11.12 세션 첨언 임선우

먼저, 오늘 나온 Autoencoder와 Variational Autoencoder의 차이를 짚고 넘어간다.

Autoencoder vs Variational Autoencoder

1) 용도의 차이

Autoencoder는 PCA에서와 같이 원래 input image X의 내재 Feature Z에 대한 이해가 목적인데 반해

VAE는 Generative Model의 종류이며 Input과 유사한 모조 이미지를 샘플링하는 것이 목적

2) Decoder

Autoencoder에서 Decoder는 X와 $\hat{X}$와의 l_2 loss를 계산하기 위해서 쓰고 학습이 끝난 후 drop

VAE에서는 $\hat{X}$를 샘플링하는 것이 주요 임무이므로 중요한 역할

1. KL Divergence

KL Divergence에 대해 이해하기 위해서는 Entropy , Cross Entropy 에 대한 이해가 필요하다!

Entropy, Cross Entropy는 정보이론에 근거한 개념인데, 정보이론은 정보를 송신자로부터 수신자에게 효율적으로 전송하기 위한 방법을 다룬다. 즉, 똑같은 양의 정보량이라면 더 적은 비용을 들이고자 한다.

(Claude Shannon의 A Mathematical Theory of Communication 논문에서 창시)

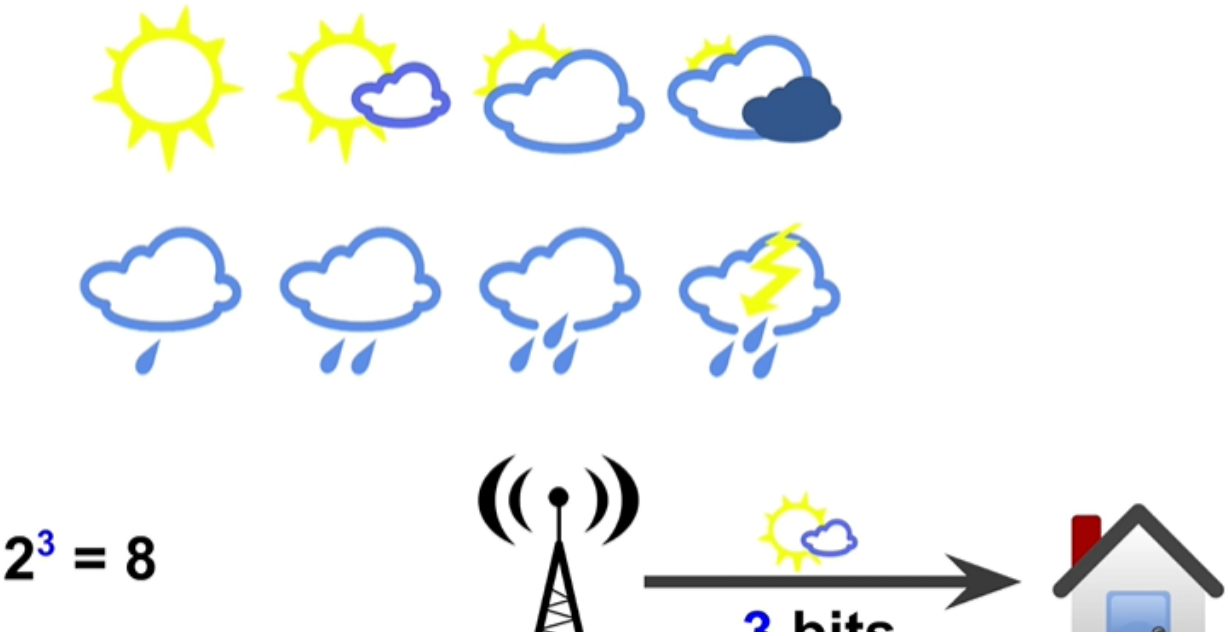
1.1 정보이론

정보의 기본 단위는 Bit (0/1)이다. 하지만 모든 정보가 유용한 것은 아니다 (Error / 중복정보).

Shannon은 소통을 할 때 있어서 유용한 정보만을 취급하고 싶었으므로 어떠한 새 정보가 원래의 불확실성을 2배로 낮추었을 때, 그 정보가 1 Bit의 정보량을 가진다고 한다.

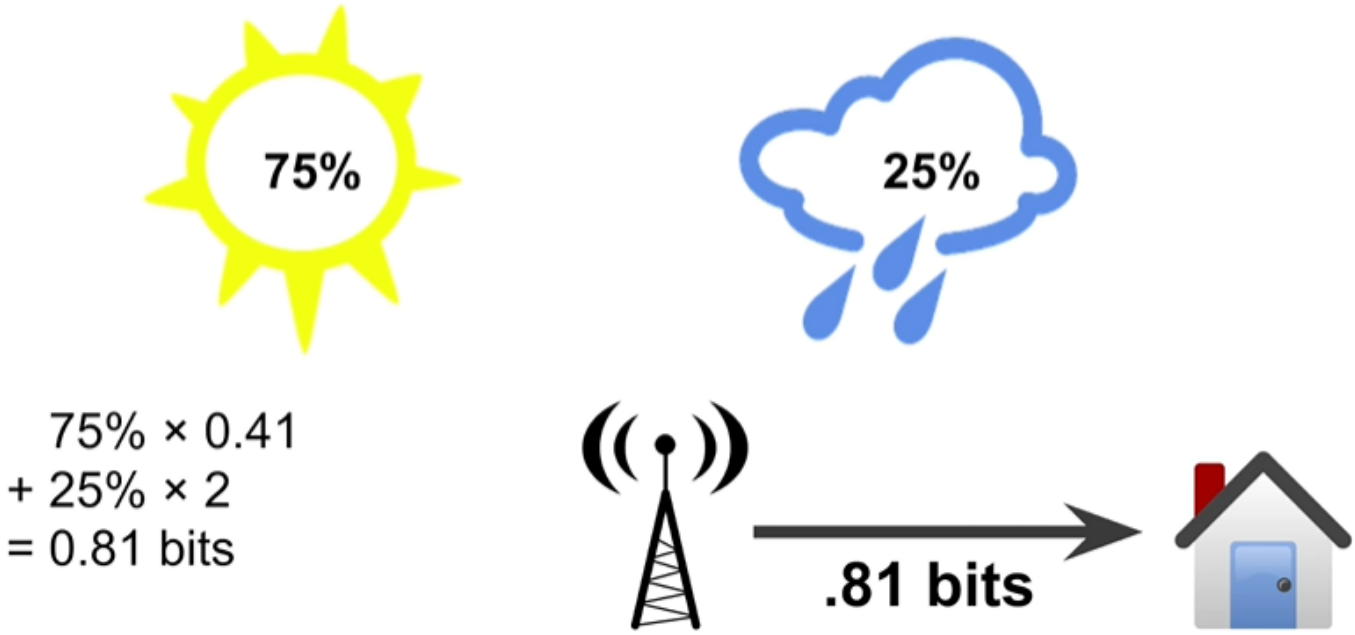
만약, 비가 올 확률 = 오지 않을 확률 = 0.5인 상황에서 기상청에서 비가 온다고 알려 주었을 때, 그 정보량은 1Bit라고 하는 것이다. "내일 비", "내일 비가 올 예정"이든 어떤 형식으로 정보를 송신하는지 무관히.

확장해서, 만약 8가지 날씨 유형이 모두 0.125의 확률질량을 갖는다고 할 때, 기상청에서 내일 맑다가 구름이라고 말을 했을 때, 3Bit의 정보를 송신한 것이다 : **2진법**



1.2 Entropy (정보이론에서)

그러면 다음과 같은 상황에서 기상청이 주는 정보량의 (가중)평균은?



" $Entropy$ " $H(x) = -\sum_x p(x)logp(x)$: 이산형 확률변수 X

" $Entropy$ " $H(x) = -\int_x f(x)logf(x)dx$: 연속형 확률변수 X

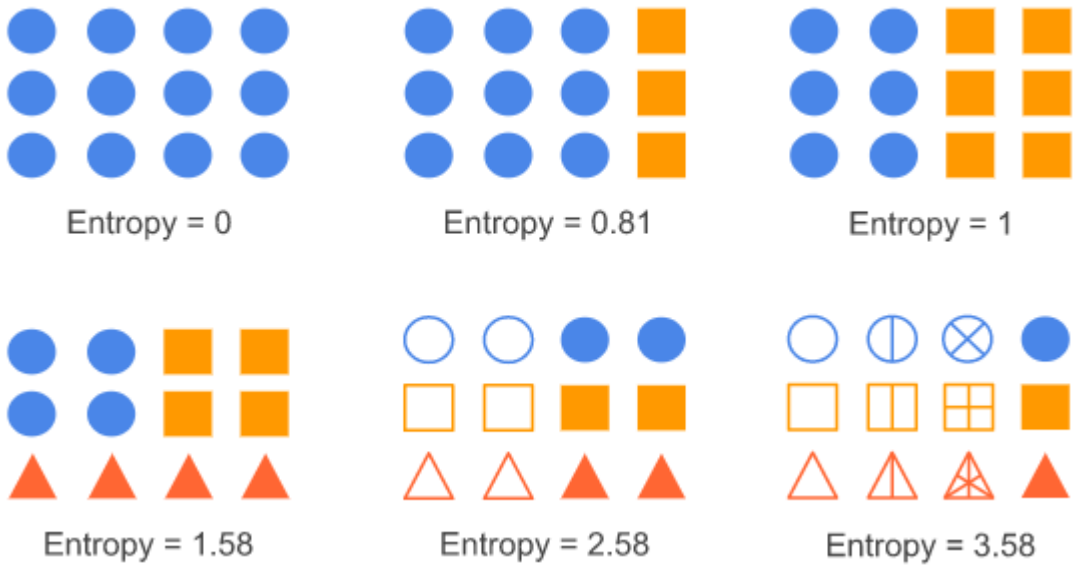
log의 밑으로는 상황에 따라 2, 10, e등을 쓴다

다른 관점, 사실 똑같은 의미

1) 송신자가 보내는 **유용한** 정보량의 가중평균

2) 확률변수가 취할 수 있는 값들을 관찰했을 때 **놀라는 정도의 가중평균**

3) 어떤 확률변수 혹은 그룹에서의 **불순도**



Details)

1) X가 Uniform 분포를 따를 때 Entropy가 가장 크다. Degenerate 분포를 따를 때 Entropy가 가장 작다.

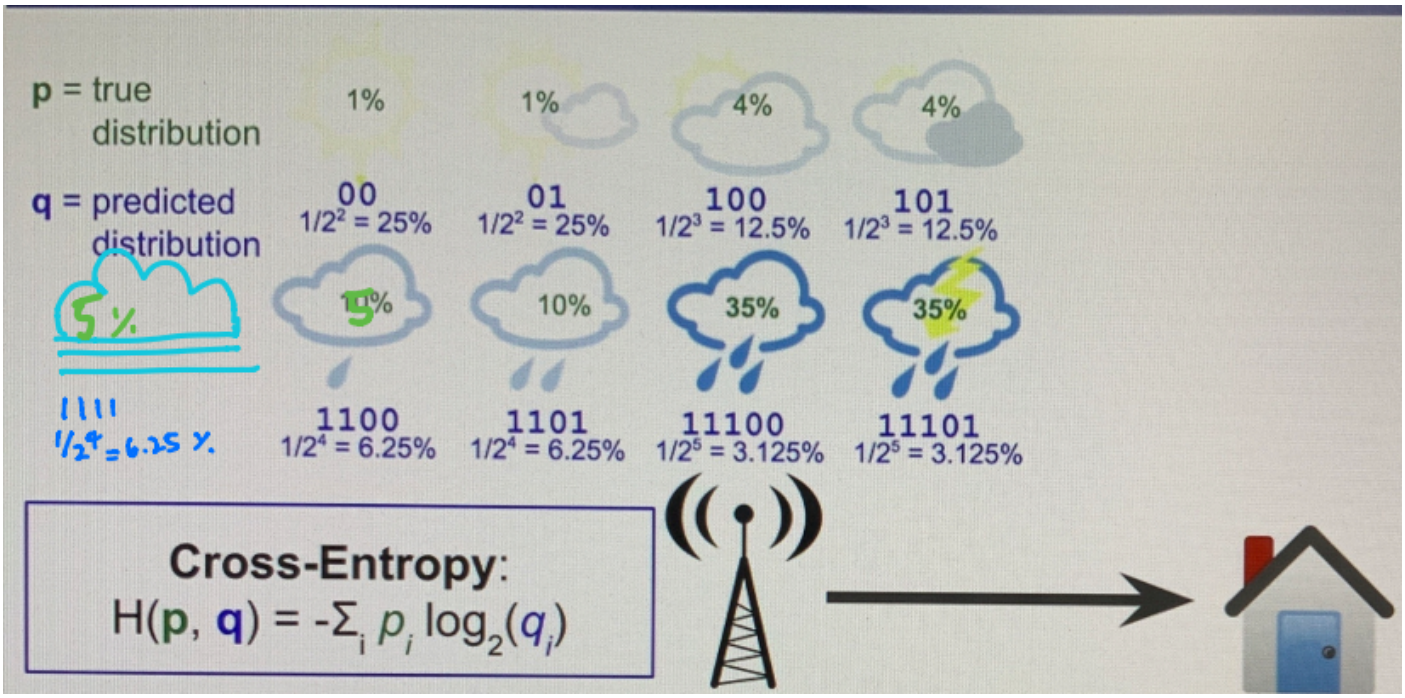
요르단의 Amman이라는 곳은 300일 이상 맑다고 하는데, 이 곳 사람은 일기예보를 볼 필요가 적을 것이다.

2) Decision Tree에서 Node Split의 기준을 노드 분리 전후의 Entropy 변화량을 토대로 하기도 한다.

1.3 Cross Entropy

질문 : 각 날씨 상태마다 얼마나 긴 메시지 를 보내고 있는가?

질문 : 그 가중평균은? 이번 질문에 대한 답이 Cross Entropy이다.



일반화하여 p는 참 분포, q는 예측 / 학습한 분포라 할 때,

" $CrossEntropy$ " $H(p,q) = -\sum_x p(x)logq(x)$: 이산형

" $CrossEntropy$ " $H(p,q) = -\int_x p(x)logq(x)dx$: 연속형

만약 필요한 정보량만큼의 message length를 가진다면 $H(p,q) = H(p,p) = H(p)$

즉, Cross Entropy는 Entropy값까지 낮아질 때 최적화된 것이다.

번외 : Binary Classification에서의 Cross Entropy = $-y_i log\hat{y}_i - (1-y_i)log(1-\hat{y}_i)$

$Cost = \sum[-y_i log\hat{y}_i - (1-y_i)log(1-\hat{y}_i)]$

익숙한 식이다.

1.4 KL Divergence = Relative Entropy

직관적 : 보내야 하는 메시지 길이 분포와 비교해서 얼마나 비효율적으로 메시지를 전달하고 있는가?

Relative Entropy or K-L Divergence

- Additional information required as a result of using $q(\mathrm{x})$ in place of $p(\mathrm{x})$

$$KL(p\,\|\,q) = -\int p(x)\ln q(x)dx - \left(\int p(x)\ln p(x)dx\right)$$

$$= -\int p(x)\ln\left\{\frac{p(x)}{q(x)}\right\}dx$$

$KL(p||q) = H(p,q) - H(p) = H(p,q) - H(p,p)$

여기서 p는 참 분포, q는 예측 / 학습한 분포

1.5 KL Divergence와 거리

직관적인 이해를 위해서는 두 분포 간의 거리라고 이해해도 좋다. 그러나,

Def) Metric

A *metric* on a set X is a function $d : X \times X \to R+$ if

1. $d(x,y) \geq 0, d(x,y) = 0 \leftrightarrow x = y$: Non negativity
2. $d(x,y) = d(y,x)$: symmetry (x에서 y의 거리 = y에서 x에서의 거리)
3. $d(x,y) \leq d(x,z) + d(z,y)$: Triangle Inequality (거쳐서 가면 멀어진다)

이 Metric은 우리한테 익숙한 두 벡터 간의 거리 뿐 아니라 분포 간의 거리도 논할 수 있게 한다.

조건 1)은 참. 조건 1)은 ELBO를 구할 때 쓰였다.

증명)

$KL(p||q) = -\int p(x) log \frac{q(x)}{p(x)} dx \geq -log \int p(x) \frac{q(x)}{p(x)} = 0$: $-log(\cdot)$는 $R++$에서 convex function이라는 사실을 통해 Jensen's Inequality 적용

그런데 2,3번 조건이 성립하지 않는다!

1.6 KL Divergence 최소화 $\leftrightarrow$ likelihood maximization

Metric의 기본적인 요소를 충족하지 못하지만 KL Divergence는 Likelihood Maximization과 equivalent라는 훌륭한 성질이 있다.

$p(x)$는 미지의 분포이며 $q(x|\theta)$ 로 근사한다고 하자. 이 때 KL Divergence

$$KL(p(x)||q(x|\theta) = E_p[log\frac{p(x)}{q(x_i|\theta)}] \approx \frac{1}{N}\sum_{i=1}^{N}[logp(X_i) - log(q(x_i|\theta)]$$

마지막 근사는 sample mean. 이 때, log(p)는 θ 와 무관하므로 뒤의 -log(q)의 크기에 따라 KL divergence가 근사적으로 변하게 된다. 즉, q분포에서 θ의 likelihood를 최대화하는 것과 KL Divergence를 최소화하는 것이 같은 문제가 된다.

무엇의 KL divergence? intractible $p(x)$와 그 근사인 tractible $q(x|\theta)$ 간의!

citing :

https://www.techleer.com/articles/496-a-short-introduction-to-entropy-cross-entropy-and-kl-divergence-aurelien-geron/ (https://www.techleer.com/articles/496-a-short-introduction-to-entropy-cross-entropy-and-kl-divergence-aurelien-geron/) : 핸즈온 머신러닝 저자의 영상

https://hyunw.kim/blog/2017/10/27/KL_divergence.html (https://hyunw.kim/blog/2017/10/27/KL_divergence.html)

https://allmodelsarewrong.github.io/tree-impurities.html (https://allmodelsarewrong.github.io/tree-impurities.html)

2. 논문 첨언

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114. : Autoencoder

https://arxiv.org/pdf/1312.6114.pdf (https://arxiv.org/pdf/1312.6114.pdf)

주제 : Latent Variable이 연속형일 때 Approximate Density에서 모수 추정 방식

방법론 : Variational Lower bound (=ELBO)의 Reparameterization 을 통해서 일반적인 Stochastic gradeint Ascent 방식으로 모수 추정 가능. 그 방식을 Stochastic Gradient Variational Bayes라고 함.

상세 :

1) 문제 세팅 : 동일한 상황

$X = \{X_i\}_{i=1}^N$: N개의 i.i.d sample.

X는 latent variable Z에 의해 생성되었다고 가정

1) *Prior* $p_\theta * (z)$: Uncorrelated Gaussian Density

2) *Encoder Network* $q_\theta(z|x)$: x를 z로 encode.

Posterior $Z|X$: Uncorrelated Gaussian Density 가정.

여기서 z값들을 Sample!!!

3) *Conditional Density* $p_\theta * (x|z_i)$: 복잡한 분포 가정.

여기서 x|z 값들을 Sample!!!

4) $p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz$: Likelihood (Intractable)

5) Intractable likelihood의 lower bound를 구해서 그 lower bound (ELBO = Variational lower bound)를 최대화

The marginal likelihood is composed of a sum over the marginal likelihoods of individual datapoints $\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(N)}) = \sum_{i=1}^{N} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})$, which can each be rewritten as:

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) = D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}^{(i)})) + \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) \quad (1)$$

The first RHS term is the KL divergence of the approximate from the true posterior. Since this KL-divergence is non-negative, the second RHS term $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)})$ is called the (variational) *lower bound* on the marginal likelihood of datapoint i, and can be written as:

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) \geq \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[-\log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) + \log p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})\right] \quad (2)$$

which can also be written as:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) = -D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z})) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z})\right] \quad (3)$$

2) 문제점 :

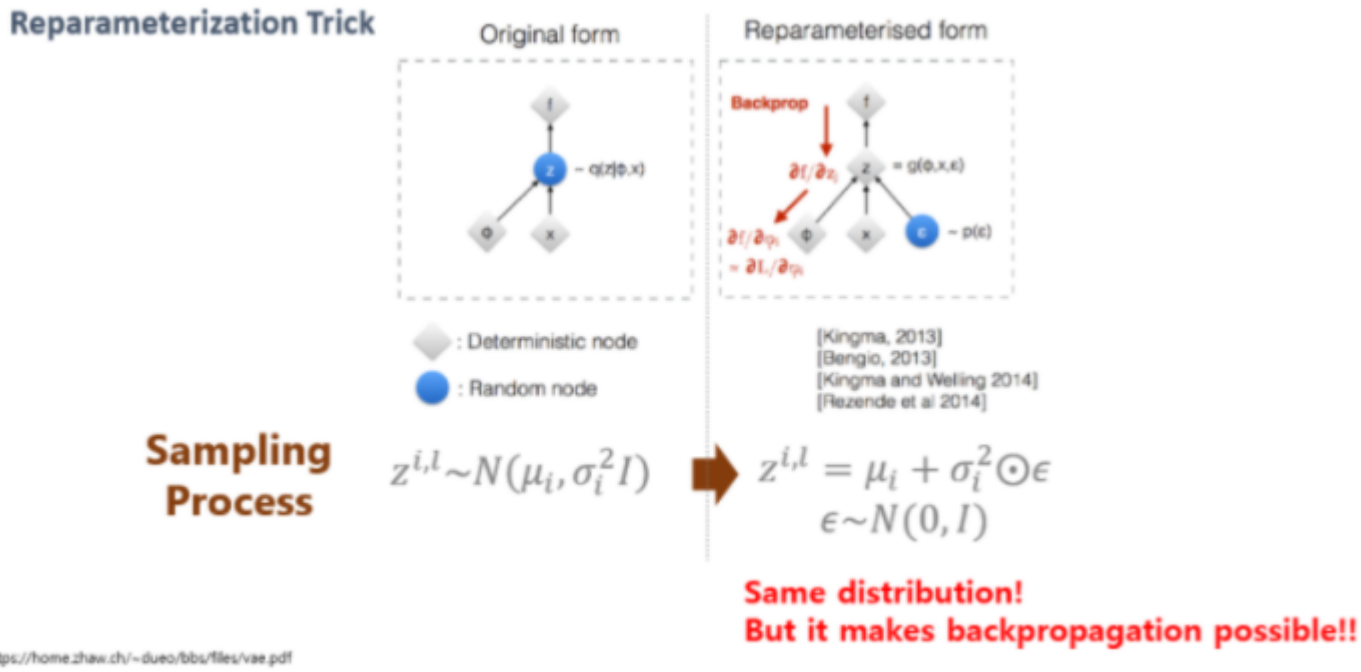
1) ELBO를 ϕ, θ에 관해서 미분을 해야 하는데, ϕ에 관해 미분을 할 때 (gradient구하기) 그 Gradient estimator의 분산이 크다는 문제가 있다.

2) 또한, VAE가 하고자 하는 것은 p(model)에서 새 데이터 생성이다.

Decoder 과정에서 똑같은 분포에서 샘플링을 해야 하는데, decoder에 Random Variable z가 있다.

Sampling 과정은 미분이 불가능하기 때문에 Backpropagation이 불가능하다. 즉, parameter estimation이 불가능!

3) 해결책 :



Decoder에 latent variable z가 있다. 이 때, z는 확률변수이다. 이 z를 랜덤성이 나오는 quantity ϵ과 랜덤성이 없는 μ, σ로 쪼개 자기 자신은 deterministic인 것처럼 parameter trick을 가한다.

As explained in section 2.4, we sample from the posterior $\mathbf{z}^{(i,l)} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})$ using $\mathbf{z}^{(i,l)} = g_{\boldsymbol{\phi}}(\mathbf{x}^{(i)}, \boldsymbol{\epsilon}^{(l)}) = \boldsymbol{\mu}^{(i)} + \boldsymbol{\sigma}^{(i)} \odot \boldsymbol{\epsilon}^{(l)}$ where $\boldsymbol{\epsilon}^{(l)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. With $\odot$ we signify an element-wise product. In this model both $p_{\boldsymbol{\theta}}(\mathbf{z})$ (the prior) and $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ are Gaussian; in this case, we can use the estimator of eq. (7) where the KL divergence can be computed and differentiated without estimation (see appendix B). The resulting estimator for this model and datapoint $\mathbf{x}^{(i)}$ is:

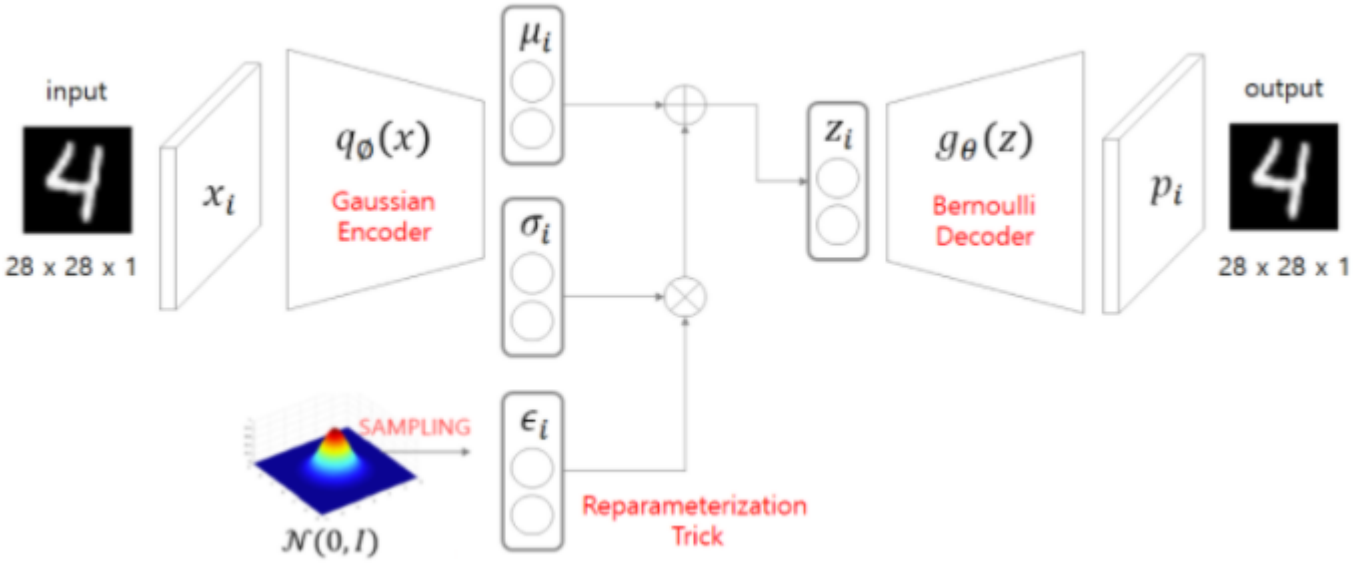
$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) \simeq \frac{1}{2} \sum_{j=1}^{J} \left(1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2\right) + \frac{1}{L} \sum_{l=1}^{L} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)})$$

$$\text{where} \quad \mathbf{z}^{(i,l)} = \boldsymbol{\mu}^{(i)} + \boldsymbol{\sigma}^{(i)} \odot \boldsymbol{\epsilon}^{(l)} \quad \text{and} \quad \boldsymbol{\epsilon}^{(l)} \sim \mathcal{N}(0, \mathbf{I}) \tag{10}$$

여기서 새로 나온 연산기호는 원소별곱 (Hadamard Product)를 말한다.

https://en.wikipedia.org/wiki/Hadamard_product_(matrices (https://en.wikipedia.org/wiki/Hadamard_product_(matrices))

4) 전체적인 과정



Citing :

https://jamiekang.github.io/2017/05/21/auto-encoding-variational-bayes/ (https://jamiekang.github.io/2017/05/21/auto-encoding-variational-bayes/)

https://taeu.github.io/paper/deeplearning-paper-vae/ (https://taeu.github.io/paper/deeplearning-paper-vae/)

https://en.wikipedia.org/wiki/Hadamard_product_(matrices (https://en.wikipedia.org/wiki/Hadamard_product_(matrices))