生成模型读书笔记四

2021年1月10日 22:37

1. **概率模型的定义**
   a. 问题设定
      假设问题的输入空间是X，输出空间是Y，训练集D就由数据样本(x,y)构成
   b. 概率模型 vs 非概率模型
      i. 概率模型
         前面提到的都可以称为概率模型，即学习目的是为了学出一个具体的分布函数(概率密度函数)，p(x,y)或者p(y|x)，得到模型后根据贝叶斯最小风险准则来得到输入对应的输出y。
         过程：先预设一个分布形式，通过对模型参数的估计算出分布函数，最后应用模型计算概率得到输出。
         (贝叶斯最小风险准则，简而言之就是因为算出来的是概率，所以y的取值有多种可能只是可能性大小不一样而已，一般会有一个自定义的决策风险表，比较不同决策的风险大小来决定最后取值，当然多数情况下默认所有决策的风险是一样的，所以就是取使得p(y|x)最大的y即可， 决策风险例子：疾病诊断，漏诊与误诊风险不一样)
      ii. 非概率模型
         直接学习输入空间到输出空间的映射关系y=h(x)，学习过程中不涉及概率密度的估计。
         H(x)通常是通过先验知识来选择的，比如用线性函数还是非线性函数(先验知识：x和y是线性还是非线性关系)，然后在这个假设空间中找出一个泛化误差最小的假设出来，这个就称为期望风险

$$h^* = \arg\min_{h \in H} \varepsilon(h) = \arg\min_{h \in H} \sum_{x,y} l(h(x), y) P(x, y) \cdots\cdots\cdots (III)$$

         L(h(x), y)就是loss函数，在这个式子中还是有p(x,y)项的，但是在非概率模型中我们不对联合分布进行建模，没法求解，所以将它改成经验误差最小，就是等概率地计算所有数据点的误差，这个叫经验风险：

$$g = \arg\min_{h \in H} \hat{\varepsilon}(h) = \arg\min_{h \in H} \frac{1}{m} \sum_{i=1}^{m} l(h(x^{(i)}), y^{(i)}) \cdots\cdots (IV)$$

         理论依据是大数定律，当训练样例无穷多的时候，假设的经验误差会依概率收敛到假设的泛化误差。
         先验知识除了设定假设空间H以外还有另一个作用，就是对loss添加正则化项（不止是常见的L1,L2等，比如motion synthesis论文中常用的相邻帧变化速度，bone length，以及一些对gradient的约束，这些都可以理解为通过先验知识给模型添加的正则项）。加了正则项的损失就叫做结构风险：

$$g = \arg\min_{h \in H} \hat{\varepsilon}(h) = \arg\min_{h \in H} \frac{1}{m}\sum_{i=1}^{m} l(h(x^{(i)}), y^{(i)}) + \lambda\Omega(h) \cdots\cdots (V)$$

iii. 两者的对应

在一定条件下，非概率模型和概率模型有以下的对应关系

非概率模型		概率模型
假设空间 H	<---------->	参数分布 P(y|x)
经验风险最小化	<---------->	极大似然估计
结构风险最小化	<---------->	极大后验概率估计
正则化项	<---------->	分布参数的先验概率

c. 参数模型 vs 非参数模型

首先，这里的参数不是指模型函数里的未知数，比如wx+b里面的w,b，而是指假设的数据分布中的参数，比如GMM中的均值方差

i. 参数模型

使用参数模型要求对要学习的问题有足够的认识，可以去假设映射函数h(x)或者分布p()的具体形式，知道属于哪一个函数族，然后用前面介绍的方法把参数估计出来就可以得到完整的模型了。

对于先验知识足够充足的情况，只需要少量数据就可以得到一个很好的模型。相应地，如果先验知识有不足，假设的模型就不能完全反应真实分布，这样无论数据量多少都无法得到好的模型。

常见的参数模型：logistic regression，linear regression

ii. 非参数模型

我们对于数据的先验知识很少，无法给出具体的形式。这时就会希望函数能够带有有数据的信息，并且数据越多越好，当数据无穷多的时候，理论上是可以逼近任意复杂的模型的。

最简单的非参数模型：KNN

常见的非参数模型： 决策树，朴素贝叶斯，神经网络，支持向量机

iii. 关于神经网络

神经网络一般会被称为半参数模型，它含有少量的超参数(hidden unit个数，深度等)，和大量的普通参数，相当于假设空间的复杂度非常高，依赖于大量的数据来进行训练，因此也就解释了为什么有大量训练数据时神经网络的效果好，因为数据越多，越能逼近模型

2. 概率模型的推断(inference of probabilistic model)

前面介绍了概率模型以及如何估计模型参数的方法，这里开始什么是概率模型的推断，如何用概率模型做推断。

问题：如何利用联合分布p(x, z)去分析数据x？如何通过对隐变量z的分解来描述数据x的分布？又如何生成x？如何从可观察到的事物来推断不可观察的事物？

举个例子： “推断----男女身高的分布可以看成一个有两个高斯混合的概率模型，性别就是隐变量z，现在从人群抽取一个人，ta的身高为一米八，那我们推断这个人很有可能是男性，那

么隐变量z=男性就解释了为什么这个人身高有一米八；“生成”----反过来，我们如果知道了一个人的性别，也可以大概推断出这个人的身高分布范围，给他生成一个身高数据

a. 关于生成

一个生成模型包括：隐含变量，观测变量，变量之间的关系

隐含变量与观测变量的关系：

i. 生成过程

前面都是在说从观测数据计算分布，那么现在反过来思考，如果知道了一个分布，怎么可以得到观测数据呢？这就是生成过程，生成模型之所以叫生成模型，也是因为如果知道了联合分布p(x, y)，我们就可以通过生成过程生成出一个数据点。

我们认为观测值x是从隐含变量组成的层次结构中生成出来的。

以GMM为例，如果现在知道了一个确定的GMM模型，那么怎么从中得到一个数据点呢?

第一步以α的概率选择一个分布k(选择性别男女)，第二步从k对应的高斯分布中采样一个数据点(根据不同性别的身高分布给ta生成一个身高)。

这就是一个具体的生成过程，这个过程也说明了观测值x和隐变量y之间的关系，回顾GMM一节中的概率图之间的依赖关系加深理解，会发现确实是上面描述的过程。

ii. 用式子表示生成过程

生成过程可以表示为 p(x | y) * p(y)

我们将这个式子变一下，应用贝叶斯公式，就得到了下式

p(x | y) * p(y) = p(y | x) * ∑p(x ,y)dy

上式中，p(y)和p(x, y)都是通过先验知识预设的，所以要完成推断和生成任务(哪个隐变量解释了x；如何生成x)，现在还未知的就是后验分布p(y|x)了

b. 后验分布

用概率模型去推断隐变量的值或者隐变量的后验分布

$$p(\mathbf{z} \mid \mathbf{x}) = \frac{p(\mathbf{x}, \mathbf{z})}{\int p(\mathbf{x}, \mathbf{z}) \mathrm{d}\mathbf{z}}.$$

联系representation learning来解释后验的话，就是说这是数据hidden representation的一个概率描述。求解后验分布需要知道三个条件：证据p(x)，证据产生的可能性p(x|z)(或者表示为p(x,z)/p(x))以及先验p(z)

后验分布通常求解困难，虽然分子部分可以通过先验知识给出假设，看作是模型的一部分，能够写出表达式易于求解，但是分母部分有积分，特别是在高维的情况下几乎不可能直接计算，因此只能用一个容易计算的方式去近似，所以通常求解后验分布通常指近似后验分布。

估计后验分布有两种方法，一是基于采样的MCMC，二是基于假设构造近似模型的变分推断

c. 马尔科夫链蒙特卡洛Markov Chain Monte Carlo(MCMC)

先来解释一下这个方法的名字，首先"Monte Carlo"是一种统计方法，基于大数定理，用采样样本来计算某个函数的期望值，写成式子就是$E_{\mathrm{z}\sim p(z)}[\mathrm{f}(\mathrm{x})] \approx$

$\sum_{i=1}^{m}(f(x_i))$，"Markov Chain"是一个随机过程，用于描述获取蒙特卡洛所需样本的采样方法。

由于马尔科夫链的特殊定义，它可以从非常困难的未经标准化的概率分布中获得样本，对标准化因子并不敏感。想想看，上面说后验分布p(z|x)的推断困难在于，用于标准化(normalization)的分母难以计算，那这个情形下是不是可以使用MCMC来采样后验分布呢?

此处阵亡。。好复杂。。也用不上。。有时间再补充。。

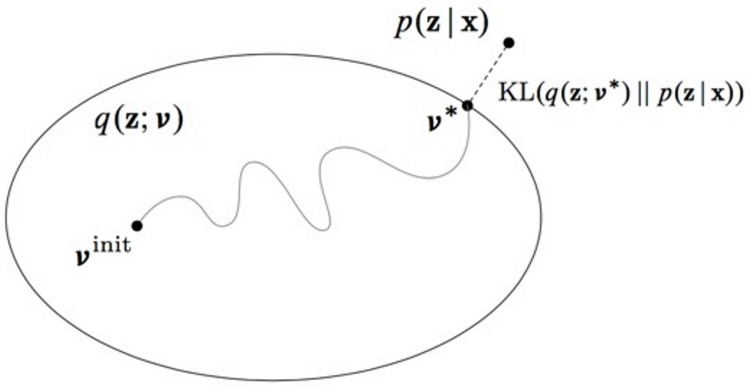
d. 变分推断VI

既然p(z|x)很难求解，那么能不能不直接求解，而是用另一个函数去靠近它呢?

这就是变分推断的思想，如果想要推断分布p，但是p比较复杂不容易表达且难以直接求解时，那我们就构造一个简单的q去近似它，并且量度p和q的距离，当p和q的差距很小时，q就可以当做p的近似分布，q叫做变分分布，因此这种方法就叫做变分推断，这样就把推断问题转化为优化问题。

过程:

我们需要构造q(z; v)，并且不断更新参数v，使得q(z;v)更接近p(z|x)。我们在构造q的时候，通常会直观地选择p可能的概率分布，这样能够更好地保证q和p的相似程度。图示如下:



- VI turns **inference into optimization.**

e. 变分推断的求解

KL散度是一个计算两个分布间距离的量度，因此在变分推断中我们使用KL来计算真实分布p和近似分布q之间的距离，变分推断的优化目标就是最小化KL(q||p)。

$$\begin{aligned}\lambda^* &= \arg\min_{\lambda} \mathrm{KL}(q(\mathbf{z}\,;\,\lambda)\,\|\,p(\mathbf{z}\mid\mathbf{x})) \\ &= \arg\min_{\lambda}\ \mathbb{E}_{q(\mathbf{z}\,;\,\lambda)}\big[\log q(\mathbf{z}\,;\,\lambda) - \log p(\mathbf{z}\mid\mathbf{x})\big].\end{aligned}$$

然而，因为上式中包含p(z|x)，这个是我们不知道的，所以无法求解，需要变形一下。

f. ELBO

目标是要最小化KL散度

KL(q(z; λ) || p(z|x)) = $E_{q(z;\,\lambda)}$[log q(z; λ) - log p(z|x)]

为了摆脱p(z|x), 从贝叶斯公式有

$$p(\mathbf{z}\mid\mathbf{x}) = \frac{p(\mathbf{x},\mathbf{z})}{\int p(\mathbf{x},\mathbf{z})\mathrm{d}\mathbf{z}}.$$

且∫p(x, z)dz = p(x)，代入KL式并展开log可以得到

KL(q(z; λ) || p(z|x)) = $E_{q(z;\lambda)}$[log q(z; λ) - log p(z|x)]

= $E_{q(z;\lambda)}$[log q(z; λ) - log p(x, z) + log p(x)] 因为p(x)与期望的q(z; λ)无关，所以可以取出来

= $E_{q(z;\lambda)}$[log q(z; λ) - log p(x, z)] + log p(x) （1）

最终有

$$\log p(\mathbf{x}) = \mathrm{KL}(q(\mathbf{z}\,;\ \lambda) \parallel p(\mathbf{z} \mid \mathbf{x})) + \ \mathbb{E}_{q(\mathbf{z}\,;\,\lambda)}\big[\log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}\,;\ \lambda)\big]$$

回顾本文最开始说的，观测到的数据就是我们的证据evidence，所以这时候上式左边观测数据x的log边缘分布logp(x)就叫做model的evidence。

又因为KL散度≥0，所以有

log p(x) - $E_{q(z;\lambda)}$[log p(x, z) - log q(z; λ)] = KL(q(z; λ) || p(z | x)) ≥ 0

log p(x) ≥ $E_{q(z;\lambda)}$[log p(x, z) - log q(z; λ)]

因此，$E_{q(z;\lambda)}$[log p(x, z) - log q(z; λ)]是证据log p(x)的一个下界，我们就将它叫做证据下界 Evidence Lower BOund，即ELBO。

$$\mathrm{ELBO}(\lambda) = \ \mathbb{E}_{q(\mathbf{z}\,;\,\lambda)}\big[\log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}\,;\ \lambda)\big].$$

且从上面的(1)式可以看出，因为证据logp(x)是确定的，它只与观测数据有关(虽然我们求不出来..)， 所以可以看成常量，因此要最小化KL散度，就相当于最大化ELBO

KL(q(z; λ) || p(z|x)) = log p(x) -$E_{q(z;\lambda)}$[log p(x, z) - log q(z; λ)]

$\min_\lambda$ KL(q(z; λ) || p(z|x)) → $\max_\lambda$ $E_{q(z;\lambda)}$[log p(x, z) - log q(z; λ)]

稍微看下这个ELBO，第一项是logp(x, z)基于q(z; λ)的期望。有没有似曾相识的感觉，回想在EM推导中那个Q函数是不是就是"似然函数logp(x,z; θ)基于分布Q(z)的期望"，是不是就和当时推出来Q函数是后验分布对上了？要是把它写成log除法的形式，是不是就和当时求和符号中的式子几乎一模一样了？

因此，让我们再看一眼EM中的式子：

E步：

$$Q_i(z_i) = \frac{p(x_i, z_i; \theta)}{\sum_z p(x_i, z_i; \theta)} = \frac{p(x_i, z_i; \theta)}{p(x_i; \theta)} = p(z_i|x_i; \theta)$$

$$\sum_{i=1}^{n} \sum_{z_i} Q_i(z_i) log \frac{p(x_i, z_i; \theta)}{Q_i(z_i)}$$

M步：

$$argmax \sum_{i=1}^{n} \sum_{z_i} Q_i(z_i) log \frac{p(x_i, z_i; \theta)}{Q_i(z_i)}$$

如果忽略参数θ和λ，求和符号里的项就正是ELBO。EM算法正是利用了ELBO，在E步假设模型的参数不变，直接q(z)=p(z|x), 计算似然的期望，在M步中再对ELBO做相对于模型θ的优化。不同的是，EM算法中假设p(z|x)是一个在给定模型参数θ后容易计算的形式，z也是比较简单的形式(如GMM中的k)，所以可以直接在E步求得并且代入在M步优

化。

自己推一下EM算法推导中用了jensen不等式的式子，将左边减右边，会发现算出来就是KL(q(z) || p(z|x))。因此，EM算法可以看作一个简化版的变分。(现在回过头想一下，这么说来是不是就可以理解为什么从KL散度的角度也可以推导出一样的解了？)

对于复杂的情况，比如z是高维的，p(z|x)是难以求解的情况，会有其他算法来求解。

有了ELBO之后，变分推断后验的目标函数就变成了

$$\lambda^* = \arg\max_{\lambda} \text{ELBO}(\lambda).$$

$$\text{ELBO}(\lambda) = \mathbb{E}_{q(\mathbf{z}\,;\,\lambda)}[\log p(\mathbf{x}, \mathbf{z})] - \mathbb{E}_{q(\mathbf{z}\,;\,\lambda)}[\log q(\mathbf{z}\,;\,\lambda)],$$

分析一下组成ELBO的两项，联合分布似然的期望加上q的熵(和前面的负号一起就是熵，所以就是这两项相加)。第一项，如果从能量的角度来看，这个能量希望q可以集中在联合概率p(x,z)大的地方；第二项，因为是加上熵，所以熵越大越好，这一项相当于防止q集中在一个点上。因此，最大化ELBO这个目标实际上是似然与熵的平衡(由此可以联想到GAN以及它的mode collapse，为了生成概率大的数据，GAN可能只会判别概率很大的那几个，从而将生成分布集中到一个很窄的范围，就造成了mode collapse，优质的GAN应该既生成概率大的数据，又能够覆盖尽量多的真实分布，也是两者的平衡，所以对抗方法比起变分方法就是差一个熵的平衡？)

g. 优化ELBO

最大化ELBO是一个优化问题，我们最熟悉的方法就是梯度下降，所以关键要求ELBO的梯度，下面介绍两种求ELBO梯度的方法

i. Score function gradient

在介绍这个方法之前，我们首先要了解一下什么是score function。

来看我们非常熟悉的log likelihood，记为 L(θ) = log p(x; θ)，它的一阶导数就叫做score function

$$S(\theta) = \frac{dL(\theta)}{d\theta}$$

score function的重要性质：期望为0，由简单的计算即可得

$$\begin{aligned}
&E_{p(x;\theta)}\left[\nabla_\theta \log p(x;\theta)\right] \\
&= \int p(x;\theta)\nabla_\theta \log p(x;\theta) \\
&= \int p(x;\theta)\frac{\nabla_\theta p(x;\theta)}{p(x;\theta)} \\
&= \int \nabla_\theta p(x;\theta) \\
&= \nabla_\theta \int p(x;\theta) \\
&= \nabla_\theta 1 \\
&= 0
\end{aligned}$$

现在我们利用这个性质来化简ELBO的梯度

$$
\begin{aligned}
&\nabla_\lambda L(\lambda) = \nabla_\lambda E_{q_\lambda(z)}[\log p(x,z) - \log q(z)] \\
&= \int \nabla_\lambda q_\lambda(z)[\log p(x,z) - \log q(z)] + q_\lambda(z)\nabla_\lambda[\log p(x,z) - \log q(z)] \\
&= \int q_\lambda(z)\nabla_\lambda \log q_\lambda(z)[\log p(x,z) - \log q(z)] - q_\lambda(z)\nabla_\lambda \log q_\lambda(z) \\
&= \int q_\lambda(z)\nabla_\lambda \log q_\lambda(z)[\log p(x,z) - \log q(z)] \\
&= E_{q_\lambda(z)}\nabla_\lambda \log q_\lambda(z)[\log p(x,z) - \log q(z)]
\end{aligned}
$$

第二行到第三行用到了log的求导公式

第三行到第四行用到了score function期望为0的性质

然后就可以使用蒙特卡洛采样来估计ELBO的梯度了，得到的是无偏估计

$$
\nabla_\lambda \text{ ELBO}(\lambda) \approx \frac{1}{S}\sum_{s=1}^{S}\left[\left(\log p(\mathbf{x}, \mathbf{z}_s) - \log q(\mathbf{z}_s\ ;\ \lambda)\right)\nabla_\lambda \log q(\mathbf{z}_s\ ;\ \lambda)\right].
$$

Algorithm 1 Black Box Variational Inference

Input: data x, joint distribution p, mean field variational family q.

Initialize $\lambda_{1:n}$ randomly, $t = 1$.

repeat

  // **Draw** S **samples from** q

  for $s = 1$ **to** S **do**

    $z[s] \sim q$

  end for

  $\rho = t$th value of a Robbins Monro sequence (Eq. 2)

  $\lambda = \lambda + \rho\frac{1}{S}\sum_{s=1}^{S}\nabla_\lambda \log q(z[s]|\lambda)(\log p(x, z[s]) - \log q(z[s]|\lambda))$

  $t = t + 1$

until change of λ is less than 0.01.

ii. Reparameterization gradient

这就是VAE中用到的重参数技巧。

我们并不直接从q(z; λ)中采样，而是转换为从标准正态分布中采样，通过线性变换得到z~q(z; λ)

(在VAE中，q(z; λ)是要求的，如果从中采样相当于梯度的传递需要经过采样这一步，这样就没法传递梯度了。那为什么会存在上一种方法，估计因为不在神经网络中应用就不需要传递梯度呀，所以上一种方法也是有用的)

所以，首先从标准正态分布中采样e~N(0,I)，再通过变换得到z = e*μ + σ

$$
\begin{aligned}
\epsilon &\sim q(\epsilon) \\
\mathbf{z} &= \mathbf{z}(\epsilon\ ;\ \lambda),
\end{aligned}
$$

这时候，ELBO的梯度就可以写成

$$\nabla_\lambda \, \mathrm{ELBO}(\lambda) = \, \mathbb{E}_{q(\epsilon)} \big[\nabla_\lambda \big(\log p(\mathbf{x}, \mathbf{z}(\epsilon \, ; \, \lambda)) - \log q(\mathbf{z}(\epsilon \, ; \, \lambda) \, ; \, \lambda) \big) \big].$$

注意，期望的下标已经换成了q(e)，所以梯度可以直接放进去

然后依旧用蒙特卡洛计算来对梯度做估计，跟上面的方法过程一样，只是改成了sample e

h. 总结：变分推断的求解，用q来近似p(z|x)，通过一些变换化简去掉p(z|x)，至于如何变换取决于要解决的问题，比如这里已知或者可求的是联合分布p(x, z)，就用联合分布去换，比如下面的VAE，用网络表示的是p(x|z)，那就用p(x|z)去换。求出ELBO之后，优化ELBO得到最优的近似。