Video Change Detection Project Update

Suya Wu

December 21, 2018

Week 12

An idea: Mixture Variational Autoencoders

Since the original VAEs cannot be directly applied to discrete variables with standard back-propagation through the the reparametrisation trick, previous works either estimate the gradient of discrete variables or adjusting the architecture of the standard VAEs. Inspired by the framework concrete VAEs Jang et al. [2016], which enables backpropagation through discrete samples, we can train classes with original VAEs by choosing either gaussian mixture model (GMM) or bernoulli mixture model (BMM) as the prior distribution for continuous latent variables or discrete latent variables respectively.

The generative model

We consider the generative model $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}, \mathbf{c}) = p_{\boldsymbol{\theta}}(\mathbf{c})p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{c})p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}, \mathbf{c})$, where an observe sample $\mathbf{x}$ is generated from a set of latent variables $\mathbf{z}$ and $\mathbf{z}$ follows the mixture distributions $w.r.t.$ the classes variables $\mathbf{c}$. They are distributed as:

$$\mathbf{x} \sim Bernoulli(\boldsymbol{\mu}_{\boldsymbol{x}}) \; or \; \mathbf{x} \sim N(\boldsymbol{\mu}_{\boldsymbol{x}}, \boldsymbol{\sigma}_{\boldsymbol{x}}\mathbf{I}) \tag{1}$$

$$\mathbf{z} \sim Bernoulli(\boldsymbol{\mu}_{\boldsymbol{z}}) \; or \; \mathbf{z} \sim N(\boldsymbol{\mu}_{\boldsymbol{z}}, \boldsymbol{\sigma}_{\boldsymbol{z}}\mathbf{I}) \tag{2}$$

$$\mathbf{c} \sim Categorical(\boldsymbol{\pi}) \tag{3}$$

The parametric space $\boldsymbol{\theta}$ therefore relavant to $\{\boldsymbol{\mu}_{\boldsymbol{x}}, \boldsymbol{\mu}_{\boldsymbol{z}}, \boldsymbol{\sigma}_{\boldsymbol{x}}, \boldsymbol{\sigma}_{\boldsymbol{z}}, \boldsymbol{\pi}\}$. We denote $q_{\boldsymbol{\Phi}}(\mathbf{z}, \mathbf{c}|\mathbf{x})$ as the variational approximation to the intractable posterior $p_{\boldsymbol{\theta}}(\mathbf{z}, \mathbf{c}|\mathbf{x})$. Then generated variables are distributed as:

$$\mathbf{c} \sim Categorical(\hat{\boldsymbol{\pi}}) \tag{4}$$

$$\mathbf{z}|\mathbf{c} \sim Bernoulli(\boldsymbol{\mu}_{\boldsymbol{z}|\boldsymbol{c}}) \; or \; \mathbf{z}|\mathbf{c} \sim N(\boldsymbol{\mu}_{\boldsymbol{z}|\boldsymbol{c}}, \boldsymbol{\sigma}_{\boldsymbol{z}|\boldsymbol{c}}\mathbf{I}) \tag{5}$$

$$\mathbf{x}|\mathbf{z}, \mathbf{c} \sim Bernoulli(\boldsymbol{\mu}_{\boldsymbol{x}|\boldsymbol{z}}) \; or \; \mathbf{x}|\mathbf{z}, \mathbf{c} \sim N(\boldsymbol{\mu}_{\boldsymbol{x}|\boldsymbol{z}}, \boldsymbol{\sigma}_{\boldsymbol{x}|\boldsymbol{z}}\mathbf{I}) \tag{6}$$

The observed sample $\mathbf{x}$ can be generated from a neural network model parametrised by $\boldsymbol{\Phi}$, which is relevant to $\{\boldsymbol{\mu}_{\boldsymbol{x}|\boldsymbol{z}}, \boldsymbol{\mu}_{\boldsymbol{z}|\boldsymbol{c}}, \boldsymbol{\sigma}_{\boldsymbol{x}|\boldsymbol{z}}, \boldsymbol{\sigma}_{\boldsymbol{z}|\boldsymbol{c}}, \hat{\boldsymbol{\pi}}\}$. By the generative process, where $\mathbf{x}$ and $\mathbf{c}$ can be treated independently with conditioning on $\mathbf{z}$, we can approximate

$$q_{\boldsymbol{\Phi}}(\mathbf{x}|\mathbf{z}, \mathbf{c}) = q_{\boldsymbol{\Phi}}(\mathbf{x}|\mathbf{z})$$

The variational lower bound

The loglikelihood of the whole observed data $\mathbf{X}$, $\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(N)}) = \sum_{i=1}^{N} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})$. Each likelihood element can be written as:

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) = KL(q_{\Phi}(\mathbf{z}, \mathbf{c}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z}, \mathbf{c}|\mathbf{x}^{(i)})) + \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\Phi}; \mathbf{x}^{(i)}) \tag{7}$$

The first RHS term is the KL-divergence of the approximate from the true posterior. Following the same argument in Kingma & Welling [2013] that the second RHS term is the lower bound we need to optimize since this KL-divergence is non-negative.

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\Phi}; \mathbf{x}^{(i)}) = E_{q_\Phi(\mathbf{z},\mathbf{c}|\mathbf{x}^{(i)})}\left[\log p_\theta(\mathbf{x}^{(i)}, \mathbf{z}, \mathbf{c}) - \log q_\Phi(\mathbf{z}, \mathbf{c}|\mathbf{x}^{(i)})\right] \tag{8}$$

which can also be written as

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\Phi}; \mathbf{x}^{(i)}) &= E_{q_{\boldsymbol{\Phi}}(\mathbf{z},\mathbf{c}|\mathbf{x}^{(i)})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z}) + \log p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{c}) + \log p_{\boldsymbol{\theta}}(\mathbf{c}) - \log q_{\boldsymbol{\Phi}}(\mathbf{z}|\mathbf{x}^{(i)}) - \log q_{\boldsymbol{\Phi}}(\mathbf{c}|\mathbf{z})] \\
&= E_{q_{\boldsymbol{\Phi}}(\mathbf{z},\mathbf{c}|\mathbf{x}^{(i)})}\left[\log p(\mathbf{x}^{(i)}|\mathbf{z})\right] + E_{q_{\boldsymbol{\Phi}}(\mathbf{z},\mathbf{c}|\mathbf{x}^{(i)})}\left[\log \frac{p(\mathbf{z}|\mathbf{c})}{q(\mathbf{z}|\mathbf{x}^{(i)})}\right] + E_{q_{\boldsymbol{\Phi}}(\mathbf{z},\mathbf{c}|\mathbf{x}^{(i)})}\left[\log \frac{p(\mathbf{c})}{q(\mathbf{c}|\mathbf{z})}\right] \\
&= \int_z \sum_c \log p(\mathbf{x}^{(i)}|\mathbf{z}) q(\mathbf{z}|\mathbf{x}^{(i)}) q(\mathbf{c}|\mathbf{z}) dz + \int_z \sum_c q(\mathbf{z}|\mathbf{x}^{(i)}) q(\mathbf{c}|\mathbf{z}) \log \frac{p(\mathbf{z}|\mathbf{c})}{q(\mathbf{z}|\mathbf{x}^{(i)})} dz \\
&\qquad + \int_z \sum_c q(\mathbf{z}|\mathbf{x}^{(i)}) q(\mathbf{c}|\mathbf{z}) \log \frac{p(\mathbf{c})}{q(\mathbf{c}|\mathbf{z})} dz \\
&\approx \frac{1}{L} \sum_{l=1}^{L} \sum_c \log p(\mathbf{x}^{(i)}|\mathbf{z}^{(l)}) q(\mathbf{c}|\mathbf{z}^{(l)}) + \sum_c KL\left(q(\mathbf{z}|\mathbf{x}^{(i)})||p(\mathbf{z}|\mathbf{c})\right) q(\mathbf{c}|\mathbf{z}) \\
&\qquad + \frac{1}{L} \sum_{l=1}^{L} KL\left(q(\mathbf{c}|\mathbf{z}^{(l)})||p(\mathbf{c})\right)
\end{aligned}$$

The last approximation may be confused, since we use only one sample (L=1) to estimate the integrate $w.r.t\ q_{\boldsymbol{\Phi}}(\mathbf{z}|\mathbf{x}^{(i)})$ following the Stochastic Gradient Variational Bayes estimator (SGVB) Kingma & Welling [2013]. Based one the one sample of $\mathbf{z}$, we also sample one $\mathbf{c}$.

1. The first term of loss is only depending on $p(\mathbf{x}^{(i)}|\mathbf{z}^{(1)})$, which can be learned by the decoder procedure, which is 2 layers linear neutral network.

2. The second term evaluate the distance between $q(\mathbf{z}|\mathbf{x}^{(i)})$ and $p(\mathbf{z}|\mathbf{c})$, where the first one is learned by the encoder procedure (2 layers linear neutral network) and the second one is assumed to be the true distribution of $\mathbf{z}$ depending on different classes $\mathbf{c}$. We updated $p(\mathbf{z}|\mathbf{c})$ by

$$\boldsymbol{\mu}_{\mathbf{z}|c=k}^{update} = \frac{\sum_{i=1}^{n} q(c=k|\mathbf{z}_i^{(l)}) \mathbf{z}_i^{(l)}}{\sum_{i=1}^{n} q(c=k|\mathbf{z}_i^{(l)})}$$

$$\boldsymbol{\Sigma}_{\mathbf{z}|c=k}^{update} = \frac{\sum_{i=1}^{n} q(c=k|\mathbf{z}_i^{(l)})(\mathbf{z}_i^{(l)} - \boldsymbol{\mu}_{\mathbf{z}|c=k})(\mathbf{z}_i^{(l)} - \boldsymbol{\mu}_{\mathbf{z}|c=k})^T}{\sum_{i=1}^{n} q(c=k|\mathbf{z}_i^{(l)})}$$

3. $q(\mathbf{c}|\mathbf{z}^{(1)})$ can be learned by 2 layers linear neutral network with gumbel-softmax distribution to estimate the discrete distribution of $\mathbf{c}$.

Then applied the Gumbel-Softmax estimator and SGVB to be capable of differentiating and optimizing the lower bound w.r.t. the variational parameters $\boldsymbol{\Phi}$ and generative parameters $\boldsymbol{\theta}$. However, the third loss term is always converging to a constant and does not contribute to the whole loss function. It cannot distinguish different classes but only learn the first term (the reconstruction term).