变分深度嵌入(Variational Deep Embedding，VaDE)

作者：凯鲁嘎吉 - 博客园 http://www.cnblogs.com/kailugaji/

这篇博文主要是对论文“Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering”的整理总结，阅读这篇博文的前提条件是：了解高斯混合模型用于聚类的算法，了解变分推断与变分自编码器。在知道高斯混合模型(GMM)与变分自编码器(VAE)之后，VaDE实际上是将这两者结合起来的一个产物。与VAE相比，VaDE在公式推导中多了一个变量c。与GMM相比，变量c就相当于是GMM中的隐变量z，而隐层得到的特征z相当于原来GMM中的数据x。下面主要介绍VaDE模型的变分下界(损失函数)L(x)的数学推导过程。推导过程用到了概率论与数理统计的相关知识。

1. 前提公式

前提公式

$$E(x)=\int xp(x)dx=c, x\sim p(x)$$

$$E(f(x,\xi))=\int f(x,\xi)p(x)dx=g(\xi), x\sim p(x)$$

$$E(f(x))=\int f(x)\cdot p(x)dx=c, x\sim p(x)$$

$$\int p(z\mid x)dz=\sum p(z\mid x)=1$$

推导见其后

若$q(z)=N(z;\tilde{\mu},\tilde{\sigma}^2 I), p(z)=N(z;\mu,\sigma^2 I)$, z是J维，

则$$\int q(z)\log p(z)dz=\sum_{j=1}^{J}\left(-\frac{1}{2}\log(2\pi\sigma_j^2)-\frac{\tilde{\sigma}_j^2}{2\sigma_j^2}-\frac{(\tilde{\mu}_j-\mu_j)^2}{2\sigma_j^2}\right)$$

$$=-\frac{J}{2}\log(2\pi)-\frac{1}{2}\sum_{j=1}^{J}\left(\log\sigma_j^2+\frac{\tilde{\sigma}_j^2}{\sigma_j^2}+\frac{(\tilde{\mu}_j-\mu_j)^2}{\sigma_j^2}\right)$$

计算过程中用到了正态分布的一阶矩与二阶矩计算公式。

$$\int q(\boldsymbol{z})\log p(\boldsymbol{z})d\boldsymbol{z} = \int N(\boldsymbol{z};\tilde{\boldsymbol{\mu}},\tilde{\boldsymbol{\sigma}}^2\boldsymbol{I})\log N(\boldsymbol{z};\boldsymbol{\mu},\boldsymbol{\sigma}^2\boldsymbol{I})d\boldsymbol{z}$$

$$\overset{(1)}{=}\int\prod_{j=1}^{J}\frac{1}{\sqrt{2\pi\tilde{\sigma}_j^2}}\exp\left(-\frac{(z_j-\tilde{\mu}_j)^2}{2\tilde{\sigma}_j^2}\right)\log\left[\prod_{j=1}^{J}\frac{1}{\sqrt{2\pi\sigma_j^2}}\exp\left(-\frac{(z_j-\mu_j)^2}{2\sigma_j^2}\right)\right]d\boldsymbol{z}$$

$$\overset{(2)}{=}\sum_{j=1}^{J}\int\frac{1}{\sqrt{2\pi\tilde{\sigma}_j^2}}\exp\left(-\frac{(z_j-\tilde{\mu}_j)^2}{2\tilde{\sigma}_j^2}\right)\log\left[\frac{1}{\sqrt{2\pi\sigma_j^2}}\exp\left(-\frac{(z_j-\mu_j)^2}{2\sigma_j^2}\right)\right]dz_j$$

$$\overset{(3)}{=}\sum_{j=1}^{J}\int\frac{1}{\sqrt{2\pi\tilde{\sigma}_j^2}}\exp\left(-\frac{(z_j-\tilde{\mu}_j)^2}{2\tilde{\sigma}_j^2}\right)\left[-\frac{1}{2}\log(2\pi\sigma_j^2)\right]dz_j-\int\frac{1}{\sqrt{2\pi\tilde{\sigma}_j^2}}\exp\left(-\frac{(z_j-\tilde{\mu}_j)^2}{2\tilde{\sigma}_j^2}\right)\frac{(z_j-\mu_j)^2}{2\sigma_j^2}dz_j$$

$$\overset{(4)}{=}\sum_{j=1}^{J}-\frac{1}{2}\log(2\pi\sigma_j^2)-\int\frac{1}{\sqrt{2\pi\tilde{\sigma}_j^2}}\exp\left(-\frac{(z_j-\tilde{\mu}_j)^2}{2\tilde{\sigma}_j^2}\right)\frac{(z_j-\tilde{\mu}_j+\tilde{\mu}_j-\mu_j)^2}{2\tilde{\sigma}_j^2}\frac{\tilde{\sigma}_j^2}{\sigma_j^2}dz_j$$

$$\overset{(5)}{=}\sum_{j=1}^{J}-\frac{1}{2}\log(2\pi\sigma_j^2)-\int\frac{1}{\sqrt{2\pi\tilde{\sigma}_j^2}}\exp\left(-\frac{(z_j-\tilde{\mu}_j)^2}{2\tilde{\sigma}_j^2}\right)\frac{(z_j-\tilde{\mu}_j)^2+2(z_j-\tilde{\mu}_j)(\tilde{\mu}_j-\mu_j)+(\tilde{\mu}_j-\mu_j)^2}{2\tilde{\sigma}_j^2}\frac{\tilde{\sigma}_j^2}{\sigma_j^2}dz_j$$

$$\overset{(6)}{=}\sum_{j=1}^{J}-\frac{1}{2}\log(2\pi\sigma_j^2)-\frac{\tilde{\sigma}_j^2}{\sigma_j^2}\int\frac{1}{\sqrt{2\pi\tilde{\sigma}_j^2}}\exp\left(-\frac{(z_j-\tilde{\mu}_j)^2}{2\tilde{\sigma}_j^2}\right)\frac{(z_j-\tilde{\mu}_j)^2}{2\tilde{\sigma}_j^2}dz_j-\int\frac{1}{\sqrt{2\pi\tilde{\sigma}_j^2}}\exp\left(-\frac{(z_j-\tilde{\mu}_j)^2}{2\tilde{\sigma}_j^2}\right)\frac{(\tilde{\mu}_j-\mu_j)^2}{2\tilde{\sigma}_j^2}dz_j$$

$$\overset{(7)}{=}\sum_{j=1}^{J}-\frac{1}{2}\log(2\pi\sigma_j^2)-\frac{\tilde{\sigma}_j^2}{2\sigma_j^2}-\frac{(\tilde{\mu}_j-\mu_j)^2}{2\tilde{\sigma}_j^2}\overset{(8)}{=}-\frac{J}{2}\log(2\pi)-\frac{1}{2}\sum_{j=1}^{J}\left(\log\sigma_j^2+\frac{\tilde{\sigma}_j^2}{\sigma_j^2}+\frac{(\tilde{\mu}_j-\mu_j)^2}{\tilde{\sigma}_j^2}\right)$$

(1) 将变量z按维度拆开

(2) Z的不同维度之间相互独立，且单个维度正态分布的积分为1

(3) 将后面log拆成两项

(4) 第一项除A是系数外，积分里面是正态分布的积分为1，第二项平方项里面加一项减一项

(5) 平方项里面拆开，(A+B)^2=A^2+2AB+B^2

(6) 交叉项中属于正态分布的1阶矩，奇数阶矩为0，E(x-μ)=0

(7) 正态分布的2阶矩，E(x-μ)^2=σ^2

➤正态分布的n阶矩

$$\text{若} X \sim N(\mu,\sigma^2), \text{令} f(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$$\because E(X-\mu)^n = \int (x-\mu)^n \cdot f(x;\mu,\sigma^2)dx$$

$$= \begin{cases} 0, & n \text{ is odd}, \\ \sigma^n (n-1)!!, & n \text{ is even}. \end{cases}$$

$$\therefore E(X-\mu) = 0,\ E(X-\mu)^2 = \int (x-\mu)^2 \cdot f(x;\mu,\sigma^2)dx = \sigma^2$$

2. VaDE损失函数公式推导过程

变分自编码器聚类(VaDE)

- 变分自编码器的损失函数

$$\max L(\boldsymbol{x}) = E_{q(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}\mid\boldsymbol{z})p(\boldsymbol{z})}{q(\boldsymbol{z}\mid\boldsymbol{x})}\right] = E_{q(\boldsymbol{z}|\boldsymbol{x})}\left[\log p(\boldsymbol{x}\mid\boldsymbol{z})\right] - KL(q(\boldsymbol{z}\mid\boldsymbol{x})\,\|\,p(\boldsymbol{z}))$$

- Variational Deep Embedding (VaDE)的损失函数

  由z已知条件下x与c独立，得p(x,z,c)= p(x|z,c)p(z|c)p(c) =p(x|z)p(z|c)p(c)

  由平均场理论，得q(z,c|x)=q(z|x)q(c|x)则

推导见
其后

$$\max L(\boldsymbol{x}) = E_{q(\boldsymbol{z},c|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}\mid\boldsymbol{z})p(\boldsymbol{z},c)}{q(\boldsymbol{z},c\mid\boldsymbol{x})}\right] = E_{q(\boldsymbol{z},c|\boldsymbol{x})}\left[\log p(\boldsymbol{x}\mid\boldsymbol{z})\right] - KL(q(\boldsymbol{z},c\mid\boldsymbol{x})\,\|\,p(\boldsymbol{z},c))$$

$$= \frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{D} x_i \log \mu_{Gi}^{(l)} + (1-x_i)\log(1-\mu_{Gi}^{(l)}) - \frac{1}{2}\sum_{c=1}^{K}\gamma_c \sum_{j=1}^{J}\left(\log \sigma_{cj}^2 + \frac{\sigma_{Ij}^2}{\sigma_{cj}^2} + \frac{(\mu_{Ij}-\mu_{cj})^2}{\sigma_{cj}^2}\right)$$

$$+\sum_{c=1}^{K}\gamma_c \log\frac{\pi_c}{\gamma_c} + \frac{1}{2}\sum_{j=1}^{J}\left(\log \sigma_{Ij}^2 + 1\right)$$

变分自编码器聚类(VaDE)

- Variational Deep Embedding (VaDE)的损失函数

$$\max L(\boldsymbol{x}) = E_{q(\boldsymbol{z},c|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z},c)}{q(\boldsymbol{z},c|\boldsymbol{x})}\right] = E_{q(\boldsymbol{z},c|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})] - KL(q(\boldsymbol{z},c|\boldsymbol{x}) \| p(\boldsymbol{z},c))$$

- 第一项 $E_{q(\boldsymbol{z},c|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})]$：重构损失

$$\boxed{\text{若} p(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_G, \boldsymbol{I}), E_{q(\boldsymbol{z},c|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})] \Rightarrow -\|\boldsymbol{x} - \boldsymbol{\mu}_G\|^2}$$

$$\boxed{\text{若} p(\boldsymbol{x}|\boldsymbol{z}) = \boldsymbol{\mu}_G^x (1-\boldsymbol{\mu}_G)^{1-x}, E_{q(\boldsymbol{z},c|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})] \Rightarrow \boldsymbol{x}\log(\boldsymbol{\mu}_G) + (1-\boldsymbol{x})\log(1-\boldsymbol{\mu}_G)}$$

- 第二项 $-KL(q(\boldsymbol{z},c|\boldsymbol{x}) \| p(\boldsymbol{z},c))$：高斯混合先验=>变分后验的KL散度

先验：$p(z,c) = p(z|c)p(c)$, $p(c) = Cat(c|\pi), p(z|c) = N(z|\mu_c, \sigma_c^2 I)$

后验：$q(z,c|x) = q(z|x)q(c|x)$

其中：$q(z|x) = N(z|\mu_I, \sigma_I^2 I)$, 令 $\gamma_c = q(c|x) = p(c|z) = \dfrac{p(c)p(z|c)}{\sum_{m=1}^{K} p(m)p(z|m)}$

最终的聚类结果是由q(c|x)得到的，q(c|x)相当于GMM中的隐变量的后验概率γ。

$q(c|x)=p(c|z)$ 详细推导

$$\max L(x) = E_{q(z,c|x)}\left[\log \frac{p(x|z)p(z,c)}{q(z,c|x)}\right] = \int_z \sum_c q(z,c|x) \log \frac{p(x|z)p(c|z)p(z)}{q(z,c|x)} dz$$

$$= \int_z \sum_c q(c|x)q(z|x) \log \frac{p(x|z)p(c|z)p(z)}{q(z|x)q(c|x)} dz = \int_z q(z|x) \sum_c q(c|x)\left[\log \frac{p(x|z)p(z)}{q(z|x)} + \log \frac{p(c|z)}{q(c|x)}\right] dz$$

$$= \int_z q(z|x) \log \frac{p(x|z)p(z)}{q(z|x)} dz - \int_z q(z|x) \sum_c q(c|x) \log \frac{q(c|x)}{p(c|z)} dz$$

$$= \int_z q(z|x) \log \frac{p(x|z)p(z)}{q(z|x)} dz - \int_z q(z|x) KL(q(c|x) \| p(c|z)) dz$$

- 由于L(x)本身与变量c无关，因此最大化下界L(x)需要满足KL(q(c|x)||p(c|z))=0，即q(c|x)=p(c|z)。
- 而p(c)，p(z|c)已知，与高斯混合模型一样，p(c|z)可以通过贝叶斯公式求得，即

$$\gamma_c = q(c|x) = p(c|z) = \frac{p(c)p(z|c)}{\sum_{m=1}^{K} p(m)p(z|m)}, \sum_{c=1}^{K} \gamma_c = \sum_{c=1}^{K} q(c|x) = 1.$$

下面将损失函数拆成5项，并一项一项进行求解。

VaDE损失函数推导

- Variational Deep Embedding (VaDE)的损失函数

$$\max L(x)=E_{q(z,c|x)}\left[\log\frac{p(x|z)p(z,c)}{q(z,c|x)}\right]=E_{q(z,c|x)}\left[\log p(x|z)\right]-KL(q(z,c|x)\|p(z,c))$$

$$=\boxed{E_{q(z,c|x)}\left[\log p(x|z)\right]}+E_{q(z,c|x)}\left[\log p(z|c)\right]+E_{q(z,c|x)}\left[\log p(c)\right]$$

$$-E_{q(z,c|x)}\left[\log q(z|x)\right]-E_{q(z,c|x)}\left[\log q(c|x)\right]$$

VaDE损失函数拆成5项，一项一项看，先看第1项，假设p(x|z)是伯努利分布。
K是聚类数，D是数据x的维度，L是采样次数，一般取1。

$$E_{q(z,c|x)}\left[\log p(x|z)\right]=\int_z\sum_{c=1}^{K}q(c|x)q(z|x)\log p(x|z)dz=\sum_{c=1}^{K}q(c|x)\cdot\int_z q(z|x)\log p(x|z)dz$$

$$=E_{q(z|x)}\left[\log p(x|z)\right]=\frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{D}x_i\log\mu_{Gi}^{(l)}+(1-x_i)\log(1-\mu_{Gi}^{(l)})$$

VaDE损失函数推导

- Variational Deep Embedding (VaDE)的损失函数

$$\max L(x) = E_{q(z,c|x)}\left[\log \frac{p(x|z)p(z,c)}{q(z,c|x)}\right] = E_{q(z,c|x)}[\log p(x|z)] - KL(q(z,c|x) \| p(z,c))$$

$$= E_{q(z,c|x)}[\log p(x|z)] + \boxed{E_{q(z,c|x)}[\log p(z|c)]} + E_{q(z,c|x)}[\log p(c)]$$
$$- E_{q(z,c|x)}[\log q(z|x)] - E_{q(z,c|x)}[\log q(c|x)]$$

VaDE损失函数拆成5项，一项一项看，看第2项。
K是聚类数，J是特征z的维度，令γc=q(c|x)。

$$E_{q(z,c|x)}[\log p(z|c)] = \int_z \sum_{c=1}^{K} q(c|x)q(z|x)\log p(z|c)dz = \sum_{c=1}^{K} q(c|x)\int_z q(z|x)\log p(z|c)dz$$

$$= \sum_{c=1}^{K} \gamma_c \int_z N(z|\mu_I, \sigma_I^2 I)\log N(z|\mu_c, \sigma_c^2 I)dz = -\sum_{c=1}^{K} \gamma_c \left[\frac{J}{2}\log(2\pi) + \frac{1}{2}\sum_{j=1}^{J}\left(\log \sigma_{cj}^2 + \frac{\sigma_{Ij}^2}{\sigma_{cj}^2} + \frac{(\mu_{Ij} - \mu_{cj})^2}{\sigma_{cj}^2}\right)\right]$$

VaDE损失函数推导

- Variational Deep Embedding (VaDE)的损失函数

$$\max L(x) = E_{q(z,c|x)}\left[\log \frac{p(x|z)p(z,c)}{q(z,c|x)}\right] = E_{q(z,c|x)}[\log p(x|z)] - KL(q(z,c|x) \| p(z,c))$$

$$= E_{q(z,c|x)}[\log p(x|z)] + E_{q(z,c|x)}[\log p(z|c)] + \boxed{E_{q(z,c|x)}[\log p(c)]}$$
$$- E_{q(z,c|x)}[\log q(z|x)] - E_{q(z,c|x)}[\log q(c|x)]$$

VaDE损失函数拆成5项，一项一项看，看第3项。

K是聚类数，令γc=q(c|x)。

$$E_{q(z,c|x)}[\log p(c)] = \int_z \sum_{c=1}^{K} q(c|x)q(z|x)\log p(c)dz = \sum_{c=1}^{K} q(c|x)\log p(c) \cdot \int_z q(z|x)dz$$

$$= \sum_{c=1}^{K} q(c|x)\log p(c) = \sum_{c=1}^{K} \gamma_c \log \pi_c$$

VaDE损失函数推导

- Variational Deep Embedding (VaDE)的损失函数

$$\max L(x)=E_{q(z,c|x)}\left[\log\frac{p(x|z)p(z,c)}{q(z,c|x)}\right]=E_{q(z,c|x)}\left[\log p(x|z)\right]-KL(q(z,c|x)\,\|\,p(z,c))$$

$$=E_{q(z,c|x)}\left[\log p(x|z)\right]+E_{q(z,c|x)}\left[\log p(z|c)\right]+E_{q(z,c|x)}\left[\log p(c)\right]$$

$$-\boxed{E_{q(z,c|x)}\left[\log q(z|x)\right]}-E_{q(z,c|x)}\left[\log q(c|x)\right]$$

VaDE损失函数拆成5项，一项一项看，看第4项。
K是聚类数，J是特征z的维度。

$$E_{q(z,c|x)}\left[\log q(z|x)\right]=\int_z\sum_{c=1}^{K}q(c|x)q(z|x)\log q(z|x)dz=\sum_{c=1}^{K}q(c|x)\cdot\int_z q(z|x)\log q(z|x)dz$$

$$=\int_z N(z|\mu_I,\sigma_I^2 I)\log N(z|\mu_I,\sigma_I^2 I)dz=-\frac{J}{2}\log(2\pi)-\frac{1}{2}\sum_{j=1}^{J}\left(\log\sigma_{Ij}^2+1\right)$$

$\mathcal{V}a\mathcal{D}\mathcal{E}$损失函数推导

- Variational Deep Embedding (VaDE)的损失函数

$$\max L(x) = E_{q(z,c|x)}\left[\log\frac{p(x|z)p(z,c)}{q(z,c|x)}\right] = E_{q(z,c|x)}[\log p(x|z)] - KL(q(z,c|x)\| p(z,c))$$

$$= E_{q(z,c|x)}[\log p(x|z)] + E_{q(z,c|x)}[\log p(z|c)] + E_{q(z,c|x)}[\log p(c)]$$

$$- E_{q(z,c|x)}[\log q(z|x)] - \boxed{E_{q(z,c|x)}[\log q(c|x)]}$$

VaDE损失函数拆成5项，一项一项看，看第5项。
K是聚类数，J是特征z的维度。

$$E_{q(z,c|x)}[\log q(c|x)] = \int_z \sum_{c=1}^{K} q(c|x)q(z|x)\log q(c|x)dz = \sum_{c=1}^{K} q(c|x)\log q(c|x)\cdot\int_z q(z|x)dz$$

$$= \sum_{c=1}^{K} q(c|x)\log q(c|x) = \sum_{c=1}^{K} \gamma_c \log\gamma_c$$

VaDE损失函数推导

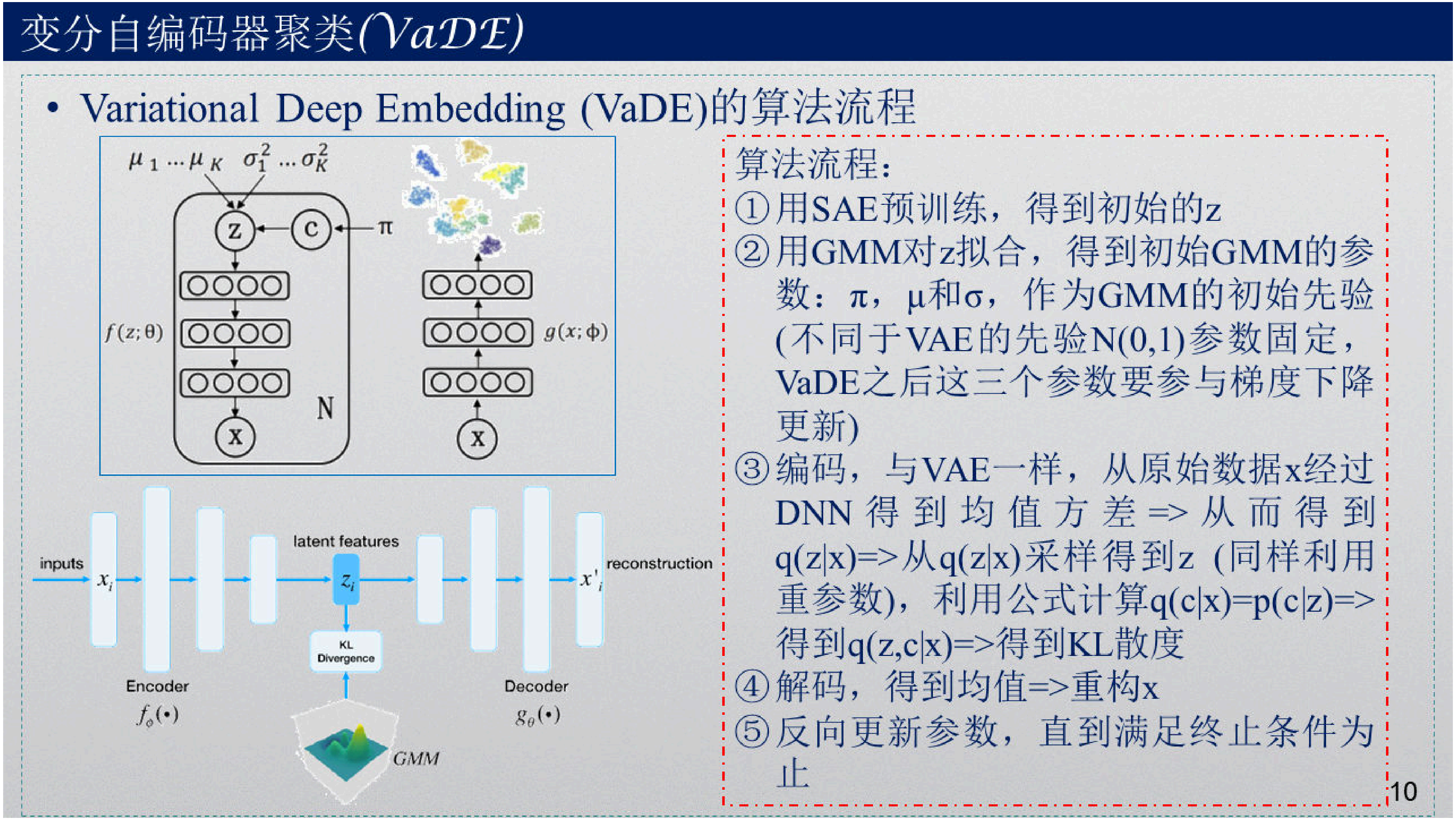
- Variational Deep Embedding (VaDE)的损失函数

$$
\max L(x) = E_{q(z,c|x)}\left[\log \frac{p(x|z)p(z,c)}{q(z,c|x)}\right] = E_{q(z,c|x)}\left[\log p(x|z)\right] - KL(q(z,c|x)\,\|\,p(z,c))
$$

$$
\begin{aligned}
&= E_{q(z,c|x)}\left[\log p(x|z)\right] + E_{q(z,c|x)}\left[\log p(z|c)\right] + E_{q(z,c|x)}\left[\log p(c)\right] \\
&\quad - E_{q(z,c|x)}\left[\log q(z|x)\right] - E_{q(z,c|x)}\left[\log q(c|x)\right] \\
&= \frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{D} x_i \log \mu_{Gi}^{(l)} + (1-x_i)\log(1-\mu_{Gi}^{(l)}) - \sum_{c=1}^{K}\gamma_c\left[\frac{J}{2}\log(2\pi) + \frac{1}{2}\sum_{j=1}^{J}\left(\log \sigma_{cj}^2 + \frac{\sigma_{Ij}^2}{\sigma_{cj}^2} + \frac{(\mu_{Ij}-\mu_{cj})^2}{\sigma_{cj}^2}\right)\right] \\
&\quad + \sum_{c=1}^{K}\gamma_c \log \pi_c + \frac{J}{2}\log(2\pi) + \frac{1}{2}\sum_{j=1}^{J}\left(\log \sigma_{Ij}^2 + 1\right) - \sum_{c=1}^{K}\gamma_c \log \gamma_c \\
&= \frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{D} x_i \log \mu_{Gi}^{(l)} + (1-x_i)\log(1-\mu_{Gi}^{(l)}) - \frac{1}{2}\sum_{c=1}^{K}\gamma_c\sum_{j=1}^{J}\left(\log \sigma_{cj}^2 + \frac{\sigma_{Ij}^2}{\sigma_{cj}^2} + \frac{(\mu_{Ij}-\mu_{cj})^2}{\sigma_{cj}^2}\right) \\
&\quad + \sum_{c=1}^{K}\gamma_c \log \frac{\pi_c}{\gamma_c} + \frac{1}{2}\sum_{j=1}^{J}\left(\log \sigma_{Ij}^2 + 1\right)
\end{aligned}
$$

9

3. VaDE算法总体流程



4. 疑问

1）GMM算法的参数pi并没有进行归一化处理，在更新过程中能保证pi的和始终为1吗？这个问题在作者评论里面有回答，说pi相比于参数miu, sigma来说，对结果影响不大，但又有人问了，如果遇到非平衡数据呢？这种情况下pi的影响还是比较大的。在代码里也不难实现，加一行代码，类似于pi/sum(pi)就行。

Amplitudes of the GMM #4

Open erlebach opened this issue on 23 Sep 2017 · 3 comments

erlebach commented on 23 Sep 2017

Hi,

How do you ensure that the amplitudes of the GMM (pi[i]) remain positive and sum to unity, when minimizing the cost function with respect to these amplitudes? Thanks.

```
Gordon
```

👍 3

slim1017 commented on 25 Sep 2017 Owner

Hi,

1. pi remain positive:
   Empirically，we found that pi always remain positive during training (note that there is a log pi in loss function, maybe this is the reason), so we don't need to worry about the constraint pi >= 0;

2. sum to unity:
   we could normalize the pi vector to make its sum equal one after each batch or epoch or entire training. The three ways achieve similar accuracy.

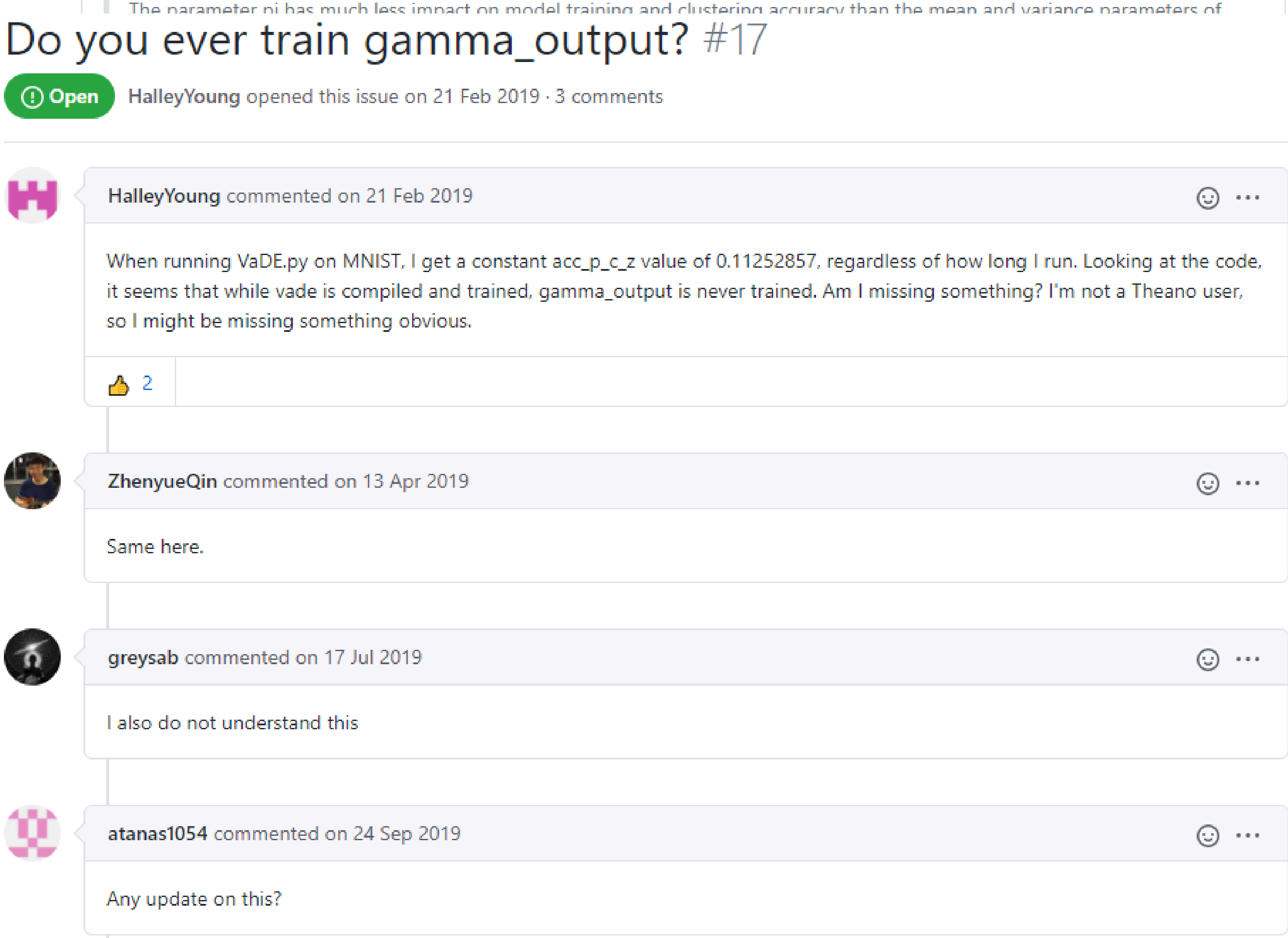
The parameter pi has much less impact on model training and clustering accuracy than the mean and variance parameters of GMM.

daib13 commented on 25 Sep 2017

Do you think using a softmax layer to produce pi is a better way or a worse way? Because sofmax layer ensures that pi is always positive and sum to 1. But it may cause problems like gradient vanishing or other problems in SGD.

colobas commented on 17 May 2019 • edited

2）后验概率γ在代码里并不参与更新，为什么不和GMM的其他参数(pi, miu, sigma)一样进行梯度下降更新呢？而是直接套公式？有什么数学依据吗？这个在[作者评论里面]有人提到过，但是未被回复。其实直接ELBO目标对γ求偏导令其为0，用这种求极值的方式求解γ也没有问题，最后记得对γ进行归一化。



Do you ever train gamma_output? #17

Open HalleyYoung opened this issue on 21 Feb 2019 · 3 comments

HalleyYoung commented on 21 Feb 2019

When running VaDE.py on MNIST, I get a constant acc_p_c_z value of 0.11252857, regardless of how long I run. Looking at the code, it seems that while vade is compiled and trained, gamma_output is never trained. Am I missing something? I'm not a Theano user, so I might be missing something obvious.

👍 2

ZhenyueQin commented on 13 Apr 2019

Same here.

greysab commented on 17 Jul 2019

I also do not understand this

atanas1054 commented on 24 Sep 2019

Any update on this?

3）预训练到底是怎么做到的，仅仅是用SAE训练得到的结果吗？原作者代码里面只给出了预训练之后得到的具体参数，并没有给出预训练的代码。预训练这个问题在[作者评论里面]有被提到。预训练阶段还是非常关键的一步，当然，有人是这样做的：预训练使用VAE模型。

Pretraining #5

Open erlebach opened this issue on 29 Sep 2017 · 12 comments

erlebach commented on 29 Sep 2017

Hi,
My team and I are trying to duplicate the results of your paper, but cannot. Would it be possible to gain access to the code that pretrains the data? That would help us a lot. Thank you.

👍 5

michelleowen commented on 10 Nov 2017

Hi I am also interested in your pre-training code. I did pre-training based on your description in your paper. However, with pre-training, gamma output will always assign the same class to all data points.

👍 4

michelleowen commented on 10 Nov 2017

Also, why you assign weights from one previous layer in pretrained AE to the layers in VaDE as below:
vade.layers[1].set_weights(ae.layers[0].get_weights())
vade.layers[2].set_weights(ae.layers[1].get_weights())
vade.layers[3].set_weights(ae.layers[2].get_weights())
vade.layers[4].set_weights(ae.layers[3].get_weights())
why not
vade.layers[1].set_weights(ae.layers[1].get_weights())
vade.layers[2].set_weights(ae.layers[2].get_weights())
vade.layers[3].set_weights(ae.layers[3].get_weights())
vade.layers[4].set_weights(ae.layers[4].get_weights())
if pre-trained ae has the same network architecture with VaDE?

👍 1

eelxpeng commented on 28 Jan 2018

4) 代码中生成z的方式与VAE一模一样，而原文中的图示以及3.1节部分中的描述是z从${\mu }_c$与${\sigma }_c$而来，而公式(14)又与VAE一致了，原文中的表述并不统一，且代码与原文部分表述不符。

论文中说的是$\pi $、${\mu }_c$与${\sigma }_c$用GMM来初始化，代码里分别初始化为取平均、零矩阵、单位矩阵。不过问题也不大，自己改一下代码就OK。

如果能解决我的疑问，欢迎在评论区回复，一起探讨~

5. 参考文献

[1] 聚类——GMM - 凯鲁嘎吉 - 博客园

[2] 变分推断与变分自编码器 - 凯鲁嘎吉 - 博客园

[3] Jiang Z , Zheng Y , Tan H , et al. Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering. 2016.

[4] VaDE代码：

GitHub - slim1017/VaDE: Python code for paper - Variational Deep Embedding : A Generative Approach to Clustering

GitHub - GuHongyang/VaDE-pytorch: the reproduce of Variational Deep Embedding : A Generative Approach to Clustering Requirements by pytorch

eelxpeng commented on 28 Jan 2018

Also having trouble replicating the results. Using the pretrained weights provided works fine, except for HAR dataset. But using pretraining code from DEC-keras, which achieves good results for AE+kmeans and DEC, does not make the VaDE model work. Also,

specification and repeat many times, the performance is significantly lower than the result reported. Is the author using the pretrain

DEC code? If not, could you provide it?

👍 3

ttgump commented on 1 Feb 2018

@michelleowen
I think they are using a Sequential model based on their json file. So the architecture of ae is like:

```
ae = Sequential()
ae.add(Dense(intermediate_dim[0], input_dim=original_dim, activation='relu'))
ae.add(Dense(intermediate_dim[1], activation='relu'))
ae.add(Dense(intermediate_dim[2], activation='relu'))
ae.add(Dense(latent_dim))
ae.add(Dense(intermediate_dim[2], activation='relu'))
ae.add(Dense(intermediate_dim[1], activation='relu'))
ae.add(Dense(intermediate_dim[0], activation='relu'))
ae.add(Dense(original_dim))
```

But even I tried to pretrain this autoencoder first, I get same problem that gamma output will always assign the same class to all data points. So I guess authors used other technic to pretrain "ae".

wangmn93 commented on 7 Apr 2018

They use VAE or AAE to pretrain the model. You need to constraint the latent space with KL divergence in the loss (Or use discriminator in AAE).
I have tried VAE for pretraining. The accuracy after 200 epoch is 86% on MNIST. The range of latent space is -5 ~ 5. While the range of latent space of the provided pretrained weights is -3 ~ 3. If you can further shrink the range of latent space, i think the result will be the same as theirs.

eelxpeng commented on 21 May 2018 • edited

@wangmn93 Could you elaborate more on the VAE pretraining? How to control the range of latent space? By setting coefficient on the KL divergence term? Also, it seems that their provided pretrain weights only have autoencoder weight, but not enc_sigma weight. It would even better if you could share your code for the pretraining. Thanks.

wangmn93 commented on 22 May 2018

Actually， i found that sometimes you can get high accuracy(around 94%)when
you just use autoencoder for pretrain training instead of vae, which means
you do not need to resitict the range of latent space. But the whole algo
is sensitive to initialization (both ae and kmean). In short, you can not
gurantee to get 94% on average. If you want to reproduce 94% acc， use their
pretrained weight. Or pretrain with ae and then use kmean in the latent
space to test the acc if it is more than 80% or higher, you may get 94% on
VaDE.

eelxpeng <notifications@github.com> 于 2018年5月21日周一 08:01写道:

...

eelxpeng commented on 22 May 2018 • edited

@wangmn93 Thank you for your reply. I actually tried many possible initializations, including ae, sdae, vae, with all kinds of random initialization. However, I haven't got one work. Could you share one code that at least sometimes works? I am trying to find out the reason of the instability, and good initialization method to make things work robustly. Your help would be much appreciated.

wangmn93 commented on 23 May 2018

i use the pretrain of dec https://github.com/XifengGuo/DEC-keras

eelxpeng <notifications@github.com> 于 2018年5月22日周二 09:06写道:

...

devyhia commented on 28 Nov 2018

@eelxpeng Did you make any progress on this problem? Did you get the DEC-Keras pre-training method to work?

I could get the AE pre-training on DEC-Keras to reach ~86% . However, once I plug that in to VaDE, accuracy drops dramatically to

I could get the AE pre-training on DEC-keras to reach ~88% ... However, once I plug that in to VaDE, accuracy drops dramatically to ~57%. Not really sure what is going on wrong there.

Zizi6947 commented on 17 Dec 2018

@wangmn93 Did you train some other datasets？ I could get 85%+ on MNIST using VaDE. But when i train the new dataset, the acc is only about 20% .

👍 1

wangmn93 commented on 20 Dec 2018

no， i only test on MNIST

Zizi6947 <notifications@github.com> 于 2018年12月17日周一 下午12:35写道：

...