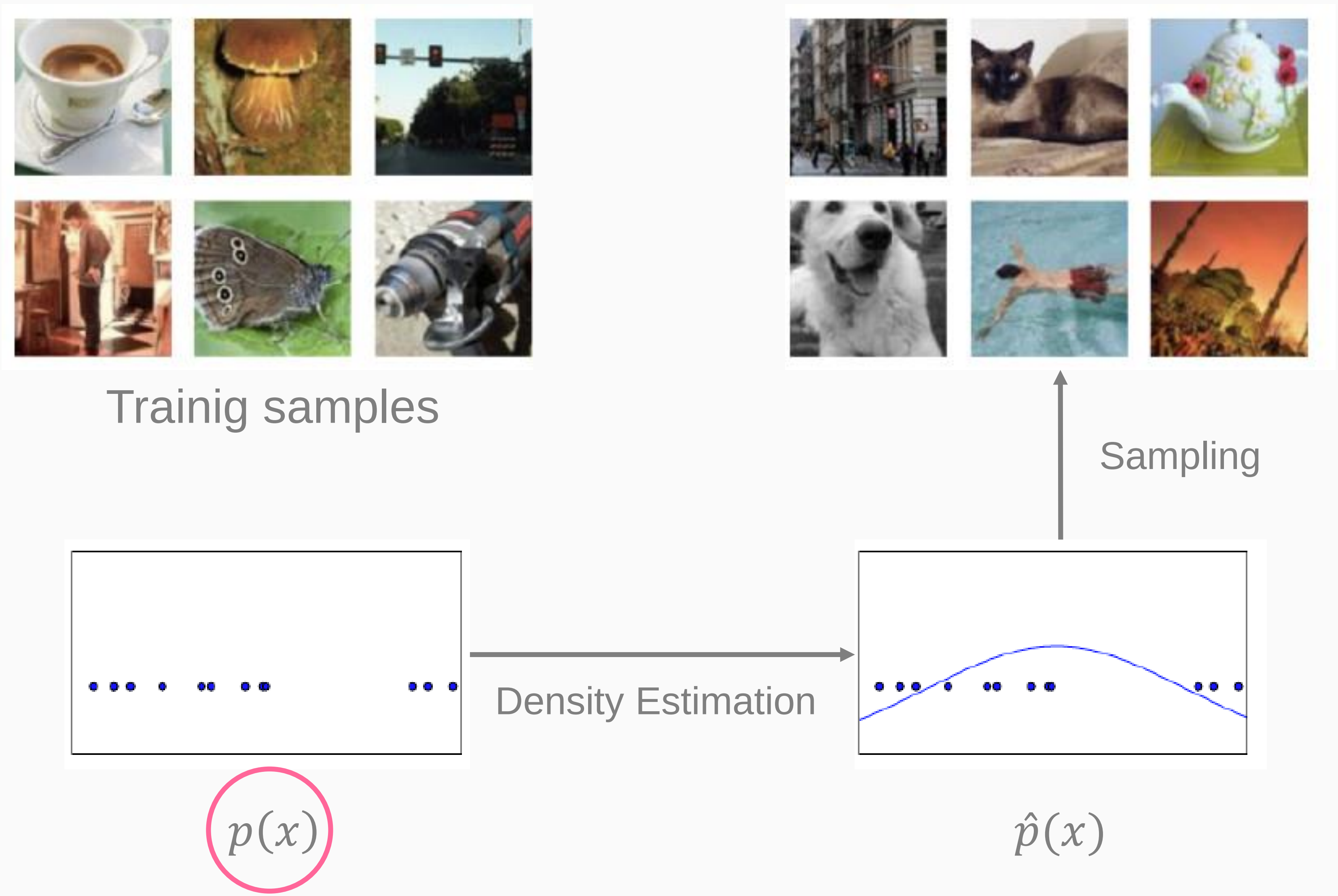
Auto-Encoding Variational Bayes

Diederik P. Kingma, Max Welling, 2014

Motivation – generative model



Problem scenario - Latent Variable Model

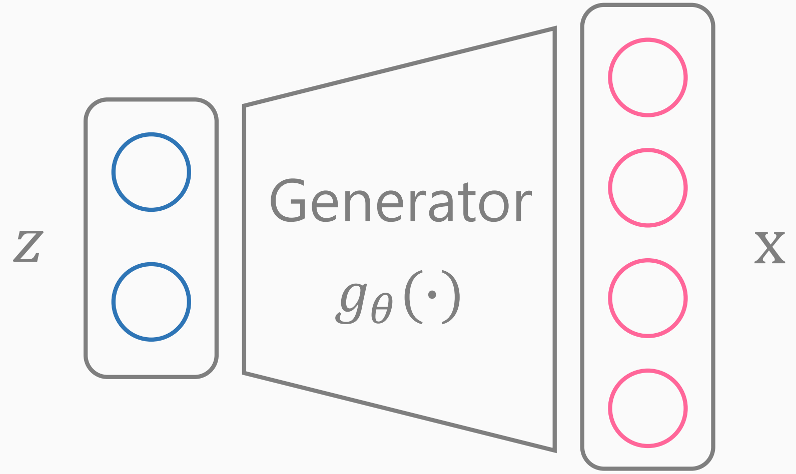
step1 : a value $z^{(i)}$ is generated from some prior distribution $p(z)$ *(normal or uniform)*

step2 : a value $\mathrm{x}^{(i)}$ is generated from some contional distribution $p(\mathrm{x}|g_{\theta^*}(z))$

- $\mathrm{X} = \{\mathrm{x}^{(i)}\}_{i=1}^{N}$ dataset
- x random variable
- z continuous random variable in space Z
- $p(\cdot)$ pdf of z
- $g_\theta(\cdot)$ a deterministic function parameterized by θ in some space Θ

  $g: Z \times \Theta \rightarrow \chi$, χ is some space
- $g(z;\theta) = g_\theta(z)$ random variable in the space χ
- $p(\mathrm{x}|g_\theta(z)) = p_\theta(\mathrm{x}|z)$ likelihood

Goal : maximize $p(\mathrm{X}) = \int p\big(\mathrm{X}|g_\theta(z)\big)p(z)dz$!!!
In VAEs, this output distribution is often istropic gaussian
i.e. $p(\mathrm{X}|g_\theta(z)) = N(\mathrm{X}|g_\theta(z), \sigma^2 I)$ (Tutorial on VAEs, 3page)



Problem1 - $p_\theta(x|z)$

- $p(\mathrm{x}) = \int p\big(\mathrm{x}|g_\theta(z)\big)p(z)dz$ the integral of the marginal likelihood

  $\rightarrow \int p\big(\mathrm{x}|g_\theta(z)\big)p(z)dz$ is intractable

Solution

- **Bad solution**

  임의의 prior distribution에서 z를 샘플링하여 likelihood가 커지도록 θ를 조절하는 방법

  → 원하는 결과를 얻지 못함. 데이터 셋의 분포를 추정하기 어려움

- **Good solution**

  $p(\mathrm{z}|\mathrm{x})$에서 z를 샘플링하여 likelihood가 커지도록 θ를 조절하는 방법

  $\rightarrow p(\mathrm{z}|\mathrm{x}) = \frac{p(\mathrm{X}|Z)p(z)}{p(\mathrm{x})}$ is intractable

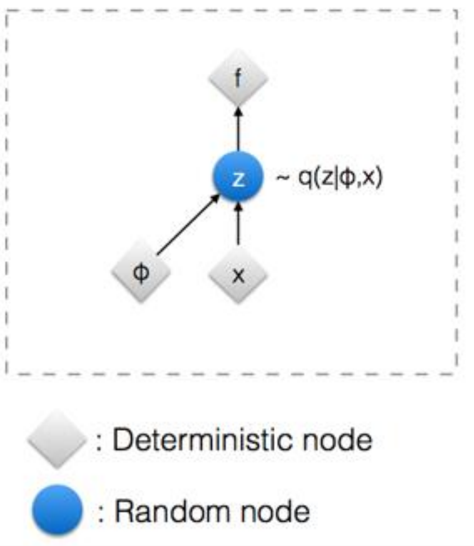
  → Variational Inference 방법을 이용하여 해결

Problem2 - Sampling

- z를 샘플링 해야한다

$$z^i \sim N(\mu_i, \sigma_i^2 I)$$

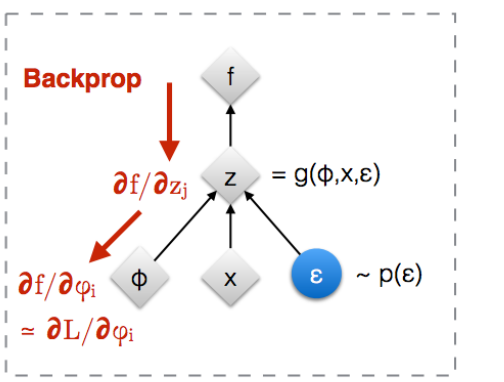
- 샘플링을 하면 gradient descent 방법을 사용할 수 없다.



Solution

- Reparameterization trick
- backpropagation 계산이 가능한 방법으로 대체한다.

$$z^i = \mu_i + \sigma_i^2 \odot \epsilon,\ \epsilon \sim N(0, I)$$



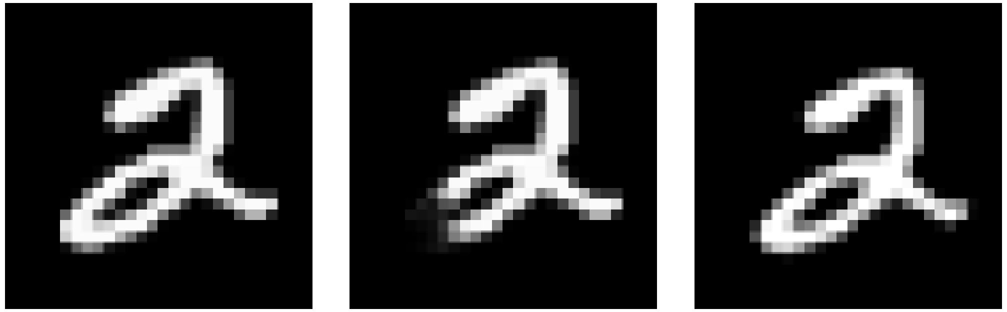
[Kingma, 2013]
[Bengio, 2013]
[Kingma and Welling 2014]
[Rezende et al 2014]

Bad Solution – maximum likelihood estimation directly

step1 : $p(z)$에서 데이터 n개를 샘플링한다. $\{z^{(1)}, \cdots, z^{(2)}\}$

step2 : $p(\mathrm{x}) \approx \sum_{i=1}^{n} p\left(\mathrm{x}|g_\theta\left(z^{(i)}\right)\right)p\left(z^{(i)}\right)$ * 단, x의 차원이 크면 n도 커야 한다.

step3 : $p(x|g_\theta(z))$를 Isotropic Gaussian($N(x|g_\theta(z), \sigma^2 I)$)으로 가정하면, -log $p(x|g_\theta(z))$는 $g_\theta(z)$와 x의 Euclidian distance와 비례관계 * MSE를 목적함수로 학습



(a) (b) (c)

문제점

$$\left\|\mathrm{x} - g_\theta\left(z^{(b)}\right)\right\|^2 < \left\|\mathrm{x} - g_\theta\left(z^{(c)}\right)\right\|^2 \rightarrow p\left(\mathrm{x}|g_\theta\left(z^{(b)}\right)\right) > p\left(\mathrm{x}|g_\theta\left(z^{(c)}\right)\right)$$

(c)의 샘플로 생성된 이미지가 의미적으로 (a)에 더 비슷하지만,
MSE관점에서는 (b)의 샘플이 $p(\mathrm{x})$에 기여하는 바가 더 크다. * 올바른 학습이 어려움

Good Solution – Variational inference

- z를 임의의 정규분포로부터 샘플링하는 것은 Bad Solution
  → x와 의미적으로 유사한 샘플을 얻기 위하여 $p(z|\mathrm{x})$로 부터 샘플링
  → $p(z|\mathrm{x})$를 알 수 없음
  → 해석하기 쉬운 확률 분포 중 하나($q_\phi(z|x)$)를 택하고,
  ϕ 값을 조정하여 $p(z|x)$와 유사하게 만들자. * Variational Inference
  → $q_\phi(z|x)$ 분포로부터 z를 샘플링

- **요약**
  1. 데이터 셋과 유사한 이미지를 생성하자
  2. $g_\theta(\cdot)$와 z를 이용해서 생성하자
  3. z는 쉬운 확률 분포(정규분포) 에서 샘플링하자
  4. 데이터셋과 의미적으로 유사한 샘플을 얻을 수 있도록 z를 $p(z|\mathrm{x})$에서 샘플링하자
  5. $p(z|\mathrm{x})$를 구하기 어렵다
  6. 쉬운 확률 분포 $q_\phi(z|\mathrm{x})$를 택하고 $p(z|\mathrm{x})$에 근사하게 파라미터를 조절하자.
  7. $q_\phi(z|\mathrm{x})$ 분포로부터 z를 샘플링하자

Variational inference – ELBO : Evidence Lower BOund

- Relationship among $p(x)$, $p(z|x)$, $q_\phi(z|x)$: Derivation 1

$$\log(p(\mathrm{x})) = \log\left(\int p(\mathrm{x}|z)p(z)\,dz\right)$$

$$= \log\left(\int p(\mathrm{x}|z)\frac{p(z)}{q_\phi(z|\mathrm{x})}q_\phi(z|\mathrm{x})dz\right) \geq \int \log\left(p(\mathrm{x}|z)\frac{p(z)}{q_\phi(z|\mathrm{x})}\right)q_\phi(z|\mathrm{x})dz$$

Jensen's Ineuqality, $\log(\cdot)$ is concave

$$\log(p(\mathrm{x})) \geq \int \log(p(\mathrm{x}|z))q_\phi(z|\mathrm{x})dz - \int \log\left(\frac{q_\phi(z|\mathrm{x})}{p(z)}\right)q_\phi(z|\mathrm{x})dz$$

$$= \underbrace{E_{q_\phi(z|\mathrm{x})}[\log(p(\mathrm{x}|z))] - KL(q_\phi(z|\mathrm{x}) \parallel p(z))}_{ELBO(\phi)}$$

Variational inference – ELBO : Evidence Lower BOund

- Relationship among $p(\mathrm{x})$, $p(z|\mathrm{x})$, $q_\phi(z|\mathrm{x})$: Derivation 2

$$
\begin{aligned}
\log(p(\mathrm{x})) &= \int \log\big(p(\mathrm{x})\big) q_\phi(z|\mathrm{x})dz \quad \longleftarrow \int q_\phi(z|\mathrm{x})dz = 1 \\
&= \int \log\left(\frac{p(\mathrm{x},z)}{p(z|\mathrm{x})}\right) q_\phi(z|\mathrm{x}) \\
&= \int \log\left(\frac{p(\mathrm{x},z)}{q_\phi(z|\mathrm{x})} \cdot \frac{q_\phi(z|\mathrm{x})}{p(z|\mathrm{x})}\right) q_\phi(z|\mathrm{x})dz \\
&= \underbrace{\int \log\left(\frac{p(\mathrm{x},z)}{q_\phi(z|\mathrm{x})}\right) q_\phi(z|\mathrm{x})dz}_{ELBO(\phi)} + \underbrace{\int \log\left(\frac{q_\phi(z|\mathrm{x})}{p(z|\mathrm{x})}\right) q_\phi(z|\mathrm{x})dz}_{KL\left(q_\phi(z|\mathrm{x}) \,\|\, p(z|\mathrm{x})\right)}
\end{aligned}
$$

KL을 최소화하는 $q_\phi(z|\mathrm{x})$의 ϕ값을 구하면 된다.

$p(z|\mathrm{x})$를 알 수 없기 때문에 KL을 최소화하는 대신 ELBO를 최대화하는 ϕ값을 찾는다.

Variational inference – ELBO : Evidence Lower BOund

- Relationship among $p(\mathrm{x}), p(z|\mathrm{x}), q_\phi(z|\mathrm{x})$: Derivation 2

$$\log(p(\mathrm{x})) = ELBO(\phi)\uparrow + KL\left(q_\phi(z|\mathrm{x}) \parallel p(z|\mathrm{x})\right)\downarrow$$

Goal : $\underset{\phi}{\operatorname{argmax}}\, ELBO(\phi)$

$$\begin{aligned} ELBO(\phi) &= \int \log\left(\frac{p(\mathrm{x}, z)}{q_\phi(z|\mathrm{x})}\right) q_\phi(z|\mathrm{x}) dz \\ &= \int \log\left(\frac{p(\mathrm{x}|z)p(z)}{q_\phi(z|\mathrm{x})}\right) q_\phi(z|\mathrm{x}) dz \\ &= \int \log(p(\mathrm{x}|z))\, q_\phi(z|\mathrm{x}) dz - \int \log\left(\frac{q_\phi(z|\mathrm{x})}{p(z)}\right) q_\phi(z|\mathrm{x}) dz \\ &= E_{q_\phi(z|\mathrm{x})}[\log(p(\mathrm{x}|z))] - KL(q_\phi(z|\mathrm{x}) \parallel p(z)) \end{aligned}$$

$$\log(p(\mathrm{x})) \geq E_{q_\phi(z|\mathrm{x})}[\log(p(\mathrm{x}|z))] - KL(q_\phi(z|\mathrm{x}) \parallel p(z))$$

Loss function

- $\log(p(\mathrm{x})) \geq E_{q_\phi(z|\mathrm{x})}[\log(p(\mathrm{x}|z))] - KL(q_\phi(z|\mathrm{x}) \parallel p(z))$

- $\log(p(\mathrm{x}|g_\theta(z))) \geq E_{q_\phi(z|\mathrm{x})}[\log(p(\mathrm{x}|g_\theta(z)))] - KL(q_\phi(z|\mathrm{x}) \parallel p(z))$

$$p(\mathrm{x})\uparrow = \int p(\mathrm{x}|g_\theta(z))\uparrow p(z)dz$$

- for data $\mathrm{x}^{(i)}$,

$$\boldsymbol{L}\left(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}\right) = -\boldsymbol{E}_{\boldsymbol{q_\phi}\left(\mathbf{z}|\mathbf{x}^{(i)}\right)}\left[\log\left(\boldsymbol{p}\left(\mathbf{x}^{(i)}|\boldsymbol{g_\theta}(\mathbf{z})\right)\right)\right] + \boldsymbol{KL}\left(\boldsymbol{q_\phi}\left(\mathbf{z}|\mathbf{x}^{(i)}\right) \parallel \boldsymbol{p}(\mathbf{z})\right)$$

Note.

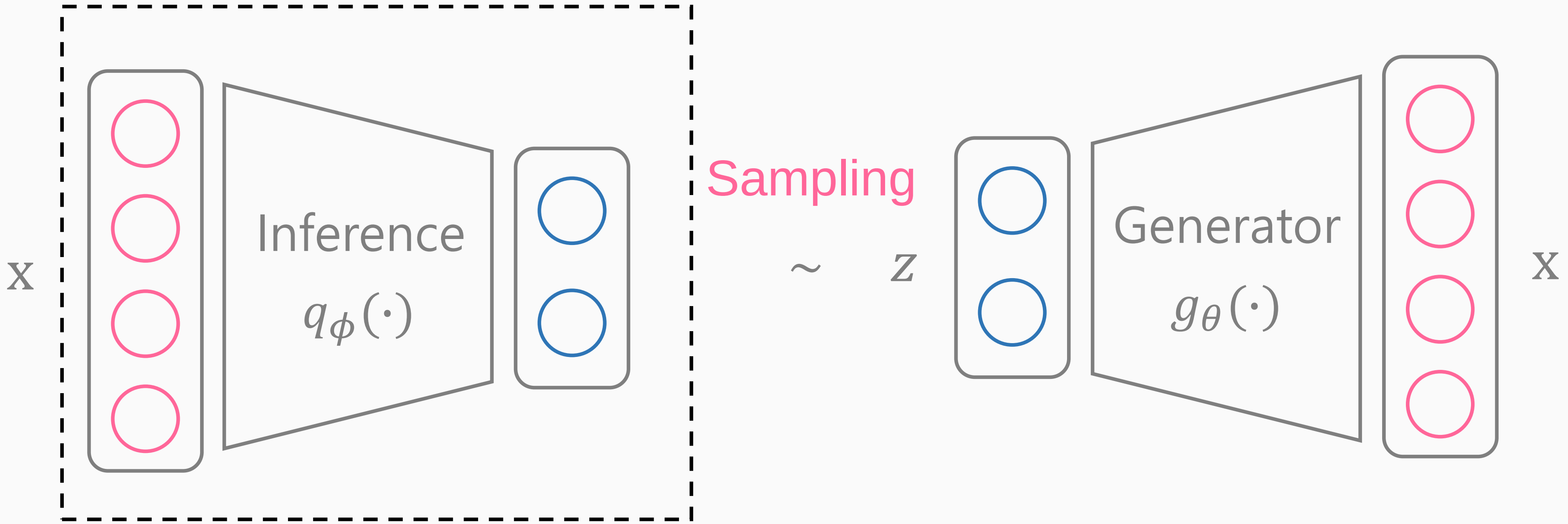
논문의 3번 식과 부호가 다르다.

논문에서의 L은 lower bound의 의미, ppt에서의 L은 loss의 의미.

논문의 lower bound를 최대화하는 것은 ppt의 loss를 최소화하는 것과 동일.

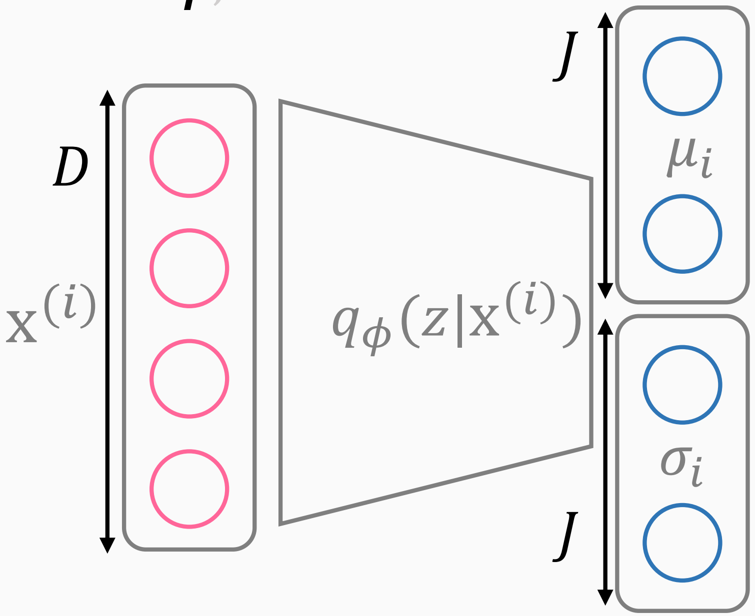
Objective function

$$\underset{\phi,\theta}{\operatorname{argmin}} \frac{1}{N}\sum_{i=1}^{N}\left(L\left(\theta,\phi;\mathrm{x}^{(i)}\right)\right) = \frac{1}{N}\sum_{i=1}^{N}\left(-E_{q_\phi\left(z|\mathrm{x}^{(i)}\right)}\left[\log\left(p\left(\mathrm{x}^{(i)}|g_\theta\right)\right)\right] + KL\left(q_\phi\left(z|\mathrm{x}^{(i)}\right) \parallel p(z)\right)\right)$$



Encoder – KL divergence

$$\underset{\boldsymbol{\phi},\,\theta}{\textbf{argmin}}\ \frac{1}{N}\sum_{i=1}^{N}\left(L\left(\theta,\phi;\mathrm{x}^{(i)}\right)\right) = \frac{1}{N}\sum_{i=1}^{N}\left(-E_{q_\phi\left(z|\mathrm{x}^{(i)}\right)}\left[\log\left(p\left(\mathrm{x}^{(i)}|g_\theta\right)\right)\right] + KL\left(q_\phi\left(z|\mathrm{x}^{(i)}\right) \parallel p(z)\right)\right)$$



$$KL\left(q_\phi\left(z|\mathrm{x}^{(i)}\right) \parallel p(z)\right) = \frac{1}{2}\left\{tr\left(\sigma_i^2 I\right) + \mu_i^T\mu_i - J + \ln\frac{1}{\prod_{j=1}^{J}\sigma_{i,j}^2}\right\}$$

$$= \frac{1}{2}\left\{\sum_{j=1}^{J}\sigma_{i,j}^2 + \sum_{j=1}^{J}\mu_{i,j}^2 - J - \sum_{j=1}^{J}\ln\left(\sigma_{i,j}^2\right)\right\}$$

$$= \frac{1}{2}\sum_{j=1}^{J}\left(\mu_{i,j}^2 + \sigma_{i,j}^2 - \ln\left(\sigma_{i,j}^2\right) - 1\right)$$

Assumption 1

Isotropic gaussian distribution

$q_\phi(z|x_i) \sim N(\mu_i, \sigma_i^2 I)$

Asuumption 2

Gaussian distribution

$p(z) \sim N(0, I)$

The Kullback-Leibler divergence from $N_0(\mu_0, \Sigma_0)$ to $N_1(\mu_1, \Sigma_1)$ for non-singular matrices Σ_0 and Σ_1 is

$$KL(N_0 \parallel N_1) = \frac{1}{2}\left\{tr(\Sigma_1^{-1}\Sigma_0) + (\mu_1 - \mu_0)^T\Sigma_1^{-1}(\mu_1 - \mu_0) - k + \ln\frac{|\Sigma_1|}{|\Sigma_0|}\right\},$$

where k is the dimension of the vector space

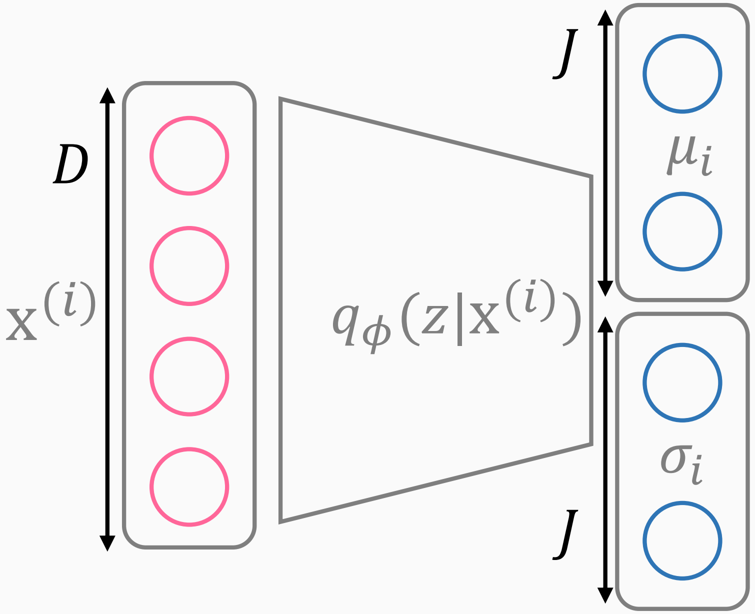
Encoder – reparameterised trick

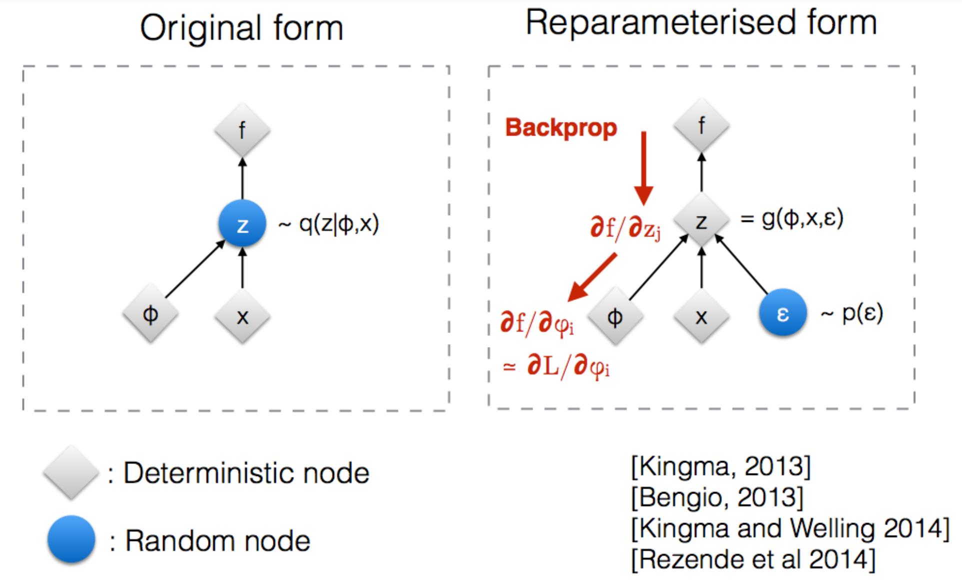
$$\underset{\phi,\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{N}\sum_{i=1}^{N}\left(L(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x}^{(i)})\right) = \frac{1}{N}\sum_{i=1}^{N}\left(-E_{q_{\phi}(z|\mathbf{x}^{(i)})}\left[\log\left(p(\mathbf{x}^{(i)}|g_{\theta})\right)\right] + KL\left(q_{\phi}(z|\mathbf{x}^{(i)}) \parallel p(z)\right)\right)$$

$$E_{q_{\phi}(z|\mathrm{x}^{(i)})}\left[\log\left(p(\mathrm{x}^{(i)}|g_{\theta}(z))\right)\right] = \int \log\left(p(\mathrm{x}^{(i)}|g_{\theta}(z))\right) q_{\phi}(z|\mathrm{x}^{(i)})dz$$

$$\approx \frac{1}{L}\sum_{z^{(i,l)}} \log\left(p(\mathrm{x}^{(i)}|g_{\theta}(z^{(i,l)}))\right)$$

monte-carlo technique





$z^{i,j} \sim N(\mu_i, \sigma_i^2 I)$ $\quad$ $z^{i,j} = \mu_i + \sigma_i^2 \odot \epsilon,\ \epsilon \sim N(0, I)$

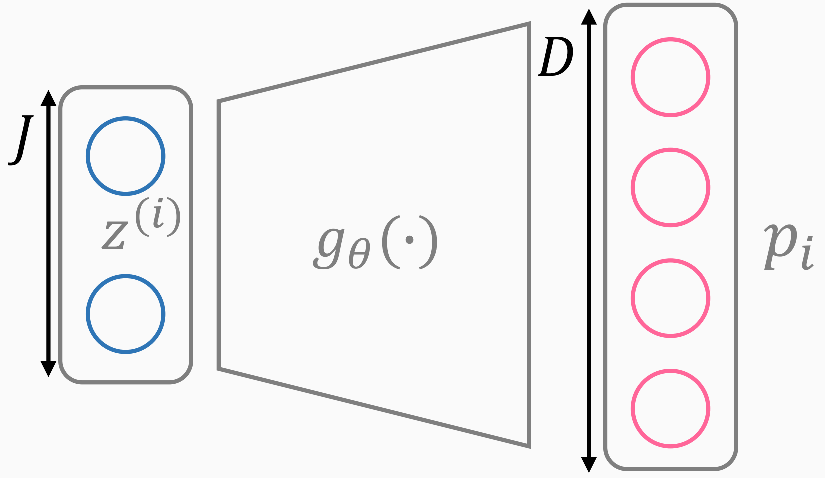
Decoder – likelihood (Bernoulli)

$$\underset{\phi,\,\theta}{\operatorname{argmin}}\ \frac{1}{N}\sum_{i=1}^{N}\left(L(\theta,\phi;\mathrm{x}^{(i)})\right) = \frac{1}{N}\sum_{i=1}^{N}\left(-E_{q_\phi(z|\mathrm{x}^{(i)})}\left[\log\left(p\left(\mathrm{x}^{(i)}|g_\theta\right)\right)\right] + KL\left(q_\phi(z|\mathrm{x}^{(i)}) \parallel p(z)\right)\right)$$

$$E_{q_\phi(z|\mathrm{x}^{(i)})}\left[\log\left(p(\mathrm{x}^{(i)}|g_\theta(z))\right)\right] = \int \log\left(p(\mathrm{x}^{(i)}|g_\theta(z))\right) q_\phi(z|\mathrm{x}^{(i)})\,dz \approx \frac{1}{L}\sum_{z^{(i,l)}} \log\left(p(\mathrm{x}^{(i)}|g_\theta(z^{(i,l)}))\right) \approx \log\left(p(\mathrm{x}^{(i)}|g_\theta(z^{(i)}))\right)$$

monte-carlo technique

$L = 1$



$$\log\left(p_\theta(\mathrm{x}^{(i)}|z^{i})\right) = \log\prod_{j=1}^{D} p_\theta\left(\mathrm{x}^{(i,j)}|z^{(i)}\right) = \sum_{j=1}^{D}\log p_\theta\left(\mathrm{x}^{(i,j)}|z^{(i)}\right)$$

$$= \sum_{j=1}^{D}\log\left((p_{i,j})^{\mathrm{x}^{(i,j)}}(1-p_{i,j})^{(1-\mathrm{x}^{(i,j)})}\right)$$

$$= \sum_{j=1}^{D}\mathrm{x}^{(i,j)}\log p_{i,j} + \left(1-\mathrm{x}^{(i,j)}\right)\log\left(1-p_{i,j}\right)$$

cross entropy

$$p_\theta\left(\mathrm{x}^{(i)}|g_\theta(z^{(i)})\right)\sim Bernoulli(p_i)$$

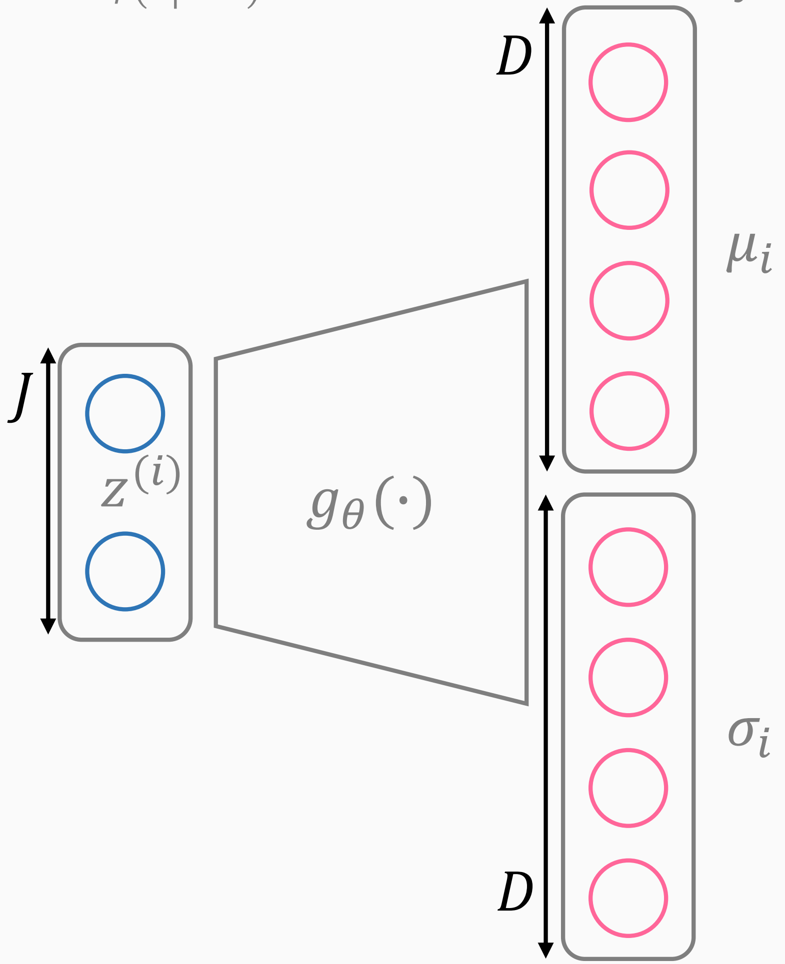
Decoder – likelihood (Gaussian)

$$\underset{\phi,\,\theta}{\mathbf{argmin}}\ \frac{1}{N}\sum_{i=1}^{N}\left(L(\theta,\phi;\mathbf{x}^{(i)})\right) = \frac{1}{N}\sum_{i=1}^{N}\left(-E_{q_\phi(z|\mathbf{x}^{(i)})}\left[\log\left(p(\mathbf{x}^{(i)}|g_\theta)\right)\right] + KL\left(q_\phi(z|\mathbf{x}^{(i)}) \parallel p(z)\right)\right)$$

$$E_{q_\phi(z|\mathrm{x}^{(i)})}\left[\log\left(p(\mathrm{x}^{(i)}|g_\theta(z))\right)\right] = \int \log\left(p(\mathrm{x}^{(i)}|g_\theta(z))\right) q_\phi(z|\mathrm{x}^{(i)})dz \approx \frac{1}{L}\sum_{z^{(i,l)}} \log\left(p(\mathrm{x}^{(i)}|g_\theta(z^{(i,l)}))\right) \approx \log\left(p(\mathrm{x}^{(i)}|g_\theta(z^{(i)}))\right)$$

monte-carlo technique

$L = 1$



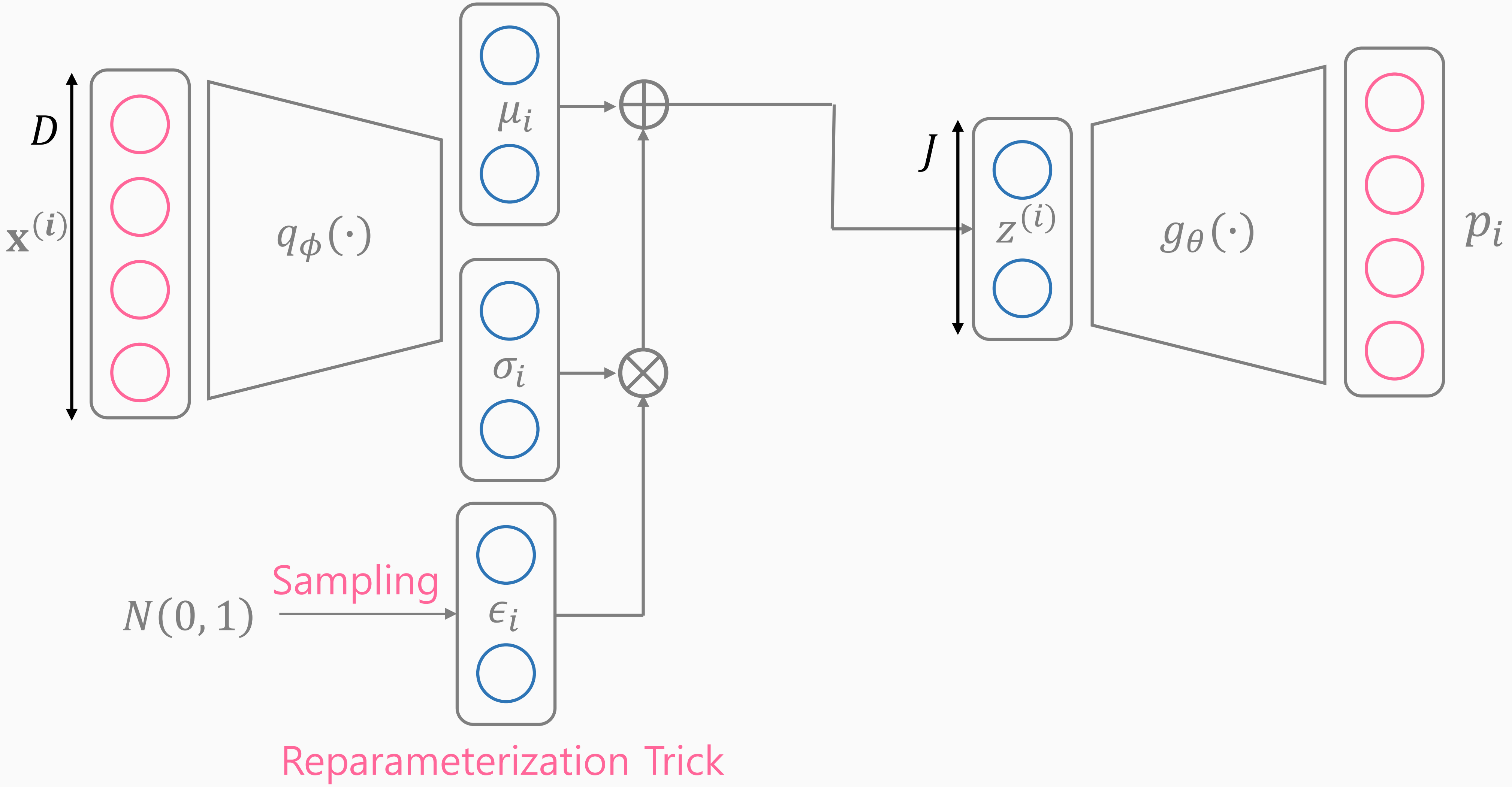
$$\log\left(p_\theta(\mathrm{x}^{(i)}|z^{(i)})\right) = \log\left(N(\mathrm{x}^{(i)};\ \mu_i, \sigma_i^2 I)\right)$$

$$= -\sum_{j=1}^{D}\frac{1}{2}\log(\sigma_{i,j}^2) + \frac{(\mathrm{x}^{(i,j)} - \mu_{i,j})^2}{2\sigma_{i,j}^2}$$

$$\log\left(p_\theta(\mathrm{x}^{(i)}|z^{(i)})\right) \propto -\sum_{j=1}^{D}(\mathrm{x}^{(i,j)} - \mu_{i,j})^2$$

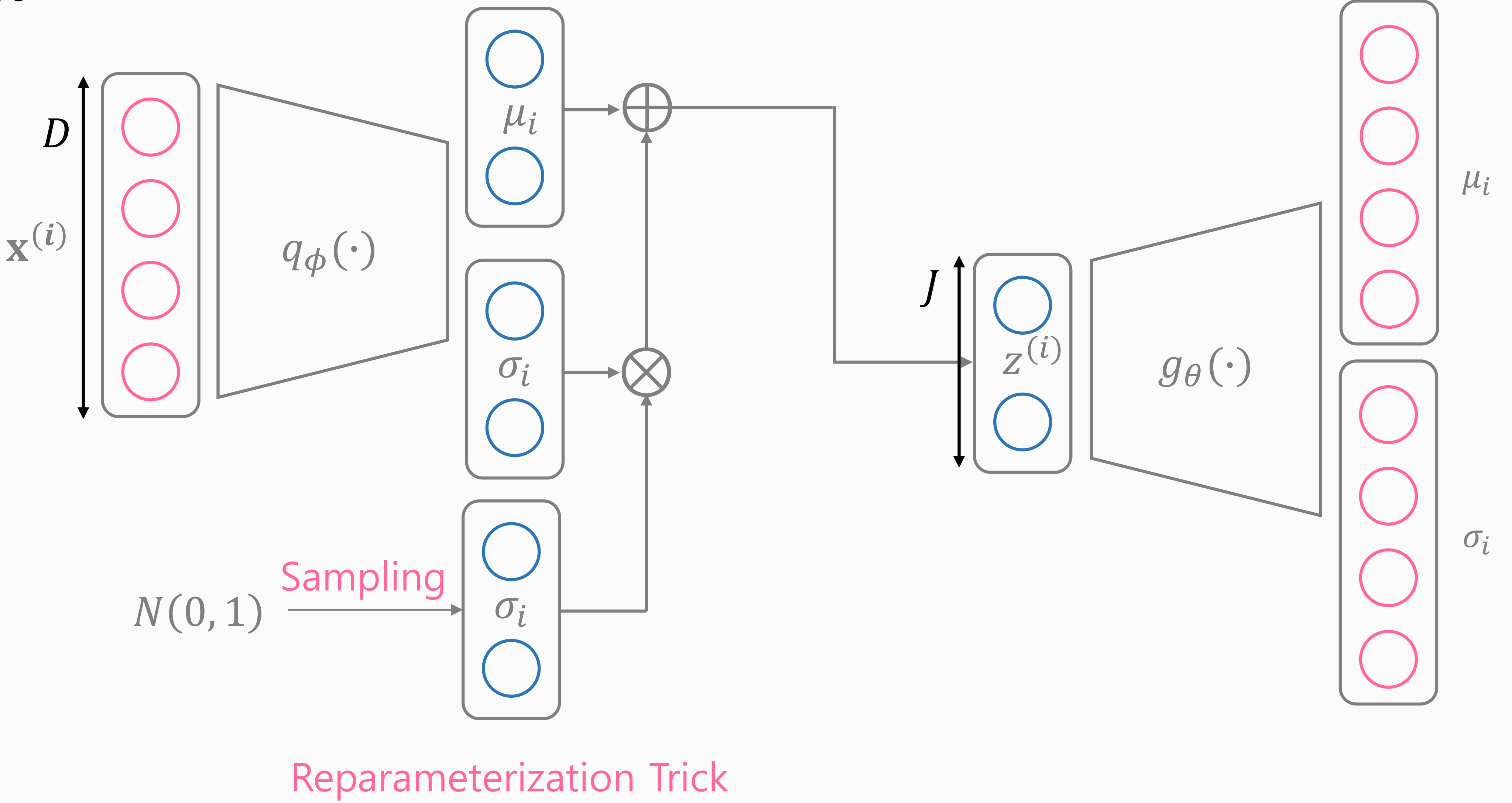
Structure 1 - Bernoulli

$$\underset{\boldsymbol{\phi},\boldsymbol{\theta}}{\textbf{argmin}}\ \frac{\mathbf{1}}{\boldsymbol{N}}\sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{N}}\left(\boldsymbol{L}\left(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x}^{(\boldsymbol{i})}\right)\right) = \frac{\mathbf{1}}{\boldsymbol{N}}\sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{N}}\left(\sum_{j=1}^{D}\mathrm{x}^{(i,j)}\log p_{i,j} + \left(1-\mathrm{x}^{(i,j)}\right)\log\left(1-p_{i,j}\right) + \frac{1}{2}\sum_{j=1}^{J}\left(\mu_{i,j}^2+\sigma_{i,j}^2-\ln\left(\sigma_{i,j}^2\right)-1\right)\right)$$



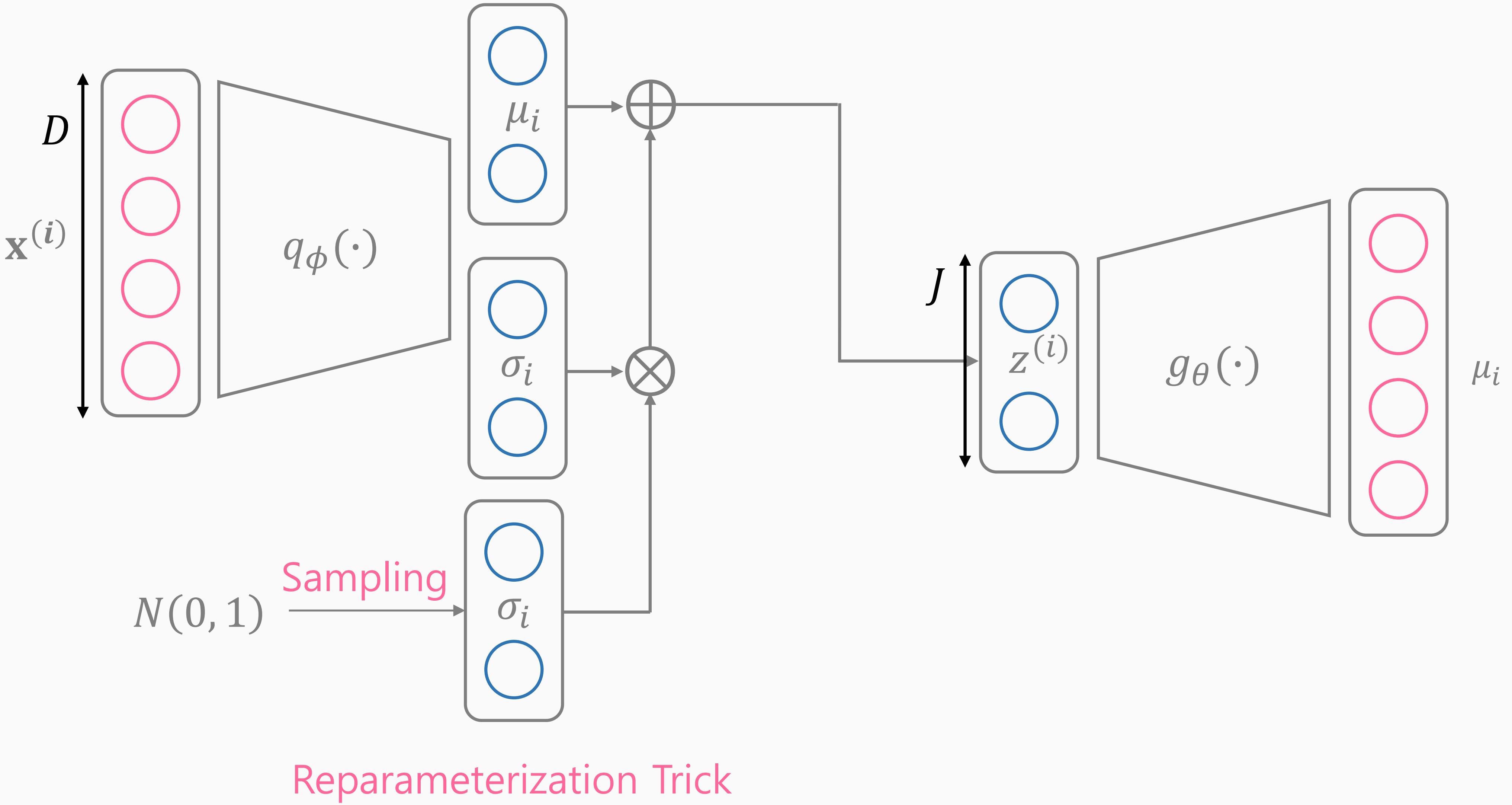
Structure 2 – Gaussian1

$$\underset{\boldsymbol{\phi},\boldsymbol{\theta}}{\text{argmin}}\ \frac{1}{N}\sum_{i=1}^{N}\left(L(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x}^{(i)})\right)=\frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=1}^{D}\frac{1}{2}\log\left(\sigma_{i,j}^{2}\right)+\frac{\left(\mathrm{x}^{(i,j)}-\mu_{i,j}\right)^{2}}{2\sigma_{i,j}^{2}}+\frac{1}{2}\sum_{j=1}^{J}\left(\mu_{i,j}^{2}+\sigma_{i,j}^{2}-\ln\left(\sigma_{i,j}^{2}\right)-1\right)\right)$$



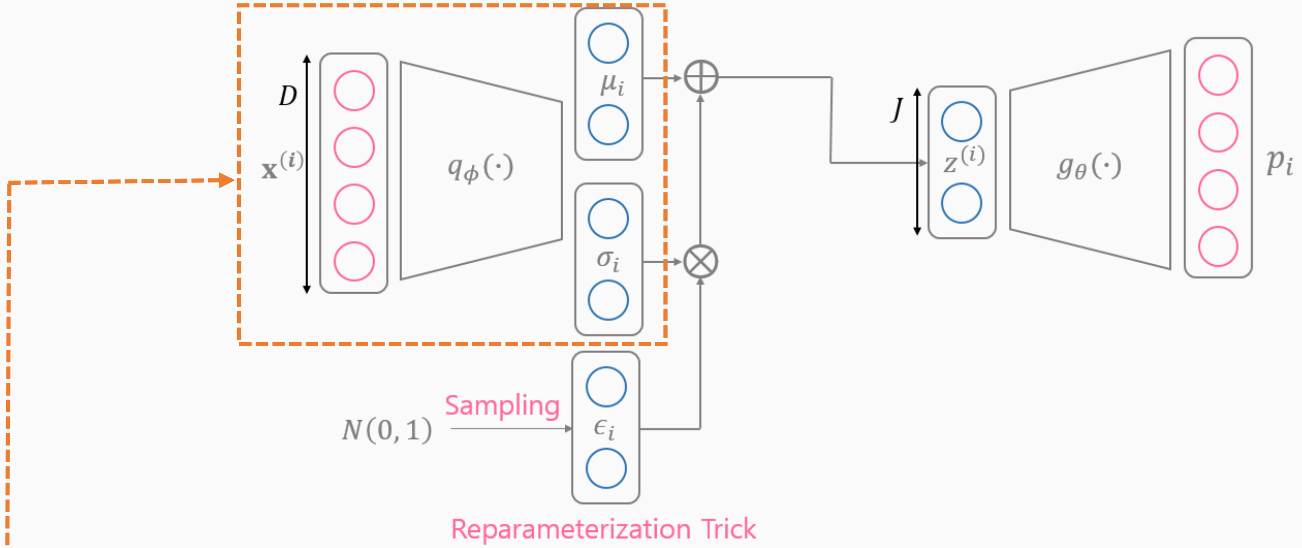
Structure 3 – Gaussian2

$$\underset{\boldsymbol{\phi},\boldsymbol{\theta}}{\text{argmin}}\ \frac{1}{N}\sum_{i=1}^{N}\left(L(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x}^{(i)})\right) = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=1}^{D}\left(\mathrm{x}^{(i,j)}-\mu_{i,j}\right)^2 + \frac{1}{2}\sum_{j=1}^{J}\left(\mu_{i,j}^2+\sigma_{i,j}^2-\ln\left(\sigma_{i,j}^2\right)-1\right)\right)$$



Tensorflow – VAE(1)

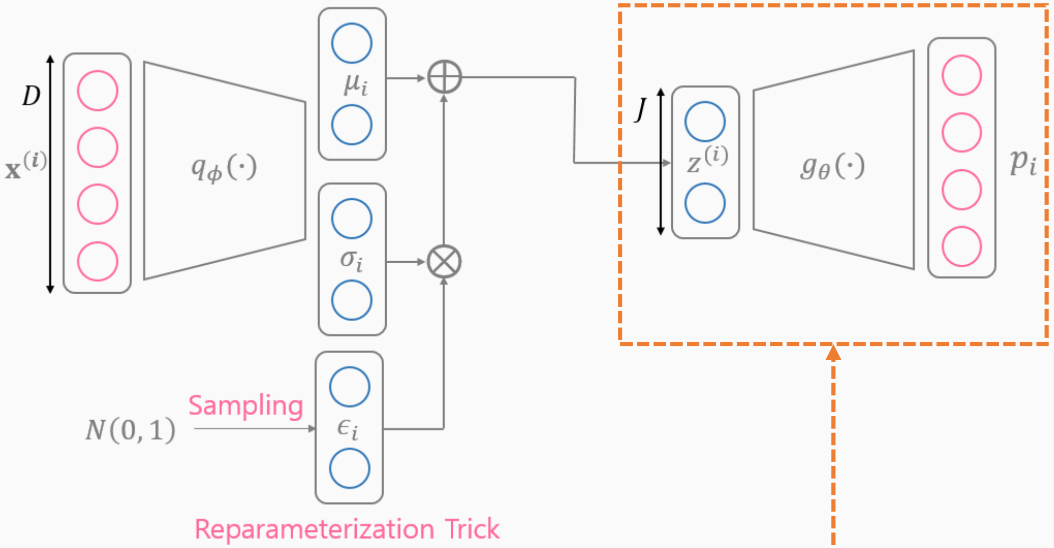
1. inference




```python
def inference(input_op, dim_z, reuse=False):
    with tf.variable_scope('inference', reuse=reuse):
        w_init = tf.contrib.layers.variance_scaling_initializer()
        b_init = tf.zeros_initializer()

        in_W1 = tf.get_variable(name='in_W1', shape=[784, 500], initializer=w_init)
        in_b1 = tf.get_variable(name='in_b1', shape=[500], initializer=b_init)
        in_h1 = tf.nn.sigmoid(tf.nn.bias_add(tf.matmul(input_op, in_W1), in_b1), name='in_h1')

        in_W2 = tf.get_variable(name='in_W2', shape=[500, 500], initializer=w_init)
        in_b2 = tf.get_variable(name='in_b2', shape=[500], initializer=b_init)
        in_h2 = tf.nn.sigmoid(tf.nn.bias_add(tf.matmul(in_h1, in_W2), in_b2), name='in_h2')

        in_W3 = tf.get_variable(name='in_W3', shape=[500, dim_z], initializer=w_init)
        in_b3 = tf.get_variable(name='in_b3', shape=[dim_z], initializer=b_init)
        z_mu = tf.nn.bias_add(tf.matmul(in_h2, in_W3), in_b3, name='z_mu')

        in_W4 = tf.get_variable(name='in_W4', shape=[500, dim_z], initializer=w_init)
        in_b4 = tf.get_variable(name='in_b4', shape=[dim_z], initializer=b_init)
        z_sigma = tf.nn.softplus(tf.nn.bias_add(tf.matmul(in_h2, in_W4), in_b4, name='z_sigma')) + EPSILON
    return z_mu, z_sigma
```

Tensorflow – VAE(2)

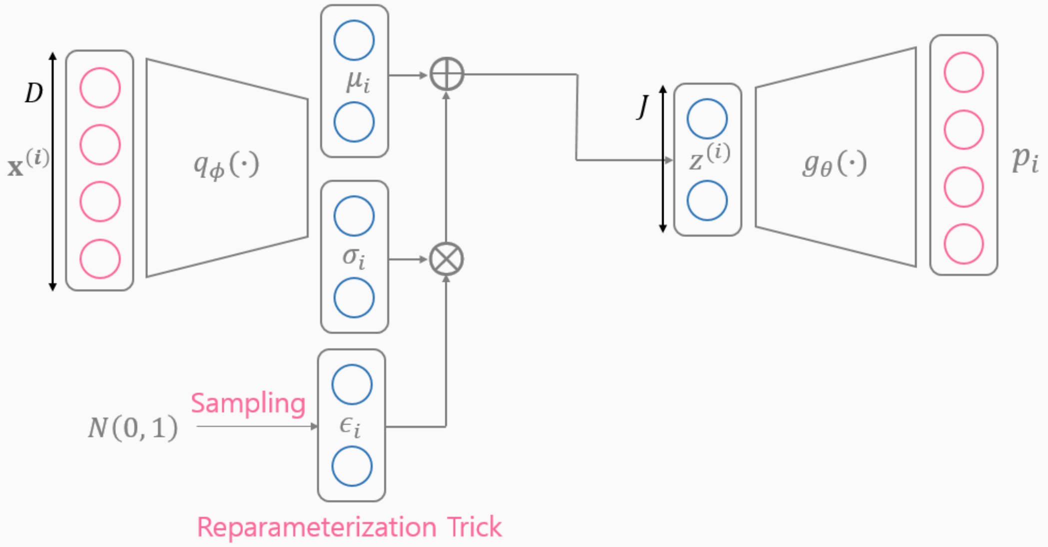
2. generator




```
def generator(z, dim_z, reuse=False):
    with tf.variable_scope("generator", reuse=reuse):
        w_init = tf.contrib.layers.variance_scaling_initializer()
        b_init = tf.zeros_initializer()

        g_W1 = tf.get_variable(name='g_W1', shape=[dim_z, 500], initializer=w_init)
        g_b1 = tf.get_variable(name='g_b1', shape=[500], initializer=b_init)
        g_h1 = tf.nn.sigmoid(tf.nn.bias_add(tf.matmul(z, g_W1), g_b1), name='g_h1')

        g_W2 = tf.get_variable(name='g_W2', shape=[500, 500], initializer=w_init)
        g_b2 = tf.get_variable(name='g_b2', shape=[500], initializer=b_init)
        g_h2 = tf.nn.sigmoid(tf.nn.bias_add(tf.matmul(g_h1, g_W2), g_b2), name='g_h2')

        g_W3 = tf.get_variable(name='g_W3', shape=[500, 784], initializer=w_init)
        g_b3 = tf.get_variable(name='g_b3', shape=[784], initializer=b_init)
        y = tf.sigmoid(tf.nn.bias_add(tf.matmul(g_h2, g_W3), g_b3), name='y')
    return y
```

Tensorflow – VAE(3)

3. _build_net



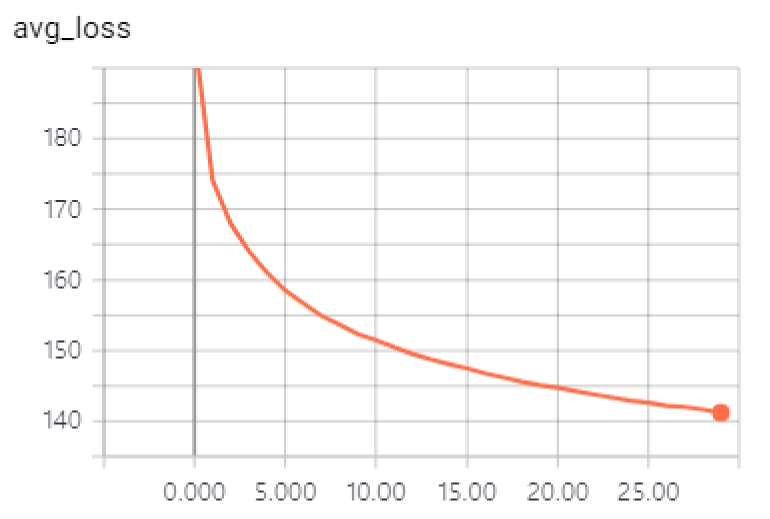

```
def _build_net(self):
    self.X = tf.placeholder(dtype=tf.float32, shape=[None, 784], name='X')
    self.z_in = tf.placeholder(tf.float32, shape=[None, self.dim_z], name='latent_variable')

    mu, sigma = inference(self.X, self.dim_z)
    self.z = mu+sigma*tf.random_normal(tf.shape(mu), 0, 1, dtype=tf.float32)
    self.y = generator(self.z, self.dim_z)
    self.output = tf.clip_by_value(self.y, EPSILON, 1-EPSILON)

    marginal_likelihood = tf.reduce_sum(self.X*tf.log(self.output)+(1-self.X)*tf.log(1-self.output), axis=1)
    KL_divergence = 0.5*tf.reduce_sum(1+tf.square(mu)+tf.square(sigma)-tf.log(tf.square(sigma)), axis=1)

    self.marginal_likelihood = tf.reduce_mean(marginal_likelihood)
    KL_divergence = tf.reduce_mean(KL_divergence)

    ELBO = self.marginal_likelihood - KL_divergence
    self.loss = -ELBO
    self.optim = tf.train.AdamOptimizer(0.001).minimize(self.loss)

    self.avg_loss = tf.placeholder(tf.float32)
    self.avg_loss_scalar = tf.summary.scalar('avg_loss', self.avg_loss)
```

reparameterization trick

Tensorflow – VAE(4)

4.result

```
>>> Start Train
Epoch : [   1/  30], cost : 194.301066, marginal_loss : 189.529322
Epoch : [   2/  30], cost : 174.079783, marginal_loss : 167.888371
Epoch : [   3/  30], cost : 167.898686, marginal_loss : 161.232583
Epoch : [   4/  30], cost : 164.027869, marginal_loss : 157.060536
Epoch : [   5/  30], cost : 161.095172, marginal_loss : 153.902977
Epoch : [   6/  30], cost : 158.553618, marginal_loss : 151.199170
Epoch : [   7/  30], cost : 156.734835, marginal_loss : 149.221140
Epoch : [   8/  30], cost : 154.952288, marginal_loss : 147.309565
Epoch : [   9/  30], cost : 153.715869, marginal_loss : 146.000891
Epoch : [  10/  30], cost : 152.372941, marginal_loss : 144.552083
Epoch : [  11/  30], cost : 151.515469, marginal_loss : 143.636646
Epoch : [  12/  30], cost : 150.451680, marginal_loss : 142.510940
Epoch : [  13/  30], cost : 149.507392, marginal_loss : 141.502457
Epoch : [  14/  30], cost : 148.735459, marginal_loss : 140.662199
Epoch : [  15/  30], cost : 148.086341, marginal_loss : 139.966753
Epoch : [  16/  30], cost : 147.483025, marginal_loss : 139.324004
Epoch : [  17/  30], cost : 146.752482, marginal_loss : 138.548264
Epoch : [  18/  30], cost : 146.202709, marginal_loss : 137.958504
Epoch : [  19/  30], cost : 145.596251, marginal_loss : 137.324189
Epoch : [  20/  30], cost : 145.088583, marginal_loss : 136.783128
Epoch : [  21/  30], cost : 144.735438, marginal_loss : 136.368010
Epoch : [  22/  30], cost : 144.253896, marginal_loss : 135.879318
Epoch : [  23/  30], cost : 143.803326, marginal_loss : 135.387492
Epoch : [  24/  30], cost : 143.371096, marginal_loss : 134.918396
Epoch : [  25/  30], cost : 142.929504, marginal_loss : 134.455940
Epoch : [  26/  30], cost : 142.609865, marginal_loss : 134.104701
Epoch : [  27/  30], cost : 142.218333, marginal_loss : 133.703764
Epoch : [  28/  30], cost : 142.027780, marginal_loss : 133.479317
Epoch : [  29/  30], cost : 141.657676, marginal_loss : 133.075992
Epoch : [  30/  30], cost : 141.207189, marginal_loss : 132.616276
```



Tensorflow – VAE(5)

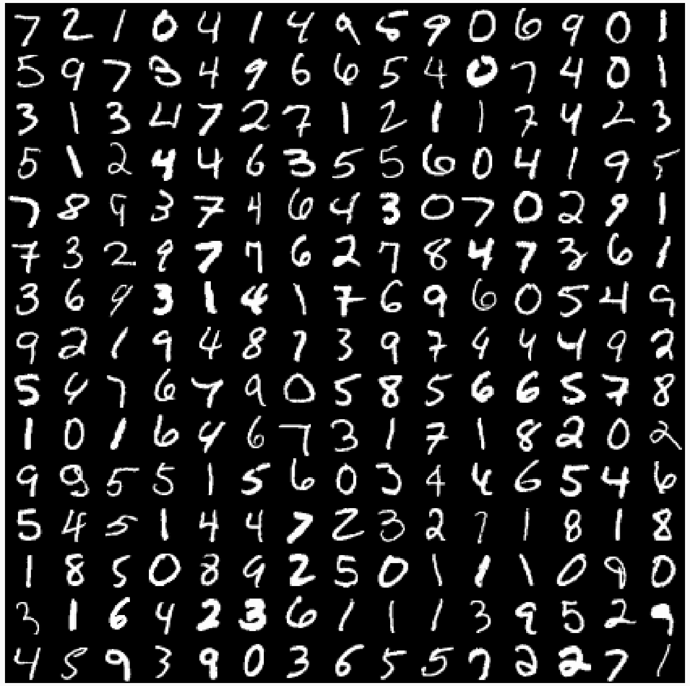
5. generated image plotting

dim_z=2로 설정하여
2차원 [-1.5, 1.5] × [-1.5, 1.5] domain의
각 점에 대하여 생성된 숫자 이미지 plot

```
x = np.linspace(-1.5, 1.5, 15)
y = np.linspace(-1.5, 1.5, 15)
cnt = 0
for i in x:
    for j in y:
        tmp = np.array([[i, j]])
        if cnt == 0:
            tmps = tmp
            cnt += 1
        else:
            tmps = np.append(tmps, tmp, axis=0)

pred = self.predict(tmps)
pred_im = plot_mnist(pred, 225, 3)
```

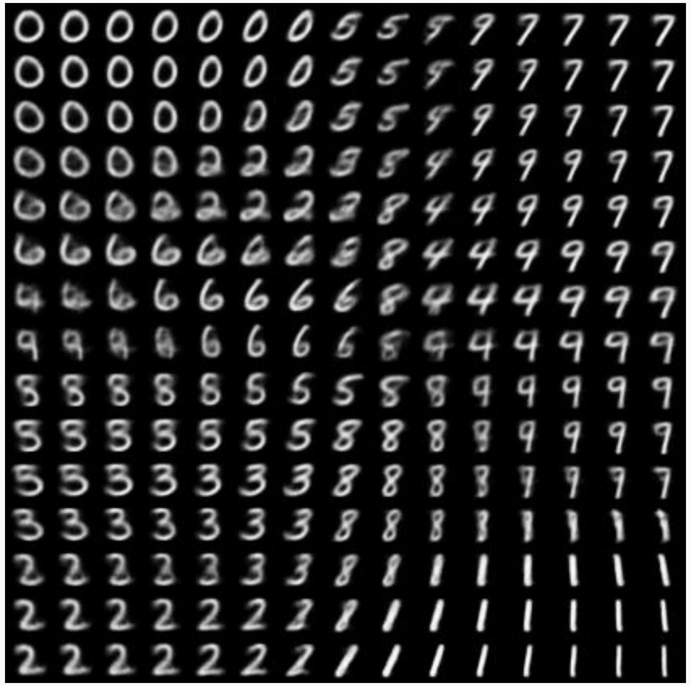
```
def predict(self, sample_z):
    y = generator(self.z_in, self.dim_z, reuse=True)
    return self.sess.run(y, feed_dict={self.z_in: sample_z})


def get_restruction(self, x_test):
    y = self.sess.run(self.y, feed_dict={self.X: x_test})
    return y
```

Tensorflow – VAE(6)



real image

recontructed image



generated image