

Machine Learning

# 11장 Autoencoder와 Variational Autoencoder

---

고려대학교 통계학과  
박유성



# Contents

- 01    Autoencoder모형**
- 02    오염제거(denoising) autoencoder**
- 03    Variational autoencoder (VAE)**

# Autoencoder와 Variational Autoencoder

- 고차원 입력자료를 정보의 손실없이 저차원의 잠재변수(latent variable)로 전환하는 것은 차원축소라는 주제의 전통적인 통계학 분야이다.
- 입력자료를 저차원으로 축약한 잠재변수로부터 재생한 입력자료가 원래의 입력자료를 그대로 구현한다는 의미는 딥러닝에서 정보의 손실이 없다는 의미이다.
- 아래 그림은 Autoencoder의 개념도이다.



# Autoencoder와 Variational Autoencoder

- 입력변수  $x$ 를 encoder  $f$ 에 적용하여 잠재변수  $z$ 를 출력하고 이를 다시 encoder를 이용하여  $\tilde{x}$ 를 출력하고 있다.
- 그러므로 autoencoder의 성능은  $x$ 와  $\tilde{x}$ 가 얼마나 가까운가로 결정된다.
- 당연히  $x$ 와  $\tilde{x}$ 는 동일한 크기의 텐서여야 한다.
- 예를 들어  $x$ 가  $28 \times 28 \times 1$ 인 흑백이미지 이면  $\tilde{x}$ 도 동일한 크기의 3D텐서여야 한다. 그러면 손실함수는

$$\ell(x, \tilde{x}) = \sum_{j=1}^{28 \times 28 \times 1} (x_j - \tilde{x}_j)^2$$

여기에서  $x_j$ 와  $\tilde{x}_j$ 는 각각의  $j$ 번째 픽셀을 의미한다.

# Autoencoder와 Variational Autoencoder

- 확률적 접근법으로 autoencoder를 설명하면 다음과 같다.
- $p(x)$ 를  $x$ 의 분포함수라고 할 때,  $z$ 의 조건부분포  $p(z|x)$ 로부터 잠재변수  $z$ 를 추출하고
- $z$ 가 주어졌다는 조건에서 조건부분포  $p(\tilde{x}|z)$ 로부터  $\tilde{x}$ 를 추출하여 이  $\tilde{x}$ 로  $p(x)$ 의  $x$ 를 구현하는 것이 autoencoder이며 목적이다.
- 그러므로  $p(z|x)$ 가 encoder이고  $p(\tilde{x}|z)$ 가 decoder가 된다.
- $p(\tilde{x}|z)$ 로부터  $\tilde{x}$ 가  $x$ 와 매우 가까우면 잠재변수  $z$ 는 차원이 축소되었는데도 불구하고 입력변수  $x$ 가 가지고 있는 정보를 거의 다 가지고 있는 좋은 잠재변수라고 할 수 있다
- Autoencoder는 목적변수가 존재하지 않으므로 비지도학습(unsupervised learning)의 영역에 해당된다.

# Autoencoder와 Variational Autoencoder

- Autoencoder의 대표적인 응용은 차원축소 이외에도 잡음제거(denoising), 색채화(colorization) 등이 있다.
- Variational autoencoder (VAE)는 autoencoder와 개념적으로 동일하지만 가장 큰 차이점은 인간의 창조적 특성을 반영하려는 시도이다.
- autoencoder는 입력변수  $x$ 를 가능한 한 그대로 재현하려는데 목적이 있지만 VAE는 입력변수  $x$ 로부터 잠재된 특성을 모형화한 후,
- 이를 이용하여 입력변수  $x$ 와 다르지만 동일한 특성을 가진 새로운 입력변수  $x$ 를 만들어 내는데 목적이 있다.

# Autoencoder와 Variational Autoencoder

- 입력자료의 재생(reconstruction)은 autoencoder와 VAE 뿐만 아니라 다음 장에서 다룰 GAN (generative adversarial networks)의 주제이다.
- 입력자료가 이미지 자료일 경우, 이미지의 재생을 위해 upsampling의 개념을 먼저 이해하여야 한다.
- CNN 아키텍처는 모형의 성능을 높이기 위해 다양성에 해당하는 channel 수를 늘리는 대신에 특성변수에 해당하는 픽셀의 개수를 strides나 pooling을 통해 줄이는 과정을 반복한다.
- 예를 들어 encoder에서 strides=2를 사용하여 픽셀의 크기를 1/2 로 줄였다면 decoder에서는 원래의 이미지를 재생하여야 하기 때문에 픽셀의 크기를 2배로 증가시켜야 한다. 이를 upsampling이라고 한다.

# Autoencoder와 Variational Autoencoder

- 여기에서 논의하는 upsampling은 제 7장에서 논의한 단순 upsampling인 UpSampling2D와는 다르며 다음의 간단한 예제를 통해 upsampling을 이해하도록 하자.
- 아래 그림은  $3 \times 3$  픽셀을  $(2,2)$  커널을 적용하여  $2 \times 2$  픽셀로 전환하고 있다.

1	2	3
4	5	6
7	8	9

 $\otimes$ 

2	1
4	3

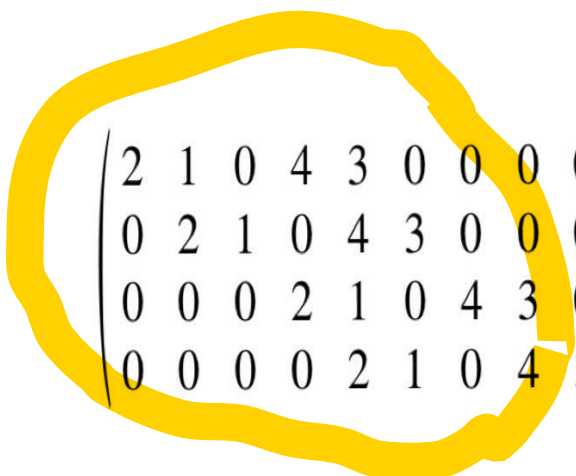
 $=$ 

35	45
65	75



# Autoencoder와 Variational Autoencoder

- 앞의 convolution계산을 다음과 같이 행렬연산으로 전환할 수 있다.


$$\begin{pmatrix} 2 & 1 & 0 & 4 & 3 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 4 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 1 & 0 & 4 & 3 & 0 \\ 0 & 0 & 0 & 0 & 2 & 1 & 0 & 4 & 3 \end{pmatrix} \times \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{pmatrix} = \begin{pmatrix} 35 \\ 45 \\ 67 \\ 75 \end{pmatrix} \longrightarrow \text{reshape } (2,2) = \begin{array}{|c|c|} \hline 35 & 45 \\ \hline 65 & 75 \\ \hline \end{array}$$

- 이제 반대로  $2 \times 2$  픽셀을  $3 \times 3$  픽셀 데이터로 전환하여 보자
- 위의  $4 \times 9$  행렬의 전치행렬을 이용하여 다음과 같이 전환한다.

# Autoencoder와 Variational Autoencoder

$$\begin{pmatrix} 2 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 4 & 0 & 2 & 0 \\ 3 & 4 & 1 & 2 \\ 0 & 3 & 0 & 1 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 3 & 4 \\ 0 & 0 & 0 & 3 \end{pmatrix} \times \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} = \begin{pmatrix} 2 \\ 5 \\ 2 \\ 10 \\ 22 \\ 10 \\ 12 \\ 25 \\ 12 \end{pmatrix} \longrightarrow \text{reshape (3,3)} =$$

2	5	2
10	22	10
12	25	12

1	2
3	4

에 (2,2) 커널을 곱하여

2	5	2
10	22	10
12	25	12

으로 upsampling 하고 있다.

- 원래의 (2,2) kernel 행렬의 전치행렬을 이용하였기 때문에 이를 **2D Transpose convolution**이라고 한다.

# Autoencoder와 Variational Autoencoder

- 다음은 stride=2인 convolution을 고려하여 보자.

$$\begin{array}{|c|c|c|c|} \hline 1 & 1 & 2 & 2 \\ \hline 1 & 1 & 2 & 2 \\ \hline 3 & 3 & 4 & 4 \\ \hline 3 & 3 & 4 & 4 \\ \hline \end{array} \otimes \begin{array}{|c|c|} \hline 1 & 2 \\ \hline 3 & 4 \\ \hline \end{array} = \begin{array}{|c|c|} \hline 10 & 20 \\ \hline 30 & 40 \\ \hline \end{array}$$

- 2D Transpose convolution을 적용하기 위해 (2,2) kernel을  $4 \times 16$  행렬로 전환하면 다음과 같고 이를 전치하여  $2 \times 2$  픽셀에 곱하면 픽셀의 크기가 2배인  $4 \times 4$  픽셀이 된다.

$$\begin{array}{|c|c|} \hline 1 & 2 \\ \hline 3 & 4 \\ \hline \end{array} \longrightarrow \begin{pmatrix} 1 & 2 & 0 & 0 & 3 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 0 & 0 & 3 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 3 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 3 & 4 \end{pmatrix}$$

# 01 Autoencoder 모형

- Autoencoder는 입력자료의 잠재변수로의 축약을 위한 encoder와 잠재변수를 입력자료로 재생하기 위한 decoder로 구성되어 있다.
- 손실함수는 입력자료와 입력자료→encoder→decoder를 걸쳐 재생된 입력자료의 차이를 측정한다.
- 다음의 예제는 이미 여러 번 사용해 왔던 0~9의 이미지자료인 MNIST 자료를 autoencoder 모형에 적용한 프로그램이다.

## 02 오염제거(denoising) autoencoder

- 오염제거 autoencoder는 오염된 입력자료를 오염되지 않은 원래의 자료로 재생하는 autoencoder모형이다.
- 기존의 autoencoder와 비교하여 입력자료를 오염된 입력자료로 대체한 것을 제외하고 거의 동일하다.

## 03 Variational Autoencoder (VAE)

- VAE의 목적은 입력변수  $x$ 의 분포와 이 분포에 포함된 모수를 동시에 추정하는데 있다.
- 여기에서 모수의 역할은 모수가 주어졌을 때, 보다 쉽게 입력변수  $x$ 를 재생할 뿐만 아니라 연속적인 입력변수  $x$ 를 생성하기 위함이다.
- 예를 들어, 입력변수의 분포가 정규분포를 따를 때 평균과 분산만 알면 표본으로 관측된 입력변수 값을 쉽게 재생할 수 있고 정규분포의 분위수별로 입력변수를 생성할 수 있다.
- 여성의 미소가 입력변수이면 입력변수의 분포를 추정하고 이 분포의 분위수별로 입력변수를 생성하여 점차적으로 변하는 여성의 미소를 만들어 낼 수 있을 것이다.

## 03 Variational Autoencoder (VAE)

- 입력변수  $x$ 의 분포를  $p_{\theta}(x)$ 로 표현할 때 이를 직접 추정하는 것은 매우 어렵거나 불가능한 경우가 대부분이다.
- VAE에서는 잠재변수  $z$ 를 모수  $\theta$ 를 통해 생성해야 한다. 이를 위해  $p_{\theta}(z|x)$ 의 근사분포로

$$q_{\phi}(z|x) = N[\mu(x), \text{diag}(\sigma^2(x))]$$

- 여기에서  $x$ 는  $k \times 1$  벡터,  $z$ 는  $p \times 1$  벡터이다.  $z$ 의 공분산이 0이므로  $p$ 개의  $z$ 는 서로 간에 독립이며 평균과 분산은 입력변수  $x$ 의 함수이다.
- 즉, 추정된  $\mu(x)$ 와  $\sigma^2(x)$ 를 이용하여 잠재변수  $z$ 를 생성하게 된다.
- 그러므로 VAE의 성능은  $q_{\phi}(z|x)$ 가  $p_{\theta}(z|x)$ 를 얼마나 잘 근사하느냐에 달려있다.

## 03 Variational Autoencoder (VAE)

- $q_{\phi}(z|x)$ 와  $p_{\theta}(z|x)$ 의 거리는 Kullback-Leibler (KL) divergence에 의해 측정되며 다음과 같이 정의된다.

$$KL(q_{\phi}(z|x) || p_{\theta}(z|x)) = \int \left( \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} \right) q_{\phi}(z|x) dz = E_q \left( \log \frac{q_{\phi}(z|x)}{p_{\theta}(z|x)} \right)$$

- KL divergence의 정의에 따라  $q_{\phi}(z|x) = p_{\theta}(z|x)$ 이면  $KL=0$ 가 되고  $\log x$ 가 concave 함수이므로 Jensen's inequality에 의해

$$-KL = \int \left( \log \frac{p_{\theta}(z|x)}{q_{\phi}(z|x)} \right) q_{\phi}(z|x) dz \leq \log \int \frac{p_{\theta}(z|x)}{q_{\phi}(z|x)} q_{\phi}(z|x) dz = \log 1 = 0$$

이므로  $KL \geq 0$ 이 된다.



## 03 Variational Autoencoder (VAE)

$$p_{\theta}(z|x) = \frac{p_{\theta}(x|z)p_{\theta}(z)}{p_{\theta}(x)}$$

이므로

$$\begin{aligned} KL(q_{\phi}(z|x)||p_{\theta}(z|x)) &= E_q(\log q_{\phi}(z|x) - \log p_{\theta}(x|z) - \log p_{\theta}(z) + \log p_{\theta}(x)) \\ &= \log p_{\theta}(x) - E_q(\log p_{\theta}(x|z)) + KL(q_{\phi}(z|x)||p_{\theta}(z)) \\ &= \log p_{\theta}(x) - \int (\log p_{\theta}(x|z)) q_{\phi}(z|x) dz + KL(q_{\phi}(z|x)||p_{\theta}(z)) \end{aligned}$$

이 된다.  $KL(q_{\phi}(z|x)||p_{\theta}(z|x))$ 를 최소화하기 위해서

$$\log p_{\theta}(x) - \int (\log p_{\theta}(x|z)) q_{\phi}(z|x) dz \quad \text{-----(1)}$$

와

$$KL(q_{\phi}(z|x)||p_{\theta}(z)) \quad \text{-----(2)}$$

를 최소화해야 한다.

## 03 Variational Autoencoder (VAE)

- (1)에서  $q_{\phi}(z|x)$ 는 encoder이고  $p_{\theta}(x|z)$ 는 decoder이므로 encoder와 decoder의 성능이 좋으면 (1)은 최소화 될 것이다.
- (1)을 재생 손실함수라고 하며 입력변수  $x$ 와 decoder를 통해 재생된  $\tilde{x}$ 의 거리로 손실함수를 평가한다.
- (2)를 추론손실(inference loss)이라고 하며  $q_{\phi}(z|x)$ 가 정규분포이므로  $p_{\theta}(z) \sim N(0, I)$ 를 선택한다.

그러므로

$$\text{KL}(q_{\phi}(z|x) || p_{\theta}(z)) = -\frac{1}{2} \sum_{j=1}^p (1 + \log \sigma_j^2 - \mu_j^2 - \sigma_j^2)$$

가 된다. 여기에서  $p$ 는  $z$ 의 크기이며  $\sigma_j^2$ 와  $\mu_j$ 는  $x$ 의 함수이다.

- Decoder  $p_{\theta}(x|z)$ 를 학습한 후, 새로운  $\tilde{x}$ 를 생성하기 위해 encoder  $q_{\phi}(z|x)$ 를 사용하지 않는다.  $p_{\theta}(z) \sim N(0, I)$ 에서 생성하기 때문이다.

Q & A