1, \dots, S_t]$$

以马尔可夫随机过程为理论基础，马尔科夫决策过程定义如下：

#### 定义 4.2. 马尔科夫决策过程

马尔科夫决策过程（马尔科夫链）是序列对  $\langle \mathcal{S}, \mathcal{P} \rangle$ ：

1.  $\mathcal{S}$  是状态的有限集；
2.  $\mathcal{P}$  为概率转移矩阵即：

$$\mathcal{P}_{ss'} = \mathbb{P}[S_{t+1} = s' | S_t = s]$$

更具体的，考虑决策行为和奖励机制，马尔科夫决策亦可扩展表示为：

#### 定义 4.3. 马尔科夫奖励机制过程

马尔科夫奖励机制过程为  $\langle \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ ：

1.  $\mathcal{S}$  是状态的有限集；
2.  $\mathcal{P}$  为概率转移矩阵即：

$$\mathcal{P}_{ss'} = \mathbb{P}[S_{t+1} = s' | S_t = s]$$

3.  $\mathcal{R}$  为奖励函数即  $\mathcal{R}_s = \mathbb{E}[R_{t+1} | S_t = s]$ ；
4.  $\gamma$  为折扣因子即  $\gamma \in [0, 1]$ ；

这就更加接近强化学习的模式了，再考虑多个时间区间的奖励，可以写出  $t$  时刻的总奖励反馈：

$$G_t = R_{t+1} + \gamma R_{t+2} + \dots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

一个自然而然的问题是，那机器人如何估计环境和策略带来的奖励 (君子知命不惧，日日自新)：

#### 定义 4.4. 状态值和决策值估计

决策函数  $\pi$  是一个基于状态预测决策的分布函数：

$$\pi(a|s) = \mathbb{P}[A_t = a | S_t = s]$$

基于马尔科夫决策过程的状态值函数  $v_\pi(s)$  是根据状态  $s$  和对应策略  $\pi$  的期望奖励：

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

基于马尔科夫决策过程的决策值函数  $q_\pi(s, a)$  是根据状态  $s$ 、采取决策  $a$  和对应策略  $\pi$  的期望奖励：

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$



至此，我们的目标是在给定当前所处的状态下，寻求最优解的决策行为，最大限度地提高环境所提供的长期预期回报。决策问题可以按照时间或空间分成多个阶段，每个阶段做出决策从而使整个过程取得效果最优的多阶段决策问题，可以用动态规划方法求解。某一阶段最优决策的问题，通过贝尔曼方程转化为下一阶段最优决策的子问题，从而初始状态的最优决策可以由终状态的最优决策 (一般易解) 问题逐步迭代求解。现实中贝尔曼方程多被用来解决马尔科夫决策过程问题，即最优决策只依赖于当前状态而和状态的历史无关：

$$v_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s]$$

$$q_\pi(s, a) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) | S_t = s, A_t = a]$$

上述方程即贝尔曼方程，它将相邻状态的价值函数联系起来，求解  $V(S_t)$  的问题可以转化为求解  $V(S_{t+1})$  的问题。

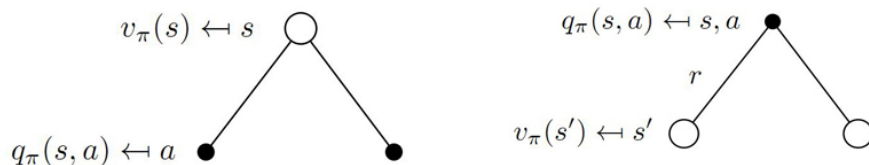
#### 思考题 4.1. Bellman 方程的表达式

请根据下图写出  $v_\pi(s)$  和  $q_\pi(s, a)$  的表达式；



## 4.2 强化学习

首先是强化学习分类问题，可分为 Model-free 和 Model-based，以 gridworld 这个机器人走迷宫的例子，即一个迷宫机器人从起点出发通过强化学习的方式选择出到达终点



的最优路径。

model-based 方式就是给机器人地图全开，事先了解好整个游戏环境根据过往的经验选取最优策略（回想动态规划），也就是说 model-based 他能够通过缓存（比如一个矩阵）来预判断接下来将要发生的所有情况，然后选择这些缓存情况中最好的那种，并依据这种情况来采取下一步的策略。

model-free 方法就是不依赖模型，这种情况下就是直接将机器人放置于迷宫中探索，机器人会根据现实环境的反馈采取下一步的动作。这种方法不对环境进行建模也能找到最优的策略。Model-free 的方法有很多，像 Q-learning, Sarsa, Policy Gradients 都是从环境中得到反馈然后从中学习。

### 4.2.1 Model-based 强化学习

Model-based 强化学习关键在于， $\langle \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma \rangle$  都是已知的（这个要求是不是有点苛刻？因此现实工程中 Model-free 强化学习更实用）。当 agent 学习的模型能够非常贴近于环境时，它就可以直接通过一些规划算法来找到最优策略，具体来说：当 agent 已知任何状态下执行任何动作获得的回报，即  $\mathcal{R}(s_t, a_t)$  已知，而且下一个状态也能通过  $\mathcal{P}(s_{t+1}|s_t, a_t)$  被计算，那么这个问题很容易就通过动态规划算法求解，尤其是当  $\mathcal{P}(s_{t+1}|s_t, a_t) = 1$  时，直接利用贪心算法，每次执行只需选择当前状态  $s_t$  下回报函数取最大值的动作  $\max_a \mathcal{R}(s, a|s = s_t)$  即可。

### 4.2.2 Model-free 强化学习和 Monte-Carlo 策略估计

但是在强化学习中，Agent 却不是那么容易知晓 MDP 中所有的元素的，比如，Agent 也许不会知道环境将会如何改变当它执行了一个动作后（状态转移概率函数  $\mathcal{P}$ ），也不会知道它执行这个动作获得即时的奖励将会是多少（奖励函数  $\mathcal{R}$ ），Agent 能做的就是：根据自己已有的策略  $\pi(a|s)$  选择关于当前状态  $s$  下自己认为好的动作  $a$ ，执行此动作给环境，观察环境给出的反馈  $r$  和下一个状态  $s'$ ，并根据这个反馈  $r$  调整更新自己的策略  $\pi(a|s)$ ，这样反复迭代，直到找到一种最优的策略  $\pi^*(a|s)$  能够最大限度获得正反馈。

事实上机器学习中的许多重要工具都基于从某种分布中采样以及用这些样本对目标量做一个 Monte-Carlo 估计 Goodfellow et al. (2016)。有许多原因使我们希望从某个分布中采样。当我们需要以较小的代价近似许多项的和或某个积分时，采样是一种很灵活的选择。

当无法精确计算和或积分（例如，和具有指数数量个项，且无法被精确简化）时，通常可以使用 Monte-Carlo 采样来近似它。这种想法把和或者积分视作某分布下的期望，然



后通过估计对应的平均值来近似这个期望。令

$$s = \sum_{\mathbf{x}} p(\mathbf{x}) f(\mathbf{x}) = E_p[f(\mathbf{x})]$$

或者

$$s = \int p(\mathbf{x}) f(\mathbf{x}) d\mathbf{x} = E_p[f(\mathbf{x})]$$

为我们所需要估计的和或者积分，写成期望的形式， $p$  是一个关于随机变量  $\mathbf{x}$  的概率分布（求和时）或者概率密度函数（求积分时）。我们可以通过从  $p$  中抽取  $n$  个样本  $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)}$  来近似  $s$  并得到一个经验平均值（大数定理可以保证它收敛到真值，其无偏性的证明留作习题）：

$$\hat{s}_n = \frac{1}{n} \sum_{i=1}^n f(\mathbf{x}^{(i)})$$

至此，可以根据上述内容来理解 Monte-Carlo 策略估计：用 Monte-Carlo 采样来估计策略函数  $\pi$ 。这可以等价于估计状态值函数（一言以蔽之：“经验平均”）：

$$v_{\pi}(m) = \frac{1}{m} \sum_j G_j = \frac{1}{m} \sum_j \sum_{k=0}^{K_j} \gamma^k R_{t+k+1}$$

#### 思考题 4.2. 和随机梯度下降的某种联系

请证明  $v_{\pi}(m) = \frac{1}{m} \sum_j G_j$  和随机梯度下降的形式等价；

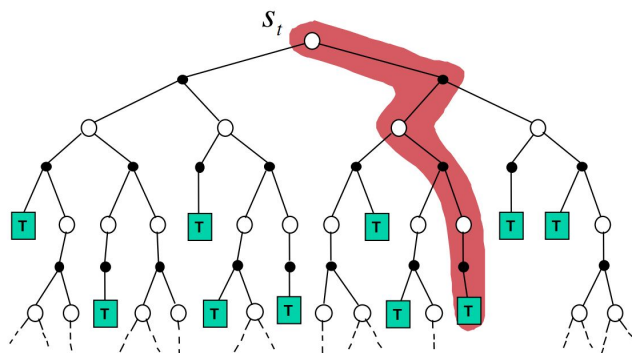


图 4.1: Monte-Carlo 估计策略函数是靠“遍历估计”的思路做，相当于预先演绎再求平均。

同理，我们还可以用“遍历估计”的思路估计其他的 MDP 中的量：

$$\hat{\phi}_{s,s'}^a = \frac{1}{N(s,a)} \sum_{k=1}^K \sum_{t=1}^{T_k} 1(s_t^k, a_t^k, s_{t+1}^k = s, a, s')$$

$$\hat{\mathcal{R}}_s^a = \frac{1}{N(s,a)} \sum_{k=1}^K \sum_{t=1}^{T_k} 1(s_t^k, a_t^k = s, a) r_t^k$$

Deep Q-learning 等一干深度强化学习的编程可以由 keras 或者 pytorch 等开源框架方便地实现:

```
def build_model(self):
    """DQN的keras实现(Q-Value的神经网络逼近)"""
    inputs = Input(shape=(4,))
    x = Dense(16, activation='relu')(inputs)
    x = Dense(16, activation='relu')(x)

    value = Dense(2, activation='linear')(x)
    a = Dense(2, activation='linear')(x)
    meam = Lambda(lambda x: K.mean(x, axis=1, keepdims=True))(a)
    advantage = Subtract()([a, meam])

    q = Add()([value, advantage])

    model = Model(inputs=inputs, outputs=q)

    model.compile(loss='mse', optimizer=Adam(1e-3))

    return model
```

### 4.3 决策模型的论文实例

在这里我们借一篇高质量的计算机论文 [Yu et al. \(2018\)](#) 来深入对决策模型的理解。

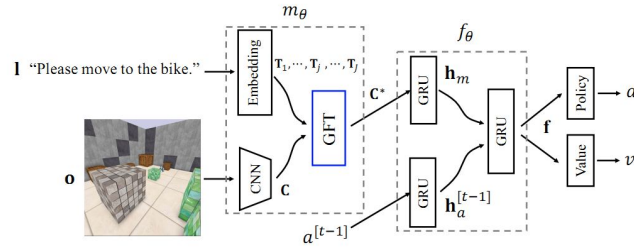
最近的研究中,研究者越来越关注在虚拟环境中训练 Agent,通过深度强化学习来执行语言导向的任务。在本文中,作者提出了一个简单但有效的神经语言模块,用于训练机器人,可以从零训练,将原始像素,非结构化语言命令和稀疏奖励作为输入。作者将语言基础过程建模为语言引导的视觉特征转换,其中潜在的句子嵌入被表征为转换矩阵。在几个语言导向的机器人寻找目标任务中,难度较大且具有可观察性、并需要简单的推理,作者的模块明显优于 state-of-the-art。作者还发布了 XWORLD3D,是一个易于定制的 3D 训练环境,可以对其进行修改以评估各种强化学习机器人。

百度这篇 paper 除了融入强化学习外,倒是实现了一种“**如何将语言嵌入到这个虚拟环境来和 Agent 交互**”的方法:具体也不难,其实就是利用强化学习的架构,并将语言标签  $l$  和第一人称图像  $O$  嵌入一个 Network。目标函数就是最大化 Reward 的期望:

$$-\mathbb{E}_{s^{[t]}, a^{[t]}, l} = [(\nabla_{\theta} \log \pi_{\theta}(a^{[t]} | o^{[t]}, h^{[t-1]}, l) + \eta \nabla_{\theta} v_{\theta}(o^{[t]}, h^{[t-1]}, l)) A^{[t]} + \kappa \nabla_{\theta} \mathcal{E}(\pi_{\theta})]$$

文章的方法架构如下:

本文的一个亮点就是作者提出了一个如何融合语义和视觉信息的方案,即其提到的



GFT 模块：

$$\mathbf{C}^{[j]} = g \left( \mathbf{T}_j \begin{bmatrix} \mathbf{C}^{[j-1]} \\ \mathbf{1}^\top \end{bmatrix} \right), \quad 1 \leq j \leq J$$

事实上这是一种依赖递归神经网络的迭代法，其中  $\{\mathbf{T}_j \in \mathbb{R}^{D \times (D+1)}\}$  是由语义信号产生的序列，最终  $\mathbf{C}$  就是融合了语言和视觉信息的特征矩阵。在其架构右端我们可以看到其对策略函数和状态值函数的拟合估计。

## 参考文献

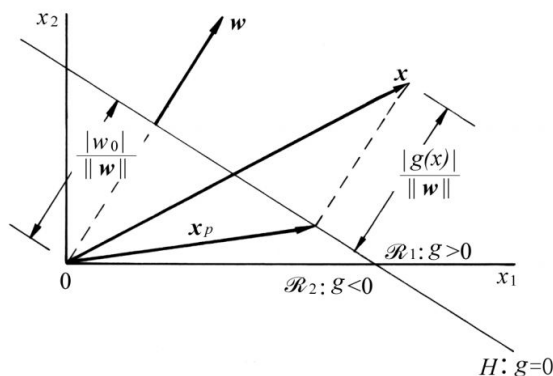
---

- BATTAGLIA P W, HAMRICK J B, BAPST V, et al., 2018. Relational inductive biases, deep learning, and graph networks[J].
- GOODFELLOW I, BENGIO Y, COURVILLE A, 2016. Deep learning[M/OL]. MIT Press. <http://www.deeplearningbook.org>.
- GOODFELLOW I J, POUGETABADIE J, MIRZA M, et al., 2014. Generative adversarial networks[J]. Advances in Neural Information Processing Systems, 3:2672-2680.
- JL S, 2018. Vae[EB/OL]. <https://kexue.fm/archives/5253>.
- JS F, 2017. bloggan[EB/OL]. <https://www.jianshu.com/p/80bd4d4c2992>.
- KRIZHEVSKY A, SUTSKEVER I, HINTON G E, 2012. Imagenet classification with deep convolutional neural networks[C]//International Conference on Neural Information Processing Systems. [S.l.: s.n.].
- TAO X, ZHANG P, HUANG Q, et al., 2017. Attngan: Fine-grained text to image generation with attentional generative adversarial networks[J].
- YU H, LIAN X, ZHANG H, et al., 2018. Guided feature transformation (gft): A neural language grounding module for embodied agents[J].

## 附录 课后习题

### A.1 练习题

- 练习 A.1 证明：关于高斯分布的方差的极大似然估计是有偏的。
- 练习 A.2 (2015 年春, 中科院自动化所考博真题) 关于神经网络：(1) 针对多层前馈神经网络，请给出反向传播算法的工作原理和训练步骤；(2) 请分析“在前馈神经网络中，隐含层数越多对分类预测可能产生的影响”。
- 练习 A.3 (2008 年秋, 中科院计算所考博真题) 样本  $x$  的类别预测后验概率可以表示为  $\mathbb{P}(\omega_i|x)$ , ( $i = 1 \dots M$ )，后验概率最大的类别表示为  $\omega_{max}$ ：
1. 证明  $\mathbb{P}(\omega_{max}|x) \leq 1/M$ ；
  2. 证明最小错误决策的错误率为  $P_E = 1 - \int \mathbb{P}(\omega_{max}|x)p(x)dx$ ；  
提示：最小错误决策即取后验概率最大；
- 练习 A.4 证明： $x$  到超平面的投影是  $x_p = x - \frac{g(x)}{\|w\|^2} w$ ；



### A.2 补充习题

- 练习 A.5 (神经网络是万能拟合器) 令  $f: [-1, 1]^n \rightarrow [-1, 1]$  是  $\rho$ -利普希茨的，取  $\epsilon > 0$ ，现在构造一个带 sigmoid 激活函数的神经网络  $N: [-1, 1]^n \rightarrow [-1, 1]$ ，证明：对任意  $\mathbf{x} \in [-1, 1]^n$ ，都有  $|f(\mathbf{x}) - N(\mathbf{x})| \leq \epsilon$ 。

$\rho$ -利普希茨的定义：


#### 定义 A.1. 利普希茨性

取  $C \subset \mathbb{R}^d$ ，称函数  $f: \mathbb{R}^d \rightarrow \mathbb{R}^k$  是  $\rho$ -利普希茨的，若对于任意  $\mathbf{w}_1, \mathbf{w}_2 \in C$  有：

$$\|f(\mathbf{w}_1) - f(\mathbf{w}_2)\| \leq \rho \|\mathbf{w}_1 - \mathbf{w}_2\|$$





 **练习 A.6** (随机梯度下降算法的一个界) 对于训练集  $\mathbf{v}_1, \dots, \mathbf{v}_T$ , 对于初始权重  $\mathbf{w}^{(1)} = \mathbf{0}$  和更新算法:  $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$ , 有:

$$\sum_{t=1}^T \left\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t \right\rangle \leq \frac{\|\mathbf{w}^*\|^2}{2\eta} + \frac{\eta}{2} \sum_{t=1}^T \|\mathbf{v}_t\|^2$$