Papers Reading Notes

Yuanxun Zhang

Contents

1 Probabilistic Latent Semantic Analysis (Hofmann, 1999)

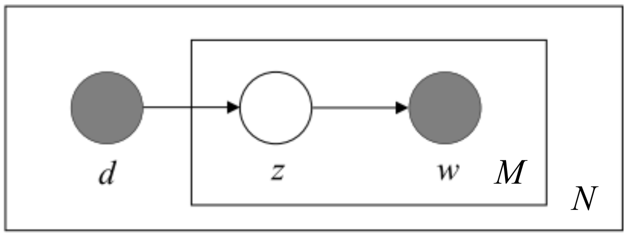

Figure 1: Graphical model representation of PLSA

As shown in Figure 1, the generative process in PLSA is as follows:

a) select a document d_i with probability $P(d_i)$

b) pick a latent class z_k with probability $P(z_k|d_i))$

c) generate a word w_j with probability $P(w_j|z_k))$

Then, the joint probability of PLSA model results in the expression,

$$P(d_i, w_j) = P(d_i)P(w_j|d_i), \qquad P(w_j|d_i) = \sum_{k=1}^{K} P(w_j|z_k)P(z_k|d_i) \tag{1}$$

The modeling goal is to identify conditional probability mass functions $P(w_j|z_k)$. Formally, we can use a maximum likelihood formulation of the learning problem,

$$\mathcal{L} = \sum_{i=1}^{N}\sum_{j=1}^{M} n(d_i, w_j) \log P(d_i, w_j) \tag{2}$$

Then, pluging Eq. 9 into Eq. 3, we got

$$\begin{aligned}\mathcal{L} &= \sum_{i=1}^{N}\sum_{j=1}^{M} n(d_i, w_j) \log\Big[P(d_i) \sum_{k=1}^{K} P(w_j|z_k)P(z_k|d_i)\Big] \qquad (3)\\ &= \sum_{i=1}^{N} n(d_i)\log P(d_i) + \sum_{i=1}^{N}\sum_{j=1}^{M} n(d_i, w_j)\log \Big[\sum_{k=1}^{K} P(w_j|z_k)P(z_k|d_i)\Big]\end{aligned}$$

Where $n(d_i)$ denotes length of doc d_i, and $n(d_i, w_j)$ denotes the number of times the term w_j occurred in document d_i.

1.1 Inference with the EM Algorithm

Basically, to derive EM algorithm, we need to: a) define $Q(\theta, \theta^{(i)})$ function; b) In **E-step**, compute $Q(\theta, \theta^{(i)})$ function based on current parameter $\theta^{(i)}$; c) In **M-step**, re-estimate parameters $\theta^{(i+1)}$ which maximizes $Q(\theta, \theta^{(i)})$,

$$\theta^{(i+1)} = \arg\max_{\theta}(\theta, \theta^{(i)}) \tag{4}$$

1.1.1 Define Q function

The Q function is defined as the expectation of the complete-data log likelihood function $\log P(Y, Z|\theta)$ with respect of the posterior distribution of unobserved latent variables $P(Z|Y, \theta^{(i)})$, which is,

$$Q(\theta, \theta^{(i)}) = E_Z[\log P(Y, Z|\theta)|Y, \theta^{(i)}] \tag{5}$$

Hence, in PLSA model, the complete-data log likelihood function will be Eq. 3. Because, in Eq. 3 the first term $\sum_{i=1}^{N} n(d_i)\log P(d_i)$ does not depends on latent variables z, we can ignore it. Then, the Q function can be defined as,

$$Q(\theta, \theta^{(i)}) = \sum_{i=1}^{N}\sum_{j=1}^{M} n(d_i, w_j) \sum_{k=1}^{K} P(z_k|d_i, w_j)\log \left[P(w_j|z_k)P(z_k|d_i)\right] \tag{6}$$

1.1.2 E-Step

To compute $Q(\theta, \theta^{(i)})$, we just need to compute $P(z_k|d_i, w_j)$, which can be computed using Bayes rule,

$$\begin{aligned} P(z_k|d_i, w_j) &= \frac{P(w_j|z_k)P(z_k|d_i)}{P(d_i, w_j)} &(7)\\ &= \frac{P(w_j|z_k)P(z_k|d_i)}{\sum_k P(w_j|z_k)P(z_k|d_i)} &(8)\end{aligned}$$

1.1.3 M-Step

In M-Step, we're going to find parameter $\theta^{(i+1)}$ that can maximize function Q. Because,

$$\sum_{j=1}^{M} P(w_j|z_k) = 1, \qquad \sum_{k=1}^{K} P(z_k|d_i) = 1 \tag{9}$$

So, the function $\mathcal{H}$ with Lagrange multipliers τ_k and ρ_i is,

$$\mathcal{H} = Q(\theta, \theta^{(i)}) + \sum_{k=1}^{K} \tau_k(1 - \sum_{j=1}^{M} P(w_j|z_k)) + \sum_{i=1}^{N}\rho_i(1 - \sum_{k=1}^{K} P(z_k|d_i)) \tag{10}$$

Then, first compute partial derivative of the function $\mathcal{H}$ with respect to the $P(w_j|z_k)$ and solve it when derivative is equal to zero.

$$\sum_{i=1}^{N} n(d_i, w_j)P(z_k|d_i, w_j)\frac{1}{P(w_j|z_k)} - \tau_k = 0 \tag{11}$$

or,

$$\sum_{i=1}^{N} n(d_i, w_j)P(z_k|d_i, w_j) - \tau_k P(w_j|z_k) = 0 \tag{12}$$

the τ_k can be solved when combining $1 \leqslant j \leqslant M$,

$$\tau_k = \sum_{i=1}^{N}\sum_{j=1}^{M} n(d_i, w_j) P(z_k|d_i, w_j) \tag{13}$$

So, the $P(w_j|z_k)$ is

$$P(w_j|z_k) = \frac{\sum_{i=1}^{N} n(d_i, w_j) P(z_k|d_i, w_j)}{\sum_{i=1}^{N}\sum_{j=1}^{M} n(d_i, w_j) P(z_k|d_i, w_j)} \tag{14}$$

Second, compute partial derivative of the function $\mathcal{H}$ with respect to the $P(z_k|d_i)$ and solve it when derivative is equal to zero.

$$\sum_{j=1}^{M} n(d_i, w_j) P(z_k|d_i, w_j) - \rho_i P(z_k|d_i) = 0 \tag{15}$$

And the ρ_i can be solved when combining $1 \leqslant k \leqslant K$, and $\sum_{k=1}^{K} P(z_k|d_i, w_j) = 1$,

$$\rho_i = \sum_{j=1}^{M} n(d_i, w_j) \tag{16}$$

So, the $P(z_k|d_i)$ is

$$P(z_k|d_i) = \frac{\sum_{j=1}^{M} n(d_i, w_j) P(z_k|d_i, w_j)}{\sum_{j=1}^{M} n(d_i, w_j)} \tag{17}$$

2 Finding scientific topics (Griffiths and Steyvers, 2004)

2.1 Derive the Eq.1 $P(\mathbf{w}|\mathbf{z})$

In paper, author use vector representation for $\mathbf{z}$ in Eq.1. For simplicity, I just use single topic assignment z_i instead of vector, and ϕ is multinomial distributions over the W words for topic assignment z_i,

$$\begin{aligned} P(\mathbf{w}|z_i) &= \int P(\mathbf{w}, \phi|z_i)\, d\phi \\ &= \int P(\mathbf{w}|\mathbf{z}, \phi)P(\phi)\, d\phi \\ &= \int \frac{\Gamma(\sum_{i=1}^{W}\beta)}{\prod_{i=1}^{W}\Gamma(\beta)} \prod_{i=1}^{W} \phi_i^{\beta-1} \times \prod_{i=1}^{W} \phi_i^{n_{z_i}^{(w)}}\, d\phi \\ &= \int \frac{\Gamma(\sum_{i=1}^{W}\beta)}{\prod_{i=1}^{W}\Gamma(\beta)} \prod_{i=1}^{W} \phi_i^{n_{z_i}^{(w)}+\beta-1}\, d\phi \\ &= \frac{\Gamma(\sum_{i=1}^{W}\beta)}{\prod_{i=1}^{W}\Gamma(\beta)} \int \prod_{i=1}^{W} \phi_i^{n_{z_i}^{(w)}+\beta-1}\, d\phi \\ &= \frac{\Gamma(W\beta)}{\Gamma(\beta)^W} \int \prod_{i=1}^{W} \phi_i^{n_{z_i}^{(w)}+\beta-1}\, d\phi \end{aligned} \tag{18}$$

in which $n_{z_i}^{(w)}$ is the number of times word w has been assigned to topic z_i. Because,

$$\begin{aligned} \int \prod_{i=1}^{W} \phi_i^{n_{z_i}^{(w)}+\beta-1}\, d\phi &= \frac{\prod_{i=1}^{W}\Gamma(n_{z_i}^{(w)}+\beta)}{\Gamma(\sum_{i=1}^{W} n_{z_i}^{(w)} + W\beta)} \\ &= \frac{\prod_{i=1}^{W}\Gamma(n_{z_i}^{(w)}+\beta)}{\Gamma(n_{z_i}^{(\cdot)} + W\beta)} \end{aligned} \tag{19}$$

Then, the Equation 18 can be written as,

$$P(\mathbf{w}|z_i) = \frac{\Gamma(W\beta)}{\Gamma(\beta)^W} \frac{\prod_{i=1}^{W}\Gamma(n_{z_i}^{(w)}+\beta)}{\Gamma(n_{z_i}^{(\cdot)} + W\beta)} \tag{20}$$

When considering the whole T topic assignment $\mathbf{z}$, we get the same equation as shown in paper Eq.1.

$$\begin{aligned} P(\mathbf{w}|\mathbf{z}) &= \prod_{j=1}^{T} p(\mathbf{w}|z_j) \\ &= \Big(\frac{\Gamma(W\beta)}{\Gamma(\beta)^W}\Big)^T \prod_{j=1}^{T} \frac{\prod_{i=1}^{W}\Gamma(n_j^{(w)}+\beta)}{\Gamma(n_j^{(\cdot)} + W\beta)} \end{aligned} \tag{21}$$

In order to avoid numerical overflow,

2.2 Derive the Eq.5 $P(z_i = j|\mathbf{z}_{-i}, \mathbf{w})$

Because,

$$P(\mathbf{z}|\mathbf{w}) = \frac{P(\mathbf{w}, \mathbf{z})}{\sum_{\mathbf{z}} P(\mathbf{w}, \mathbf{z})} \tag{22}$$

Then,

$$\begin{aligned} P(z_i = j|\mathbf{z}_{-i}, \mathbf{w}) &= \frac{P(\mathbf{w}, \mathbf{z})}{P(\mathbf{w}, \mathbf{z}_{-i})} \\ &= \frac{P(\mathbf{w}|\mathbf{z})P(\mathbf{z})}{P(\mathbf{w}|\mathbf{z}_{-i})P(\mathbf{z}_{-i})} \end{aligned} \tag{23}$$

So, we can put Eq.2 and Eq.3 of the original paper into Equation 23, and use Gamma function property $\Gamma(x+1) = x\Gamma(x)$ by cancellation of terms then,

$$\begin{aligned} P(z_i = j|\mathbf{z}_{-i}, \mathbf{w}) &= \frac{P(\mathbf{w}|\mathbf{z})P(\mathbf{z})}{P(\mathbf{w}|\mathbf{z}_{-i})P(\mathbf{z}_{-i})} \\ &= \frac{\left[\left(\frac{\Gamma(W\beta)}{\Gamma(\beta)^W}\right)^T \prod_{j=1}^T \frac{\prod_{i=1}^W \Gamma(n_j^{(w)}+\beta)}{\Gamma(n_j^{(\cdot)}+W\beta)}\right] \times \left[\left(\frac{\Gamma(T\alpha)}{\Gamma(\alpha)^T}\right)^D \prod_{d=1}^D \frac{\prod_{j=1}^T \Gamma(n_j^{(d)}+\alpha)}{\Gamma(n_{\cdot}^{(d)}+T\alpha)}\right]}{\left[\left(\frac{\Gamma(W\beta)}{\Gamma(\beta)^W}\right)^T \prod_{j=1}^T \frac{\prod_{i=1}^W \Gamma(n_{-i,j}^{(w)}+\beta)}{\Gamma(n_{-i,j}^{(\cdot)}+W\beta)}\right] \times \left[\left(\frac{\Gamma(T\alpha)}{\Gamma(\alpha)^T}\right)^D \prod_{d=1}^D \frac{\prod_{j=1}^T \Gamma(n_{-i,j}^{(d)}+\alpha)}{\Gamma(n_{-i,\cdot}^{(d)}+T\alpha)}\right]} \\ &= \frac{n_{-i,j}^{(wi)} + \beta}{n_{-i,j}^{(\cdot)} + W\beta} \frac{n_{-i,j}^{(di)} + \alpha}{n_{-i,\cdot}^{(di)} + T\alpha} \end{aligned} \tag{24}$$

2.3 Model selection for computing $P(\mathbf{w}|T)$

In paper, author approximate $P(\mathbf{w}|T)$ by taking the harmonic mean of a set of values of $P(\mathbf{w}|\mathbf{z}^{(i)}, T)$ when $\mathbf{z}^{(i)}$ is sampled from the posterior $P(\mathbf{z}|\mathbf{w}, T)$, which means,

$$P(\mathbf{w}|T) \approx \left\{\frac{1}{m}\sum_{i=1}^{m} P(\mathbf{w}|\mathbf{z}^{(i)}, T)^{-1}\right\}^{-1} \tag{25}$$

Raftery *et al.* in papers (Newton and Raftery, 1994; Kass and Raftery, 1995) explain this idea by using the concept of importance sampling for model section.

In this example, we have several models $\{T_i : i = 10, 20, ..., 1000\}$, then Bayesian inference needs to compute the posterior probabilities given data $\mathbf{w}$,

$$P(T_i|\mathbf{w}) = \frac{P(\mathbf{w}|T_i)P(T_i)}{\sum_i^T P(\mathbf{w}|T_i)p(T_i)} \tag{26}$$

And the likelihood function $P(\mathbf{w}|T_i)$ is crucial component that needs to integrate out all topic assignment $\mathbf{z}$ then,

$$P(\mathbf{w}|T_i) = \int P(\mathbf{w}|\mathbf{z}, T_i)P(\mathbf{z}|T_i)\, d\mathbf{z} \tag{27}$$

So, the problem becomes how to approximate $P(\mathbf{w}|T_i)$.

Recall the basic Monte Carlo integration is to approximate $p(x) = \int p(x|\theta)p(\theta)d\theta$, when $p(\theta)$ is hard to integrate and the simple Monte Carlo approximation method is

$$\hat{I} = \frac{1}{m}\sum_{i=1}^{m} p(x|\theta^{(i)})) \tag{28}$$

However, the weakness of this simple method is that the estimation is dominated by a few large values of the small likelihood.

Another method (called importance sampling) is to generate samples $\{\theta^{(i)} : i = 1, ..., m\}$ from a proposal density function $q(\theta)$, and compute importance weight $w_i = \frac{p(\theta)}{q(\theta)}$. Then, the approximation is written as,

$$\hat{I} = \frac{\sum_{i=1}^{m} w_i p(x|\theta^{(i)})}{\sum_{i=1}^{m} w_i} \tag{29}$$

which is also known as importance sampling without normalization constants. Raftery *et al.* mentioned in papers (Newton and Raftery, 1994) that $q(\theta)$ can be approximately drawn from the their posterior density,

$$q(\theta) \approx p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)} \tag{30}$$

Subsistution into Equation 29 yields, an an estimate for $p(x)$,

$$p(x) \approx \hat{p}(x) = \left\{\frac{1}{m}\sum_{i=1}^{m} p(x|\theta^{(i)})^{-1}\right\}^{-1} \tag{31}$$

In this example, we need to approximate $P(\mathbf{w}|T_i)$, and we sample $\mathbf{z}^{(i)}$ from posterior distribution $P(\mathbf{w}|\mathbf{z}, T_i)$, then we got

$$P(\mathbf{w}|T_i) \approx \left\{\frac{1}{m}\sum_{i=1}^{m} P(\mathbf{w}|\mathbf{z}^{(i)}, T_i)^{-1}\right\}^{-1} \tag{32}$$

3 On the importance of initialization and momentum in deep learning (Sutskever et al., 2013)

In this paper, authors mentioned two momentum-based optimization methods for deep learning: a). classical momentum (CM) and b). Nesterov's accelerated gradient (NAG).

3.1 Gradient Descent

The basic gradient descent is defined as,

$$\theta_{t+1} = \theta_t - \epsilon \nabla f(\theta_t) \tag{33}$$

where the θ is learning parameters or weights. and the ϵ is learning rate.

3.2 Classical Momentum (CM)

The CM method is defined as,

$$v_{t+1} = \mu v_t + (1-\mu)\nabla f(\theta_t) \tag{34}$$
$$\theta_{t+1} = \theta_t - \epsilon v_{t+1} \tag{35}$$

where ϵ is the learning rate, $\mu \in [0, 1]$ is momentum coefficient. The notations are same with paper, but the equation is slightly different from the original paper.

3.3 Nesterov's Accelerated Gradient (NAG)

The NAG method is defined as,

$$v_{t+1} = \mu v_t + (1-\mu)\nabla f(\theta_t + \mu v_t) \tag{36}$$
$$\theta_{t+1} = \theta_t - \epsilon v_{t+1} \tag{37}$$

4 Auto-Encoding Variational Bayes (Kingma and Welling, 2013)

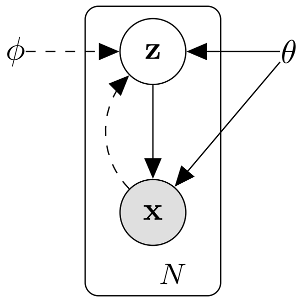

Figure 2: Variational inference of graphical model

VAE is popular generative model that performs an auto-encoder manner with reparameterization trick.

4.1 The Variational Bound

Considering some dataset $X = \{x^{(i)}\}_{i=1}^{N}$ consisting of N i.i.d. samples of some continuous or discrete variable $\mathbf{x}$. We assume that the data are generated by some random process, involving an unobserved continuous random variable $\mathbf{z}$. Because, to compute $p(\mathbf{x})$ is intractable, that involves the integral of the marginal distribution $p(\mathbf{x}) = \int p(\mathbf{z})p(\mathbf{x}|\mathbf{z})d\mathbf{z}$. Hence, to infer posterior density $p(\mathbf{z}|\mathbf{x}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z})/p(\mathbf{x})$ is also intractable. So, EM algorithm is intractable.

To solve this problem, authors introduce a recognition model $q(\mathbf{z}|\mathbf{x})$ to approximate true posterior $p(\mathbf{z}|\mathbf{x})$ and method to learn the recognition model parameters ϕ jointly with the generative model parameters θ. And the important definition of this paper is that they refer the recognition model $q_\phi(\mathbf{z}|\mathbf{x})$ as a probabilistic ***encoder***, and generative model, and refer $p_\theta(\mathbf{x}|\mathbf{z})$ as as a probabilistic ***decoder***. The Figure 3 illustrates the encoder-decoder framework in VAE.

Then, this problem can be treated as optimization problem that is to minimize the divergence between $q_\phi(\mathbf{z}|\mathbf{x})$ and $p_\theta(\mathbf{z}|\mathbf{x})$, which is $D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))$. Then,

$$\begin{aligned}
&D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) \qquad (38)\\
&\quad= -\sum q_\phi(\mathbf{z}|\mathbf{x})\log\frac{p_\theta(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})}\\
&\quad= -\sum q_\phi(\mathbf{z}|\mathbf{x})\log p_\theta(\mathbf{z}|\mathbf{x}) + \sum q_\phi(\mathbf{z}|\mathbf{x})\log q_\phi(\mathbf{z}|\mathbf{x})\\
&\quad= -\sum q_\phi(\mathbf{z}|\mathbf{x})\log\frac{p_\theta(\mathbf{x},\mathbf{z})}{p_\theta(\mathbf{x})} + \sum q_\phi(\mathbf{z}|\mathbf{x})\log q_\phi(\mathbf{z}|\mathbf{x})\\
&\quad= -\sum q_\phi(\mathbf{z}|\mathbf{x})\log p_\theta(\mathbf{x},\mathbf{z}) + \sum q_\phi(\mathbf{z}|\mathbf{x})\log p_\theta(\mathbf{x}) + \sum q_\phi(\mathbf{z}|\mathbf{x})\log q_\phi(\mathbf{z}|\mathbf{x})
\end{aligned}$$

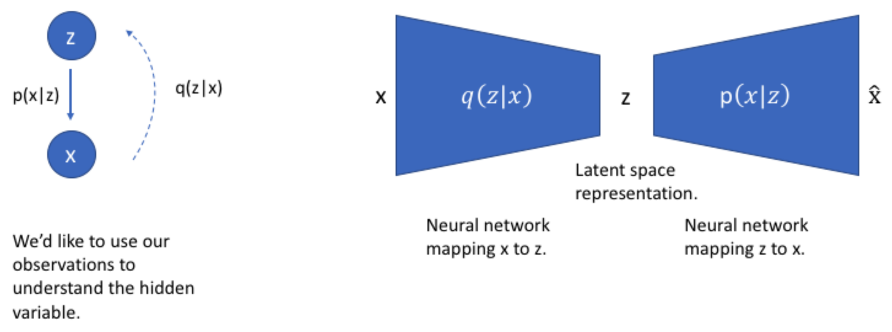
z
p(x|z)
q(z|x)
x
We'd like to use our observations to understand the hidden variable.
x
$q(z|x)$
z
$p(x|z)$
$\hat{x}$
Latent space representation.
Neural network mapping x to z.
Neural network mapping z to x.

Figure 3: Encoder-Decoder framework in VAE

$$= \sum q_\phi(\mathbf{z}|\mathbf{x})\log p_\theta(\mathbf{x}) - \sum q_\phi(\mathbf{z}|\mathbf{x})\log\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}$$
$$= \log p_\theta(\mathbf{x}) - \sum q_\phi(\mathbf{z}|\mathbf{x})\log\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}$$

Hence,

$$\log p_\theta(\mathbf{x}) = D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) + \sum q_\phi(\mathbf{z}|\mathbf{x})\log\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \tag{39}$$

We define second term in Eq. 39 as variational lower bound $\mathcal{L}(\theta,\phi;x)$. Then,

$$\log p_\theta(\mathbf{x}) = D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) + \mathcal{L}(\theta,\phi;\mathbf{x}) \tag{40}$$

And,

$$\begin{aligned}\mathcal{L}(\theta,\phi;\mathbf{x}) &= \sum q_\phi(\mathbf{z}|\mathbf{x})\log\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \\ &= \sum q_\phi(\mathbf{z}|\mathbf{x})\log\frac{p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \\ &= \sum q_\phi(\mathbf{z}|\mathbf{x})\left[\log p_\theta(\mathbf{x}|\mathbf{z}) - \log\frac{p_\theta(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] \\ &= \sum q_\phi(\mathbf{z}|\mathbf{x})\log p_\theta(\mathbf{x}|\mathbf{z}) - \sum q_\phi(\mathbf{z}|\mathbf{x})\log\frac{p_\theta(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} \\ &= -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right]\end{aligned} \tag{41}$$

So, to maximize the $\log p_\theta(\mathbf{x})$ is also equal to maximize lower bound $\mathcal{L}(\theta,\phi;\mathbf{x})$. And, because, $D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}))$ is hard to compute, we can just maximize the second term $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right]$ to increase lower bound. Authors also mentioned the navie method Monte Carlo gradient estimator to approximate compute derivative of

$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right]$, which is

$$\begin{aligned}
\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] &= \nabla_\phi \int_{\mathbf{z}} q_\phi(\mathbf{z}|\mathbf{x}) \log p_\theta(\mathbf{x}|\mathbf{z}) d\mathbf{z} \\
&= \int_{\mathbf{z}} \log p_\theta(\mathbf{x}|\mathbf{z}) \nabla_\phi \left[q_\phi(\mathbf{z}|\mathbf{x})\right] d\mathbf{z} \\
&= \int_{\mathbf{z}} \log p_\theta(\mathbf{x}|\mathbf{z}) q_\phi(\mathbf{z}|\mathbf{x}) \nabla_\phi \left[\log q_\phi(\mathbf{z}|\mathbf{x})\right] d\mathbf{z} \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z}) \nabla_\phi \left[\log q_\phi(\mathbf{z}|\mathbf{x})\right]\right]
\end{aligned} \tag{42}$$

Then, it can approximate this expectation using Monte Carlo integration

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = \mathbb{E}_{q_\phi(\mathbf{z})}\left[f(\mathbf{z}) \nabla_{q_\phi(\mathbf{z})} \log q_\phi(\mathbf{z})\right] \simeq \frac{1}{L}\sum_{l=1}^{L} f(\mathbf{z}) \nabla_{q_\phi\left(z^{(l)}\right)} \log q_\phi\left(z^{(l)}\right) \tag{43}$$

Where $z^{(l)} \sim q_\phi(\mathbf{z}|\mathbf{x})$ and $\log p_\theta(\mathbf{x}|\mathbf{z})$ is simplified with with $f(\mathbf{z})$, And this gradient estimator exhibits exhibits very high variance (Paisley et al., 2012). **So, the problem becomes that can we differentiate through the sampling process?**

4.2 The SGVB estimator and AEVB algorithm

In this paper, author approximate posterior $q_\phi(\mathbf{z}|x)$ using **reparameterize** the random variable $\widetilde{\mathbf{z}} \sim q_\phi(\mathbf{z}|\mathbf{x})$ using a differentiable transformation $g_\phi(\epsilon, \mathbf{x})$

$$\widetilde{\mathbf{z}} = g_\phi(\epsilon, \mathbf{x}) \quad \text{with} \quad \epsilon \sim p(\boldsymbol{\epsilon}) \tag{44}$$

Then, use Monte Carlo estimates of expectations of some function $f(\mathbf{z})$ w.r.t. $q_\phi(\mathbf{z}|\mathbf{x})$ as follows:

$$\mathbb{E}_{q_\phi\left(\mathbf{z}|\mathbf{x}^{(i)}\right)}[f(\mathbf{z})] = \mathbb{E}_{p(\epsilon)}\left[f\left(g_\phi\left(\boldsymbol{\epsilon}, \mathbf{x}^{(i)}\right)\right)\right] \simeq \frac{1}{L}\sum_{l=1}^{L} f\left(g_\phi\left(\boldsymbol{\epsilon}^{(l)}, \mathbf{x}^{(i)}\right)\right) \quad \text{where} \quad \boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon}) \tag{45}$$

Because, from Eq. 41 the $\mathcal{L}(\theta, \phi; \mathbf{x})$ can be written as

$$\mathcal{L}(\theta, \phi; \mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})\right] \tag{46}$$

Apply Eq. 45 to Eq. 46 for approximating lower bound, yielding our generic Stochastic Gradient Variational Bayes (SGVB) estimator $\widetilde{\mathcal{L}}^A\left(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}\right) \simeq \mathcal{L}\left(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}\right)$

$$\begin{gathered}
\widetilde{\mathcal{L}}^A\left(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}\right) = \frac{1}{L}\sum_{l=1}^{L}\left[\log p_{\boldsymbol{\theta}}\left(\mathbf{x}^{(i)}, \mathbf{z}^{(i,l)}\right) - \log q_\phi\left(\mathbf{z}^{(i,l)}|\mathbf{x}^{(i)}\right)\right] \\
\text{where} \quad \mathbf{z}^{(i,l)} = g_\phi\left(\boldsymbol{\epsilon}^{(i,l)}, \mathbf{x}^{(i)}\right) \quad \text{and} \quad \boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon})
\end{gathered} \tag{47}$$

And author also provide another version of SGVB estimator that considers KL divergence if it can be computed using Eq. 41 as follows:

$$\widetilde{\mathcal{L}}^B\left(\boldsymbol{\theta},\phi;\mathbf{x}^{(i)}\right) = -D_{KL}\left(q_\phi\left(\mathbf{z}|\mathbf{x}^{(i)}\right)\|p_{\boldsymbol{\theta}}(\mathbf{z})\right) + \frac{1}{L}\sum_{l=1}^{L}\left(\log p_{\boldsymbol{\theta}}\left(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)}\right)\right) \tag{48}$$
$$\text{where } \mathbf{z}^{(i,l)} = g_\phi\left(\boldsymbol{\epsilon}^{(i,l)},\mathbf{x}^{(i)}\right) \quad \text{and} \quad \boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon})$$

And first term can be treated as a regularizer. Then given dataset X with N datapoints, we can construct an estimator of SGVB, as follows:

$$\mathcal{L}(\boldsymbol{\theta},\phi;\mathbf{X}) \simeq \widetilde{\mathcal{L}}^M\left(\boldsymbol{\theta},\phi;\mathbf{X}^M\right) = \frac{N}{M}\sum_{i=1}^{M}\widetilde{\mathcal{L}}\left(\boldsymbol{\theta},\phi;\mathbf{x}^{(i)}\right) \tag{49}$$

Therefore, the **auto-encoder procedure** will be

a) **Encoder**: choose function $g_\phi(.)$ that maps datapoint $\mathbf{x}^{(i)}$ and a ranodom noise $\boldsymbol{\epsilon}^{(l)}$ to a sample from the approximate posterior for that datapoint $\mathbf{z}^{(i,l)} = g_\phi\left(\boldsymbol{\epsilon}^{(l)},\mathbf{x}^{(i)}\right)$ where $\mathbf{z}^{(i,l)} \sim q_\phi\left(\mathbf{z}|\mathbf{x}^{(i)}\right)$

b) **Decoder**: use sample $\mathbf{z}^{(i,l)}$ as input for function $\log p_{\boldsymbol{\theta}}\left(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)}\right)$ to generate $\mathbf{x}^{(i)}$

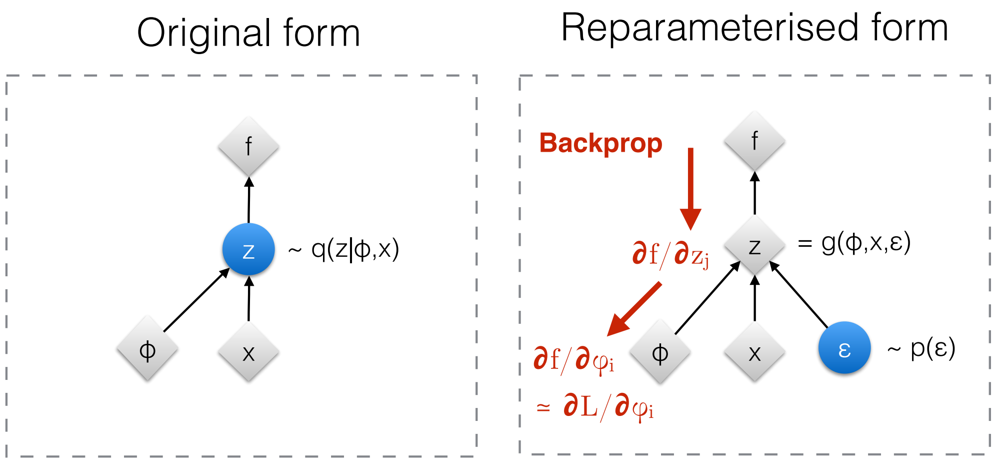

Figure 4: Reparameterization trick

4.3 The reparameterization trick

The naive method for computing $\mathcal{L} = \mathbb{E}_{q_\phi(z|x)}\left[f_\phi(z)\right]$ that requires backpropagation through random sampling process $\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})$ that leads high variance . Hence, the author uses reparameterization trick to solve this problem. That express the random variable $\mathbf{z}$ as a deterministic variable $\mathbf{z} = g_\phi(\boldsymbol{\epsilon}, \mathbf{x})$, where ϵ is an auxiliary variable with independent marginal $p(\epsilon)$. For example,

$$\text{Original: } z \sim p(z|x) = \mathcal{N}\left(\mu, \sigma^2\right)$$
$$\text{Reparameterization: } z = \mu + \sigma\epsilon \quad \text{and} \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0,1)$$

Then,

$$\mathbb{E}_{\mathcal{N}(\mu,\sigma^2)}\left[f_(z)\right] = \mathbb{E}_{\mathcal{N}(\epsilon;0,1)}\left[f(\mu+\sigma\epsilon)\right] \simeq \frac{1}{L}\sum_{l=1}^{L} f(\mu+\sigma\epsilon^{(l)}) \quad \text{and} \quad \epsilon^{(l)} \sim \mathcal{N}(0,1)$$

Therefor, the reparameterization trick can be summarized as follows:

a) sample $\epsilon^{(l)}$ from $p(\epsilon)$
b) $z^{(l)} \sim g_\phi(\boldsymbol{\epsilon})$, such that $z^{(l)} \sim q_\phi(z^{(l)}|x)$
c) $\mathcal{L} \simeq f_\phi(x, z^{(l)})$, here the number of sample is 1
d) $\nabla_\phi \mathcal{L} \simeq \nabla_\phi f_\phi(x, z^{(l)})$

5 Generative Adversarial Nets (Goodfellow et al., 2014)

GAN (Generative Adversarial Nets) is one of the most popular deep generative model published in 2014.

In VAE model described in Section 4,

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{ data }}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \tag{50}$$

6 InfoGAN (Chen et al., 2016)

Due to the latent representations of z in GAN are usually uncontrollable and unexplainable, InfoGAN is proposed as an extension to Generative Adversarial Nets (GAN) (Goodfellow et al., 2014), that is able to learn disentangled representations in an unsupervised manner. The original GAN defines the minmax function between discriminator D and generator G,

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim P_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim P_z}[\log(1-D(G(z)))] \tag{51}$$

The goal of InfoGAN is to learn the learn interpretable and meaningful latent representations $c_1, c_2, ..., c_L$. Then, the generator network needs to consider both the noise z and the latent code c, which becomes $G(z,c)$.

Authors solve this problem using mutual information theory, which means mutual information between c and $G(z,c)$: $I(c; G(z,c))$ should be high. According to mutual information theory, it can be written as,

$$I(c; G(z,c)) = H(c) - H(c|G(z,c))$$

It measures the average reduction in uncertainty about c that results from learning the value of $G(z,c)$. Therefore, it can be solved the following information-regularized minimax game,

$$\min_G \max_D V_I(D,G) = V(D,G) - \lambda I(c; G(z,c)) \tag{52}$$

6.1 Variational Mutual Information Maximization

However, the $I(c; G(z,c))$ is hard to maximize, because there is no close form for $P(c|x)$. Using variational inference theory, it can be approximated with an auxiliary distribution $Q(c|x)$, hence

$$\begin{aligned}
I(c;&G(z,c)) = H(c) - H(c|G(z,c)) \\
&=H(c) + \sum_{x\in G(z,c)} \sum_{c'\in P(c|x)} P(c=c', x=G(z,c)) \log P(c=c'|x=G(z,c)) \\
&= \sum_{x\in G(z,c)} \sum_{c'\in P(c|x)} P(c',x) \log P(c'|x) + H(c) \\
&= \sum_{x\in G(z,c)} \sum_{c'\in P(c|x)} P(x)P(c'|x) \log P(c'|x) + H(c) \\
&= \sum_{x\in G(z,c)} P(x) \sum_{c'\in P(c|x)} P(c'|x) \log \frac{Q(c'|x)P(c'|x)}{Q(c'|x)} + H(c) \\
&= \sum_{x\in G(z,c)} P(x) \sum_{c'\in P(c|x)} P(c'|x) \log \frac{P(c'|x)}{Q(c'|x)} + \sum_{x\in G(z,c)} P(x) \sum_{c'\in P(c|x)} P(c'|x) \log Q(c'|x) + H(c) \\
&= \sum_{x\in G(z,c)} P(x) D_{KL}(P(c'|x)||Q(c'|x)) + \sum_{x\in G(z,c)} P(x) \sum_{c'\in P(c|x)} P(c'|x) \log Q(c'|x) + H(c)
\end{aligned}$$

$$
\begin{aligned}
&\geq \sum_{x \in G(z,c)} P(x) \sum_{c' \in P(c|x)} P(c'|x) \log Q(c'|x) + H(c) \\
&= \mathbb{E}_{x \sim G(z,c)} \left[\mathbb{E}_{c' \sim P(c|x)} \left[\log Q\left(c'|x\right) \right] \right] + H(c)
\end{aligned}
$$

Because the $H(c)$ is constant, then the question is to maximize the term

$$
\mathbb{E}_{x \sim G(z,c)} \left[\mathbb{E}_{c' \sim P(c|x)} \left[\log Q\left(c'|x\right) \right] \right] \tag{53}
$$

In The Equation 53, we still need to access to $P(c|x)$, which is hard to compute. But we can transform the Equation 53 into,

$$
\begin{aligned}
&\mathbb{E}_{x \sim G(z,c)} \left[\mathbb{E}_{c' \sim P(c|x)} \left[\log Q\left(c'|x\right) \right] \right] \\
&= \sum_{x \in G(z,c)} P(x) \sum_{c' \in P(c|x)} P(c'|x) \log Q(c'|x) \\
&= \sum_{x \in G(z,c)} P(x) \log Q(c'|x) \sum_{c' \in P(c|x)} P(c'|x) \\
&= \sum_{x \in G(z,c)} P(x) \log Q(c'|x) \qquad \text{(change variable } c' \text{ to } c\text{)} \\
&= \sum_{x \in G(z,c)} P(x) \log Q(c|x) \\
&= \sum_{x \in G(z,c)} \sum_{c \in P(c)} P(x,c) \log Q(c|x) \\
&= \sum_{x \in G(z,c)} \sum_{c \in P(c)} P(c) P(x|c) \log Q(c|x) \\
&= \sum_{x \in G(z,c)} P(c) \sum_{c \in P(c)} P(x|c) \log Q(c|x) \\
&= \mathbb{E}_{c \sim P(c), x \sim G(z,c)} [\log Q(c|x)]
\end{aligned}
$$

Then, we can define a variational lower bound as $L_1(G,Q)$

$$
L_1(G,Q) = \mathbb{E}_{c \sim P(c), x \sim G(z,c)} [\log Q(c|x)] + H(c) \leq I(c; G(z,c)) \tag{54}
$$

In Equation 54, c is drawn from prior distribution (Gaussian or Categorical). x is generated by our generator. And, the $Q(c|x)$ is computed by our auxiliary network.

7 Sequence to Sequence Learning with Neural Networks (Sutskever et al., 2014)

In NLP, Seq2Seq models are popular model widely used in Machine Translation, Text Summarization, Conversational Modeling, Image Captioning, and so on. And Seq2Seq model can be also used in time series analysis (such as stock price prediction, traffic prediction).

DNN model achieves excellent performance in image recognition. However, the limitation of DNN is that DNN requires fixed size of inputs and outputs, which is hard to apply to those tasks with unknown inputs/outputs length, e.g., speech, text.

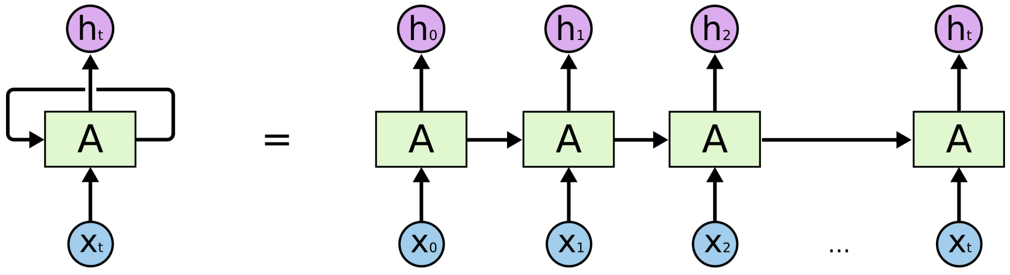

Figure 5: A Basic RNN model

As shown in Figure 5, the basic RNN model can be formulated as,

$$h_t = \tanh(W_{hh}h_{t-1} + W_{hx}x_t + b_h)$$
$$y_t = \sigma(W_{hy}h_t + b_y)$$

References

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems*, pages 2172–2180, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf`.

Thomas L. Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(suppl 1):5228–5235, 2004. doi: 10.1073/pnas.0307752101. URL `http://www.pnas.org/content/101/suppl_1/5228.abstract`.

Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 289–296. Morgan Kaufmann Publishers Inc., 1999.

Robert E Kass and Adrian E Raftery. Bayes factors. *Journal of the american statistical association*, 90(430):773–795, 1995.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Michael A Newton and Adrian E Raftery. Approximate bayesian inference with the weighted likelihood bootstrap. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 3–48, 1994.

John Paisley, David Blei, and Michael Jordan. Variational bayesian inference with stochastic search. *arXiv preprint arXiv:1206.6430*, 2012.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1139–1147, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL `http://proceedings.mlr.press/v28/sutskever13.html`.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.